# Constructing features using a hybrid genetic algorithm

Ioannis G. Tsoulos[*]

Department of Informatics and Telecommunications, University of Ioannina, Greece

**Abstract**

A hybrid procedure that incorporates Grammatical Evolution and a weight decaying technique is proposed here for various problems, classification or regression. The procedure is divided into two phases: during the first phase new features are created from the original ones and in second phase a hybrid method based on a genetic algorithm is used to train neural networks on the previously created features. The proposed procedure is applied on a wide range of datasets from the relevant literature and the results are reported and discussed.

**Keywords**: Genetic algorithm, machine learning, neural networks, Grammatical Evolution.

## 1 Introduction

Artificial Neural networks (ANNs) are programming tools [1, 2], based on a series of parameters that commonly called weights or processing units. They have been used in a variety of problems from different scientific areas such as physics [3, 4, 5], chemistry [6, 7, 8], economics [9, 10, 11], medicine [12, 13] etc. A common way to express a neural network is a function $N(\overrightarrow{x},\overrightarrow{w})$, with $\overrightarrow{x}$ the input vector (commonly called pattern) and $\overrightarrow{w}$ the weight vector. A method that trains a neural network should be used to estimate the vector $\overrightarrow{w}$ for the certain problem. The training procedure can be formulated also an optimization problem, where the target is to minimize the so called error function:

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

In equation 1 the set $\left(\overrightarrow{x_i},y_i\right),\ i=1,...,M$ is the dataset used to train the neural network, with $y_i$ being the actual output for the point $\overrightarrow{x_i}$ . The neural network

[*]Corresponding author. Email: itsoulos@uoi.gr

$N(\overrightarrow{x},\overrightarrow{w})$ can be modeled as a summation of processing units as proposed in [14]:

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

where $H$ is the number of processing units of the neural network and $d$ is the dimension of vector $\overrightarrow{x}$. The function $\sigma(x)$ is the sigmoid function defined as:

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

From the equation 2 one can obtain that the dimension of weight vector $w$ is computed as: $w=(d+2)H$. The function of equation 1 has been minimized with a variety of optimization methods during the past years such as: the Back Propagation method [15, 16], the RPROP method [17, 18, 19], Quasi Newton methods [20, 21], Genetic Algorithms [22, 23], Particle Swarm Optimization [24, 25] etc. All the previously mentioned methods have to overcome two major problems:

- Excessive computational times, because they require processing time proportional to the dimension of the objective problem and the number of processing units as well. For example, a neural network of $H=10$ processing units applied to a test data with $d=3$, is considered as an optimization problem with dimension $w=(d+2)H=50$. An extensive discussion on the problems occurred by the dimensionality on neural networks is presented in [26]. A common approach to overcome this problem is to use the PCA technique to reduce the dimensionality of the objective problem [27, 28, 29] i.e. the parameter $d$.

- The ovetfitting problem. It is quite common for these methods to produce poor results, when they are applied to data (test data) not previously used in the training procedure. This problem is discussed in detail in the article of Geman et all [30] as well as in the article of Hawkins [31]. A variety of methods have been proposed to overcome this problem such as are weight sharing [32], pruning [33, 34, 35], the dropout technique [36], early stopping [37, 38], and weight decaying [39, 40].

This article proposes a method that tackle both the above problems using two major steps. During the first step a new set of features is created from the initial features using a procedure based on the Grammatical Evolution technique [41]. This procedure was introduced in the work of Gavrilis et al [42] and it has been used with success in Spam Identification [43], Fetal heart classification [44], epileptic oscillations in clinical intracranial electroencephalograms [45] etc. The outcomes of the first phase are the modified training and testing data according to the created features. During the second step, a genetic algorithm that incorporates a weight decaying procedure is used to train a neural network on the modified data of the first step.

The rest of this paper is organized as follows: in section 2 the proposed method is described in detail, in section 3 the proposed method is tested on a series of well know datasets from the relevant literature and the results are compared to those of a simple genetic algorithm and finally in section 4 some conclusions are presented.

## 2 Method description

The proposed method has two major phases. In the a procedure that exploits the Grammatical Evolution technique is used, in order to create new features from the old ones. The new features are evaluated using an RBF [46] neural network with $H$ hidden nodes. The RBF network is used during this phase instead of a neural network because the training procedure for RBF networks are must faster than these of neural networks. In the second phase, a hybrid genetic algorithm trains a neural network using the constructed features of the first phase.

### 2.1 The usage of Grammatical Evolution

Grammatical evolution is a evolutionary procedure, where the chromosomes are represent production rules from a BNF grammar. The production procedure initiates from the start symbol of the BNF grammar and iteratively produces programs by replacing non terminal symbols with the right hand of the production rules that will be selected according to the value of each element in the chromosome**.** In the proposed method the BNF grammar of the Figure 1 was used to create a new feature from the initial features. The parameter N denotes the number of original features. $[0, 255]$. Take for example the chromosome $x = [9, 8, 6, 4, 16, 10, 17, 23, 8, 14]$ and $N = 3$. The valid expression $f(x) = x_2 + \cos(x_3)$ is created using a series of production steps shown in Table 1. Each number in the parentheses stands for the sequence number of the production rule. Hence, the process to produce $N_f$ features from the original have as follows:

1. Every chromosome $Z$ is split into $N_f$ parts. Each part $g_i$ will be used to construct a feature.

2. For every part $g_i$ construct a feature $t_i$ using the grammar given in 1

3. Create a mapping function

$$G(\overrightarrow{x}, Z) = \left(t_1\left(\overrightarrow{x}, Z\right), t_2\left(\overrightarrow{x}, Z\right), \ldots, t_{N_f}\left(\overrightarrow{x}, Z\right)\right) \tag{4}$$

   where $\overrightarrow{x}$ is a pattern from the original set and $Z$ is the chromosome.

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)
```

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

| String | Chromosome | Operation |
|---|---|---|
| <expr> | 9,8,6,4,16,10,17,23,8,14 | $9 \mod 3 = 0$ |
| (<expr><op><expr>) | 8,6,4,16,10,17,23,8,14 | $8 \mod 3 = 2$ |
| (<terminal><op><expr>) | 6,4,16,10,17,23,8,14 | $6 \mod 2 = 0$ |
| (<xlist><op><expr>) | 4,16,10,17,23,8,14 | $4 \mod 3 = 1$ |
| (x2<op><expr>) | 16,10,17,23,8,14 | $16 \mod 4 = 0$ |
| (x2+<expr>) | 10,17,23,8,14 | $10 \mod 3 = 1$ |
| (x2+<func>(<expr>)) | 17,23,8,14 | $17 \mod 4 = 1$ |
| (x2+cos(<expr>)) | 23,8,14 | $23 \mod 2 = 1$ |
| (x2+cos(<terminal>)) | 8,14 | $8 \mod 2 = 0$ |
| (x2+cos(<xlist>)) | 14 | $14 \mod 3 = 2$ |
| (x2+cos(x3)) | | |

## 2.2 Feature construction

The steps of the algorithm for the first phase are:

1. **Initialization step**

   (a) **Set** iter=0, generation number.

   (b) **Construct** the set TR $= \{(\overrightarrow{x_1}, y_1), (\overrightarrow{x_2}, y_2), \ldots, (\overrightarrow{x_M}, y_M)\}$ the original train set.

   (c) **Set** $N_c$ the number of chromosomes and $N_f$ the number of desired constructed features.

   (d) **Initialize** randomly in range $[0, 255]$ the integer chromosomes $Z_i, i = 1 \ldots N_c$

   (e) **Set** $N_g$ as the maximum number of generations allowed.

   (f) **Set** $p_s \in [0, 1]$ as the selection rate and $p_m \in [0, 1]$ the mutation rate.

2. **Termination check. If** iter$>=N_g$ **goto** step 6.

3. **Estimate** the fitness $f_i$ of every chromosome $Z_i$ with the following procedure:

   (a) **Use** the procedure described in subsection 2.1 and create $N_f$ features.

   (b) **Create** a modified training set

   $$\text{TN} = \{(G(\overrightarrow{x_1}, Z_i), y_1), (G(\overrightarrow{x_2}, Z_i), y_2), \ldots, (G(\overrightarrow{x_M}, Z_i), y_M)\} \quad (5)$$

   (c) **Train** an RBF neural network $C$ with $H$ processing units on the modified training set TN using the following train error

   $$f_i = \sum_{j=1}^{M} \left(C\left(G\left(\overrightarrow{x_j}, Z_i\right)\right) - y_j\right)^2 \quad (6)$$

4. **Genetic Operators**

   (a) **Selection procedure:** Initially the chromosomes are sorted according to their fitness value. The best chromosomes are placed in the beginning of the population and the worst at the end. The best $(1-p_s) \times N_c$ chromosomes are transferred to the next generation intact. The remain chromosomes are substituted by offsprings created through the crossover and mutation procedures.

   (b) **Crossover procedure:** For every produced offspring two mating chromosomes (parents) are selected from the previous population using tournament selection. The tournament selection is a rather simple selection mechanism defined as: First a set of $K > 1$ randomly selected chromosomes is constructed and subsequently the chromosome with the best fitness value in the previous set is selected as mating chromosome. Having selected the two parents for the offrpsing, the offspring is formed using the one point crossover. In one - point crossover a random point is selected for the two parents and their right-hand side subchromosomes are exchanged.

   (c) **Mutation procedure:** For every element of each chromosome a random number $r \in [0,1]$ is taken. If $r \leq p_m$ then this element is altered randomly by producing a new integer number.

5. **Set** iter=iter+1 and **goto** Step 2.

6. **Get** the best chromosome in the population defined as $Z_l$ with the corresponding fitness value $f_l$ and **Terminate**.

## 2.3 Weight decay mechanism

The quantity $x$ in the equation 3 of the sigmoid function is calculated through many calculations involving the input patterns as well as the weight vector. If the value within the function is too large, then the sigmoid function tends to 0 and this will result in the neural network losing what generalization possibilities it has. In order to estimate the effect of this issue, the quantity $B\left(N\left(\overrightarrow{x},\overrightarrow{w}\right),F\right)$ is defined as shown in the Algorithm 1**.**

## 2.4 Application of genetic algorithm

The main steps of the hybrid genetic algorithm used in the second phase are:

1. **Initialization step**

   (a) **Set** iter=0, the generation number.

   (b) **Set** TN the modified training set, where

$$\mathrm{TN} = \left\{\left(G\left(\overrightarrow{x_1},Z_l\right),y_1\right),\left(G\left(\overrightarrow{x_2},Z_l\right),y_2\right),\ldots,\left(G\left(\overrightarrow{x_M},Z_l\right),y_M\right)\right\} \quad (7)$$

**Algorithm 1 Calculation of the bounding quantity for neural network $N(x,w)$.**

```
1. Define $b=0$
2. For $i=1..K$ Do
    (a) For $j=1..M$ Do
        i. Define $v=\sum_{kT=1}^{d} w_{(d+2)i-(d+i)+k}x_{jk}+w_{(d+2)i}$
        ii. If $|v|>F$ set $b=b+1$
    (b) EndFor
3. EndFor
4. Return $\frac{b}{K\star M}$
```

```
    (c) Initialize randomly the double precision chromosomes $D_i, i=1\ldots N_c$ in range $[L_N,R_N]$. The size of each chromosome is set to $W=(N_f+2)H$
2. Termination check. If iter>=$N_g$ goto step 6
3. Fitness calculation step.
    (a) For every chromosome $D_i$
        i. Calculate the quantity $B_i=\sum_{x\in \mathrm{TN}}\left(B\left(N\left(x,D_i\right),F\right)\right)$ using the algorithm 1.
        ii. Calculate the quantity $E_i=\sum_{(x,y)\in \mathrm{TN}}\left(N\left(x,D_i\right)-y\right)^2$ , the training error of the neural network where the chromosome $D_i$ is used as the weight vector.
        iii. Set $f_i=-E_i\left(1+\lambda B_i^2\right)$, where $\lambda>0$ as the fitness of $D_i$.
    (b) End For
4. Genetic operations Step. Apply the same genetic operations as in the first algorithm of subsection 2.2.
5. Set iter=iter+1 and goto step 2
6. Local Search step.
    (a) Get the best chromosome $D^*$ of the population.
    (b) For i=1..W Do
        i. Set $p_i=D_i^*$
        ii. Set $LM_i=-\alpha\left|p_i\right|$
        iii. Set $RM_i=\ \alpha\left|p_i\right|$ , $\alpha>1$ .
```

Table 2: Experimental parameters.

| PARAMETER | VALUE |
|---|---|
| $H$ | 10 |
| $N_c$ | 500 |
| $N_f$ | 2 |
| $p_s$ | 0.10 |
| $p_m$ | 0.05 |
| $N_g$ | 200 |
| $L_N$ | -10.0 |
| $R_N$ | 10.0 |
| $F$ | 20.0 |
| $\lambda$ | 100.0 |
| $\alpha$ | 5.0 |

(c) **EndFor**

(d) **Set** $L^* = \mathcal{L}\left(D^*, LM, RM\right)$ where $\mathcal{L}()$ is a local optimization method procedure that searches for a local optimum of $N\left(x, D^*\right)$ inside the bounds $\left[\overrightarrow{LM}, \overrightarrow{RM}\right]$. The TOLMIN [47] local optimization procedure used in the above algorithm, which is modified version BFGS local optimization procedure[48].

(e) **Apply** the optimized neural network $N\left(x, D^*\right)$ to the test set, that has been modified using the same transformation procedure as in the train set and report the final results.

## 3 Experiments

The proposed method was tested against neural network trained by a genetic algorithm (denoted as MLP GEN) on some classification and regression datasets from the relevant literature. The software for the algorithm was coded using ANSI C++ and all the experiments were conducted using the OpenMP library [49] for parallelization purposes. The experiments were conducted 30 times with different seed for the random generator each time and averages were taken. The random generator used was the function drand48() of the C programming language. The classification error is reported for classification datasets on the test set and the average mean squared error for regression datasets. Also, for more reliability in the results the common used method of 10 - fold cross validation was used. The values for the parameters of the used algorithms are reported in Table 2.

## 3.1 Experimental datasets

A series of well - known datasets were used in the experiments found in the Machine Learning Repository `http://www.ics.uci.edu/~mlearn/MLRepository.html` in Keel repository `https://sci2s.ugr.es/keel/`:

1. **Balance** dataset [50], used in psychological experiments.
2. **Dermatology** dataset [51], which is used for differential diagnosis of erythemato-squamous diseases.
3. **Glass** dataset. This dataset contains glass component analysis for glass pieces that belong to 6 classes.
4. **Hayes Roth** dataset [52].
5. **Heart** dataset [53], used to detect heart disease.
6. **Ionosphere** dataset, a meteorological dataset used in various research papers [54, 55].
7. **Parkinsons** dataset,[56] which is created using a range of biomedical voice measurements from 31 people, 23 with Parkinson's disease (PD).The dataset has 22 features.
8. **Pima** dataset, related to the diabetes disease.
9. **PopFailures** dataset [57], used in meteorology.
10. **Spiral** dataset, which is an artificial dataset with two classes. The features in the first class are constructed as:: $x_1 = 0.5t\cos\left(0.08t\right),\ x_2 = 0.5t\cos\left(0.08t + \frac{\pi}{2}\right)$ and for the second class the used equations are : $x_1 = 0.5t\cos\left(0.08t + \pi\right),\ x_2 = 0.5t\cos\left(0.08t + \frac{3\pi}{2}\right)$
11. **Wine**: dataset, which is related to chemical analysis of wines and it has been used in comparison in various research papers [58, 59].
12. **Wdbc** dataset: which contains data for breast tumors.
13. As an real word example, consider an EEG dataset described in [60, 61] is used here. The dataset consists of five sets (denoted as Z, O, N, F and S) each containing 100 single-channel EEG segments each having 23.6 sec duration. With different combinations of these sets the produced datasets are Z_F_S, ZO_NF_S, ZONF_S.

The regression datasets are available from the Statlib URL `ftp://lib.stat.cmu.edu/datasets/index.html` and other sources:

1. **BK** dataset. This dataset comes from Smoothing Methods in Statistics [62] and is used to estimate the points scored per minute in a basketball game. The dataset has 96 patterns of 4 features each.

2. **BL** dataset: This dataset can be downloaded from StatLib. It contains data from an experiment on the affects of machine adjustments on the time to count bolts. It contains 40 patters of 7 features each.

3. **Housing** dataset, described in [63].

4. **Laser** dataset, which is related to laser experiments.

5. **NT** dataset [64], which is related to the body temperature measurements.

6. **Quake** dataset, used to estimate the strength of a earthquake.

7. **FA** dataset dataset, which contains percentage of body fat and ten body circumference measurements. The goal is to fit body fat to the other measurements.

8. **PY** dataset [65], used to learn Quantitative Structure Activity Relationships (QSARs).

## 3.2 Experimental results

The table 3 represents the comparative results for the classification datasets and the table 4 shows the results for the regression problems. The column MLP GEN stands for the artificial neural network with $H$ hidden nodes that was trained by a genetic algorithm with $N_g$ chromosomes and the column FC MLP denotes the proposed method. Also, the column GAIN indicates the percentage gain in test error (classification or regression) obtained by the suggested method. Judging by the results produced, it is obvious that the proposed technique is significantly superior to a neural network trained with a genetic algorithm. Also the percentage gain is extremely high in many of the cases. On the other hand the proposed method is slower than a simple genetic algorithm because it relies on two steps and every step involves the usage of a genetic algorithm.

## 4 Conclusions

A hybrid method was proposed here for classification and regression problems. The method is composed by two steps: during the first step a recently introduced method was used to produce artificial features from the initial features. The feature construction method has utilized the commonly used technique of Grammatical Evolution to produce artificial features. The outcome of the first step was the modified train and test sets of the objective problem. During the second step, a hybrid genetic algorithm which preserves the generalization abilities of the neural network was incorporated. The method suggested here was tested on a variety of classification and regression problems and the results were very promising.

Table 3: Experimental results for classification datasets.

| DATASET | MLP GEN | FC MLP | **GAIN** |
|---|---|---|---|
| BALANCE | 8.23% | 0.30% | **96.35%** |
| DERMATOLOGY | 10.01% | 4.98% | **50.25%** |
| GLASS | 58.03% | 45.84% | **21.00%** |
| HAYES ROTH | 35.26% | 23.26% | **34.03%** |
| HEART | 25.46% | 17.71% | **30.44%** |
| IONOSPHERE | 13.67% | 8.42% | **38.41%** |
| PARKINSONS | 17.47% | 10.10% | **36.46%** |
| PIMA | 32.98% | 23.76% | **27.96%** |
| POPFAILURES | 7.66% | 4.66% | **39.16%** |
| SPIRAL | 45.71% | 26.53% | **41.96%** |
| WINE | 20.82% | 7.31% | **64.89%** |
| WDBC | 6.32% | 3.47% | **45.09%** |
| Z_F_S | 9.42% | 5.52% | **41.40%** |
| Z_O_N_F_S | 60.38% | 31.20% | **48.33%** |
| ZO_NF_S | 8.06% | 4.00% | **50.37%** |

Table 4: Experiments for regression datasets.

| DATASET | MLP GEN | FC MLP | **GAIN** |
|---|---|---|---|
| BK | 0.21 | 0.03 | **85.71%** |
| BL | 0.84 | 0.005 | **99.40%** |
| Housing | 30.05 | 10.77 | **64.16%** |
| Laser | 0.003 | 0.002 | **33.33%** |
| NT | 1.11 | 0.01 | **99.10%** |
| Quake | 0.07 | 0.03 | **57.14%** |
| FA | 0.04 | 0.01 | **75.00%** |
| PY | 0.21 | 0.02 | **90.47%** |

## 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] P. Baldi, K. Cranmer, T. Faucett et al, Parameterized neural networks for high-energy physics, Eur. Phys. J. C **76**, 2016.

[4] 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

[5] G. Carleo,M. Troyer, Solving the quantum many-body problem with artificial neural networks, Science **355**, pp. 602-606, 2017.

[6] 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.

[7] 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.

[8] 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.

[9] Lukas Falat and Lucia Pancikova, Quantitative Modelling in Economics with Advanced Artificial Neural Networks, Procedia Economics and Finance **34**, pp. 194-201, 2015.

[10] 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.

[11] G. Tkacz, Neural network forecasting of Canadian GDP growth, International Journal of Forecasting **17**, pp. 57-69, 2001.

[12] 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.

[13] Ronadl Bartzatt, Prediction of Novel Anti-Ebola Virus Compounds Utilizing Artificial Neural Network (ANN), Chemistry Faculty Publications **49**, pp. 16-34, 2018.

[14] I.G. Tsoulos, D. Gavrilis, E. Glavas, Neural network construction and training using grammatical evolution, Neurocomputing **72**, pp. 269-277, 2008.

[15] D.E. Rumelhart, G.E. Hinton and R.J. Williams, Learning representations by back-propagating errors, Nature **323**, pp. 533 - 536 , 1986.

[16] T. Chen and S. Zhong, Privacy-Preserving Backpropagation Neural Network Learning, IEEE Transactions on Neural Networks **20**, , pp. 1554-1564, 2009.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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

[23] X. Yao, Evolving artificial neural networks, Proceedings of the IEEE, 87(9), pp. 1423-1447, 1999.

[24] 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.

[25] Jianbo Yu, Shijin Wang, Lifeng Xi, Evolving artificial neural networks using an improved PSO and DPSO **71**, pp. 1054-1060, 2008.

[26] Verleysen M., Francois D., Simon G., Wertz V., 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.

[27] 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.

[28] 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.

[29] 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

[30] S. Geman, E. Bienenstock and R. Doursat, Neural networks and the bias/variance dilemma, Neural Computation 4 , pp. 1 - 58, 1992.

[31] Douglas M. Hawkins, The Problem of Overfitting, J. Chem. Inf. Comput. Sci. **44**, pp. 1–12, 2004.

[32] S.J. Nowlan and G.E. Hinton, Simplifying neural networks by soft weight sharing, Neural Computation 4, pp. 473-493, 1992.

[33] 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.

[34] 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.

[35] M. Augasta and T. Kathirvalavakumar, Pruning algorithms of neural networks — a comparative study, Central European Journal of Computer Science, 2003.

[36] 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.

[37] Lutz Prechelt, Automatic early stopping using cross validation: quantifying the criteria, Neural Networks **11**, pp. 761-767, 1998.

[38] X. Wu and J. Liu, A New Early Stopping Algorithm for Improving Neural Network Generalization, 2009 Second International Conference on Intelligent Computation Technology and Automation, Changsha, Hunan, 2009, pp. 15-18.

[39] N. K. Treadgold and T. D. Gedeon, Simulated annealing and weight decay in adaptive learning: the SARPROP algorithm,IEEE Transactions on Neural Networks **9**, pp. 662-668, 1998.

[40] 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.

[41] M. O'Neill, C. Ryan, Grammatical evolution, IEEE Trans. Evol. Comput. **5,**pp. 349–358, 2001.

[42] Dimitris Gavrilis, Ioannis G. Tsoulos, Evangelos Dermatas, Selecting and constructing features using grammatical evolution, Pattern Recognition Letters **29**,pp. 1358-1365, 2008.

[43] Dimitris Gavrilis, Ioannis G. Tsoulos, 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.

[44] George Georgoulas, Dimitris Gavrilis, Ioannis G. Tsoulos, 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

[45] Otis Smart, Ioannis G. Tsoulos, Dimitris Gavrilis, George Georgoulas, Grammatical evolution for features of epileptic oscillations in clinical intracranial electroencephalograms, Expert Systems with Applications **38**, pp. 9991-9999, 2011

[46] J. Park and I. W. Sandberg, Universal Approximation Using Radial-Basis-Function Networks, Neural Computation 3, pp. 246-257, 1991.

[47] M.J.D. Powell, A Tolerant Algorithm for Linearly Constrained Optimization Calculations, Mathematical Programming **45**, pp 547, 1989.

[48] R. Fletcher, A new approach to variable metric algorithms, Computer Journal **13**, pp. 317-322, 1970.

[49] R. Chandra, L. Dagum, D. Kohr, D. Maydan,J. McDonald and R. Menon, Parallel Programming in OpenMP, Morgan Kaufmann Publishers Inc., 2001.

[50] T. Shultz, D. Mareschal, W. Schmidt, Modeling Cognitive Development on Balance Scale Phenomena, Machine Learning **16**, pp. 59-88, 1994.

[51] 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.

[52] 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.

[53] I. Kononenko, E. Šimec, M. Robnik-Šikonja, Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF, Applied Intelligence **7**, pp. 39–55, 1997.

[54] J.G. Dy , C.E. Brodley, Feature Selection for Unsupervised Learning, The Journal of Machine Learning Research **5**, pp 845–889, 2004.

[55] S. J. Perantonis, V. Virvilis, Input Feature Extraction for Multilayered Perceptrons Using Supervised Principal Component Analysis, Neural Processing Letters **10**, pp 243–252, 1999.

[56] Max A. Little, Patrick E. McSharry, Eric J. Hunter, Lorraine O. Ramig (2008), 'Suitability of dysphonia measurements for telemonitoring of Parkinson's disease', IEEE Transactions on Biomedical Engineering **56**, pp. 1015-1022, 2009.

[57] 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.

[58] 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.

[59] P. Zhong, M. Fukushima, Regularized nonsmooth Newton method for multi-class support vector machines, Optimization Methods and Software **22**, pp. 225-236, 2007.

[60] 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.

[61] 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

[62] J.S. Simonoff, Smooting Methods in Statistics, Springer - Verlag, 1996.

[63] D. Harrison and D.L. Rubinfeld, Hedonic prices and the demand for clean ai, J. Environ. Economics & Management **5**, pp. 81-102, 1978.

[64] Mackowiak, P.A., Wasserman, S.S., Levine, M.M., 1992. A critical appraisal of 98.6 degrees f, the upper limit of the normal body temperature, and other legacies of Carl Reinhold August Wunderlich. J. Amer. Med. Assoc. 268, 1578–1580

[65] R.D. King, S. Muggleton, R. Lewis, M.J.E. Sternberg, Proc. Nat. Acad. Sci. USA **89**, pp. 11322–11326, 1992.