MDPI

Article

A feature construction method that combines Particle Swarm Optimization and Grammatical Evolution

Ioannis G. Solos[1,*], Alexandros Tzallas[2]

[1] Department of Informatics and Telecommunications, University of Ioannina, Greece;itsoulos@uoi.gr
[2] Department of Informatics and Telecommunications, University of Ioannina, Greece;tzallas@uoi.gr
* Correspondence: itsoulos@uoi.gr;

Abstract: The problem of data classification or data fitting is widely applicable in a multitude of scientific areas, and for this reason a number of machine learning models have been developed. However, in many cases, these models present problems of overfitting and cannot generalize satisfactorily to unknown data. Furthermore, in many cases, many of the features of the input data do not contribute to learning, or there may even be hidden correlations between the features of the dataset. The purpose of the proposed method is to significantly reduce data classification or regression errors through the usage of a technique that utilizes the Particle Swarm Optimization method and Grammatical Evolution. This method is divided into two phases: in the first phase, artificial features are constructed using Grammatical Evolution and the progress of the creation of these features is controlled by the Particle Swarm Optimization method. In addition, this new technique utilizes penalty factors to limit the generated features to a range of values to make training machine learning models more efficient. In the second phase of the proposed technique, these features are exploited to transform the original data set and then any machine learning method can be applied to this data set. The performance of the proposed method was measured on some benchmark datasets from the relevant literature. Also, the method was tested against a series of widely used machine learning models. Experiments performed show a significant improvement of 30% on average in classification datasets and an even greater improvement of 60% in data fitting datasets.

Keywords: Particle swarm optimization, Grammatical Evolution, Evolutionary techniques, Stochastic methods.

Citation: Solos, I.G.; Tzallas A Creating features using particle swarm optimization. *Journal Not Specified* **2022**, *1*, 0. https://doi.org/

Received:
Accepted:
Published:

1. Introduction

A multitude of everyday problems from various sciences can be treated as a problem of classification or data fitting problems, such as problems that appear in the fields of physics [1–4], chemistry [5–7], economics [8,9], environmental problems [10–12], medical problems [13,14] etc. In the relevant literature there is a wide range of techniques that one can use to handle such problems, such as the k nearest neighbors model (k-NN) [15,16], artificial neural networks (ANNs) [17,18], Radial Basis Function (RF) networks [19–21], Support Vector Machines (SVM) [22,23], decision trees [24,25] etc. Also, many practical problems have been tackled using machine learning approaches, such as prediction of non-breaking waves [26], energy conservation problems [27], prediction of scour depth at seawalls using Genetic Programming and neural networks [28]. Furthermore, machine learning models have been used in various complex tasks such as neural machine translation [29], oil distribution [30], image processing [31], robotics [32], hydracarbon production [33] etc. A brief description of the methods that can be used for classification datasets is given in the publication of Kotsiantis et al [34].

In the majority of cases, the machine learning models have a number of parameters that should be determined through some algorithms. These parameters include, for example, the weights of artificial neural networks, which can be estimated with techniques such as the Back Propagation method [35,36] or for example Genetic algorithms [37–39] but also

the hyperparameters of learning models that require different handling [40–42]. However, most of the time there are some problems in the parameterization of machine learning models:

- Requirement for large training time, which is proportional to the dimension of the input data. For example, in a neural network with one hidden layer equipped with 10 processing nodes and a provided dataset with 10 inputs, more than $N = 100$ parameters are required to build the neural network. Therefore, the size of the network will grow proportionally to the problem and therefore, longer training times will be required for the model.
- Another important problem presented in machine learning techniques is the fact that many models require significant storage space in the computer's memory for their parameters, and in fact, this space increases significantly with the increase in the dimension of the objective problem. For example, in the Bfgs [43] optimization method, $O(N^2)$ storage space will be required for the training model and for the partial derivatives required by the optimization method. This issue is thoroughly discussed in the paper of Verleysen et al [44]. Some common approaches proposed in order to reduce the dimension of the input datasets are the Principal Component Analysis (PCA) method [45–47] as well as the Minimum redundancy feature selection (MRMR) technique [48,49]. Moreover, Pourzangbar proposed a feature selection method [50] based on Genetic Programming for the determination of the most effective parameters on scour depth at seawalls. The proposed technique essentially, in addition to creating artificial features, also selects features at the same time, since it can remove from the final features those that will not bring significant benefits to the learning of the objective problem. Furthermore, Wang et al proposed an auto - encoder reduction method, applied on a series of large datasets[51].
- The problem of reduced performance of models on unknown data, is also known as the overfitting problem. The paper of Geman et al [52] as well the article of Hawkins [53] thoroughly discuss the topic of overfitting. Examples of techniques proposed to tackle this problem are the weight sharing methods [54,55], the pruning methods [56,57], weight elimination [58–60], weight decaying methods [61,62] etc.

This paper recommends a two-phase method for data classification or regression problems. In the first phase, a global optimization method directs the production of artificial features from the existing ones with the help of Grammatical Evolution [63]. Grammatical Evolution is a variation of genetic programming, where the chromosomes are production rules of the target BNF grammar and it has been used successfully in a variety of applications, such as music composition [64], economics [65], symbolic regression [66], robotics [67], caching algorithms [68] etc. The global optimization method used in this work was the particle swarm optimization (PESO) method [69,72,73]. The PESO method has been selected as the optimization method due to its simplicity and the small number of parameters that should be set. Also, the PESO method has been used in many difficult problems in all areas of the sciences, such as problems that arise in physics [74,75], chemistry [76,77], medicine [78,79], economics [80] etc. Furthermore, the PESO method was successfully applied recently in many practical problems such as flow shop scheduling [81], successful development of electric vehicle charging strategies [82], emotion recognition [83], robotics [84], optimal design of brace-viscous damper and pendulum tuned mass damper [85], application to high - dimensional expensive industrial problems [86], RFID readers [87], etc. The generated artificial features are nonlinear combinations of the original ones and any machine learning model can be used to effectively estimate their dynamics. In the present implementation, the RF network was used, since it is a widely tested machine learning model but also because its training is much faster compared to other models. In the second phase, the best features obtained from the first phase are also used to modify the test set of the objective problem and a machine learning method can be used to estimate the error in the control set.

The idea of creating artificial features using Grammatical Evolution was first introduced in the paper of Gavrilis et al [88] and it has been successfully applied on a series

of problems, such as Spam Identification [89], Fetal heart classification [90], epileptic oscillations[91], construction of Covid-19 predictive models [92], performance and early drop prediction for higher education students [93] etc.

Feature selection using neural networks has been also proposed in a series of papers, such as the work of Verikas and Bacauskiene [94] or the work of Kabir et al [95]. Moreover, Devi utilized a Simulated Annealing approach [96] to select the most important features for classification datasets. Also, Neshatian et al [97] developed a genetic algorithm that produces features using an entropy based fitness function.

The rest of this article is divided as follows: in section 2 the steps of the proposed method are fully described, in section 3 the used experimental datasets as well as the results obtained by the incorporation of the proposed method are outlined and finally in section 4 some conclusions are listed.

2. The proposed method

This section will introduce the main parts of the proposed two-step method. The first subsection will introduce the basics of Grammatical Evolution and give a complete example of building a valid function from a chromosome. Next, the process by which the Grammatical Evolution chromosomes can be used to create artificial features from existing ones will be presented in subsection 2.2 The procedure by which the fitness of each chromosome can be assessed is presented in subsection 2.3 and finally in subsection 2.4 the overall algorithm is presented along with a flowchart for its graphical representation.

2.1. The technique of Grammatical Evolution

The process of Grammatical Evolution uses chromosomes that represent production rules of the underlying BNF (Backus–Naur form) grammar[98] of the objective problem. BNF grammars have been widely used to describe the syntax of programming languages. Any BNF grammar is a set $G = (N, T, S, P)$, where

- The set N represents the non-terminal symbols of the grammar. Any non - terminal symbol is analyzed to a series of terminal symbols using the production rules of the grammar.
- T is the set of terminal symbols.
- The non - terminal symbol S represents the start symbol of the grammar.
- The set P contains the production rules of the grammar. Typically, any production rule is expressed in the form $A \to a$ or $A \to aB,\ A, B \in N,\ a \in T$.

The process that creates a valid program, starts from the symbol S and gradually replaces non-terminal symbols with the right hand of the selected production rule from the provided chromosome. The rule is selected with the following steps:

- In the first step, the next element is taken from the current chromosome. Let us denote it as V.
- The next production rule is selected through

$$\text{Rule} = V \bmod N_R$$

  where N_R is the total number of production rules for the current non – terminal symbol.

The BNF grammar for the proposed method is shown in Figure 1. Symbols in < > brackets denote non - terminal symbols that belong to set N. In every line of the grammar a production rule is shown for every non - terminal symbol. The numbers in parentheses represent the sequence number of the production rule for the corresponding non-terminal symbol. For example the non-terminal symbol <op> has 4 production rules, each leading to a terminating arithmetic operation symbol. The constant N is the dimension of the input dataset.

Figure 1. BNF grammar of the proposed method.

```
S::=<expr>   (0)
<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)
           .........
           | xN (N)
<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)
```

An example that produces a valid expression for the chromosome

$$x = [10, 12, 20, 8, 11]$$

with $N = 3$ is shown in Table 1. This chromosome represents a series of sequential numbers of production rules from the above grammar. The Grammar Evolution method takes the elements of the chromosome one by one and finds the corresponding production rule by taking the remainder of the division by the number of symbols of each non-terminal symbol. The final expression that created is $f(x) = \sin(x_3)$.

Table 1. Steps to produce a valid expression from the BNF grammar.

Expression	Chromosome	Operation
<expr>	10,12,20,8,11	$10 \mod 3 = 1$
<func>(<expr>)	12,20,8,11	$12 \mod 4 = 0$
sin(<expr>)	20,8,11	$20 \mod 3 = 2$
sin(<terminal>)	8,11	$8 \mod 2 = 0$
sin(<xlist>)	11	$11 \mod 3 = 2$
sin(x3)		

2.2. Feature construction

In the proposed technique the chromosomes of Grammatical Evolution are used as a set of functions that create artificial features as nonlinear combinations of the existing ones. This process can also be considered as a feature selection method, since possibly only a part of the original features can be used in the generated features. The proposed method creates N_f artificial features from the original ones and the process for any chromosome p is as follows:

1. **Divide** p into N_f parts. Every part is denoted as the p_i sub - particle.
2. **For** each sub - particle p_i a new artificial feature $g_i(\overrightarrow{x}, p_i)$ is constructed with the grammar of Figure 1 as a non - linear combination of the original set of features $\overrightarrow{x}$.

The final set of features will be considered as mapping functions of the original ones. For example the set:

$$g(\overrightarrow{x}, p) = \begin{cases} g_1(\overrightarrow{x}, p_1) & = & x_1^2 + 2x_3 \\ g_2(\overrightarrow{x}, p_2) & = & 3\cos(x_2) \end{cases}$$

is a set of mapping functions for the original features $\overrightarrow{x} = (x_1, x_2, x_3)$. However, sometimes the generated features can lead to extreme values and this will result in generalization problems from the used machine learning models. For this reason and in the present work, penalty factors are used so that the mapping functions do not lead to extreme values. These penalty factors also modify the fitness function that the Particle Swarm Optimization technique will minimize each time and are considered next.

2.3. Fitness calculation

Each chromosome in grammatical evolution produces a series of artificial features, which are nonlinear functions of existing features. However, an evaluation and a distinction should be made between those sets of features which will yield more in the learning process and those which will yield less. This is done by assessing the appropriateness of these features. In order to be able to compute the fitness of each group of features, the original training set should be reduced using the artificial features that have been produced and the following steps should be executed for any given chromosome p.

1. **Denote as** TO $= \left\{ \left(\overrightarrow{x_1}, y_1\right), \left(\overrightarrow{x_2}, y_2\right), \ldots, \left(\overrightarrow{x_M}, y_M\right) \right\}$ the original train set.
2. **Set** $V = 0$, the penalty factor
3. **Compute** the mapping function $g(\overrightarrow{x}, p)$ as suggested in subsection 2.2
4. **Set** TF=$\emptyset$,the modified train set
5. **For** $i = 1, \ldots, M$ **do**
   (a) **Set** $\widetilde{x}_i = g\left(\overrightarrow{x_i}, p\right)$
   (b) **Set** TF = TF $\cup$ $(\widetilde{x}_i, y_i)$
   (c) **If** $\|\widetilde{x}_i\| > L_{\max}$, **then** $V = V + 1$, where $L_{\max}$ a predefined positive value.
6. **End For**
7. **Train** an RF $C(x)$ with H processing NODES on TF and obtain the following error:

$$f_p = \sum_{j=1}^{M} \left(C(\widetilde{x}_i) - y_j\right)^2 \tag{1}$$

8. **Compute** the final fitness value:

$$f_p = f_p \times \left(1 + \lambda V^2\right) \tag{2}$$

where $\lambda > 0$.

2.4. The used PESO method

The mains steps for this algorithm are outlined in detail in Algorithm 1.

Algorithm 1 The base PESO algorithm executed in one processing unit.

1. **Initialization Step** .
   (a) **Set** iter $= 0$.
   (b) **Set** m as the total number of particles.
   (c) **Set** $\text{iter}_{\max}$ as the maximum number of iterations allowed.
   (d) **Initialize** randomly, the positions $\overrightarrow{p_1}, \overrightarrow{p_2}, ..., \overrightarrow{p_m}$ for the particles. For the grammatical evolution, every particle is a vector of randomly selected integers.
   (e) **Initialize** randomly the velocities $\overrightarrow{u_1}, \overrightarrow{u_2}, ..., \overrightarrow{u_m}$. For the current work every vector of velocities is a series of randomly selected integers in a the range $[u_{\min}, u_{\max}]$. In the current work $u_{\min} = -5$, $u_{\max} = 5$.
   (f) **For** $i = 1..m$ do $\overrightarrow{b_i} = \overrightarrow{p_i}$. The vector $\overrightarrow{b_i}$ denotes the best located position of particle $\overrightarrow{p_i}$.
   (g) **Set** $\overrightarrow{p}_{\text{best}} = \arg\min_{i \in 1..m} f(\overrightarrow{p_i})$
2. **Termination Check Step** . If iter $\geq \text{iter}_{\max}$ then got step 8.
3. **For** $i = 1..m$ **Do**
   (a) **Compute** the velocity $\overrightarrow{u_i}$ as a combination of the vectors $\overrightarrow{u_i}$, $\overrightarrow{p_i}$, $\overrightarrow{b_i}$ and $\overrightarrow{p}_{\text{best}}$
   (b) **Set** the new position for the particle as: $\overrightarrow{p_i} = \overrightarrow{p_i} + \overrightarrow{u_i}$
   (c) **Calculate** the fitness $f(p_i)$ for particle p_i using the procedure described in subsection 2.3.
   (d) **If** $f(\overrightarrow{p_i}) \leq f\left(\overrightarrow{b_i}\right)$ then $\overrightarrow{b_i} = \overrightarrow{p_i}$
4. **End For**
5. **Set** $\overrightarrow{p}_{\text{best}} = \arg\min_{i \in 1..m} f(\overrightarrow{p_i})$
6. **Set** iter $=$ iter $+ 1$.
7. **Goto** Step 2
8. **Test step**. Apply the mapping function of the best particle $\overrightarrow{p}_{\text{best}}$ to the test set of the problem and apply a machine learning model obtaining the corresponding test error.

The above calculates at every iteration the new position of the particle i using:

$$\overrightarrow{p_i} = \overrightarrow{p_i} + \overrightarrow{u_i} \tag{3}$$

In most cases the new velocity could be a linear combination of the previously computed velocity and the corresponding vectors for the best values $\overrightarrow{b_i}$ and $\overrightarrow{p}_{\text{best}}$ and it can be defined as:

$$\overrightarrow{u}_i = \omega \overrightarrow{u}_i + r_1 c_1 \left(\overrightarrow{b}_i - \overrightarrow{p_i}\right) + r_2 c_2 (\overrightarrow{p}_{\text{best}} - \overrightarrow{p_i}) \tag{4}$$

where

1. The variables r_1, r_2 are random numbers defined in $[0, 1]$.
2. The constants c_1, c_2 are defined in range $[1, 2]$.
3. The variable ω, commonly called inertia, was suggested by Shim and Earhart [69]. In the original paper, they proposed the idea that large values for the inertia coefficient can lead to a better exploration of the search space while smaller values of the coefficient lead to the method being concentrated around regions likely to contain the global minimum. Hence, in their work, the value of the inertia factor generally starts with large values and decreases with repetition. In the current work was computed through the following equation

$$\omega = 0.5 + \frac{r}{2} \tag{5}$$

The variable r is a a random number with $r \in [0, 1]$. This inertia calculation was proposed in [70]. With this calculation of the inertia variable, an even better exploration of the research

space is achieved with the randomness it introduces, something that was also found in the publication of Charolais and Solos [71].

A flowchart for the overall process is shown in Figure 2.

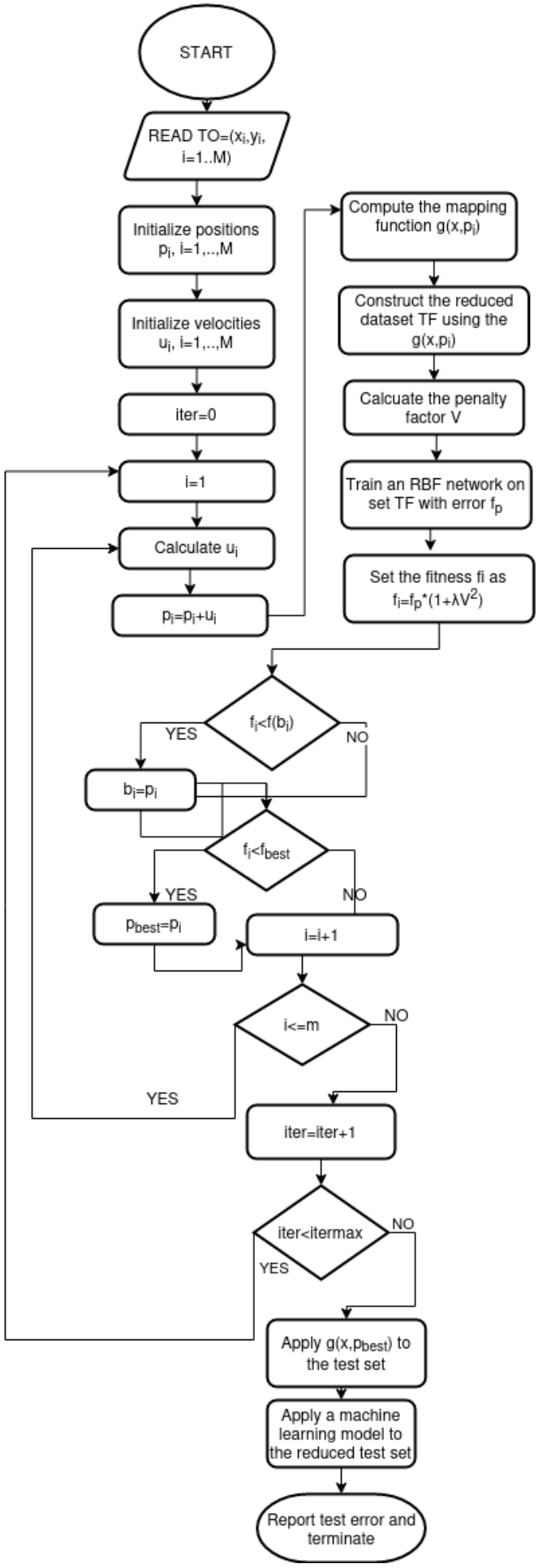

Figure 2. The flowchart for the proposed method.

3. Experiments

The ability of the proposed technique to produce effective artificial features for class prediction and feature learning will be measured in this section on a some datasets from the relevant literature. These problems have been studied by various researchers in the relevant literature and cover a wide range of research areas from Physics to Economics. These datasets comes from the relevant websites:

1. CI dataset repository, https://archive.ics.uci.edu/ml/index.php
2. Keel repository, https://sci2s.ugr.es/keel/datasets.php[99].
3. The Stat lib URL ftp://lib.stat.cmu.edu/datasets/index.html.

The proposed technique will be compared with a series of known machine learning techniques and the experimental results are then presented in the relevant tables.

3.1. Experimental datasets

The classification problems used in the experiments have as follows:

1. **Appendicitis**, a medical dataset [100,101].
2. **Australian** dataset [102], an dataset concerning economical transactions in banks.
3. **Balance** dataset, a dataset generated to model psychological experimental results[103].
4. **Bands** dataset, a dataset used in rotogravure printing [104].
5. **Dermatology** dataset [105], a medical dataset used to detect the type of Eryhemato-Squamous Disease.
6. **Hayes Roth** dataset [107].
7. **Heart** dataset [106], a medical dataset used to detect heart diseases.
8. **House Votes** dataset [108], a dataset related to the Congressional voting records of USA.
9. **Ionosphere** dataset, used to classify measurements from the ionosphere and it has been examined in a variety of research papers [109,110].
10. **Liver disorder** dataset [111,112], a dataset used for medical purposes.
11. **Mammography** dataset [113], a medical dataset used for breast cancer diagnosis.
12. **Parkinson's** dataset [114,115], a dataset used to detect the Parkinson's decease using voice measurements.
13. **Puma** dataset [116], a dataset used for medical purposes.
14. **Pop failures** dataset [117], a dataset related to meteorological data.
15. **Regions2** dataset, a medical dataset for liver biopsy images [118].
16. **Suharto** dataset [119], a medical dataset.
17. **Segment** dataset [120], a dataset related to image segmentation.
18. **BC** dataset [121] used in breast tumors.
19. **Wine** dataset, a dataset related to chemical analysis of wines [122,123].
20. **EEG** datasets [124,125], it is an EEG dataset and the following cases were used in the experiments:
    (a) Z_F_S,
    (b) ZOE_NF_S
    (c) ZONE_S.
21. **Zoo** dataset [126].

The regression datasets used in the relevant experiments have as follows:

1. **Abalone** dataset [127], a dataset used to predict the age of abalones.
2. **Airfoil** dataset, a dataset provided by NASA [128] obtained from a series of aerodynamic and acoustic tests.
3. **Baseball** dataset, a dataset related to the salary of baseball players.
4. **BK** dataset [129], a dataset that was used to calculate the points in a basketball game.
5. **BL** dataset, used in machine problems.
6. **Concrete** dataset [130], a civil engineering dataset to calculate The concrete compressive strength

7. **Dee** dataset, used to estimate the daily average price of KWh electricity energy in Spain.
8. **Diabetes** dataset, a medical dataset.
9. **Housing** dataset [131].
10. **FA** dataset, used to fit body fat to other measurements.
11. **MB** dataset [132].
12. **MORTGAGE** dataset, holding economic data from USA. The goal is to predict the 30-Year Conventional Mortgage Rate.
13. **PU** dataset, (Pyrimidines problem)[133].
14. **Quake** dataset, used to approximate the strength of a earthquake given its the depth of its focal point, its latitude and its longitude.
15. **Treasure** dataset, which contains Economic data information of USA, where the the goal is to predict 1-Month CD Rate.

3.2. Experimental results

In order to give greater credibility to the experiments carried out, the method of 10 - fold cross validation was incorporated for every experimental dataset. Every experiment was repeated 30 times using different seeds for the random generator each time. All the used code was implemented in ANSI C++ using the OPTIMUMS programming library for optimization purposes, freely available from https://github.com/itsoulos/OPTIMUMS/. For the classification datasets, the average classification error as measured in the test set is reported, while for the regression datasets the average regression error is reported. Here, by the term classification error, we mean the percentage of patterns in the test set that are classified into a different class than expected. Also, in every table, an additional column denoted as AVERAGE is added to show the average classification or regression error for the corresponding datasets. The values for the experimental parameters are shown in Table 2.

Table 2. The values for every parameter used in the experiments.

PARAMETER	MEANING	VALUE
m	Particles or Chromosomes	200
H	Number of hidden nodes	10
$\text{iter}_{\max}$	Maximum number of iterations	200
$L_{\max}$	Limit used in penalty calculation	100
λ	Penalty factor	100

In all techniques the same parameter sets and the same random numbers have been used in order to have a fair comparison of the experimental results.

In the first phase the proposed technique will generate new artificial features from the existing ones with the help of a technique guided by the partnership of Particle Swarm Optimization and Grammatical Evolution. In the second phase of the technique, these features will be used to modify the original test set to which any machine learning method can now be applied. In the second phase of the current work, two different techniques will be used: an RF neural network and an artificial neural network which will be trained using a genetic algorithm. This is done in order to establish the potential of the proposed procedure to improve the performance of both simple machine learning models and complex models. The proposed technique that created artificial features is compared on the same datasets against a series of well - known method from the relevant literature:

1. A genetic algorithm with m chromosomes, denotes as GENETIC in the experimental tables. This genetic algorithm is used to train an artificial neural network with H hidden nodes. After the termination of the genetic algorithm the local optimization method BEFOGS is applied to the best chromosome of the population.
2. The Radial Basis Function (RF) network [135] with H processing nodes.

3. The optimization method Adam [136], used to train an artificial neural network with H hidden nodes.
4. The Prop optimization method [137,138], used to train an artificial neural network with H hidden nodes.
5. The NEAT method (Revolution of Augmenting Typologies) [139].

The experimental results using the above methods on the classification datasets are shown in Table 3 and the results for the regression datasets are illustrated in Table 4.

Table 3. Average classification error for the classification datasets using the well - known methods.

DATASET	GENETIC	RF	ADAM	PROP	NEAT
Appendicitis	18.10%	12.23%	16.50%	16.30%	17.20%
Australian	32.21%	34.89%	35.65%	36.12%	31.98%
Balance	8.97%	33.42%	7.87%	8.81%	23.14%
Bands	35.75%	37.22%	36.25%	36.32%	34.30%
Dermatology	30.58%	62.34%	26.14%	15.12%	32.43%
Hayes Roth	56.18%	64.36%	59.70%	37.46%	50.15%
Heart	28.34%	31.20%	38.53%	30.51%	39.27%
House Votes	6.62%	6.13%	7.48%	6.04%	10.89%
Ionosphere	15.14%	16.22%	16.64%	13.65%	19.67%
Liver disorder	31.11%	30.84%	41.53%	40.26%	30.67%
Lymography	23.26%	25.31%	29.26%	24.67%	33.70%
Mammography	19.88%	21.38%	46.25%	18.46%	22.85%
Parkinson's	18.05%	17.42%	24.06%	22.28%	18.56%
Puma	32.19%	25.78%	34.85%	34.27%	34.51%
Pop failures	5.94%	7.04%	5.18%	4.81%	7.05%
Regions2	29.39%	38.29%	29.85%	27.53%	33.23%
Suharto	34.86%	32.19%	34.04%	34.90%	34.51%
Segment	57.72%	59.68%	49.75%	52.14%	66.72%
BC	8.56%	7.27%	35.35%	21.57%	12.88%
Wine	19.20%	31.41%	29.40%	30.73%	25.43%
Z_F_S	10.73%	13.16%	47.81%	29.28%	38.41%
ZOE_NF_S	8.41%	9.02%	47.43%	6.43%	43.75%
ZONE_S	2.60%	4.03%	11.99%	27.27%	5.44%
ZOO	16.67%	21.93%	14.13%	15.47%	20.27%
AVERAGE	**22.94%**	**26.78%**	**30.24%**	**24.60%**	**28.63%**

Table 4. Average regression error using the well - known methods for the regression datasets.

DATASET	GENETIC	RF	ADAM	PROP	NEAT
ABALONE	7.17	7.37	4.30	4.55	9.88
AIRFOIL	0.003	0.27	0.005	0.002	0.067
BASEBALL	103.60	93.02	77.90	92.05	100.39
BK	0.027	0.02	0.03	1.599	0.15
BL	5.74	0.01	0.28	4.38	0.05
CONCRETE	0.0099	0.011	0.078	0.0086	0.081
DEE	1.013	0.17	0.63	0.608	1.512
DIABETES	19.86	0.49	3.03	1.11	4.25
HOUSING	43.26	57.68	80.20	74.38	56.49
FA	1.95	0.02	0.11	0.14	0.19
MB	3.39	2.16	0.06	0.055	0.061
MORTGAGE	2.41	1.45	9.24	9.19	14.11
PU	1.21	0.02	0.09	0.039	0.075
QUAKE	0.04	0.071	0.06	0.041	0.298
TREASURY	2.929	2.02	11.16	10.88	15.52
AVERAGE	**12.84**	**10.30**	**11.70**	**12.44**	**12.70**

The results using the proposed method and for the construction of 2,3 and 4 artificial features are presented in the relevant tables, 5 and 6. The RF column represents the experimental results in which, after the construction of the artificial features, a RF network with H processing nodes is applied on the modified dataset. Also, the column GENETIC in tables 5, 6 stands for the results obtained by the application of a genetic algorithm with m chromosomes to the modified dataset, when the feature creation procedure has been finished.

Table 5. Experimental results for the classification datasets using the proposed method. Number in cells denote average classification error as measured on the test set.. The variable f corresponds to the number of artificial features created by the proposed method.

	$f = 2$		$f = 3$		$f = 4$	
DATASET	RF	GENETIC	RF	GENETIC	RF	GENETIC
APPENDICITIS	15.40%	14.33%	16.90%	15.77%	15.97%	17.30%
AUSTRALIAN	15.49%	14.48%	14.53%	15.33%	14.75%	15.97%
BALANCE	16.67%	2.89%	22.54%	4.94%	17.26%	4.62%
BANDS	38.09%	38.13%	37.09%	39.22%	37.22%	35.51%
DERMATOLOGY	41.58%	30.37%	35.46%	25.44%	40.45%	21.97%
HAYES ROTH	37.41%	27.92%	38.10%	25.74%	39.59%	25.82%
HEART	21.53%	17.13%	17.64%	16.87%	19.63%	15.69%
HOUSE VOTES	6.36%	3.78%	7.17%	3.25%	4.25%	3.52%
IONOSPHERE	10.32%	10.17%	10.12%	10.01%	11.42%	9.02%
LIVERDISORDER	34.23%	32.33%	35.84%	32.97%	35.93%	30.74%
LYMOGRAPHY	34.93%	28.67%	32.00%	23.00%	29.00%	23.83%
MAMMOGRAPHIC	16.92%	16.51%	16.47%	16.35%	17.54%	16.50%
PARKINSONS	11.14%	13.00%	11.30%	11.11%	12.95%	9.42%
PIMA	22.85%	22.76%	24.93%	24.67%	24.25%	24.20%
POPFAILURES	7.32%	7.41%	6.96%	7.62%	5.96%	5.67%
REGIONS2	28.52%	26.84%	24.91%	25.28%	25.35%	24.93%
SAHEART	29.29%	28.63%	28.92%	30.31%	28.25%	30.25%
SEGMENT	52.69%	45.59%	46.83%	41.06%	50.15%	39.52%
WDBC	5.00%	4.66%	5.76%	4.97%	5.13%	3.84%
WINE	8.92%	7.22%	6.76%	5.75%	6.00%	5.86%
Z_F_S	7.91%	8.37%	7.89%	7.67%	5.21%	6.86%
ZOE_NF_S	6.90%	6.85%	6.95%	5.65%	6.24%	5.28%
ZONE_S	3.08%	3.40%	2.44%	2.52%	3.47%	3.33%
ZOO	26.47%	7.83%	31.73%	10.03%	28.70%	11.57%
AVERAGE	**20.79%**	**17.47%**	**20.39%**	**16.90%**	**20.19%**	**16.30%**

Table 6. Experimental results on the regression datasets using the proposed method. The number in cells denote average regression error as measured on the test set.The variable f corresponds to the number of artificial features created by the proposed method.

	$f = 2$		$f = 3$		$f = 4$	
DATASET	RF	GENETIC	RF	GENETIC	RF	GENETIC
ABALONE	4.361	3.518	4.159	3.839	4.859	3.786
AIRFOIL	0.003	0.001	0.003	0.001	0.003	0.001
BASEBALL	66.00	53.74	60.79	57.04	66.19	61.69
BK	0.022	0.031	0.021	0.029	0.019	0.023
BL	0.413	0.0001	0.019	0.007	0.043	0.011
CONCRETE	0.008	0.006	0.007	0.005	0.008	0.004
DEE	0.259	0.252	0.339	0.286	0.609	0.5
DIABETES	0.611	0.832	0.634	1.411	0.857	1.157
HOUSING	22.387	15.583	18.614	13.602	14.83	13.208
FA	0.056	0.011	0.015	0.011	0.015	0.012
MB	0.258	0.087	0.115	0.078	0.342	0.072
MORTGAGE	0.621	0.046	0.65	0.037	0.078	0.04
PU	2.894	0.14	0.936	0.029	0.724	0.031
QUAKE	0.069	0.036	0.057	0.037	0.04	0.037
TREASURY	0.912	0.088	0.874	0.084	0.173	0.076
AVERAGE	**6.59**	**4.96**	**5.82**	**5.10**	**5.92**	**5.38**

The experimental results are of great interest, as one can see from their careful study, the proposed technique is able to significantly reduce the error in the corresponding test sets. Especially in the case of regression problems, the reduction in error is on average greater than 50%. Moreover, the usage of a neural network trained by a genetic algorithm on the modified datasets, gives clearly better results than the use of an RF neural network, especially in the classification datasets. An additional test was performed for the regression datasets, where the number of particles in PESO algorithm increases from 100 to 400 and the results are graphically illustrated in Figure 3.

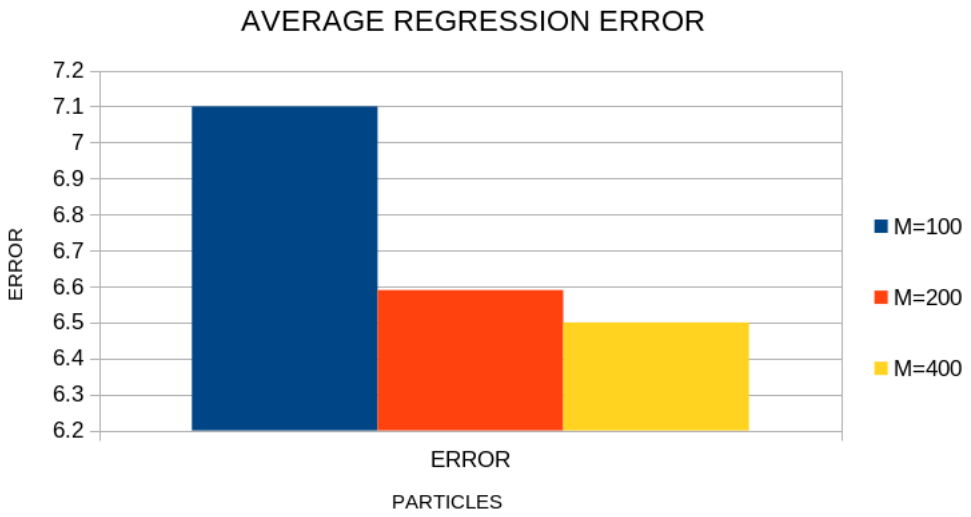

Figure 3. Average regression error for all regression datasets using the proposed method. The number of PESO particles increases from 100 to 400 and the number of constructed features was set to $f = 2$.

Judging from the results, we can observe that the selection of 200 particles in the experimental results was an optimal choice and a compromise between the speed and efficiency of the method, as adding another 200 particles to the Particle Swarm Optimization did not significantly improve the efficiency of the proposed method.

Moreover, a graphical comparison between the genetic algorithm when applied to artificial datasets and the genetic algorithm when applied to the original classification datasets is given in Figure 4. The same graphical comparison is also shown for the RF model in Figure 5. And in these figures, the ability of the proposed method to drastically reduce learning error through the construction of artificial features is evident.

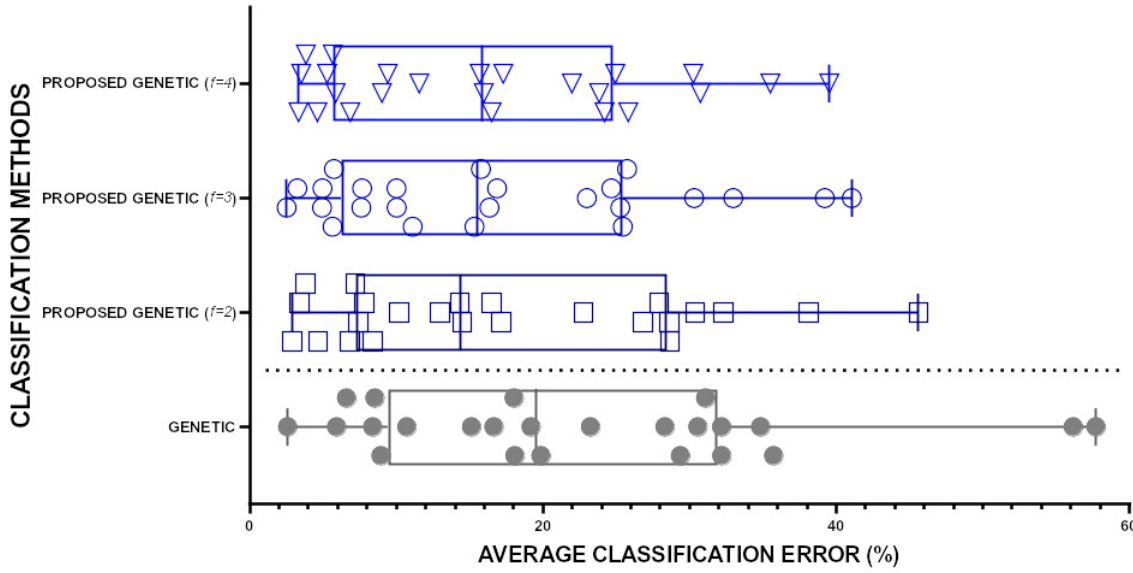

Figure 4. Comparison of the proposed Genetic classification algorithm for the construction of 2,3 and 4 artificial features (blue color) vs. a welkin Genetic classification algorithm (Grey color) to the 24 different datasets.

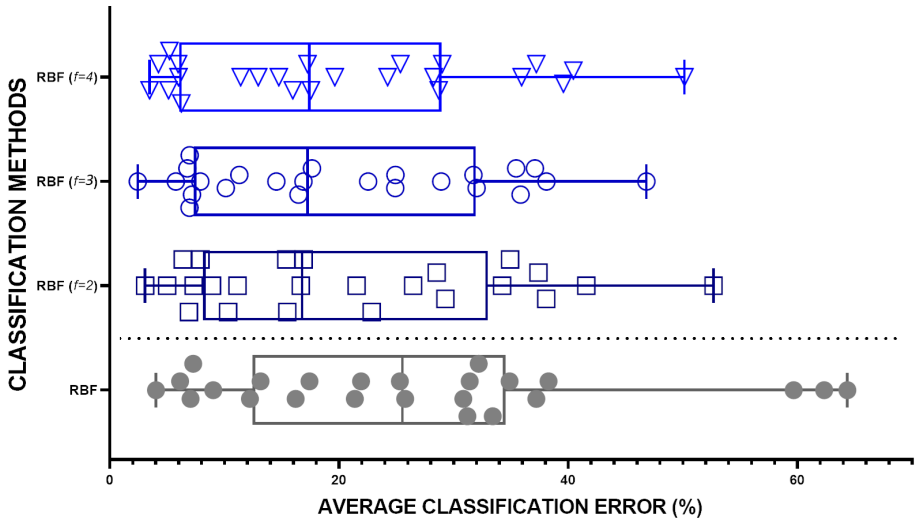

Figure 5. Comparison of the proposed RF classification network for the construction of 2,3 and 4 artificial features (blue color) vs. a welkin RF classification network (Grey color) to the 24 different datasets.

Also, in figure 6 a comparison for the regression datasets is outlined between the proposed method with the application of the genetic algorithm in second phase and the genetic algorithm on the original regression datasets.

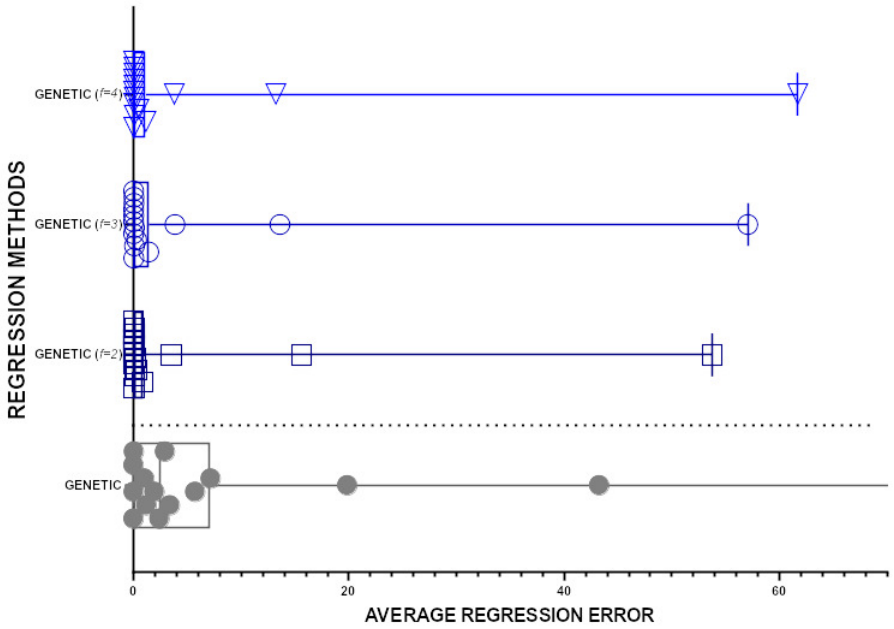

Figure 6. Comparison of the proposed algorithm for the construction of 2,3 and 4 artificial features (blue color) vs. the genetic algorithm (Grey color) for the regression datasets.

Finally, using the Wilcox on sandbank test a comparison is made between the proposed method and all mentioned machine learning methods for the classification datasets. This comparison is graphically outlined in Figure 7.

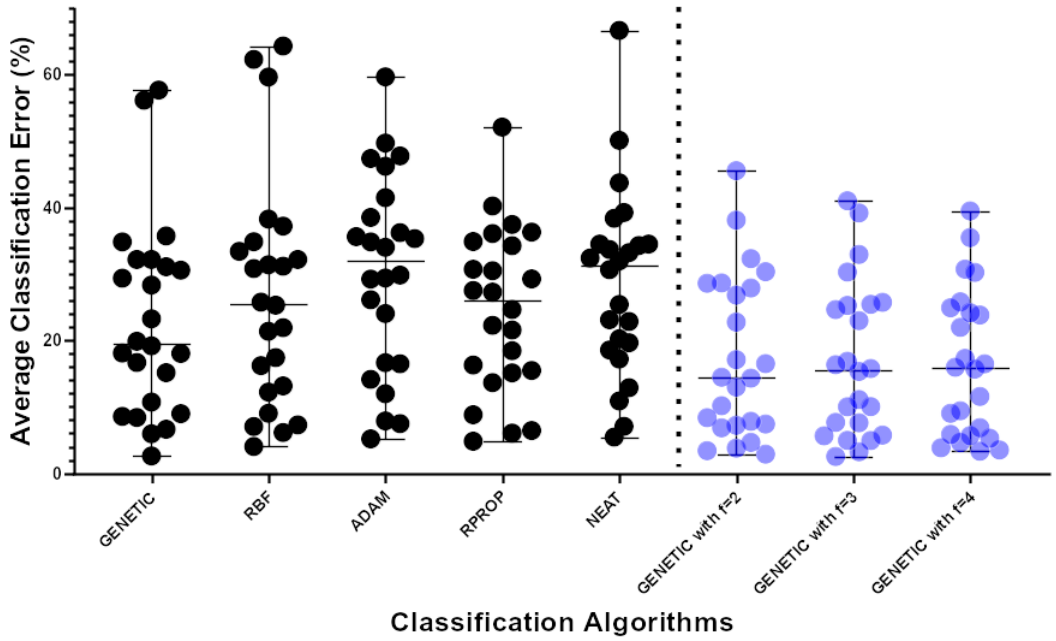

Figure 7. The comparison between the proposed method for constructing 2, 3, and 4 artificial features (blue colour) and several welkin classification methods (black colour) on 24 different datasets revealed that the proposed method demonstrated superior performance with statistical significance. The results were obtained using the Wilcox on sandbank test. Across all 24 datasets, the proposed method consistently outperformed the welkin classification methods regarding classification error .

These results suggest that the proposed method has a distinct advantage over the welkin classification methods when it comes to constructing artificial features for classification tasks. It offers improved performance and provides a more effective solution for these datasets

4. Conclusions

A hybrid technique that utilizes a Particle Swarm Optimizer and a feature creation method using Grammatical Evolution was introduced here. The proposed method can identify possible dependencies between the original features and can also reduce the number of required features to a limited number. Also, the method can remove from the set of features those features that may not contribute to the learning of the data set by some machine learning model. In addition, to make learning more efficient, the values of the generated features are bounded within a value interval using penalty factors. The constructed features are evaluated in terms of their effectiveness with the help of a fast machine learning model such as the RF network, even though other more effective models could also be used. Among the advantages of the proposed procedure is the fact that it does not require any prior knowledge of the data set to which it will be applied and furthermore, the procedure is exactly the same whether it is a data classification problem or a data fitting problem. The Particle Swarm Optimization method was used for the production of the characteristics as it has been proven by the relevant literature to be an extremely efficient technique and has a limited number of parameters that must be defined by the user.

The current work was applied on an extended series of widely used datasets from various fields and was compared against some machine learning models on the same datasets. From the experimental results, it was seen that the proposed technique dramatically improves the performance of traditional learning techniques when applied to artificial features. The proposed two-stage technique generated artificial features in the first stage guided by the Particle Swarm Optimization technique, and in the second stage, either a neural network trained by genetic algorithm or a RF network was used in the modified test set. In both cases, the improvement from artificial feature generation in control error was significant for each learning model. This improvement reaches an average of 30% for data classification and 50% for data fitting problems. In fact, in many cases, the improvement in test error exceeds 75%. Moreover, the method appears quite robust, since increasing the number of particles in the particle swarm optimization method did not show to significantly reduce the average error in the test sets. Furthermore, increasing the number of features constructed does not seem to have a dramatic effect on the performance of the method, which means that the method is able to achieve good generalization results even with a limited number of features, which in turn leads to greatly reducing the number of dimensions of the original problem. Future work on the method may include the use of parallel techniques for feature construction to drastically reduce the required execution time.

Author Contributions: I.G.T. and A.T. conceived of the idea and the methodology and I.G.T has implemented the corresponding software. I.G.T. conducted the experiments, employing objective functions as test cases, and provided the comparative experiments. A.T. has performed the necessary statistical tests. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Institutional Review Board Statement: Not applicable.

Acknowledgments: The experiments of this research work were performed at the high performance computing system established at Knowledge and Intelligent Computing Laboratory, Department of Informatics and Telecommunications, University of Ioannina, acquired with the project “Educational

Laboratory equipment of TEI of Epirus" with MIS 5007094 funded by the Operational Programme "Epirus" 2014–2020, by ERDF and national funds.

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

Sample Availability: Not applicable.

References

1. E.M. Metodiev, B. Nachman, J. Thaler, Classification without labels: learning from mixed samples in high energy physics. J. High Energ. Phys. 2017, article number 174, 2017.
2. P. Baldi, K. Cranmer, T. Faucett et al, Parameterized neural networks for high-energy physics, Eur. Phys. J. C **76**, 2016.
3. A. K. Aniyan, K. Thorat, Classifying Radio Galaxies with the Convolutional Neural Network, The Astrophysical Journal Supplement Series **230**, 20, 2017.
4. G. Carleo,M. Troyer, Solving the quantum many-body problem with artificial neural networks, Science **355**, pp. 602-606, 2017.
5. J.N. Wei, D. Duvenaud, A. A. Guzik, Neural Networks for the Prediction of Organic Chemistry Reactions, ACS Central Science **2**, pp. 725-732, 2016.
6. C.Qi, A. Fourie, Q. Chen, Neural network and particle swarm optimization for predicting the unconfined compressive strength of cemented paste backfill, Construction and Building Materials **159**, pp. 473-478, 2018.
7. H. Gao, T. J. Struble, C.W. Coley, Y. Wang, W.H. Green, K. F. Jensen, Using Machine Learning To Predict Suitable Conditions for Organic Reactions, ACS Central Science **4**, pp. 1465-1476, 2018.
8. R. Hafezi, J. Shahrabi, E. Hadavandi, A bat-neural network multi-agent system (BNNMAS) for stock price prediction: Case study of DAX stock price, Applied Soft Computing **29**, pp. 196-210, 2015.
9. X. Pang, Y. Zhou, P. Wang et al, An innovative neural network approach for stock market prediction, J Supercomput **76**, pp. 2098–2118, 2020.
10. A. Russo, P.G. Lind, F. Raischel, R. Trigo, M. Mendes, Neural network forecast of daily pollution concentration using optimal meteorological data at synoptic and local scales, Atmospheric Pollution Research **6**, pp. 540-549, 2015.
11. A. Azid, H. Juahir, M.E. Toriman et al., Prediction of the Level of Air Pollution Using Principal Component Analysis and Artificial Neural Network Techniques: a Case Study in Malaysia, Water Air Soil Pollut **225**, pp. 2063, 2014.
12. H. Maleki, A. Sorooshian, G. Goudarzi et al., Air pollution prediction by using an artificial neural network model, Clean Techn Environ Policy **21**, pp. 1341–1352, 2019.
13. Igor I. Baskin, David Winkler and Igor V. Tetko, A renaissance of neural networks in drug discovery, Expert Opinion on Drug Discovery **11**, pp. 785-795, 2016.
14. Ronadl Bartzatt, Prediction of Novel Anti-Ebola Virus Compounds Utilizing Artificial Neural Network (ANN), Chemistry Faculty Publications **49**, pp. 16-34, 2018.
15. Z.-ga Liu, Q. Pan, J. Dezert, A new belief-based K-nearest neighbor classification method, Pattern Recognition **46**, pp. 834-844, 2012.
16. Z. Deng, X. Zhu, D. Cheng, M. Zong, S. Zhang, Efficient kNN classification algorithm for big data, Neurocomputing **195**, pp. 143-148, 2016.
17. D. Graupe, Principles of artificial neural networks. Vol. 7. World Scientific, 2013.
18. S. Samarasinghe, Neural networks for applied sciences and engineering: from fundamentals to complex pattern recognition. Crc Press, 2016.
19. H. Yu, T. Xie, S. Paszczynski, B. M. Wilamowski, Advantages of Radial Basis Function Networks for Dynamic System Design, in IEEE Transactions on Industrial Electronics **58**, pp. 5438-5450, 2011.
20. D. Chen, Research on Traffic Flow Prediction in the Big Data Environment Based on the Improved RF Neural Network, IEEE Transactions on Industrial Informatics **13**, pp. 2000-2008, 2017.
21. Z. Yang, M. Mourshed, K. Liu, X. Xu, S. Feng, A novel competitive swarm optimized RF neural network model for short-term solar power generation forecasting, Neurocomputing **397**, pp. 415-421, 2020.
22. A. Iranmehr, H. Masnadi-Shirazi, N. Vasconcelos, Cost-sensitive support vector machines, Neurocomputing **343**, pp. 50-64, 2019.
23. J. Cervantes, F.G. Lamont, L.R. Mazahua, A. Lopez, A comprehensive survey on support vector machine classification: Applications, challenges and trends, Neurocomputing **408**, pp. 189-215, 2020.
24. S.B. Kotsiantis, Decision trees: a recent overview, Artif Intell Rev **39**, pp. 261–283, 2013.
25. D. Bertsimas, J. Dunn, Optimal classification trees, Mach Learn **106**, pp. 1039–1082, 2017.
26. A. Pourzangbar, M.A. Losada, A. Saber, L. Rasoul Ahari, P. Larroudé, M. Vaezi, M. Brocchini, Prediction of non-breaking wave induced scour depth at the trunk section of breakwaters using Genetic Programming and Artificial Neural Networks, Coastal Engineering **121**, pp. 107-118, 2017.
27. F. Afsarian, A. Saber, A. Pourzangbar, A.G. Olabi, M.A. Khanmohammadi, Analysis of recycled aggregates effect on energy conservation using M5´ model tree algorithm, Energy 156, pp. 264-277, 2018.
28. A. Pourzangbar, A. Saber, A.Y. Bakhtiary, L.R. Ahari, Predicting scour depth at seawalls using GP and ANNs, Journal of Hydroinformatics **19**, pp. 349–363, 2017.

29. X. Liu, J. He, M. Liu Z. Yin, L. Yin, W.A. Zheng, Scenario-Generic Neural Machine Translation Data Augmentation Method. Electronics **12**, 2320, 2023.
30. X. Xu, Z. Lin, X. Li, C. Shang, Q. Shen, Multi-objective robust optimisation model for MDVRPLS in refined oil distribution, International Journal of Production Research **60**, pp. 6772-6792, 2022.
31. S. Lu, Y. Ding, M. Liu, et al, Multiscale Feature Extraction and Fusion of Image and Text in VQA. Int J Comput Intell Syst **16**, 54, 2023.
32. C. Zheng, Y. An, Z. Wang, H. Wu, X. Qin, B. Eynard, Y. Zhang, Hybrid offline programming method for robotic welding systems, Robotics and Computer-Integrated Manufacturing **73**, 102238, 2022.
33. K. Zhang, Z. Wang, G. Chen, L. Zhang, Y. Yang, C. Yao, J. Wang, J. Yao, Training effective deep reinforcement learning agents for real-time life-cycle production optimization, Journal of Petroleum Science and Engineering **208** Part E, 109766, 2022.
34. S.B. Kotsiantis, I.D. Zaharakis, P.E. Pintelas, Machine learning: a review of classification and combining techniques, Artif Intell Rev **26**, pp. 159–190, 2006.
35. J. Li, Jh Cheng, Jy. Shim, F. Huang, Brief Introduction of Back Propagation (BP) Neural Network Algorithm and Its Improvement. In: Jin, D., Lin, S. (eds) Advances in Computer Science and Information Engineering. Advances in Intelligent and Soft Computing, vol 169. Springer, Berlin, Heidelberg, 2012.
36. T. Chen and S. Zhong, Privacy-Preserving Backpropagation Neural Network Learning, IEEE Transactions on Neural Networks **20**, , pp. 1554-1564, 2009.
37. A. Sedki, D. Ouazar, E. El Mazoudi, Evolving neural network using real coded genetic algorithm for daily rainfall–runoff forecasting, Expert Systems with Applications **36**, pp. 4523-4527, 2009.
38. F. Ruehle, Evolving neural networks with genetic algorithms to study the string landscape, J. High Energ. Phys. **2017**, 38, 2017.
39. A. Majdi, M. Beiki, Evolving neural network using a genetic algorithm for predicting the deformation modulus of rock masses, International Journal of Rock Mechanics and Mining Sciences **47**, pp. 246-253, 2010.
40. A. Lentzas, C. Nalmpantis, D. Vrakas, Hyperparameter Tuning using Quantum Genetic Algorithms, In: 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI), Portland, OR, USA, pp. 1412-1416, 2019.
41. I.D. Raji, H. Bello-Salau, I.J. Umoh, A.J. Onumany, M.A. Adegboye, A.T. Salawudeen, Simple Deterministic Selection-Based Genetic Algorithm for Hyperparameter Tuning of Machine Learning Models. Applied Sciences **12**, 1186, 2022.
42. D.L. Shanthi, N. Chethan, Genetic Algorithm Based Hyper-Parameter Tuning to Improve the Performance of Machine Learning Models. SN COMPUT. SCI. **4**, 119, 2023.
43. M.J.D Powell, A Tolerant Algorithm for Linearly Constrained Optimization Calculations, Mathematical Programming **45**, pp. 547-566, 1989.
44. M. Verleysen, D. Francois, G. Simon, V. Wertz, On the effects of dimensionality on data analysis with neural networks. In: Mira J., Álvarez J.R. (eds) Artificial Neural Nets Problem Solving Methods. IWANN 2003. Lecture Notes in Computer Science, vol 2687. Springer, Berlin, Heidelberg. 2003.
45. Burcu Erkmen, Tülay Yıldırım, Improving classification performance of sonar targets by applying general regression neural network with PCA, Expert Systems with Applications **35**, pp. 472-475, 2008.
46. Jing Zhou, Aihuang Guo, Branko Celler, Steven Su, Fault detection and identification spanning multiple processes by integrating PCA with neural network, Applied Soft Computing **14**, pp. 4-11, 2014.
47. Ravi Kumar G., Nagamani K., Anjan Babu G., A Framework of Dimensionality Reduction Utilizing PCA for Neural Network Prediction. In: Borah S., Emilia Balas V., Polkowski Z. (eds) Advances in Data Science and Management. Lecture Notes on Data Engineering and Communications Technologies, vol 37. Springer, Singapore. 2020.
48. Hanchuan Peng, Fuhui Long, and Chris Ding, Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy, IEEE Transactions on Pattern Analysis and Machine Intelligence **27**, pp.1226-1238, 2005.
49. M. Radovic, M. Ghalwash, N. Filipovic, Minimum redundancy maximum relevance feature selection approach for temporal gene expression data. BMC Bioinformatics **18**, 9, 2017.
50. A. Pourzangbar, Determination of the most effective parameters on scour depth at seawalls using genetic programming (GP). In The 10th International Conference on Coasts, Ports and Marine Structures (ICOPMASS 2012), Tehran, Iran, 9p, November 2012.
51. Y. Wang, H. Yao, S. Zhao, Auto-encoder based dimensionality reduction, Neurocomputing **184**, pp. 232-242, 2016.
52. S. Geman, E. Bienenstock and R. Doursat, Neural networks and the bias/variance dilemma, Neural Computation 4 , pp. 1 - 58, 1992.
53. Douglas M. Hawkins, The Problem of Overfitting, J. Chem. Inf. Comput. Sci. **44**, pp. 1–12, 2004.
54. J.K. Kim, M.Y. Lee, J.Y. Kim, B.J. Kim, J.H. Lee, An efficient pruning and weight sharing method for neural network. In 2016 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia) (pp. 1-2). IEEE, 2016.
55. W. Roth, F. Pernkopf, Bayesian Neural Networks with Weight Sharing Using Dirichlet Processes, IEEE Transactions on Pattern Analysis and Machine Intelligence **42**, pp. 246-252, 2020.
56. M. Augasta and T. Kathirvalavakumar, Pruning algorithms of neural networks — a comparative study, Central European Journal of Computer Science, 2003.
57. N.M. Hewahi, Neural network pruning based on input importance. Journal of Intelligent & Fuzzy Systems **37**, pp. 2243-2252, 2019.

58. F. Hergert, W. Finnoff, H. G. Zimmermann, A comparison of weight elimination methods for reducing complexity in neural networks, In: Proceedings 1992] IJCNN International Joint Conference on Neural Networks, Baltimore, MD, USA, pp. 980-987 vol.3, 1992.
59. M. Cottrell, B. Girard, Y. Girard, M. Mangeas, C. Muller, Neural modeling for time series: A statistical stepwise method for weight elimination, IEEE Transactions on Neural Networks **6**, pp. 1355-1364, 1995.
60. C. M. Ennett, M. Frize, Weight-elimination neural networks applied to coronary surgery mortality prediction, IEEE Transactions on Information Technology in Biomedicine **7**, pp. 86-92, 2003.
61. 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.
62. I.G. Solos, A. Tzallas, D. Tsalikakis, Evolutionary Based Weight Decaying Method for Neural Network Training. Neural Process Lett **47**, pp. 463–473, 2018.
63. M. O'Neill, C. Ryan, Grammatical evolution, IEEE Trans. Evol. Comput. **5,**pp. 349–358, 2001.
64. R. Loughran, J. McDermott, M. O'Neill, Tonality driven piano compositions with grammatical evolution, In: 2015 IEEE Congress on Evolutionary Computation (CEC), Sendai, Japan, pp. 2168-2175, 2015.
65. P. Gabrielsson, U. Johansson, R. König, Co-evolving online high-frequency trading strategies using grammatical evolution, In: 2014 IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr), London, UK, pp. 473-480, 2014.
66. M.S. Ali, M. Kshirsagar, E. Naredo, C. Ryan, Towards Automatic Grammatical Evolution for Real-world Symbolic Regression. In IJCCI (pp. 68-78), 2021.
67. 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.
68. J. Díaz Álvarez, J.M. Colmenar, J.L. Risco-Martín et al, Optimizing L1 cache for embedded systems through grammatical evolution. Soft Comput **20**, pp. 2451–2465, 2016.
69. J. Kennedy and R. Earhart, "Particle swarm optimization," Proceedings of ICNN'95 - International Conference on Neural Networks, 1995, pp. 1942-1948 vol.4, doi: 10.1109/ICNN.1995.488968.
70. R.C. Earhart, Y.H. Shim, Tracking and optimizing dynamic systems with particle swarms, in: Congress on Evolutionary Computation, Korea, 2001.
71. V. Charolais, I.G. Solos, Toward an Ideal Particle Swarm Optimizer for Multidimensional Functions, Information **13**, 217, 2022.
72. Riccardo Poli, James Kennedy kennedy, Tim Blackwell, Particle swarm optimization An Overview, Swarm Intelligence **1**, pp 33-57, 2007.
73. Ioan Cristian Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection, Information Processing Letters **85**, pp. 317-325, 2003.
74. Anderson Alvarenga de Moura Meneses, Marcelo Dornellas, Machado Roberto Schirru, Particle Swarm Optimization applied to the nuclear reload problem of a Pressurized Water Reactor, Progress in Nuclear Energy **51**, pp. 319-326, 2009.
75. Y. Wang, M. Miao, J. Lv, L. Zhu, K. Yin, H. Liu, Y. Ma, An effective structure prediction method for layered materials based on 2D particle swarm optimization algorithm. The Journal of chemical physics **137**, 224108, 2012.
76. X. Chen, W. Du, R. Qi, F. Qian, H. Tianfield, Hybrid gradient particle swarm optimization for dynamic optimization problems of chemical processes. Asia-Pac. J. Chem. Eng **8**, pp. 708-720, 2013.
77. H. Fang, J. Zhou, Z. Wang et al, Hybrid method integrating machine learning and particle swarm optimization for smart chemical process operations, Front. Chem. Sci. Eng. **16**, pp. 274–287, 2022.
78. P.C. Chang, J.J. Lin, C.H. Liu, An attribute weight assignment and particle swarm optimization algorithm for medical database classifications, Computer Methods and Programs in Biomedicine **107**,pp. 382-392, 2012.
79. R. Radha, R. Gopalakrishnan, A medical analytical system using intelligent fuzzy level set brain image segmentation based on improved quantum particle swarm optimization, Microprocessors and Microsystems **79**, 103283, 2020.
80. Jong-Bae Park, Yun-Won Jeong, Joong-Rin Shin, Kwang Y. Lee, An Improved Particle Swarm Optimization for Nonconvex Economic Dispatch Problems, IEEE Transactions on Power Systems **25**, pp. 156-16**216**6, 2010.
81. B. Liu, L. Wang, Y.H. Jin, An Effective PSO-Based Memetic Algorithm for Flow Shop Scheduling, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **37**, pp. 18-27, 2007.
82. J. Yang, L. He, S. Fu, An improved PSO-based charging strategy of electric vehicles in electrical distribution grid, Applied Energy **128**, pp. 82-92, 2014.
83. K. Mistry, L. Zhang, S. C. Neoh, C. P. Lim, B. Fielding, A Micro-GA Embedded PESO Feature Selection Approach to Intelligent Facial Emotion Recognition, IEEE Transactions on Cybernetics. **47**, pp. 1496-1509, 2017.
84. S. Han, X. Shan, J. Fu, W. Xu, H. Mi, Industrial robot trajectory planning based on improved pso algorithm, J. Phys.: Conf. Ser. **1820**, 012185, 2021.
85. A. Pourzangbar, M. Vaezi, Optimal design of brace-viscous damper and pendulum tuned mass damper using Particle Swarm Optimization, Applied Ocean Research **112**, 102706, 2021.
86. J. Tian, M. Hou, H. Bian et al, Variable surrogate model-based particle swarm optimization for high-dimensional expensive problems. Complex Intell. Syst., 2022.

87. B. Cao, Y. Gu, Z. Lv, S. Yang, J. Zhao, Y. Li, RFID Reader Anticollision Based on Distributed Parallel Particle Swarm Optimization, IEEE Internet of Things Journal **8**, pp. 3099-3107, 2021.
88. Dimitris Gavrilis, Ioannis G. Solos, Evangelos Dermatas, Selecting and constructing features using grammatical evolution, Pattern Recognition Letters **29**,pp. 1358-1365, 2008.
89. Dimitris Gavrilis, Ioannis G. Solos, Evangelos Dermatas, Neural Recognition and Genetic Features Selection for Robust Detection of E-Mail Spam, Advances in Artificial Intelligence Volume 3955 of the series Lecture Notes in Computer Science pp 498-501, 2006.
90. George Georgoulas, Dimitris Gavrilis, Ioannis G. Solos, Chrysostomos Stylios, João Bernardes, Peter P. Groumpos, Novel approach for fetal heart rate classification introducing grammatical evolution, Biomedical Signal Processing and Control **2**,pp. 69-79, 2007
91. Otis Smart, Ioannis G. Solos, Dimitris Gavrilis, George Georgoulas, Grammatical evolution for features of epileptic oscillations in clinical intracranial electroencephalograms, Expert Systems with Applications **38**, pp. 9991-9999, 2011.
92. I.G. Solos, C. Stylios, V. Charalampous, COVID-19 Predictive Models Based on Grammatical Evolution, SN COMPUT. SCI. **4**, 191, 2023.
93. V. Christou, I.G. Solos, 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.
94. A. Verikas, M. Bacauskiene, Feature selection with neural networks, Pattern Recognition Letters **23**, pp. 1323-1335, 2002.
95. Md. Monirul Kabir, Md. Monirul Islam, K. Murase, A new wrapper feature selection approach using neural network, Neurocomputing **73**, pp. 3273-3283, 2010.
96. V.S. Devi, Class Specific Feature Selection Using Simulated Annealing. In: Prasath, R., Vuppala, A., Kathirvalavakumar, T. (eds) Mining Intelligence and Knowledge Exploration. MIKE 2015. Lecture Notes in Computer Science(), vol 9468. Springer, Cham. 2015.
97. K. Neshatian, M. Zhang, P. Andreae, A Filter Approach to Multiple Feature Construction for Symbolic Learning Classifiers Using Genetic Programming, IEEE Transactions on Evolutionary Computation **16**, pp. 645-661, 2012.
98. 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.
99. 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.
100. 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.
101. M. Wang, Y.Y. Zhang, F. Min, Active learning through multi-standard optimization, IEEE Access **7**, pp. 56772–56784, 2019.
102. J.R. Quinlan, Simplifying Decision Trees. International Journal of Man-Machine Studies **27**, pp. 221-234, 1987.
103. T. Shultz, D. Mareschal, W. Schmidt, Modeling Cognitive Development on Balance Scale Phenomena, Machine Learning **16**, pp. 59-88, 1994.
104. B. Evans, D. Fisher, Overcoming process delays with decision tree induction. IEEE Expert **9**, pp. 60-66, 1994.
105. 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.
106. I. Kononenko, E. Šimec, M. Robnik-Šikonja, Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF, Applied Intelligence **7**, pp. 39–55, 1997
107. 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.
108. 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.
109. J.G. Dy , C.E. Brodley, Feature Selection for Unsupervised Learning, The Journal of Machine Learning Research **5**, pp 845–889, 2004.
110. S. J. Perantonis, V. Virvilis, Input Feature Extraction for Multilayered Perceptrons Using Supervised Principal Component Analysis, Neural Processing Letters **10**, pp 243–252, 1999.
111. J. Mcdermott, R.S. Forsyth, Diagnosing a disorder in a classification benchmark, Pattern Recognition Letters **73**, pp. 41-43, 2016.
112. J. Garcke, M. Griebel, Classification with sparse grids using simplicial basis functions, Intell. Data Anal. **6**, pp. 483-502, 2002.
113. 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.
114. 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.
115. 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.
116. 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.
117. 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.

118. 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.
119. T. Hastie, R. Tibshirani, Non-parametric logistic and proportional odds regression, JRSS-C (Applied Statistics) **36**, pp. 260–276, 1987.
120. M. Dash, H. Liu, P. Scheuermann, K. L. Tan, Fast hierarchical clustering and its validation, Data & Knowledge Engineering **44**, pp 109–138, 2003.
121. 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.
122. 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.
123. P. Zhong, M. Fukushima, Regularized nonsmooth Newton method for multi-class support vector machines, Optimization Methods and Software **22**, pp. 225-236, 2007.
124. 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.
125. 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
126. M. Koivisto, K. Sood, Exact Bayesian Structure Discovery in Bayesian Networks, The Journal of Machine Learning Research **5**, pp. 549–573, 2004.
127. W. J Nash, T.L. Sellers, S.R. Talbot, A.J. Cawthor, W.B. Ford, 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 (ISSN 1034-3288), 1994.
128. T.F. Brooks, D.S. Pope, A.M. Marcolini, Airfoil self-noise and prediction. Technical report, NASA RP-1218, July 1989.
129. J.S. Simonoff, Smooting Methods in Statistics, Springer - Verlag, 1996.
130. I.Cheng Yeh, Modeling of strength of high performance concrete using artificial neural networks, Cement and Concrete Research. **28**, pp. 1797-1808, 1998.
131. D. Harrison and D.L. Rubinfeld, Hedonic prices and the demand for clean ai, J. Environ. Economics & Management **5**, pp. 81-102, 1978.
132. J.S. Simonoff, Smooting Methods in Statistics, Springer - Verlag, 1996.
133. R.D. King, S. Muggleton, R. Lewis, M.J.E. Sternberg, Proc. Nat. Acad. Sci. USA **89**, pp. 11322–11326, 1992.
134. R. Fletcher, A new approach to variable metric algorithms, Computer Journal **13**, pp. 317-322, 1970.
135. H. Yu, T. Xie, S. Paszczynski, B.M. Wilamowski, Advantages of Radial Basis Function Networks for Dynamic System Design, IEEE Transactions on Industrial Electronics **58**, pp. 5438-5450, 2011.
136. 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.
137. T. Pajchrowski, K. Zawirski and K. Nowopolski, Neural Speed Controller Trained Online by Means of Modified PROP Algorithm, IEEE Transactions on Industrial Informatics **11**, pp. 560-568, 2015.
138. Rinda Parama Satya Hermanto, Suharjito, Diana, Ariadi Nugroho, Waiting-Time Estimation in Bank Customer Queues using PROP Neural Networks, Procedia Computer Science **135**, pp. 35-42, 2018.
139. K. O. Stanley, R. Miikkulainen, Evolving Neural Networks through Augmenting Typologies, Evolutionary Computation **10**, pp. 99-127, 2002.