Article

Training neural networks using neural networks

Ioannis G. Tsoulos[1,*]

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

Abstract: Artificial neural networks are parametric machine learning models, which have been used in recent decades in a number of practical problems in various scientific fields, such as Physics, Chemistry, Medicine, etc. Such problems can be reduced to pattern recognition problems and then modeled from artificial neural networks, whether these problems are classification problems or data fitting problems. To achieve the goal of neural networks, their parameters should be adjusted appropriately and this process is often achieved by using global optimization methods. In this work, the application of a recent global minimization technique is proposed for the adjustment of neural network parameters. In this technique, an approximation of the objective function to be minimized is created using artificial neural networks and then sampling is done from the approximation function and not the original one. Therefore, in the present work, learning of the parameters of artificial neural networks is done using other neural networks. The new training method was tested on a series of well-known problems from the relevant literature and a comparative study was made against other neural network parameter tuning techniques and the results were more than promising.

Keywords: Global optimization; Neural networks; Stochastic methods

Citation: Tsoulos, I.G. Training neural networks using neural networks. *Journal Not Specified* **2022**, *1*, 0. https://doi.org/

Received:
Accepted:
Published:

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

1. Introduction

Artificial Neural networks (ANNs) are parametric machine learning models [1–3] widely used in pattern recognition problems. Many practical problems from the fields of physics [4–6], chemistry [7–9], economics [10–12], medicine [13,14] etc. can be transformed to pattern recognition problems and then solved using Artificial Neural Networks. Also, neural networks have been used with success to solve differential equations [15–17], solar radiation prediction [18,19], Spam detection [20–22] etc. Moreover, variations of artificial neural networks have been employed to solve agricultural problems [23,24], facial expression recognition [25], prediction of the speed of wind [26], the gas consumption problem [27], intrusion detection [28], hydrological systems [29] etc. Also, Swales and Yoon discuss the application of artificial neural networks to investment analysis in their work [30].

A neural network can be denoted as a function $N(\overrightarrow{x},\overrightarrow{w})$ where the vector $\overrightarrow{x}$ stands for the input vector and the vector $\overrightarrow{w}$ is the set of the parameters of the neural network that should be estimated. The input vector is usually called pattern in the relevant literature and the vector $\overrightarrow{w}$ is usually called the weight vector. Artificial neural network training methods adjust the vector of weights in order to minimize the following quantity:

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

In the previous equation, that will be called training error, the set $\left(\overrightarrow{x_i},y_i\right),\ i=1,\ldots,M$ is the input train dataset for the neural network with M patterns. The value y_i is the expected output for the the pattern $\overrightarrow{x_i}$. The equation 1 can be minimized with respect to the weight vector using any local or global optimization method from the relevant literature such as the Back Propagation method [31,32], the Hill Climbing method [33], the RPROP method [34–36], Quasi Newton methods [38,39], Simulated Annealing [40,41], Genetic Algorithms

[42,43], Particle Swarm Optimization [44,45],Differential Optimization methods [46,47], Evolutionary Computation [48], the Whale optimization algorithm [49] etc. Furthermore, Cui et al suggested the usage of a new stochastic optimization algorithm that simulates the plant growing process for neural network training. Also recently, the Bird Mating Optimizer [51] was suggested as a training method for artificial neural networks [50]. Also, hybrid methods have been developed by various researchers to optimize the weight vector, such as the work of Yaghini et al [53] that combined particle swarm optimization with a back propagation algorithm to minimize the error function. Moreover, Chen at al [52] has used a hybrid technique that combines particle swarm optimization and Cuckoo Search [54] to optimize the weight vector of neural networks.

In addition to the training of artificial neural networks, many researchers also employ techniques for initializing the parameters of neural networks, such as the incorporation of decision trees [55], an initialization method using the Cauchy's inequality [56], incorporation of discriminant learning [57], methods based on genetic algorithms [58,59] etc. A systematic review of weight initialization strategies can be found in the paper by Narkhede et al [60].

Moreover, various groups of researchers are dealing with the issue of constructing the structure of artificial neural networks, such as the incorporation of genetic algorithms [61], the usage of the Grammatical Evolution method [62] for the construction of neural networks[63], a constructing and pruning approach to optimize the structure of ANNs [64], usage of cellular automata [65] etc. In addition, since the training of artificial neural networks by optimization methods requires significantly larger computing time, parallel techniques have been developed that take advantage of modern parallel computing units[66–68].

Another field of research in the field of artificial neural networks that attracts a multitude of researchers is that of dealing with the problem of overfitting that occurs in many cases. In this problem, although the artificial neural network has achieved a satisfactory level of training this is not reflected in unknown patterns that were not present during training. This set of patterns will be called as test set in the remaining of this work. Commonly used methods that tackle the overfitting problem are weight sharing [69,70], methods that reduce the number of parameters (weight pruning) [71–73], the method of dropout [74,75], weight decaying methods [76,77], the Sarporp method [78], positive correlation methods [79] etc.

In this paper, the use of a recent global minimization technique [80] called NeuralMinimizer, is proposed to find the optimal set of parameters for artificial neural networks. This innovative global minimization technique constructs an approximation of the objective function to be minimized using a limited number of its samples. These limited samples form the training set of an artificial neural network that can be trained with any optimization method. Subsequently, the sampling for the continuation of the global optimization method is not done by the objective function but by the already trained artificial neural network. The samples obtained by artificial neural networks before being fed to the global minimization method into are classified and those with the smallest functional value will finally be input to the global minimization method. From the experimental results, it was shown that this global minimization method requires a limited number of samples from the objective function to find the global minimum and is also more efficient than other techniques to discover the global minimum. Therefore, this paper proposes using artificial neural networks to train other artificial neural networks. This new procedure will be tested on a series of known problems in the relevant literature, in order to evaluate its effectiveness.

The rest of this article is organized as follows: the section 2 described the proposed method, the section 3 list the experimental datasets and the results obtained by the incorporation of various methods and finally the section 4 discusses some conclusions.

2. The proposed method

In this section, some basic principles for artificial neural networks are presented and then a new training method that incorporates a modified version of the NeuralMinimizer global optimization technique is outlined.

2.1. Preliminaries

Let us consider that we have an artificial neural network with a processing layer, in which the sigmoid function is used as an activation function. The output value for every node in this layer is calculated as:

$$o_i(x) = \sigma\left(p_i^T x + \theta_i\right), \tag{2}$$

where the value p_i is the weight vector and θ_i denotes the bias for the node i. The sigmoid function is defined as:

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

and it is graphically illustrated in Figure 1.

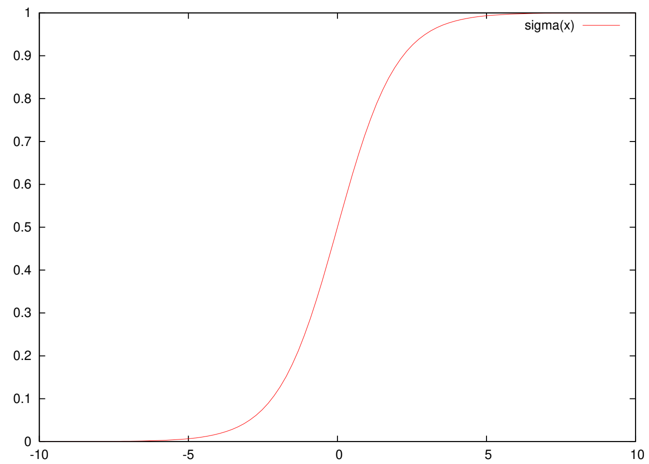

Figure 1. The sigmoid function $\sigma(x)$.

When the neural network has H processing nodes, then output can be formulated as:

$$N(x) = \sum_{i=1}^{H} v_i o_i(x), \tag{4}$$

where v_i stands for the output weight for node i. Hence, by using one vector for all the parameters (weights and biases) the neural network can be written in the following form:

$$N(\overrightarrow{x}, \overrightarrow{w}) = \sum_{i=1}^{H} w_{(d+2)i-(d+1)} \sigma\left(\sum_{j=1}^{d} x_j w_{(d+2)i-(d+1)+j} + w_{(d+2)i}\right) \tag{5}$$

where d is the dimension of vector $\overrightarrow{x}$. From the equation 5 we can conclude that the dimension of the weight vector w is calculated as:

$$d_w = (d+2)H \tag{6}$$

2.2. The modified NeuralMinimizer method

In its original version, the method NeuralMinimizer employed RBF neural networks [81] to build a model of the objective function. Even though RBF networks have been used with success in a variety of problems [82–85], it is not possible to apply them to the training of the parameters of an artificial neural network due to the large dimension of the problem, as shown in Equation 6. Hence, in current work, the RBF network has been replaced by an artificial neural network that implements the equation 5. The training of the artificial neural network was done using a local minimization technique that is not particularly demanding in calculations and storage space, such as Limited Memory BFGS (L-BFGS)[86]. The L-BFGS method is variation of BFGS method [87] using a limited amount of computer memory. This local minimization method has found wide application in difficult and memory-intensive optimization problems such as image reconstruction [88], inverse eigenvalue problems [89], seismic waveform tomography [90] etc. Because of the application of this technique to large-dimensional problems, a number of modifications have been proposed that make use of modern parallel computing systems [91–93]. A numerical study on the limited memory BFGS methods is provided in the work of Morales [94].

In the following the main steps of the modified NeuralMinimizer method for the training of neural networks are listed. In this steps the neural network used by the NeuralMinimizer method will be called $N_N(x,w)$.

1. **Initialization** step.
   (a) **Set** H the number of weights used in the neural network. In the current method the same number of weights was used for both $N(x,w)$ and $N_N(x,w)$ artificial neural networks.
   (b) **Set** N_S as the initial samples drawn from $N(x,w)$. At this stage, the training error for the artificial neural network will be used as an objective function to minimize
   (c) **Set** N_T as the the number of points that will be used as local minimization method starters in every iteration.
   (d) **Set** N_R the number of samples that will be taken from the $N_N(x,w)$ network in each iteration.
   (e) **Set** N_G as the number of iterations allowed.
   (f) **Set** Iter=0, the current iteration number.
   (g) **Set** (w^*, y^*) as the global minimum discovered by the method. Initially $y^* = \infty$
2. **Creation** Step.
   (a) **Set** $T = \emptyset$, the training set for the $N_N(x,w)$ neural network.
   (b) **For** $i = 1,..,N_S$ do
      i. **Draw** a new sample w_i from $N(x,w)$.
      ii. **Calculate** $y_i = f(w_i)$ using equation 1
      iii. $T = T \cup (w_i, y_i)$
   (c) **EndFor**
   (d) **Train** the $N_N(x,w)$ neural network on set T using the L-BFGS method.
3. **Sampling** Step
   (a) **Set** $T_R = \emptyset$
   (b) **For** $i = 1,..,N_R$ **do**
      i. **Get** a new random sample (w_i, y_i) from $N_N(w,x)$ neural network
      ii. **Set** $T_R = T_R \cup (x_i, y_i)$
   (c) **EndFor**
   (d) **Sort** T_R in ascending order according to the values y_i
4. **Optimization** Step.
   (a) **For** $i = 1,..,N_T$ do
      i. **Get** the sample (w_i, y_i) from T_R.
      ii. **Train** the neural network $N(w_i, x)$ on the train set of the objective problem, using the L-BFGS method and get the corresponding training error y_i.
      iii. **Update** $T = T \cup (w_i, y_i)$

iv. **Train** again the network $N_N(w,x)$ on the modified set T. In this step the original train set used by $N_N(x,w)$ is updated to include the new discovered local minimum. This operation is used in order to construct a more accurate approximation of the real objective function.
v. **If** $y_i \le y^*$ then $w^* = w_i, y^* = y_i$
vi. **If** the termination rule proposed in [95] then apply the neural network $N(w^*,x)$ on the test set of the objective problem, report the test error and **terminate**.

(b) **EndFor**

5. **Set** iter=iter+1
6. **Goto** to Sampling step.

3. Experiments

The efficiency of the proposed artificial neural network training technique was evaluated using a series of data sets from the relevant literature which are freely available from the following websites:

1. The UCI repository, https://archive.ics.uci.edu/ml/index.php
2. The Keel repository, https://sci2s.ugr.es/keel/datasets.php[96].
3. The Statlib URL **ftp://lib.stat.cmu.edu/datasets/index.html**. This repository is used mainly for the regression datasets.

3.1. Experimental datasets

The following classification datasets from the relevant literature were used in the experiments:

1. **Appendictis** which is a medical dataset found in [97,98].
2. **Australian** dataset [99], which is related to economical transactions in banks.
3. **Balance** dataset [100], which is related to psychological experiments.
4. **Cleveland** dataset, a medical dataset, used in a variety of research papers[101,102].
5. **Bands** dataset, a dataset related to printing problems [103].
6. **Dermatology** dataset [104], a dataset related to dermatology problems.
7. **Hayes roth** dataset [106].
8. **Heart** dataset [105], a medical dataset used to detect heart diseases.
9. **HouseVotes** dataset [107], related to the Congressional voting records of USA.
10. **Ionosphere** dataset, related to measurements from the ionosphere an thoroughly studied in a series of research papers [108,109].
11. **Liverdisorder** dataset [110,111], a medical dataset.
12. **Lymography** dataset [112].
13. **Mammographic** dataset [113], a medical dataset related to breast cancer diagnosis.
14. **Page Blocks** dataset [114], related to documents.
15. **Parkinsons** dataset [115,116], a dataset related to Parkinson's decease.
16. **Pima** dataset [117], a medical dataset.
17. **Popfailures** dataset [118], a dataset related to meteorological data.
18. **Regions2** dataset, a medical dataset for liver biopsy images [119].
19. **Saheart** dataset [120], a medical dataset related to heart diseases.
20. **Segment** dataset [121], a dataset related to image segmentation.
21. **Wdbc** dataset [122], a dataset about breast tumors.
22. **Wine** dataset, a dataset related to chemical analysis of wines [123,124].
23. **Eeg** datasets [125,126], it is an EEG dataset and the following cases were used in the experiments:
    (a) Z_F_S,
    (b) ZO_NF_S
    (c) ZONF_S.
24. **Zoo** dataset [127], suggested for classification of animals into seven categories.

The following regression datasets were used:

1. **Abalone** dataset [129], proposed to predict the age of abalones.
2. **Airfoil** dataset, a dataset provided by NASA [130], created from a series of aerodynamic and acoustic tests.
3. **Baseball** dataset, a dataset used to predict the amount of salary for some baseball players.
4. **BK** dataset [131], used to predict the points scored in a basketball game.
5. **BL** dataset, used in machine problems.
6. **Concrete** dataset [132], a dataset proposed to calculate the concrete compressive strength
7. **Dee** dataset, used to estimate the daily average price of TkWhe electricity energy in Spain.
8. **Diabetes** dataset, a medical dataset.
9. **Housing** dataset [133].
10. **FA** dataset, used to fit body fat to other measurements.
11. **MB** dataset [134].
12. **Mortgage** dataset. The goal is to predict the 30-Year Conventional Mortgage Rate.
13. **PY** dataset, (Pyrimidines problem)[135].
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.
16. **Wankara** dataset, a weather dataset.

3.2. Experimental results

The proposed method was compared against five other approaches found in the relevant literature on the previous mentioned classification and regression datasets. For greater reliability of the experimental results, the 10 - fold validation technique was employed for every classification or regression dataset. Every experiment was executed 30 times, with different initialization for the random generator each time. Also, the srand48() random generator of the C - programming language was utilized. The used code was implemented in ANSI C++ using the freely available OPTIMUS optimization library available from https://github.com/itsoulos/OPTIMUS/. For the case of classification datasets, the average classification error is measured for every method and for regression datasets the average regression error is measured in the test set. The number of hidden nodes for the neural networks was set to $H = 10$ for every method. All the experiments were performed using an AMD Ryzen 5950X with 128GB of RAM. The running operating system was Debian Linux. The methods used in the experimental results are the following:

1. A genetic algorithm with 200 chromosomes used to train a neural network with H hidden nodes. This method was denoted as GENETIC in the tables holding the experimental results.
2. A Radial Basis Function (RBF) network [81] having H weights.
3. The Adam optimization method[136]. Here the method is used to minimize the train error of a neural network with H hidden nodes.
4. The resilient back propagation (RPROP) optimization method[34–36] was employed also to train a neural network with H hidden nodes.
5. The NEAT method (NeuroEvolution of Augmenting Topologies) [137].

The values used for every parameter are listed in Table 1 and they are similar to the values used in the original publication of the NeuralMinimizer method.

Table 1. Experimental settings.

PARAMETER	MEANING	VALUE
H	Number of weights	10
N_S	Start samples	50
N_T	Starting points	100
N_R	Samples drawn from the first network	$10 \times N_T$
N_G	Maximum number of iterations	200

The experimental results for the classification datasets are shown in Table 2 and for the regression datasets in Table 3. The column PROPOSED stands for the usage of the proposed method to train a neural network with H hidden nodes. In both tables, the last row (denoted as AVERAGE) represents the mean error for each method. Furthermore, the average classification error for every used method is graphically displayed in Figure 2 and a similar graph is shown for the average regression error of each method in Figure 3.

Table 2. Experimental results for the classification datasets. The numbers in cells denote average classification error of 30 independent runs.

DATASET	GENETIC	RBF	ADAM	RPROP	NEAT	PROPOSED
Appendicitis	18.10%	12.23%	16.50%	16.30%	17.20%	22.30%
Australian	32.21%	34.89%	35.65%	36.12%	31.98%	21.59%
Balance	8.97%	33.42%	7.87%	8.81%	23.14%	5.46%
Bands	35.75%	37.22%	36.25%	36.32%	34.30%	33.06%
Cleveland	51.60%	67.10%	67.55%	61.41%	53.44%	45.41%
Dermatology	30.58%	62.34%	26.14%	15.12%	32.43%	4.14%
Hayes Roth	56.18%	64.36%	59.70%	37.46%	50.15%	35.28%
Heart	28.34%	31.20%	38.53%	30.51%	39.27%	17.93%
HouseVotes	6.62%	6.13%	7.48%	6.04%	10.89%	5.78%
Ionosphere	15.14%	16.22%	16.64%	13.65%	19.67%	16.31%
Liverdisorder	31.11%	30.84%	41.53%	40.26%	30.67%	33.02%
Lymography	23.26%	25.31%	29.26%	24.67%	33.70%	25.64%
Mammographic	19.88%	21.38%	46.25%	18.46%	22.85%	16.37%
PageBlocks	8.06%	10.09%	7.93%	7.82%	10.22%	5.44%
Parkinsons	18.05%	17.42%	24.06%	22.28%	18.56%	14.47%
Pima	32.19%	25.78%	34.85%	34.27%	34.51%	25.61%
Popfailures	5.94%	7.04%	5.18%	4.81%	7.05%	5.57%
Regions2	29.39%	38.29%	29.85%	27.53%	33.23%	22.73%
Saheart	34.86%	32.19%	34.04%	34.90%	34.51%	34.03%
Segment	57.72%	59.68%	49.75%	52.14%	66.72%	37.28%
Wdbc	8.56%	7.27%	35.35%	21.57%	12.88%	5.01%
Wine	19.20%	31.41%	29.40%	30.73%	25.43%	7.14%
Z_F_S	10.73%	13.16%	47.81%	29.28%	38.41%	7.09%
ZO_NF_S	8.41%	9.02%	47.43%	6.43%	43.75%	5.15%
ZONF_S	2.60%	4.03%	11.99%	27.27%	5.44%	2.35%
ZOO	16.67%	21.93%	14.13%	15.47%	20.27%	4.20%
AVERAGE	**23.47%**	**27.69%**	**30.81%**	**25.37%**	**28.87%**	**17.63%**

Table 3. Average regression error for the regression datasets.

DATASET	GENETIC	RBF	ADAM	RPROP	NEAT	PROPOSED
ABALONE	7.17	7.37	4.30	4.55	9.88	4.50
AIRFOIL	0.003	0.27	0.005	0.002	0.067	0.003
BASEBALL	103.60	93.02	77.90	92.05	100.39	56.16
BK	0.027	0.02	0.03	1.599	0.15	0.02
BL	5.74	0.01	0.28	4.38	0.05	0.0004
CONCRETE	0.0099	0.011	0.078	0.0086	0.081	0.003
DEE	1.013	0.17	0.63	0.608	1.512	0.30
DIABETES	19.86	0.49	3.03	1.11	4.25	1.24
HOUSING	43.26	57.68	80.20	74.38	56.49	18.30
FA	1.95	0.02	0.11	0.14	0.19	0.01
MB	3.39	2.16	0.06	0.055	0.061	0.05
MORTGAGE	2.41	1.45	9.24	9.19	14.11	3.50
PY	105.41	0.02	0.09	0.039	0.075	0.03
QUAKE	0.04	0.071	0.06	0.041	0.298	0.039
TREASURY	2.929	2.02	11.16	10.88	15.52	3.72
WANKARA	0.012	0.001	0.02	0.0003	0.005	0.002
AVERAGE	**18.55**	**10.30**	**11.70**	**12.44**	**12.70**	**5.49**

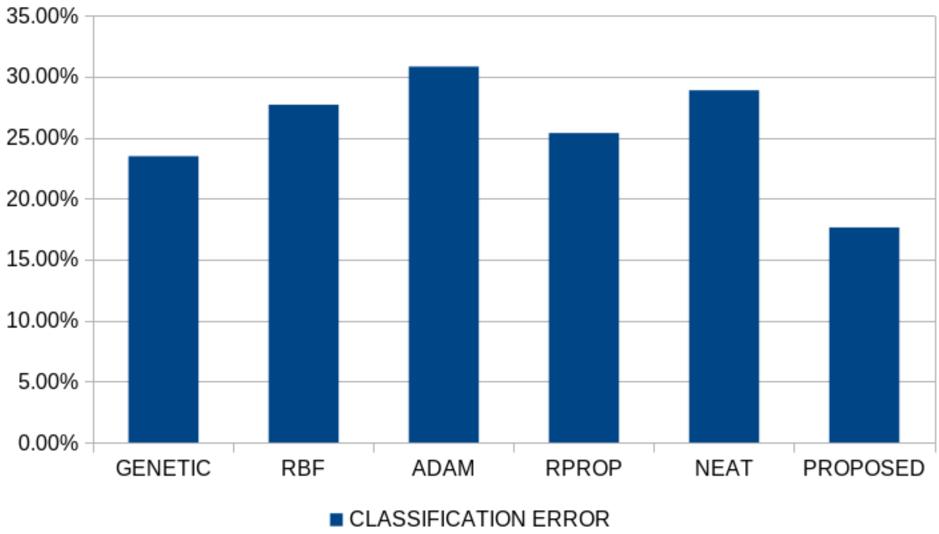

Figure 2. The average classification error for all mentioned methods.

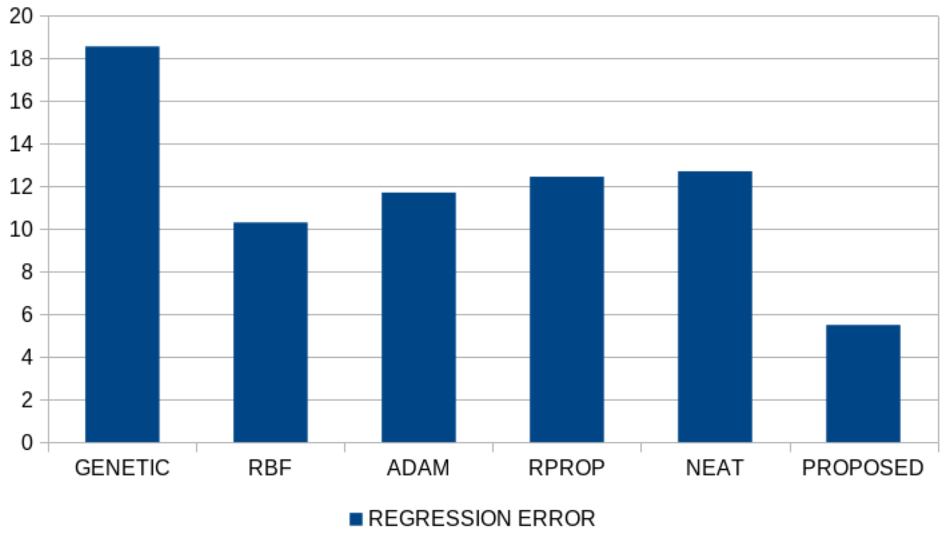

Figure 3. Average regression error for all mentioned methods.

The experimental results and their graphical representation demonstrate the superiority of the proposed technique over the others in terms of the average error, as measured in the test set. For example, in the case of datasets used for classification, the proposed method outperforms the remaining techniques in 19 out of 26 datasets (73% percent). Also, in several cases, the percentage reduction in error exceeds 50%. For the classification problems, the immediate most effective training method after the proposed one is the genetic algorithm and on average, the proposed technique achieves lower classification error than the genetic algorithm error by 24%. Moreover, in data fitting problems, the next most effective method after the proposed one is the RBF neural network with small differences from the ADAM optimizer. However, in the case of data fitting problems the improvement in average error by using the proposed technique exceeds 49%. Of course, the proposed technique is quite time-consuming since it requires continuous training of an artificial neural network.

4. Conclusions

In this work, the application of a recent global minimization method for the training of artificial neural networks was proposed. The application of this method was used in artificial neural networks both for classification problems and for data fitting problems. This new global minimization method constructs an approximation of the objective function using neural networks. This construction is done with a limited number of samples from the objective function. However each time a local minimization takes place, this approximation is readjusted. Subsequently, the sampling for the minimization is done from the approximate function and not from the objective one, even taking samples from the approximation with the smallest function value, in order to speed up the finding of the global minimum. In this particular case, the artificial neural network of the global minimization method is used to train the artificial neural network. However, due to the large time and storage requirements of artificial neural networks, the RBF network of the original NeuralMinimizer method was replaced with an artificial neural network that was trained using the local minimization method L-BFGS. The new artificial neural network training technique is tested on a wide collection of classification and data fitting problems from the relevant literature and is shown to significantly improve the learning error over other established artificial neural network training techniques. This improvement is 25% on average for the case of classification problems and rises significantly to 50% for data fitting problems.

Nevertheless, the proposed procedure can be extremely slow, especially as the size of the artificial neural network increases. The size of the artificial neural network directly depends on the dimension of the input dataset. Future improvements to the methodology may include the use of parallel programming techniques, such as parallel implementations of the L-BFGS optimization method, in order to accelerate the training of artificial neural networks by taking advantage of modern computing structures. Also, in the present phase as a minimization method in step 4 of the proposed training method, a local minimization method is used. Future extensions could explore the possibility of also using global minimization techniques in this step, although care should be taken to make use of parallel computing techniques to avoid long execution times.

Author Contributions: I.G.T. conducted the experiments, employing several datasets and provided the comparative experiments. I.G.T have performed the statistical analysis and prepared 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: This research has been co-financed by the European Regional Development Fund of the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call Research -Create-Innovate, project name "Create a system of recommendations and augmented reality applications in a hotel" (project code:T1EDK-03745).

Sample Availability: Not applicable.

References

1. C. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1995.
2. G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of Control Signals and Systems **2**, pp. 303-314, 1989.
3. O.I. Abiodun, A. Jantan, A. E. Omolara, K.V. Dada, N. A. Mohamed, H. Arshad, State-of-the-art in artificial neural network applications: A survey, Heliyon **4**, e00938, 2018.
4. P. Baldi, K. Cranmer, T. Faucett et al, Parameterized neural networks for high-energy physics, Eur. Phys. J. C **76**, 2016.
5. J. J. Valdas and G. Bonham-Carter, Time dependent neural network models for detecting changes of state in complex processes: Applications in earth sciences and astronomy, Neural Networks **19**, pp. 196-207, 2006
6. G. Carleo,M. Troyer, Solving the quantum many-body problem with artificial neural networks, Science **355**, pp. 602-606, 2017.
7. Lin Shen, Jingheng Wu, and Weitao Yang, Multiscale Quantum Mechanics/Molecular Mechanics Simulations with Neural Networks, Journal of Chemical Theory and Computation **12**, pp. 4934-4946, 2016.
8. Sergei Manzhos, Richard Dawes, Tucker Carrington, Neural network-based approaches for building high dimensional and quantum dynamics-friendly potential energy surfaces, Int. J. Quantum Chem. **115**, pp. 1012-1020, 2015.
9. Jennifer N. Wei, David Duvenaud, and Alán Aspuru-Guzik, Neural Networks for the Prediction of Organic Chemistry Reactions, ACS Central Science **2**, pp. 725-732, 2016.
10. Lukas Falat and Lucia Pancikova, Quantitative Modelling in Economics with Advanced Artificial Neural Networks, Procedia Economics and Finance **34**, pp. 194-201, 2015.
11. Mohammad Namazi, Ahmad Shokrolahi, Mohammad Sadeghzadeh Maharluie, Detecting and ranking cash flow risk factors via artificial neural networks technique, Journal of Business Research **69**, pp. 1801-1806, 2016.
12. G. Tkacz, Neural network forecasting of Canadian GDP growth, International Journal of Forecasting **17**, pp. 57-69, 2001.
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. I. E. Lagaris, A. Likas, D. I. Fotiadis, Artificial neural networks for solving ordinary and partial differential equations, IEEE Transactions on Neural Networks **9**, pp. 987-1000, 1998.
16. S. Effati, M. Pakdaman, Artificial neural network approach for solving fuzzy differential equations, Information Sciences **180**, pp. 1434-1457, 2010.
17. F. Rostami, A. Jafarian, A new artificial neural network structure for solving high-order linear fractional differential equations, International Journal of Computer Mathematics **95**, pp. 528-539, 2018.

18. A.K. Yadav, S.S. Chandel, Solar radiation prediction using Artificial Neural Network techniques: A review, Renewable and Sustainable Energy Reviews **33**, pp 772-781, 2014.
19. A. Qazi, H. Fayaz, A. Wadi, R.G. Raj, N.A. Rahim, W.A. Khan, The artificial neural network for solar radiation prediction and designing solar systems: a systematic literature review, Journal of Cleaner Production **104**, pp 1-12, 2015.
20. C.H. Wu, Behavior-based spam detection using a hybrid method of rule-based techniques and neural networks, Expert Systems with Applications **36**, pp. 4321-4330, 2009.
21. Y. Ren, D. Ji, Neural networks for deceptive opinion spam detection: An empirical study, Information Sciences **385–386**, pp. 213-224, 2017.
22. S. Madisetty, M.S. Desarkar, A Neural Network-Based Ensemble Approach for Spam Detection in Twitter, IEEE Transactions on Computational Social Systems **5**, pp. 973-984, 2018.
23. A. Topuz, Predicting moisture content of agricultural products using artificial neural networks, Advances in Engineering Software **41**, pp. 464-470, 2010.
24. A. Escamilla-García, G.M. Soto-Zarazúa, M. Toledano-Ayala, E. Rivas-Araiza, A. Gastélum-Barrios, Abraham,Applications of Artificial Neural Networks in Greenhouse Technology and Overview for Smart Agriculture Development, Applied Sciences **10**, Article number 3835, 2020.
25. H. Boughrara, M. Chtourou, C. Ben Amar et al, Facial expression recognition based on a mlp neural network using constructive training algorithm. Multimed Tools Appl **75**, pp. 709–731, 2016.
26. H. Liu, H.Q Tian, Y.F. Li, L. Zhang, Comparison of four Adaboost algorithm based artificial neural networks in wind speed predictions, Energy Conversion and Management **92**, pp. 67-81, 2015.
27. J. Szoplik, Forecasting of natural gas consumption with artificial neural networks, Energy **85**, pp. 208-220, 2015.
28. H. Bahram, N.J. Navimipour, Intrusion detection for cloud computing using neural networks and artificial bee colony optimization algorithm, ICT Express 5, pp. 56-59, 2019.
29. Y.S. Chen, F.J. Chang, Evolutionary artificial neural networks for hydrological systems forecasting, Journal of Hydrology **367**, pp. 125-137, 2009.
30. G.S. Swales, Y.Yoon, Applying Artificial Neural Networks to Investment Analysis, Financial Analysts Journal **48**, pp. 78-80, 1992.
31. D.E. Rumelhart, G.E. Hinton and R.J. Williams, Learning representations by back-propagating errors, Nature **323**, pp. 533 - 536 , 1986.
32. T. Chen and S. Zhong, Privacy-Preserving Backpropagation Neural Network Learning, IEEE Transactions on Neural Networks **20**, , pp. 1554-1564, 2009.
33. S. Chalup, F. Maire, A study on hill climbing algorithms for neural network training, In: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), Washington, DC, USA, 1999, pp. 2014-2021 Vol. 3.
34. M. Riedmiller and H. Braun, A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP algorithm, Proc. of the IEEE Intl. Conf. on Neural Networks, San Francisco, CA, pp. 586–591, 1993.
35. T. Pajchrowski, K. Zawirski and K. Nowopolski, Neural Speed Controller Trained Online by Means of Modified RPROP Algorithm, IEEE Transactions on Industrial Informatics **11**, pp. 560-568, 2015.
36. Rinda Parama Satya Hermanto, Suharjito, Diana, Ariadi Nugroho, Waiting-Time Estimation in Bank Customer Queues using RPROP Neural Networks, Procedia Computer Science **135**, pp. 35-42, 2018.
37. Neural Networks, Procedia Computer Science **135**, pp. 35-42, 2018.
38. B. Robitaille and B. Marcos and M. Veillette and G. Payre, Modified quasi-Newton methods for training neural networks, Computers & Chemical Engineering **20**, pp. 1133-1140, 1996.
39. Q. Liu, J. Liu, R. Sang, J. Li, T. Zhang and Q. Zhang, Fast Neural Network Training on FPGA Using Quasi-Newton Optimization Method,IEEE Transactions on Very Large Scale Integration (VLSI) Systems **26**, pp. 1575-1579, 2018.
40. A. Yamazaki, M. C. P. de Souto,T. B. Ludermir, Optimization of neural network weights and architectures for odor recognition using simulated annealing, In: Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 **1**, pp. 547-552 , 2002.
41. Y. Da, G. Xiurun, An improved PSO-based ANN with simulated annealing technique, Neurocomputing **63**, pp. 527-533, 2005.
42. F. H. F. Leung, H. K. Lam, S. H. Ling and P. K. S. Tam, Tuning of the structure and parameters of a neural network using an improved genetic algorithm, IEEE Transactions on Neural Networks **14**, pp. 79-88, 2003
43. X. Yao, Evolving artificial neural networks, Proceedings of the IEEE, 87(9), pp. 1423-1447, 1999.
44. C. Zhang, H. Shao and Y. Li, Particle swarm optimisation for evolving artificial neural network, IEEE International Conference on Systems, Man, and Cybernetics, , pp. 2487-2490, 2000.
45. Jianbo Yu, Shijin Wang, Lifeng Xi, Evolving artificial neural networks using an improved PSO and DPSO **71**, pp. 1054-1060, 2008.
46. J. Ilonen, J.K. Kamarainen, J. Lampinen, Differential Evolution Training Algorithm for Feed-Forward Neural Networks, Neural Processing Letters **17**, pp. 93–105, 2003.
47. A. Slowik, M. Bialko, Training of artificial neural networks using differential evolution algorithm, In: 2008 Conference on Human System Interactions, Krakow, Poland, pp. 60-65, 2008.
48. M. Rocha, P. Cortez, J. Neves, Evolution of neural networks for classification and regression, Neurocomputing **70**, pp. 2809-2816, 2007.

49. I. Aljarah, H. Faris, S. Mirjalili, Optimizing connection weights in neural networks using the whale optimization algorithm, Soft Comput **22**, pp. 1–15, 2018.
50. Z. Cui , C. Yang, S. Sanyal, Training artificial neural networks using APPM, International Journal of Wireless and Mobile Computing **5**, pp. 168-174, 2012.
51. A. Askarzadeh, A. Rezazadeh, Artificial neural network training using a new efficient optimization algorithm, Applied Soft Computing **13**, pp 1206-1213, 2013.
52. J.F. Chen, Q.H. Do,H.N. Hsieh, Training Artificial Neural Networks by a Hybrid PSO-CS Algorithm, Algorithms **8**, pp. 292-308, 2015.
53. M. Yaghini, M.M. Khoshraftar, M. Fallahi, A hybrid algorithm for artificial neural network training, Engineering Applications of Artificial Intelligence **26**, pp 293-301, 2013.
54. X.S. Yang, S. Deb, Engineering Optimisation by Cuckoo Search, Int. J. Math. Model. Numer. Optim. **1**, 330–343, 2010.
55. I. Ivanova, M. Kubat, Initialization of neural networks by means of decision trees, Knowledge-Based Systems **8**, pp. 333-344, 1995.
56. J.Y.F. Yam, T.W.S. Chow, A weight initialization method for improving training speed in feedforward neural network, Neurocomputing **30**, pp. 219-232, 2000.
57. K. Chumachenko, A. Iosifidis, M. Gabbouj, Feedforward neural networks initialization based on discriminant learning, Neural Networks **146**, pp. 220-229, 2022.
58. F. Itano, M. A. de Abreu de Sousa, E. Del-Moral-Hernandez, Extending MLP ANN hyper-parameters Optimization by using Genetic Algorithm, In: 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 2018, pp. 1-8, 2018.
59. F. Itano, M. A. de Abreu de Sousa, E. Del-Moral-Hernandez, Extending MLP ANN hyper-parameters Optimization by using Genetic Algorithm," 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, pp. 1-8, 2018.
60. M.V. Narkhede, P.P. Bartakke, M.S. Sutaone, A review on weight initialization strategies for neural networks, Artif Intell Rev **55**, pp. 291–322, 2022.
61. F. H. F. Leung, H. K. Lam, S. H. Ling, P. K. S. Tam, Tuning of the structure and parameters of a neural network using an improved genetic algorithm, IEEE Transactions on Neural Networks **14**, pp. 79-88, 2003.
62. M. O'Neill, C. Ryan, Grammatical evolution, IEEE Trans. Evol. Comput. **5**, pp. 349–358, 2001.
63. I.G. Tsoulos, D. Gavrilis, E. Glavas, Neural network construction and training using grammatical evolution, Neurocomputing 72, pp. 269-277, 2008.
64. H.G. Han, J.F. Qiao, A structure optimisation algorithm for feedforward neural network construction, Neurocomputing **99**, pp 347-357, 2013.
65. K.J. Kim, S.B. Cho, Evolved neural networks based on cellular automata for sensory-motor controller, Neurocomputing **69**, pp. 2193-2207, 2006.
66. M. Martínez-Zarzuela, F.J. Díaz Pernas, J.F. Díez Higuera, M.A. Rodríguez, Fuzzy ART Neural Network Parallel Computing on the GPU. In: Sandoval, F., Prieto, A., Cabestany, J., Graña, M. (eds) Computational and Ambient Intelligence. IWANN 2007. Lecture Notes in Computer Science, vol 4507. Springer, Berlin, Heidelberg, 2007.
67. X. Sierra-Canto, F. Madera-Ramirez, V. Uc-Cetina, Parallel Training of a Back-Propagation Neural Network Using CUDA, In: 2010 Ninth International Conference on Machine Learning and Applications, Washington, DC, USA, pp. 307-312, 2010.
68. A.A. Huqqani, E. Schikuta, S. Ye Peng Chen, Multicore and GPU Parallelization of Neural Networks for Face Recognition, Procedia Computer Science **18**, pp. 349-358, 2013.
69. S.J. Nowlan and G.E. Hinton, Simplifying neural networks by soft weight sharing, Neural Computation 4, pp. 473-493, 1992.
70. 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), Seoul, Korea (South), pp. 1-2, 2016.
71. S.J. Hanson and L.Y. Pratt, Comparing biases for minimal network construction with back propagation, In D.S. Touretzky (Ed.), Advances in Neural Information Processing Systems, Volume 1, pp. 177-185, San Mateo, CA: Morgan Kaufmann, 1989.
72. M.C. Mozer and P. Smolensky, Skeletonization: a technique for trimming the fat from a network via relevance assesment. In D.S. Touretzky (Ed.), Advances in Neural Processing Systems, Volume 1, pp. 107-115, San Mateo CA: Morgan Kaufmann, 1989.
73. M. Augasta and T. Kathirvalavakumar, Pruning algorithms of neural networks — a comparative study, Central European Journal of Computer Science, 2003.
74. Nitish Srivastava, G E Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan R Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, Journal of Machine Learning Research **15**, pp. 1929-1958, 2014.
75. A. Iosifidis, A. Tefas, I. Pitas, DropELM: Fast neural network regularization with Dropout and DropConnect, Neurocomputing **162**, pp. 57-66, 2015.
76. A. Gupta, S.M. Lam, Weight decay backpropagation for noisy data, Neural Networks **11**, pp. 1127-1138, 1998.
77. 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.
78. N.K. Treadgold, T.D. Gedeon, Simulated annealing and weight decay in adaptive learning: the SARPROP algorithm, IEEE Trans. on Neural Networks 9, pp. 662-668, 1998.
79. M.D. Shahjahan, M. Kazuyuki, Neural network training algorithm with possitive correlation, IEEE Trans. Inf & Syst. **88**, pp. 2399-2409, 2005.

80. I.G. Tsoulos, A. Tzallas, E. Karvounis, D. Tsalikakis, NeuralMinimizer: A Novel Method for Global Optimization, Information **14**, 66, 2023.
81. J. Park and I. W. Sandberg, Universal Approximation Using Radial-Basis-Function Networks, Neural Computation **3**, pp. 246-257, 1991.
82. Nam Mai-Duy, Thanh Tran-Cong, Numerical solution of differential equations using multiquadric radial basis function networks, Neural Networks 14, pp. 185-199, 2001.
83. N. Mai-Duy, Solving high order ordinary differential equations with radial basis function networks. Int. J. Numer. Meth. Engng. **62**, pp. 824-852, 2005.
84. C. Laoudias, P. Kemppi and C. G. Panayiotou, Localization Using Radial Basis Function Networks and Signal Strength Fingerprints in WLAN, GLOBECOM 2009 - 2009 IEEE Global Telecommunications Conference, Honolulu, HI, 2009, pp. 1-6, 2009.
85. M. Azarbad, S. Hakimi, A. Ebrahimzadeh, Automatic recognition of digital communication signal, International journal of energy, information and communications **3**, pp. 21-33, 2012.
86. D.C. Liu, J. Nocedal, On the Limited Memory Method for Large Scale Optimization, Mathematical Programming B. **45**, pp. 503–528, 1989.
87. R. Fletcher, A new approach to variable metric algorithms, Computer Journal 13, pp. 317-322, 1970.
88. H. Wang, H. Gemmeke, T. Hopp, J. Hesser, Accelerating image reconstruction in ultrasound transmission tomography using L-BFGS algorithm, In: Medical Imaging 2019: Ultrasonic Imaging and Tomography; 109550B (2019) https://doi.org/10.1117/12.2512654 Event: SPIE Medical Imaging, 2019, San Diego, California, United States.
89. Z. Dalvand, M. Hajarian, Solving generalized inverse eigenvalue problems via L-BFGS-B method, Inverse Problems in Science and Engineering **28**, pp. 1719-1746, 2020.
90. Y. Rao, Y. Wang, Seismic waveform tomography with shot-encoding using a restarted L-BFGS algorithm, Scientific Reports **7**, pp. 1-9, 2017.
91. Y. Fei, G. Rong, B. Wang, W. Wang, Parallel L-BFGS-B algorithm on GPU, Computers & Graphics **40**, pp. 1-9, 2014.
92. L. D'Amore, G. Laccetti, D. Romano, G. Scotti, A. Murli, Towards a parallel component in a GPU–CUDA environment: a case study with the L-BFGS Harwell routine, International Journal of Computer Mathematics **92**, pp. 59-76, 2015.
93. M.M. Najafabadi, T.M. Khoshgoftaar, F. Villanustre et al, Large-scale distributed L-BFGS, J Big Data **4**, 22, 2017.
94. J.L. Morales, A numerical study of limited memory BFGS methods, Applied Mathematics Letters **15**, pp. 481-487, 2002.
95. I.G. Tsoulos, Modifications of real code genetic algorithm for global optimization, Applied Mathematics and Computation **203**, pp. 598-607, 2008.
96. 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.
97. 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.
98. M. Wang, Y.Y. Zhang, F. Min, Active learning through multi-standard optimization, IEEE Access **7**, pp. 56772–56784, 2019.
99. J.R. Quinlan, Simplifying Decision Trees. International Journal of Man-Machine Studies **27**, pp. 221-234, 1987.
100. T. Shultz, D. Mareschal, W. Schmidt, Modeling Cognitive Development on Balance Scale Phenomena, Machine Learning **16**, pp. 59-88, 1994.
101. 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.
102. R. Setiono , W.K. Leow, FERNN: An Algorithm for Fast Extraction of Rules from Neural Networks, Applied Intelligence **12**, pp. 15-25, 2000.
103. B. Evans, D. Fisher, Overcoming process delays with decision tree induction. IEEE Expert **9**, pp. 60-66, 1994.
104. 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.
105. I. Kononenko, E. Šimec, M. Robnik-Šikonja, Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF, Applied Intelligence **7**, pp. 39–55, 1997
106. 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.
107. 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.
108. J.G. Dy , C.E. Brodley, Feature Selection for Unsupervised Learning, The Journal of Machine Learning Research **5**, pp 845–889, 2004.
109. S. J. Perantonis, V. Virvilis, Input Feature Extraction for Multilayered Perceptrons Using Supervised Principal Component Analysis, Neural Processing Letters **10**, pp 243–252, 1999.
110. J. Garcke, M. Griebel, Classification with sparse grids using simplicial basis functions, Intell. Data Anal. **6**, pp. 483-502, 2002.
111. J. Mcdermott, R.S. Forsyth, Diagnosing a disorder in a classification benchmark, Pattern Recognition Letters **73**, pp. 41-43, 2016.
112. G. Cestnik, I. Konenenko, I. Bratko, Assistant-86: A Knowledge-Elicitation Tool for Sophisticated Users. In: Bratko, I. and Lavrac, N., Eds., Progress in Machine Learning, Sigma Press, Wilmslow, pp. 31-45, 1987.

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. F. Esposito F., D. Malerba, G. Semeraro, Multistrategy Learning for Document Recognition, Applied Artificial Intelligence **8**, pp. 33-84, 1994.
115. 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.
116. 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.
117. 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.
118. 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.
119. 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.
120. T. Hastie, R. Tibshirani, Non-parametric logistic and proportional odds regression, JRSS-C (Applied Statistics) **36**, pp. 260–276, 1987.
121. M. Dash, H. Liu, P. Scheuermann, K. L. Tan, Fast hierarchical clustering and its validation, Data & Knowledge Engineering **44**, pp 109–138, 2003.
122. 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.
123. 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.
124. P. Zhong, M. Fukushima, Regularized nonsmooth Newton method for multi-class support vector machines, Optimization Methods and Software **22**, pp. 225-236, 2007.
125. 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.
126. 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
127. M. Koivisto, K. Sood, Exact Bayesian Structure Discovery in Bayesian Networks, The Journal of Machine Learning Research **5**, pp. 549–573, 2004.
128. 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.
129. 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.
130. T.F. Brooks, D.S. Pope, A.M. Marcolini, Airfoil self-noise and prediction. Technical report, NASA RP-1218, July 1989.
131. J.S. Simonoff, Smooting Methods in Statistics, Springer - Verlag, 1996.
132. I.Cheng Yeh, Modeling of strength of high performance concrete using artificial neural networks, Cement and Concrete Research. **28**, pp. 1797-1808, 1998.
133. D. Harrison and D.L. Rubinfeld, Hedonic prices and the demand for clean ai, J. Environ. Economics & Management **5**, pp. 81-102, 1978.
134. J.S. Simonoff, Smooting Methods in Statistics, Springer - Verlag, 1996.
135. R.D. King, S. Muggleton, R. Lewis, M.J.E. Sternberg, Proc. Nat. Acad. Sci. USA **89**, pp. 11322–11326, 1992.
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. K. O. Stanley, R. Miikkulainen, Evolving Neural Networks through Augmenting Topologies, Evolutionary Computation **10**, pp. 99-127, 2002.