Improved Evolutionary Extreme Learning Machines Based on Particle Swarm Optimization and Clustering Approaches

Luciano D. S. Pacifico, Informatics Center, Federal University of Pernambuco, Recife, Brazil

Teresa B. Ludermir, Informatics Center, Federal University of Pernambuco, Recife, Brazil

ABSTRACT

Extreme Learning Machine (ELM) is a new learning method for single-hidden layer feedforward neural network (SLFN) training. ELM approach increases the learning speed by means of randomly generating input weights and biases for hidden nodes rather than tuning network parameters, making this approach much faster than traditional gradient-based ones. However, ELM random generation may lead to non-optimal performance. Particle Swarm Optimization (PSO) technique was introduced as a stochastic search through an n-dimensional problem space aiming the minimization (or the maximization) of the objective function of the problem. In this paper, two new hybrid approaches are proposed based on PSO to select input weights and hidden biases for ELM. Experimental results show that the proposed methods are able to achieve better generalization performance than traditional ELM in real benchmark datasets.

Keywords: *Artificial Neural Networks, Extreme Learning Machine, Hybrid Systems, Particle Swarm Optimization, Population Stereotyping, Selection Operator*

1. INTRODUCTION

Artificial neural networks (ANNs) are known as universal approximators and computational models with remarkable properties such as adaptability, capacity of learning by examples and the ability to generalize data (Haykin, 1998).

Neural networks are of great use in pattern classification applications, and through a supervised learning perspective, they are considered a general method for constructing mappings between a group of sample vectors (training set) and the corresponding classes, allowing the classification of unseen data as one of the classes learned in the training process.

One of the most used ANN models is the well-known Multi-Layer Perceptron (MLP). The training process of MLPs for pattern classification consists of two main tasks: selection of an appropriate architecture for the problem and the adjustment of the connection weights of the network.

DOI: 10.4018/jncr.2012070101

Gradient-based learning strategies, such as Backpropagation (BP) and its variant Levenberg-Marquardt (LM-BP), have been extensively used in the training of MLPs, but these approaches are usually slower than required in learning, and may also get stuck in local minima (Zhu et al., 2005).

Extreme learning machine (ELM) was proposed as an efficient learning algorithm for single-hidden layer feedforward neural network (SLFN) (Huang et al., 2006). ELM increases the learning speed by means of randomly generating weights and biases for hidden nodes, differently from gradient-based approaches, which commonly tune iteratively the network parameters.

Although ELM is fast and presents good generalization performance, as the output weights are computed based on the prefixed input weights and hidden biases using the Moore-Penrose (MP) generalized inverse, there may exist a set of non-optimal input weights and hidden biases, and it might suffer from the *overfitting* as the learning model will approximate all training samples well.

Global search techniques, such as Tabu Search (TS) (Glover, 1986), Evolutionary Algorithms (EAs, like Genetic Algorithm - GA) (Eiben & Smith, 2003), Differential Evolution (DE) (Storn & Price, 1995; Storn & Price, 1997), Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995; Kennedy & Eberhart, 2001; van den Bergh, 2002) and Group Search Optimization (GSO) (He et al., 2006; He et al., 2009), are widely used in scientific and engineering problems, and these strategies have been combined with ANNs to perform various tasks, such as connection weight initialization, connection weight training and architecture design.

In this paper, we present two new hybrid evolutionary approaches based on Particle Swarm Optimization technique to select input weights and hidden biases for Extreme Learning Machine neural network: PSO-ELM-CS_2 and GCPSO-CS_2. These methods are extensions from the PSO-ELM-CS_1 and GCPSO-ELM-CS_1 approaches, respectively, presented in Pacifico and Ludermir (2012). The Particle Swarm Optimization (PSO) consists of a stochastic global search originated from the attempt to graphically simulate the social behavior of a flock of birds looking for resources.

For the proposed methods, individuals in the PSO population were divided into groups by a clustering algorithm, following the idea of "population stereotyping" presented in Kennedy (2000). The *lbest* topology was adopted in a way that PSO particles will update according to individuals from its neighborhood. A selection operator was also applied to all strategies, based on the ideas of Angeline (1999).

Some evolutionary strategies have been adopted for the ELM context. Zhu et al. (2006) introduces a hybrid form of differential evolutionary (DE) algorithm to search for optimal input weights and hidden biases for ELM, called E-ELM to train SLFN with more compact networks.

Xu and Shu (2006) presented a new evolutionary ELM based on PSO for prediction task. In Saraswathi et al. (2011) a combination of Integer Coded Genetic Algorithm (ICGA) and Particle Swarm Optimization (PSO), coupled with the ELM has been used for gene selection and cancer classification, where the ICGA and PSO-ELM selected an optimal set of genes which are then used to build a classifier to develop an algorithm (ICGA_PSO_ELM) that could handle sparse data and sample imbalance.

In Cho and Lee (2007), an optimization method based on Bacterial Foraging (BF) algorithm was proposed to adjust the input weights and hidden biases for the ELM.

Lahoz et al. (2011) presented a bi-objective micro genetic ELM (μG-ELM) to generate the hidden weights and biases for ELM, and it also used a regression based strategy to select the appropriate number of hidden nodes.

In Silva et al. (2011a), the ELM was combined with Group Search Optimization (GSO) algorithm (GSO-ELM), and four different forms of handling individuals (members) that fly out of the search space bounds were used. The GSO was used to optimize the input weights and hidden biases for ELM, and also has found a more compact architecture than ELM for four of the six tested datasets.

Silva et al. (2011b) presented four new hybrid approaches to optimize ELM input weights, hidden biases and architecture, based on cooperative PSO variants (van den Bergh & Engelbrecht, 2004; Carvalho & Ludemir, 2006a; Carvalho & Ludemir, 2006b): Guaranteed Convergence PSO (GCPSO-ELM), Cooperative PSO (CPSO-S_k-ELM), Hybrid CPSO (CPSO-H_k-ELM) and a combination of GCPSO and CPSO-H_k (GC-CPSO-H_k-ELM).

Although Silva et al. (2011b) also presented PSO variations using *lbest* neighborhood approach, the local groups were fixed (i.e., the individuals do not change groups, keeping the same group throughout the algorithm execution) and followed a "divide-and-conquer" strategy, trying to update only its reduced set of variables (i.e., a reduced set of the search space dimensions), aiming to give its contribution towards a global best solution.

In this work, each cluster updates trying to find its own local best solution, exploring different regions of the search space, and each cluster individual has a complete vision of the search space dimension set, trying to find only a local best solution. Clusters may also vary throughout the algorithm execution, with individuals leaving their former clusters to join into another ones. In this work, each individual belongs to one cluster only at each iteration.

This paper is organized as follows. The next section (Section 2) presents the Extreme Learning Machine (ELM), Particle Swarm Optimization (PSO), Guaranteed Convergence Particle Swarm Optimization (GCPSO) (Carvalho & Ludemir, 2006a; Carvalho & Ludemir, 2006b) and the clustering scheme for population stereotyping adopted. Next, the proposed hybrid PSO approaches (Section 3) and the experimental results (Section 4) are shown. Finally, the conclusions and suggestions for future works are given (Section 5).

2. PRELIMINARIES

2.1. Extreme Learning Machine

Extreme learning machine (ELM) was proposed in Huang, *et al.* (2006). The main concept behind the ELM lies in the random initialization of the input weights and hidden biases. Suppose we are training SLFNs with N hidden neurons and activation function $f(x)$ to learn M distinct samples $(\mathbf{x}_i, \mathbf{t}_i)$, where $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{ik}]^T \in \Re^k$ and $\mathbf{t}_i = [t_{i1}, t_{i2}, \ldots, t_{id}]^T \in \Re^d$. By doing so, the nonlinear system has been converted to a linear system:

$$\mathbf{H}\beta = \mathbf{T}$$

where $\mathbf{H}$ is the hidden-layer output matrix denoted by:

$$\mathbf{H} = \begin{pmatrix} f\left(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1\right) & \cdots & f\left(\mathbf{w}_N \cdot \mathbf{x}_1 + b_N\right) \\ \vdots & \ddots & \vdots \\ f\left(\mathbf{w}_1 \cdot \mathbf{x}_M + b_1\right) & \cdots & f\left(\mathbf{w}_N \cdot \mathbf{x}_M + b_N\right) \end{pmatrix}$$

where $\mathbf{w}_j = [w_{j1}, w_{j2}, \ldots, w_{jk}]^T$ $(j = 1, \ldots, N)$ is the weight vector connecting jth hidden neuron and input neurons, and b_j denotes the bias of jth hidden neuron; $\mathbf{w}_j \cdot \mathbf{x}_i$ $(i = 1, \ldots, M)$ denotes the inner product of w_j and x_i; $\beta = [\beta_1, \beta_2, \ldots, \beta_N]^T$ is the matrix of output weights and $\beta_j = [\beta_{j1}, \beta_{j2}, \ldots, \beta_{jd}]^T$ $(j = 1, \ldots, N)$ denotes the weight vector connecting the jth hidden neuron and output neurons; $\mathbf{T} = [\hat{\mathbf{t}}_1, \hat{\mathbf{t}}_2, \ldots, \hat{\mathbf{t}}_M]^T$ is the matrix of targets (desired output). In the case where the SLFN perfectly approximates the data, the errors between the estimated outputs $\hat{\mathbf{t}}_i$ and the actual outputs $\mathbf{t}_i$ are zero and the relation is:

$$\mathbf{t}_i = \sum_{j=1}^{N} \beta_j f\left(\mathbf{w}_j \cdot \mathbf{x}_i + b_j\right)$$

Thus, the determination of the output weights (linking the hidden layer to the output layer) is determined by the least-square solution to the linear system represented by Equation (1). The minimum norm least-square (LS) solution to the linear system is:

$$\hat{\boldsymbol{\beta}} = \mathbf{H}^{\dagger}\,\mathbf{T} \tag{1}$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose (MP) generalized inverse of matrix $\mathbf{H}$. The minimum norm LS solution is unique and has the smallest norm among all the LS solutions. As analyzed by Huang, *et al*. [1], ELM using such MP inverse method tends to obtain good generalization performance with dramatically increased learning speed. A pseudocode for the ELM algorithm is presented in Figure 1.

2.2. Particle Swarm Optimization

The Particle Swarm Optimization (PSO) technique was introduced by Kennedy & Eberhart (1995) as a stochastic search through an *n*-dimensional problem space aiming the minimization (or maximization) of the objective function of the problem.

The PSO was built through the attempt to graphically simulate the choreography of a flock of birds flying to resources. Later, looking for theoretical foundations, studies were realized concerning the way individuals in groups interact, exchanging information and reviewing personal concepts improving their adaptation to the environment (Kennedy & Eberhart, 2001).

The PSO is a population based technique, where the population is called swarm. Each individual, called particle, represents a potential solution to the task at hand. Each particle $i \in \Re^n$ ($1 \le i \le s$, where *s* is the swarm size) keeps its position $\mathbf{x}_i(t)$, its velocity $\mathbf{v}_i(t)$ and its best position found so far $\mathbf{y}_i(t)$. The swarm also keeps track of its global best position found so far $\hat{\mathbf{y}}(t)$.

During each iteration, the new velocity of the *i*th particle is determined according to its best position found so far $\mathbf{y}_i$ (t) and the global best position $\hat{\mathbf{y}}(t)$. The new velocity $\mathbf{v}_i(t+1)$ $= [\, v_{i1}\left(t+1\right), ..., \, v_{in}\left(t+1\right)]$ and the new position $\mathbf{x}_i(t+1)$ for the *i*th particle are determined by equations Equation (2) and Equation (3), respectively.

$$\begin{aligned} v_{ij}\left(t+1\right) &= w v_{ij}\left(t\right) + c_1 r_1 \left(y_{ij}\left(t\right) - x_{ij}\left(t\right)\right) \\ &+ c_2 r_2 \left(\hat{y}_j\left(t\right) - x_{ij}\left(t\right)\right) \end{aligned} \tag{2}$$

$$\begin{aligned} &\mathbf{x}_i\left(t+1\right) = \mathbf{x}_i\left(t\right) + \mathbf{v}_i \;\; (t+1) \\ &1 \le i \le s, 1 \le j \le n \end{aligned} \tag{3}$$

where c_1 and c_2 local and global are acceleration coefficients, respectively, typically set to equal values usually in the interval $0 < c_1, c_2 \le 2.0$, r_1 and r_2 are random values taken from an uniform distribution $U(0,1)$ and w is the inertial weight (momentum term), usually vary linearly from 0.9 to 0.4 during PSO iterations.

Figure 1. Pseudocode for the ELM algorithm

Initialization: Randomly initialize the input weights and hidden biases;
Calculate the hidden-layer output matrix **H**;
Estimate the $\mathbf{H}^{\dagger}$ as the MP generalized inverse obtained from **H**;
Calculate the output weights matrix $\hat{\beta}$

The local best position visited so far for the *i*th particle $\mathbf{y}_i(t+1)$ is updated according to Equation (4), and the swarm global best position visited so far $\hat{\mathbf{y}}(t+1)$ is updated according to Equation (5).

$$\mathbf{y}_i(t+1) = \begin{cases} \mathbf{y}_i(t), if\ f(\mathbf{x}_i(t+1)) \geq f(\mathbf{y}_i(t)) \\ \mathbf{x}_i(t+1),\ otherwise \end{cases}$$
$$1 \leq i \leq s \tag{4}$$

$$\hat{\mathbf{y}}\left(t+1\right) = \underset{\mathbf{y}_i(t+1), 1 \leq i \leq s}{\arg\min} f(\mathbf{y}_i(t+1)) \tag{5}$$

The inertial weight is similar to the momentum term in a gradient descent neural network training algorithm and to the temperature adjustment schedule found in Simulated Annealing heuristic (Kirkpatrick et al., 1983).

The pseudocode for the PSO algorithm is presented in Figure 2.

2.3. Guaranteed Convergence Particle Swarm Optimization

In PSO, if in iteration *t* the *i*th particle reaches the global best point ever found by the swarm (i.e., $\mathbf{x}_i(t) = \mathbf{y}_i(t) = \hat{\boldsymbol{y}}(t)$), the velocity update equation (Equation (1)) is entirely dependent on the inertial term $w\mathbf{v}_i$. If the previous velocity of that particle is very close to zero then the particle will stop moving, pushing the particles to that point and causing the premature convergence of the swarm.

The Guaranteed Convergence PSO (GCPSO) (Carvalho & Ludemir, 2006a; Carvalho & Ludemir, 2006b), introduces a modification to the velocity equation for standard PSO, which will affect only the particles that reached the global best position of the search space, making them avoid the premature convergence of the swarm and, at the same time, they will look for better solutions at the vicinity of the current global best position $\hat{\boldsymbol{y}}(t)$. The other particles of the swarm continue to use the standard velocity update equation, i.e. the Equation (1).

The new velocity equation for the current best particle $\mathbf{x}_i(t)$ is given by Equation (6):

$$v_{ij}\left(t+1\right) = -x_{ij}\left(t\right) + \hat{y}_j\left(t\right) + wv_{ij}\left(t\right) + \rho(t)(1 - 2r\left(t\right)) \tag{6}$$

where $r(t)$ is a random uniform number taken from $U(0, 1)$ and $\rho(t)$ is an adaptive scaling factor that makes the PSO to perform a random

Figure 2. Pseudocode for the PSO algorithm

```
Initialization: Randomly initialize all particles of the swarm;
While the termination conditions are not met do
 For each particle of the swarm do
  Calculate its fitness function f(x_i);
  If f(x_i) < f(y_i) then
   y_i = x_i;
  End_If
 End_For
 ŷ = arg min_{y_i, 1≤i≤s} f(y_i)
 Update velocity and position of each particle;
End_While
```

search surrounding the best particle of the swarm.

The next $\rho(t)$ value is determined by the expression Equation(7), in which *numSuccesses* and *numFailures* denote, respectively, the number of consecutive successes and failures of the search in minimizing the objective function, and s_c and f_c are threshold parameters.

Whenever the *numSuccesses* counter exceeds the success threshold s_c, this means that the area surrounding the best position may be enlarged leading to the doubling of the $\rho(t)$ value. Similarly, when the *numFailures* counter exceeds the failure threshold f_c, it means that the area surrounding the global best position is too big and need to be reduced as can be seen in Equation (7):

$$\rho(t+1)\begin{cases} 2\rho(t), if\ numSucesses \geq s_c \\ 0.5\rho(t), if\ numFailures \geq f_c \\ \rho(t), \quad otherwise \end{cases} \quad (7)$$

Every time that the success or failure counters exceed their corresponding thresholds, s_c and f_c, respectively, the threshold exceeded is increased. Every iteration that the search succeeds in minimize the current best position, the *numSuccesses* counter is increased and the *numFailures* counter is reset to zero; every iteration that the best global position $\hat{\mathbf{y}}(t)$ is not updated, the *numFailures* counter is increased and the *numSuccesses* counter is reset to zero.

The pseudocode for the GCPSO algorithm is presented in Figure 3.

A constriction factor was presented (Clerc, 1999; Corne et al., 1999; Clerc & Kennedy, 2002) to help ensure the swarm convergence, given by the following equation (Equation (8)):

$$v_{ij}(t+1) = \lambda[v_{ij}(t) + c_1 r_1 (y_{ij}(t) - x_{ij}(t)) + c_2 r_2 (\hat{y}_j(t) - x_{ij}(t))] \quad (8)$$

where,

$$\lambda = \frac{2}{\left|2 - \psi - \sqrt{\psi^2 - 4\psi}\right|}, \psi = c_1 + c_2, \psi > 4$$

2.4. Population Stereotyping by Clustering Analysis

In PSO, there are two main behaviors concerning particles' update according to the influence of their neighbors: *gbest* and *lbest* approaches. Traditional PSO adopts the *gbest* population strategy, in which the trajectory of each particle's search is influenced by the best point found so far by any individual of the swarm.

The *lbest* population allows each individual to be influenced by some smaller number of

Figure 3. Pseudocode for the GCPSO algorithm

Initialization: Randomly initialize all particles of the swarm;
While the termination conditions are not met **do**
 For each particle of the swarm **do**
  Calculate its fitness function $f(\mathbf{x}_i)$ $f(\mathbf{x}_i)$;
  If $f(\mathbf{x}_i) < f(\mathbf{y}_i)$ **then**
   $\mathbf{y}_i = \mathbf{x}_i$;
  End_If;
 End_For;
 $\hat{\mathbf{y}} = \arg\min_{\mathbf{y}_i, 1 \leq i \leq s} f(\mathbf{y}_i)$
 Update velocity and position of each particle; For the best particle found so far, update the velocity according to Equation (6).
 Update ρ according to Equation (7), and, if necessary, *numSuccesses*, *numFailures*, s_c and f_c.
End_While

adjacent particles belonging to its neighborhood. In *lbest*, each neighborhood maintains its own local best solution. The *gbest* approach has the advantage of faster convergence than *lbest*, but it can be trapped in local minima points, while *lbest* can avoid this problem by exploring different regions of the search space with each of its subpopulations.

In (Kennedy, 2000), the *lbest* neighborhood was adopted in such a way that particles were grouped according to their similarities by a clustering algorithm, forming "social stereotypical groups". The cluster center $\mathbf{g}_c$ of each group *C* was calculated and used to modify the new velocity equation of *i*th particle in three different ways: the individual local best position found so far was replaced with its cluster center (Equation (9)); the global best position found so far was substituted by its cluster center $\mathbf{g}_{ij}$ (Equation (10)); both best terms were replaced with their cluster centers (Equation (11)). These substitutions are used to simulate stereotypical behaviors, so the individual's behavior is guided by the average behavior of its group (cluster center):

$$v_{ij}\left(t+1\right)=wv_{ij}\left(t\right)+c_1r_1\left(g_{cj}\left(t\right)-x_{ij}\left(t\right)\right)+c_2r_2\left(\hat{y}_j\left(t\right)-x_{ij}\left(t\right)\right) \quad (9)$$

$$v_{ij}\left(t+1\right)=wv_{ij}\left(t\right)+c_1r_1\left(y_{ij}\left(t\right)-x_{ij}\left(t\right)\right)+c_2r_2\left(g_{\hat{y}j}\left(t\right)-x_{ij}\left(t\right)\right) \quad (10)$$

$$v_{ij}\left(t+1\right)=wv_{ij}\left(t\right)+c_1r_1\left(g_{cj}\left(t\right)-x_{ij}\left(t\right)\right)+c_2r_2\left(g_{\hat{y}j}\left(t\right)-x_{ij}\left(t\right)\right) \quad (11)$$

Different PSO *lbest* methods can be found in literature. Suganthan (1999) investigated the use of spatial topologies for PSO. In Kennedy and Mendes (2002), neighborhood topologies were used to improve the performance of PSO.

3. PROPOSED METHODS

This section presents two new hybrid Particle Swarm Optimization approaches, used to select input weights and hidden biases for Extreme Learning Machine neural network: PSO-ELM-CS_2 and GCPSO-ELM-CS_2. These methods are extensions from the PSO-ELM-CS_1 and GCPSO-ELM-CS_1, respectively. All methods are based on the idea of population stereotyping (Kennedy, 2000), so each individual will update its velocity according to its cluster (neighborhood), and each cluster will explore a different area of the problem search space, looking for a specific local best position.

The chosen clustering method was the Hard K-Means algorithm (MacQueen, 1967). The clustering method was executed only for a few number of iterations (*maxItClust*) to reduce the computational cost added by its execution, so the final cluster were not necessarily the best ones, and it was executed only in iterations where at least one of the local best positions associated to each particle was changed (Kennedy, 2000). The Hard K-Means pseudocode is presented in Figure 4.

The proposed strategies also use a selection operator based on the work of Angeline (1999). Angeline presented a form of tournament selection based on the particles' current fitness, so that the properties that make some solutions superior are transferred directly to less effective individuals of the swarm. The current positions and velocities of the better half of the swarm are copied onto the worse half. This way, only the local best solutions found so far by the worse half of the population are not changed. The tournament selection operator performs as described in Figure 5.

The modifications adopted to this work are the following. There is no score system, and the particles are sorted according to their current fitness value only. The best half of the swarm replaces the worst one in such a way that the best particle uses its current position and velocity to replace the current position and velocity of the worst particle; the second best particle

Figure 4. Pseudocode for the Hard K-Means algorithm

Initialization: Pick randomly *C* particles as the initial cluster centers $\mathbf{g}_c$;
$t = 0$;
While $t <$ *maxItClust* **do**
  $t = t + 1$;
  For each particle $\mathbf{x}_i$ of the swarm **do**
    Calculate the distance between $\mathbf{x}_i$ and each cluster center $\mathbf{g}_c$;
    Assign $\mathbf{x}_i$ to the nearest cluster center $\mathbf{g}_c$;
  End_For
  Update each cluster center $\mathbf{g}_c$ as the mean point of its cluster;
End_While

Figure 5. Pseudocode for Angeline's tournament selection operator

Calculate the fitness function for all particles of the swarm, according to their current positions;
For each particle $\mathbf{x}_i$ of the swarm **do**
  For *k* different particles $\mathbf{x}_j$ of the swarm
  $(i \neq j)$ **do**
    If $f(\mathbf{x}_i) < f(\mathbf{x}_j)$ **then**
      $score_{\mathbf{x}_i} = score_{\mathbf{x}_i} + 1$;
    End_If
  End_For
End_For
Sort all particles according to their scores, having the highest scores appearing at the head of the population;
Replace the current position and velocity for each particle of the worse half of the swarm, using the positions and velocities of the best half of the swarm.

uses its current position and velocity to replace the current position and velocity of the second worst particle; and so on. The pseudocode for the current selection scheme is presented in Figure 6.

For all proposed algorithms, the initial swarm is randomly generated. Each particle $\mathbf{x}_i$ in the swarm is composed of a set of input weights and hidden biases:

$$\mathbf{x}_i = [w_{11}, w_{12}, \ldots, w_{N1}, w_{21}, w_{22}, \ldots, w_{1k}, w_{2k}, \ldots, w_{Nk}, \ldots, b_1, b_2, \ldots, b_N]$$

All w_{ij} and b_j are randomly initialized within the range of [-1, 1]. For each particle, the corresponding output weights matrix is computed using MP generalized inverse. The fitness function adopted is the root mean squared error (RMSE) on the validation set (Equation (12))(Zhu *et al*., 2005):

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{M}\sum_{j=1}^{N}\beta_j f(\mathbf{w}_j \cdot \mathbf{x}_i + b_j) - \mathbf{t}_{i2}^{\,2}}{d \times M}} \quad (12)$$

In PSO-ELM-CS$_1$ and GCPSO-ELM-CS$_1$ algorithms, the local best position found so far $\mathbf{y}_i$ is replaced with the cluster center $\mathbf{g}_c$ for

Figure 6. Pseudocode for the adopted selection operator

Calculate the fitness function for all particles of the swarm, according to their current positions;
Sort all particles according to their current fitness value, having the highest fitness appearing at the head of the population;
For i = 1to $\lfloor s/2 \rfloor$ **do**
$\mathbf{x}_{s-i+1} = \mathbf{x}_i$;
$\mathbf{v}_{s-i+1} = \mathbf{v}_i$
End_For

the new velocity determination, where $\mathbf{x}_i \in c$, like in Equation (8) (Kennedy, 2000).

For the new PSO-ELM-CS$_2$ and GCPSO-ELM-CS$_2$ strategies, the local best position found so far $\mathbf{y}_i$ is replaced with the local best position found so far $\mathbf{y}_{bstN}$ by a member of the cluster c ($\mathbf{x}_i \in c$) for the new velocity determination of the ith particle, according to Equation (13). These approaches also used a modified scheme, adapted from (Zhu *et al*., 2005), to select the local best position $\mathbf{y}_i$ found so far by a particle $\mathbf{x}_i$ (Equation (14)), replacing the PSO traditional approach, represented by Equation (4). This scheme uses the output weight norm β to improve the generalization performance of the ELM represented by each particle of the swarm (Bartlett, 1998).

$$v_{ij}(t+1) = wv_{ij}(t) + c_1 r_1 \left(y_{bstNj}(t) - x_{ij}(t)\right) + c_2 r_2 \left(\hat{y}_j(t) - x_{ij}(t)\right) \quad (13)$$

$$\mathbf{y}_i(t+1) = \begin{cases} \mathbf{x}_i, if\left(f\left(\mathbf{y}_i(t)\right) - f\left(\mathbf{x}_i(t+1)\right)\right) \\ \quad > \mu f(\mathbf{y}_i(t)) \\ \mathbf{x}_i, if\left(f\left(\mathbf{y}_i(t)\right) - f\left(\mathbf{x}_i(t+1)\right)\right) \\ \quad < \mu f\left(\mathbf{y}_i(t)\right) \\ \quad and\, \beta_{\mathbf{x}_i(t+1)} < \beta_{\mathbf{y}_i(t)} \\ \mathbf{y}_i,\ otherwise \end{cases} \quad (14)$$

where $\beta_{\mathbf{x}_i(t+1)}$ and $\beta_{\mathbf{y}_i(t)}$ are the output weight matrixes related to the new position of the ith particle $\mathbf{x}_i(t+1)$ and the old local best position $\mathbf{y}_i(t)$ found so far, respectively, and μ is a tolerance rate (Zhu *et al*., 2005).

The Guaranteed Convergence approach for the PSO algorithm is applied to the GCPSO-ELM-CS$_2$ method, just like in GCPSO-ELM-CS$_1$ (Pacifico & Ludermir, 2012). The pseudocodes for the PSO-ELM-CS$_1$, GCPSO-ELM-CS$_1$ and for the proposed approaches are presented in Figures 7 to Figure 10.

4. EXPERIMENTAL RESULTS

In this section, the experimental results are presented. The proposed methods (PSO-ELM-CS$_2$ and GCPSO-ELM-CS$_2$) are compared with traditional ELM, the Levenberg-Marquardt Backpropagation (LM-BP), E-ELM (Zhu *et al*., 2005), PSO-ELM (Xu & Shu, 2006), GCPSO-ELM (Silva *et al*., 2011b) and their predecessors PSO-ELM-CS$_1$ and GCPSO-ELM-CS$_1$ (Pacifico & Ludermir, 2012). All programs run in a MATLAB 6.0 environment, and the LM-BP algorithm is provided in the neural networks toolbox of MATLAB. A validation set is used in all evaluated methodologies to prevent *overfitting*.

For evaluating all of these algorithms six benchmark classification datasets (Diabetes, E. coli, Glass, Heart, Iris and Wine), obtained from UCI Machine Learning Repository (Frank &

Figure 7. Pseudocode for the PSO-ELM-CS$_1$ algorithm

Define: The number of clusters *C* and the maximum number of iterations for the clustering algorithm *maxItClust*.
Initialization: Randomly initialize all particles of the swarm;
Clustering step: execute the clustering strategy, following the steps described in Figure 4;
While the termination conditions are not met **do**
 For each particle *i* of the swarm **do**
  Calculate its fitness function $f(\mathbf{x}_i)$ as the RMSE obtained by the ELM on the validation set using Equation (11);
  If $f(\mathbf{x}_i) < f(\mathbf{y}_i)$ **then**
   $\mathbf{y}_i = \mathbf{x}_i$;
  End_If
 End_For
 $\hat{\mathbf{y}} = \arg\min_{\mathbf{y}_i, 1 \le i \le s} f(\mathbf{y}_i)$;
 Update velocity and position of each particle, according to Equation (8) and Equation (3), respectively;
 Selection step: execute the selection operator, according to Figure 6;
 If any local best position $\mathbf{y}_i$ has changed
 then
  Execute the clustering algorithm (Figure 4);
 End_If
End_While

Figure 8. Pseudocode for the GCPSO-ELM-CS$_1$ algorithm

Define: The number of clusters *C* and the maximum number of iterations for the clustering algorithm *maxItClust*.
Initialization: Randomly initialize all particles of the swarm; initialize ρ, s_c and f_c *numSucesses* = 0, *numFailures* = 0;
Clustering step: execute the clustering strategy, following the steps described in Figure 4;
While the termination conditions are not met **do**
 For each particle *i* of the swarm **do**
  Calculate its fitness function $f(\mathbf{x}_i)$ as the RMSE obtained by the ELM on the validation set using Equation 11;
  If $f(\mathbf{x}_i) < f(\mathbf{y}_i)$ **then**
   $\mathbf{y}_i = \mathbf{x}_i$;
  End_If
 End_For
 $\hat{\mathbf{y}} = \arg\min_{\mathbf{y}_i, 1 \le i \le s} f(\mathbf{y}_i)$;
 Update velocity and position of each particle, according to Equation (8) and Equation (3), respectively; for the current global best particle, the new velocity is given by Equation 6;
 Update the ρ according to Equation 7; update s_c, f_c; *numSucesses* and *numFailures*, when needed;
 Selection step: execute the selection operator, according to Figure 6;
 If any local best position $\mathbf{y}_i$ has changed
 then
  Execute the clustering algorithm (Figure 4);
 End_If
End_While

Asuncion, 2012) are used. These datasets present different degrees of difficulties and different number of classes. The evaluation metrics used are an empirical analysis over the average test accuracies and training times and a paired hypothesis test of type *t*-test (DeGroot, 1989), considering a 95% degree of confidence, over the test accuracies obtained by each method in each dataset.

In our experiments, all inputs (attributes) have been normalized into the range [0, 1], while the outputs (targets) have been normalize into [-1, 1]. The input weights and the biases have been obtained into the range [-1, 1]. The ELM activation function used was the sigmoid function $g(x) = 1 / (1 + exp(-x))$.

The tests were divided in two steps. In first step an evaluation is made aiming to select the

Figure 9. Pseudocode for the PSO-ELM-CS$_2$ algorithm

Define: The number of clusters *C* and the maximum number of iterations for the clustering algorithm *maxItClust*.
Initialization: Randomly initialize all particles of the swarm;
Clustering step: execute the clustering strategy, following the steps described in Figure 4;
While the termination conditions are not met **do**
 For each particle *i* of the swarm **do**
  Calculate its fitness function $f(\mathbf{x}_i)$ as the RMSE obtained by the ELM on the validation set using Equation 11;
  Update the local best position found so far $\mathbf{y}_i$ according to Equation (13);
 End_For
 $\hat{\mathbf{y}} = \arg\min_{\mathbf{y}_i, 1 \le i \le s} f(\mathbf{y}_i)$;
 Update velocity and position of each particle, according to Equation (12) and Equation (3), respectively;
 Selection step: execute the selection operator, according to Figure 6;
 If any local best position $\mathbf{y}_i$ has changed
 then
  Execute the clustering algorithm (Figure 4);
 End_If
End_While

Figure 10. Pseudocode for the GCPSO-ELM-CS$_2$ algorithm

Define: The number of clusters *C* and the maximum number of iterations for the clustering algorithm *maxItClust*.
Initialization: Randomly initialize all particles of the swarm; initialize ρ, s_c and f_c; *numSucesses* = 0, *numFailures* = 0;
Clustering step: execute the clustering strategy, following the steps described in Figure 4;
While the termination conditions are not met **do**
 For each particle *i* of the swarm **do**
  Calculate its fitness function $f(\mathbf{x}_i)$ as the RMSE obtained by the ELM on the validation set using Equation (11);
  Update the local best position found so far $\mathbf{y}_i$ according to Equation (13);
 End_For
 $\hat{\mathbf{y}} = \arg\min_{\mathbf{y}_i, 1 \le i \le s} f(\mathbf{y}_i)$;
 Update velocity and position of each particle, according to Equation (12) and Equation (3), respectively; for the current global best particle, the new velocity is given by Equation (6);
 Update the ρ according to Equation 7; update s_c, f_c; *numSucesses* and *numFailures*, when needed;
 Selection step: execute the selection operator, according to Figure 6;
 If any local best position $\mathbf{y}_i$ has changed
 then
  Execute the clustering algorithm (Figure 4);
 End_If
End_While

best number of hidden nodes (*N*) for the ELM and LM-BP algorithms, the maximum number of iterations (*maxIter*) and the population size (*s*) for the PSO based approaches and E-ELM method; after that, the final tests were executed (second step).

For LM-BP algorithm, the maximum number of epochs was set to 200. The remaining parameters for all tested algorithms were obtained from the literature (Carvalho & Ludermir, 2006a; Kennedy, 2000; Zhu *et al*., 2005). Table 1 shows the list of parameters for all tested algorithms.

Each dataset was divided in training, validation and testing sets, as specified in Table 2. For all algorithms, 50 independent executions were done with each dataset. Table 3 to Table 8 show the average training time and average test accuracy obtained for each method for each dataset. The training, validation and testing sets were randomly generated at each trial of simulations. The best results (according to the empirical analysis) are emphasized in bold.

Table 1. List of parameter for all tested methodologies

Method	Parameter	Value
LM-BP & ELM	N	15
LM-BP	*Number of Epochs*	200
E-ELM & PSO	s	50
	maxIter	100
	Crossover Rate	0.8
E-ELM	F	1.0
	μ	0.02
	c_1	2.0
PSO	c_2	2.0
	w	0.9 to 0.4
	ρ	1.0
GCPSO	s_c	5
	f_c	5
Hard K-Means	C	5
	maxItClust	5

Table 2. Dataset specifications

Classes	Attrib.	Train.	Validat.	Test.
Diabetes:				
2	8	252	258	258
E. coli:				
8	7	180	78	78
Glass:				
6	9	114	50	50
Heart:				
2	13	130	70	70
Iris:				
3	4	70	40	40
Wine:				
3	13	78	50	50

Table 3. Results for diabetes dataset

Method	Training Time (s)	Test Accuracy (%)
LM-BP	0.51358	71.44±5.81
ELM	0.00092	76.63±2.25
E-ELM	39.5844	75.07±2.21
PSO-ELM	44.1723	76.56±2.24
GCPSO-ELM	43.9440	76.83±2.34
PSO-ELM-CS_1	44.4792	76.74±2.47
GCPSO-ELM-CS_1	45.8464	**77.13±2.01**
PSO-ELM-CS_2	43.2369	77.08±2.28
GCPSO-ELM-CS_2	45.1401	76.83±1.87

4.1. Diabetes Dataset

This dataset consists of two possible diagnostics (classes) given to a sample of 768 females at least 21 years old of Pima Indian heritage, concerning about when a patient shows signs of diabetes according to World Health Organization criteria or not.

The classes (1 for a healthy patient, and 2 for a patient interpreted as "tested positive for diabetes") have, respectively, 500 and 268 instances (Frank & Asuncion, 2012). Each class is described by eight real-valued attributes: number of times pregnant, plasma glucose concentration a 2 hours in an oral glucose tolerance test, diastolic blood pressure (mm Hg), triceps skin fold thickness (mm), 2-Hour serum insulin (mu U/ml), body mass index, diabetes pedigree function and age (in years).

In an empirical analysis (Table 3), all proposed methods (PSO-ELM-CS_1, GCPSO-ELM-CS_1, PSO-ELM-CS_2 and GCPSO-ELM-CS_2) obtained better results than traditional ELM, E-ELM, PSO-ELM and LM-BP, but the GCPSO-ELM-CS_2 performed similarly to GCPSO-ELM. The worst result was achieved by LM-BP algorithm, which showed a high degree of instability.

The hypothesis tests (paired *t*-tests with a 95% degree of confidence) showed that all ELM-based strategies achieved a better result than LM-BP algorithm, and the traditional ELM and all PSO-based methods achieved a better result than E-ELM.

4.2. E. Coli Dataset

This dataset gives characteristics of each ORF (potential gene) in the E. coli genome, providing their sequence, homology (similarity to other genes), structural information, and function (if known) (Frank & Asuncion, 2012).

The dataset is divided in eight classes, representing the localization site of each gene: cytoplasm (1), inner membrane without signal sequence (2), perisplasm (3), inner membrane, uncleavable signal sequence (4), outer membrane (5), outer membrane lipoprotein (6), inner membrane lipoprotein (7) and inner membrane, cleavable signal sequence (8).

The classes (1, 2, 3, 4, 5, 6, 7 and 8) have, respectively, 143, 77, 52, 35, 20, 5, 5 and 2 instances. Each instance is described by seven real-valued attributes: mcg (McGeoch's method for signal sequence recognition), gvh (von Heijne's method for signal sequence recognition), lip (von Heijne's Signal Peptidase II consensus sequence score), chg (Presence of charge on N-terminus of predicted lipoproteins), aac (score of discriminant analysis of the amino acid content of outer membrane and periplasmic proteins), alm1(score of the ALOM membrane spanning

region prediction program) and alm2 (score of ALOM program after excluding putative cleavable signal regions from the sequence).

In an empirical analysis (Table 4), all PSO-based methods performed better than traditional ELM, LM-BP and E-ELM, and the PSO-ELM-CS_1, GCPSO-ELM-CS_1 and GCPSO-ELM-CS_2 algorithms achieved the better results for this dataset.

The paired *t*-tests (95% degree of confidence showed that all ELM-based approaches performed better than LM-BP algorithm. All PSO-bases approaches and the traditional ELM performed better than E-ELM method, according to the hypothesis tests.

4.3. Glass Dataset

This dataset consists in a study of the classification of types of glass motivated by criminological investigation. This dataset consists of seven types (classes) of glasses: building windows float processed (1), building windows non-float processed (2), vehicle windows float processed (3), vehicle windows non-float processed (4, none in this database), containers (5), tableware (6) and headlamps (7).

The classes (1, 2, 3, 4, 5, 6 and 7) have, respectively, 70, 17, 76, 0, 13, 9 and 29 instances. Each instance is described by nine real-valued attributes: RI (refractive index), Na (Sodium), Mg (Magnesium), Al (Aluminum), Si (Silicon), K (Potassium), Ca (Calcium), Ba (Barium) and Fe (Iron) (Frank & Asuncion, 2012).

In an empirical analysis (Table 5), the LM-BP learning algorithm achieved the worst result and the highest degree of instability than all tested methods. All proposed methods (PSO-ELM-CS_1, GCPSO-ELM-CS_1, PSO-ELM-CS_2, PSO-ELM-CS_1 and GCPSO-ELM-CS_2) performed better than traditional ELM, E-ELM, PSO-ELM and GCPSO-ELM according to the empirical analysis.

The hypothesis test showed that all ELM-based approaches achieved better results than LM-BP and only the GCPSO-ELM-CS_1 performed better than E-ELM algorithm.

4.4. Heart Dataset

This dataset consists of two possible diagnoses (classes) concerning the absence (class 1) or presence (class 2) of heart disease in a group of 270 individuals.

The classes (1 and 2) contain 150 and 120 individuals, respectively. Each individual is described by thirteen real-valued variables: age, sex, chest pain type, resting blood pressure, serum cholestoral (in mg/dl), fasting blood sugar (>120mg/dl), resting electrocardiographic

Table 4. Results for e. coli dataset

Method	Training Time (s)	Test Accuracy (%)
LM-BP	3.41676	50.23±22.3
ELM	0.00094	85.54±3.76
E-ELM	24.7448	81.38±4.56
PSO-ELM	26.6807	85.82±3.96
GCPSO-ELM	26.4952	85.95± 3.41
PSO-ELM-CS_1	26.3323	**86.69±4.04**
GCPSO-ELM-CS_1	26.4923	86.13±3.72
PSO-ELM-CS_2	27.5780	85.92±3.11
GCPSO-ELM-CS_2	26.7100	86.33±3.93

Table 5. Results for glass dataset

Method	Training Time (s)	Test Accuracy (%)
LM-BP	0.00716	41.08±18.63
ELM	0.00000	61.20±6.89
E-ELM	18.8546	61.84±6.66
PSO-ELM	18.9793	62.88±6.63
GCPSO-ELM	17.2143	62.40±5.95
PSO-ELM-CS_1	17.7372	63.48 ± 5.94
GCPSO-ELM-CS_1	18.0371	**63.92 ± 5.38**
PSO-ELM-CS_2	17.6573	63.44 ± 5.61
GCPSO-ELM-CS_2	18.2136	63.28 ± 6.18

results, maximum heart rate achieved, exercise induced angina, oldpeak (ST depression induced by exercise relative to rest), the slope of the peak exercise ST segment, number of major vessels colored by flourosopy and thal value (Frank & Asuncion, 2012).

In an empirical analysis (Table 6), the proposed PSO-ELM-CS_2 achieved the better results, followed by the PSO-ELM-GS_1, GCPSO-ELM-CS_1 and GCPSO-ELM methods. The worse results were achieved by the LM-BP and E-ELM algorithms.

The paired hypothesis tests of type *t*-test (95% of confidence) were executed over the results, and showed that all methods achieved better results than LM-BP and E-ELM methods.

4.5. Iris Dataset

This dataset consists of three types (classes) of iris plants: iris setosa, iris versicolour and iris virginica. The three classes each have 50 instances. One class is linearly separable from the other two; the latter two are not linearly separable from each other. Each class is described by four real valued attributes: sepal length, sepal width, petal length and petal width (Frank & Asuncion, 2012).

Table 6. Results for heart dataset

Method	Training Time (s)	Test Accuracy (%)
LM-BP	0.28960	68.69±14.50
ELM	0.00064	81.46±4.28
E-ELM	18.8052	79.09±4.06
PSO-ELM	19.0143	81.31±3.76
GCPSO-ELM	21.6880	82.57±3.65
PSO-ELM-CS_1	19.7496	82.57±2.89
GCPSO-ELM-CS_1	20.2834	82.57±4.36
PSO-ELM-CS_2	22.3084	**82.63±3.61**
GCPSO-ELM-CS_2	20.5506	82.49±4.18

Table 7. Results for iris dataset

Method	Training Time (s)	Test Accuracy (%)
LM-BP	0.16442	74.55±24.25
ELM	0.00094	95.45±3.30
E-ELM	11.0290	95.60±3.22
PSO-ELM	10.9440	96.10±3.24
GCPSO-ELM	10.9846	96.25±3.20
PSO-ELM-CS_1	12.2531	96.80±3.03
GCPSO-ELM-CS_1	11.2857	96.45±2.82
PSO-ELM-CS_2	11.6463	96.55±3.06
GCPSO-ELM-CS_2	11.5195	**97.00±2.42**

Table 8. Results for wine dataset

Method	Training Time (s)	Test Accuracy (%)
LM-BP	0.520320	79.96±23.29
ELM	0.00032	96.56±3.16
E-ELM	11.7101	91.60±4.66
PSO-ELM	11.7262	97.24±2.02
GCPSO-ELM	12.1815	97.32±2.67
PSO-ELM-CS_1	12.2304	97.08±2.29
GCPSO-ELM-CS_1	12.8079	97.00±2.60
PSO-ELM-CS_2	11.9664	97.32±2.08
GCPSO-ELM-CS_2	12.8676	**97.84±1.89**

In an empirical analysis (Table 7), all PSO-based methods outperformed the traditional ELM, E-ELM and LM-BP methods. The proposed methods (PSO-ELM-CS_1, GCPSO-ELM-CS_1, PSO-ELM-CS_2 and GCPSO-ELM-CS_2) outperformed all others PSO-based approaches. The *t*-tests pointed that all ELM-based methods were better than LM-BP approach (95% degree of confidence), and PSO-ELM-CS_1 and GCPSO-ELM-CS_2 outperformed the traditional ELM algorithm. The proposed GCPSO-ELM-CS_2 also outperformed the E-ELM method.

4.6. Wine Dataset

This dataset consists of three types (classes) of wines grown in the same region in Italy, but derived from three different cultivars.

The classes (1, 2 and 3) have, respectively, 59, 71 and 48 instances. Each wine is described by 13 real valued attributes representing the quantities of 13 components found in each of the three types of wines. These attributes are: (1) alcohol; (2) malic acid; (3) ash; (4) alkalinity of ash; (5) magnesium; (6) total phenols; (7) flavonoids; (8) non-flavonoid phenols; (9) proanthocyanins; (10) color intensity; (11) hue;

(12) OD280/OD315 of diluted wines and (13) proline (Frank & Asuncion, 2012).

In an empirical analysis (Table 8), the PSO-ELM-CS_2 and GCPSO-ELM-CS_2 achieved better results for this dataset. The PSO-ELM-CS_1 and GCPSO-ELM-CS_1 were worse than PSO-ELM and GCPSO-ELM. All PSO-based methods achieved better results than traditional ELM, E-ELM and LM-BP.

The paired *t*-tests showed that all ELM-based approaches achieved better results than LM-BP, which still presented a high degree of instability. All PSO-based approaches and the traditional ELM performed better than E-ELM. The GCPSO-ELM-CS_2 achieved better results than traditional ELM according to the paired *t*-test.

5. CONCLUSION AND FUTURE WORKS

In this paper, two new hybrid methods were proposed, based on Particle Swarm Optimization (PSO) strategy to select the input weights and hidden biases to ELM algorithm, named PSO-ELM-CS_2 and GCPSO-ELM-CS_2, as extensions from the PSO-ELM-CS_1 and GCPSO-ELM-CS_1, respectively, presented in a previous work. These approaches use the concept of population stereotyping, to group the particles of the swarm in different clusters, so that each cluster searches for a specific local best solution, exploring different regions of the problem search space.

The performance of the tested methods was evaluated with well known benchmark classification datasets (Diabetes, E. coli, Glass, Heart, Iris and Wine), obtained from UCI Machine Learning Repository.

Experimental results show that the new hybrid PSO-ELM-CS_2 and GCPSO-ELM-CS_2 approaches obtained better generalization performance than Levenberg-Marquardt Backpropagation (LM-BP), traditional ELM, E-ELM and PSO-ELM for all datasets, and the former PSO-ELM-CS_1 and GCPSO-ELM-CS_1 outperformed LM-BP, ELM, E-ELM and PSO-ELM for five of the six tested datasets, in an empirical analysis over the average test accuracies.

The paired *t*-test hypothesis test was executed for all datasets. For all datasets, the LM-BP was outperformed by all ELM-based approaches, and for four of the six datasets the E-ELM was outperformed by all PSO-based approaches and traditional ELM. GCPSO-ELM-CS_2 achieved superior results than traditional ELM for two datasets (Iris and Wine) according to this test. All proposed approaches achieved equivalent results to ELM for Diabetes, E. Coli, Glass and Heart datasets.

The computational costs demanded by the new strategies were relevant when compared with the execution time achieved by traditional ELM algorithm, because of their evolutionary nature (i.e., they need a considerable number of evaluations throughout the method execution), but the execution time achieved by these methods is according to evolutionary approaches used to optimize ELM found in recent literature, such as the E-ELM, PSO-ELM and GCPSO-ELM. As can be observed from Table 3 to Table 8, the clustering and selection steps did not influence the computational costs considerably.

Although the local best approaches presented in this work showed better results than ELM only for two cases according to *t*-test hypothesis test, the empirical analysis showed that these approaches are slightly better than traditional ELM and PSO-ELM for most of the cases, fact that encourages the study and exploration of local PSO methods for ELM optimization.

As future works, a deeper investigation will be done to evaluate the influence of population clusters in performance of the hybrid PSO-ELM-based approaches, using a broader number of problems, so the effective power of local approaches could be measured. Also, different selection operators and topologies for the PSO population will be tested.

ACKNOWLEDGMENT

The authors would like to thank FACEPE, CNPq and CAPES (Brazilian Research Agencies) for their financial support.

REFERENCES

Angeline, P. J. (1999). Using selection to improve particle swarm optimization. In *Proceeding of the 1999 IEEE International Joint Conference on Neural Networks (IJCNN'99)*, Washington, DC (pp. 84-89). Los Alamitos, CA: IEEE Computer Society.

Bartlett, P. L. (1998). The sample complexity of pattern classification with neural networks: The size of the weights is more important than the size of the network. *IEEE Transactions on Information Theory*, *44*(2), 525–536. doi:10.1109/18.661502.

Carvalho, M., & Ludemir, T. B. (2006a). An analysis of PSO hybrid algorithms for feed-forward neural networks training. In *Proceedings of the Ninth Brazilian Symposium on Neural Networks (SBRN'06),* Ribeirão Preto, Brazil (pp. 6-11). Los Alamitos: IEEE Computer Society.

Carvalho, M., & Ludemir, T. B. (2006b). Particle swarm optimization of feed-forward neural networks with weight decay. In *Proceedings of the Sixth International Conference on Hybrid Intelligent Systems (HIS'06)*, Auckland, New Zealand. Los Alamitos, CA: IEEE Computer Society.

Cho, J.-H., & Lee, D.-J. (2007). *Parameter optimization of extreme learning machine using bacterial foraging algorithm. Korea Electrical Engineering and Science Research Institute* (pp. 742–747). South Korea: EESRI.

Clerc, M. (1999). The swarm and the Queen: Towards a deterministic and adaptive particle swarm optimization. In *Proceedings of the Congress on Evolutionary Computation*, Washington, DC (pp. 1951-1957). Piscataway, NJ: IEEE Service Center.

Clerc, M., & Kennedy, J. (2002). The particle swarm: Explosion, stability and convergence in a multi-dimensional complex space. *IEEE Transactions on Evolutionary Computation*, *6*(1), 58–73. doi:10.1109/4235.985692.

Corne, D. W., Dorigo, M., & Glover, F. (1999). *New ideas in optimization*. New York, NY: McGraw-Hill.

M. H. DeGroot (Ed.). (1989). *Probability and statistics*. Boston, MA: Addison Wesley Publishing Company.

Eiben, E., & Smith, J. E. (2003). *Introduction to evolutionary computing*. Berlin, Germany: Springer-Verlag. doi:10.1007/978-3-662-05094-1.

Frank, A., & Asuncion, A. (2013). *UCI machine learning repository*. University of California, School of Information and Computer Science, Irvine, CA. Retrieved March 15, 2013, from http://archive.ics.uci.edu/ml

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operation Research - Special Issue. Applications of Integer Programming*, *13*(5), 533–549.

S. Haykin (Ed.). (1998). *Neural networks: A comprehensive foundation*. Englewood Cliffs, NJ: Prentice-Hall, Inc..

He, S., Wu, H., & Saunders, J. R. (2006). A novel group search optimizer inspired by animal behavioural ecology. In *Proceedings of the 2006 IEEE Congress on Evolutionary Computation (CEC'2006)*, Vancouver, Canada (pp. 1272-1278). Los Alamitos, CA: IEEE Computer Society.

He, S., Wu, H., & Saunders, J. R. (2009). Group search optimizer: An optimization algorithm inspired by animal searching behaviour. *IEEE Transactions on Evolutionary Computation*, *13*(5), 973–990. doi:10.1109/TEVC.2009.2011992.

Huang, G. B., Zhu, Q. Y., & Siew, C. K. (2006). Extreme learning machine: Theory and applications. *Neurocomputing*, *4*, 489–501. doi:10.1016/j.neucom.2005.12.126.

Kennedy, J. (2000). Stereotyping: Improving particle swarm performance with cluster analysis. In *Proceedings of the 2000 Congress on Evolutionary Computing,* Washington, DC (Vol. 2, pp. 1507-1512). Los Alamitos, CA: IEEE Computer Society.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of the 1999 IEEE International Conference on Neural Networks*, Perth, Australia (pp. 1942-1948). Los Alamitos, CA: IEEE Computer Society.

Kennedy, J., & Eberhart, R. (2001). *Swarm intelligence*. San Francisco, CA: Morgan Kaufmann Publishers, Inc..

Kennedy, J., & Mendes, R. (2002). Population structure and particle swarm performance. In *Proceedings of the 2002 World Congress on Computational Intelligence,* Honolulu, HI (pp. 1671-1676). Los Alamitos, CA: IEEE Computer Society.

Kirkpatrick, S., Gellat, C. D. Jr, & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*, 671–680. doi:10.1126/science.220.4598.671 PMID:17813860.

Lahoz, D., Lacruz, B., & Mateo, P. M. (2011). A bi-objective micro genetic extreme learning machine. In *Proceedings of the 2011 IEEE Workshop on Hybrid Intelligent Models and Applications (HIMA)* (pp. 68-75).

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematics, Statistics and Probability*, Berkeley, CA (Vol. 1, pp. 281-296). Berkeley, CA: University of California Press.

Pacifico, L. D. S., & Ludermir, T. B. (2012). Improving ELM neural networks through PSO with selection (in Portuguese). *Anais do IX Encontro Nacional de Inteligência Artificial (ENIA 2012)*, Curitiba, Brazil (pp. 1-11). Porto Alegre, Brazil: Sociedade Brasileira de Computação.

Saraswathi, S., Sundaram, S., Sundararajan, N., Zimmermann, M., & Nilsen-Hamilton, M. (2011). ICGA-PSO-ELM approach for accurate multiclass cancer classification resulting in reduced gene sets in which genes encoding secreted proteins are highly represented. *IEEE Transactions on Computational Biology and Bioinformatics*, *8*(2), 452–463. doi:10.1109/TCBB.2010.13 PMID:21233525.

Silva, D. N. G., Pacifico, L. D. S., & Ludermir, T. B. (2011a). An evolutionary extreme learning machine based on group search optimization. In *Proceedings of the 2011 Congress on Evolutionary Computing (CEC 2011)*, New Orleans, FL (pp. 2297-2304). Los Alamitos, CA: IEEE Computer Society.

Silva, D. N. G., Pacifico, L. D. S., & Ludermir, T. B. (2011b). Extreme learning machine based on cooperative PSO (in Portugues). In *Proceedings of the 10th Brazilian Congress on Computational Intelligence (CBIC2011)*, Fortaleza, Brazil.

Storn, R., & Price, K. (1995). *Differential evolution – A simple and efficient adaptive scheme for global optimization over continuos spaces* (Tech. Rep. TR-95-012). Berkerley, CA: International Computer Science Institute.

Storn, R., & Price, K. (1997). Differential evolution – A simple and efficient heuristic for global optimization over continuos spaces. *Journal of Global Optimization*, *11*, 341–359. doi:10.1023/A:1008202821328.

Suganthan, P. N. (1999). Particle swarm optimizer with neighborhood operator. In *Proceedings of the 1999 Congress on Evolutionary Computing (CEC'99)*, Washington, DC (pp. 1958-1962). Los Alamitos, CA: IEEE Computer Society.

van den Bergh, F. (2002). *An analysis of particle swarm optimizers*. Unpublished Doctoral dissertation, Faculty of Natural and Agricultural Sciences, University of Pretoria, Pretoria, South Africa.

van den Bergh, F., & Engelbrecht, A. P. (2004). A cooperative approach to particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, *8*(3), 225–239. doi:10.1109/TEVC.2004.826069.

Xu, Y., & Shu, Y. (2006). Evolutionary extreme learning machine - Based on particle swarm optimization. In J. Wang, Z. Yi, J. M. Zurada, B.-L. Lu, & H. Yin (Eds.), Lecture Notes in Computer Science: Vol. 3971. *Advances in neural networks - ISNN 2006* (pp. 644–652). Berlin, Germany: Springer-Verlag. doi:10.1007/11759966_95.

Zhu, Q. Y., Qin, A. K., Suganthan, P. N., & Huang, G. B. (2005). Evolutionary extreme learning machine. *Pattern Recognition*, *38*, 1759–1763. doi:10.1016/j.patcog.2005.03.028.

Luciano D. S. Pacifico received the bachelor degree in computer science (fuzzy partitioning dynamic clustering analysis) and M. Sc. degree in computer science (clustering analysis with self-organizing maps) from Federal University of Pernambuco, Pernambuco, Brazil, in 2009 and 2012, respectively. He is currently a PhD student (neural networks optimization with evolutionary strategies) from Federal University of Pernambuco, Pernambuco, Brazil. He maintains an active interest in the field of evolutionary computing, focusing on numerical and neural networks optimization. Further research interests include pattern recognition, clustering analysis, computer vision, interaction design, game design and gamification, with publications in some of those fields.

Teresa Ludermir received the PhD degree in Artificial Neural Networks in 1990 from Imperial College, University of London, UK. From 1991 to 1992, she was a lecturer at Kings College London. She joined the Center of Informatics at Federal University of Pernambuco, Brazil, in September 1992, where she is currently a Professor and head of the Computational Intelligence Group. She has published over a 200 articles in scientific journals and conferences, three books in Neural Networks and organized two of the Brazilian Symposium on Neural Networks. Her research interests include weightless Neural Networks, hybrid neural systems and applications of Neural Networks.