Evolutionary Optimization of Neural Networks for Estimating Mechanical Properties of Lightweight Aggregate Concretes

Jonata J. Andrade

Leonardo Goliatt

Michele R. C. Farage

Lucas N. Silva

Rogerio. G. N. Santos

leonardo.goliatt@ufjf.edu.br

michele.farage@ufjf.edu.br

jonata.jefferson@ice.ufjf.br

lucas.neves@engenharia.ufjf.br

rogerio.gns@gmail.com

Federal University of Juiz de Fora

Juiz de Fora, MG 36300-330, Brazil

Abstract. *In this paper, neural networks with parameters adjusted by a Particle Swarm Optimization algorithm is used to predict mechanical properties of lightweight aggregate concretes. Unlike the approaches found in the literature, the proposed procedure estimates simultaneously the compressive strength and elasticity modulus. These properties were modeled as a function of four variables: water/cement fraction, lightweight aggregate volume, cement quantity and lightweight aggregate density. A Particle Swarm Optimization algorithm performs the model selection and automatically tunes the number of neurons in the hidden layer and the activation function. The results are compared with a model selection based on exhaustive search on the parameter space. The proposed approach arises as an alternative and competitive tool for estimating simultaneously the mechanical properties of lightweight aggregate concretes.*

Keywords: *Neural Networks, lightweight aggregate concretes, Particle Swarm Optimization*

1 INTRODUCTION

Reinforced concrete has been the most used structural material worldwide. As a result of rapid economic growth and urbanization, cement demand and cement production have grown rapidly over the last decades (Gao et al., 2017; Shen et al., 2017; Igliski and Buczkowski, 2017). Moreover, when compared with other construction and building materials, reinforced concrete has some advantages: (1) the possibility to adapt to any kind of shape, allowing independence to the architectural conception. (2) solution to obtain monolithic and hyperstatic structures. (3) excellent durability and low maintenance and preservation costs. (4) endurance to thermal, atmospheric and mechanical wear. On the other hand, a disadvantage is its own self-weight. In that context, the use of lightweight aggregate concretes emerges as an alternative solution. The lightweight aggregate concrete has refractory properties of thermal insulation and its specific weight is, approximately, two-thirds of the self-weight of the normal aggregate concrete. At the United States (ACI, 1999), for instance, the structural lightweight concrete is defined as the material with compressive strength over 17 MPa at 28 days and specific mass not over 1850 kg/m^3. In Brazil, the NBR NM 35 Standards (ABNT, 1995) establish the following relation between the compressive strength and specific mass for structural lightweight concrete: a) Compressive strength of 28 Mpa at 28 days and specific mass not over 1840 Kg/m^3; b) Compressive strength over 21Mpa at 28 days and specific mass not over 1760 Kg/m^3; c) Compressive strength of 17 Mpa at 28 days and specific mass not over 1680 Kg/m^3.

The specific weight of the normal concretes varies between 2200 Kg/m^3 and 2600 Kg/m^3, and the structural lightweight concrete's between 1350 Kg/m^3 and 1850 Kg/m^3. Though it has disadvantages such as the greater cost, more care in placing, greater porosity and more drying shrinkage, the demand for lightweight aggregate concretes have been increasing (Nilson et al., 1986). The use of lightweight aggregates reduces the dead load of a concrete structure, which then makes savings in reinforcement, footing and other load bearing elements. As an example, the structure's final cost can be lower due to the more economic scaling of the foundations.

The modulus of elasticity has a great importance for the structural lightweight concrete given its influence over deformations of the parts subject to bending, over the distribution of internal forces and over the critical load in the case of parts subject to buckling. In normal concretes, the cement paste's modulus of elasticity is, usually, much lower than the aggregate's modulus. On the other hand, in the structural lightweight concrete, the values of the light aggregate's and cement paste's modulus of elasticity are quite close. Laboratory tests prove that, for the same compressive strength, the lightweight concrete's modulus of elasticity is considerably lower than that values obtained for a normal concrete. That difference is given for the lower value of the lightweight aggregate's modulus of elasticity in proportion with the normal aggregate. Therefore, the lightweight concrete build structure's deformations are higher than those build with normal concrete.

Properties such as compressive strength are not only determined by the water/cement ratio, but also by other materials used in the mixture. Lightweight aggregate concretes contains other ingredients such as the type and the quantity of lightweight aggregate, chemical additives, and the mortar's composition. The various ingredients, besides the nonlinear concrete's structures, can make difficult the calculation of the compressive strength, besides other properties. In that way, mechanistic models (analytical) of concrete properties may be not enough to meet the requirements of the lightweight concrete design. Several studies have shown that the compressive

strength is determined, regardless of the type of concrete, not only for the water to cement ratio but also for other components (Papadakis and Tsimas, 2002; Chandra and Berntsson, 2003; Kockal and Ozturan, 2011; Duan et al., 2013). Most of the mathematical models used to study the behavior of concrete mixes consist of mathematical rules and expressions that capture the relationship between components of concrete mixes. It is known that as the cement has a specific weight higher than the lightweight aggregate, for a given aggregate, the compressive strength increases with the increase of its specific mass (Bogas and Gomes, 2013). In that way, most of the standards and specifications show mathematical models to estimate the lightweight concrete compressive strength as a function of specific mass. In the recent years, computational intelligence and machine learning techniques have been used to approximate non-linear and complex relationships in material sciences.

Considering the research scenario of the lightweight aggregate concretes, some studies have explored different predictive techniques. Artificial neural networks were used to predict the compressive strength of lightweight concretes in Altun et al. (2008) and Alshihri et al. (2009). An artificial neural network approach was used to predict the compressive strength of lightweight and semi-lightweight concretes in Bingl et al. (2013). Multiple linear regression, clustering, decision trees and neural networks were evaluated to predict the modulus of elasticity in Garciaa et al. (2015).

The purpose of this paper is to assess the performance of Extreme Learning Machines and Multilayer Perceptron neural networks combined with a Particle Swarm Optimization algorithm to predict mechanical properties of lightweight aggregate concretes. This model can provides researchers with information regarding the mechanical behavior lightweight concrete without the need of performing destructive tests such as the usual compressive tests, reducing the number of laboratory tests and reworking.

2 MATERIALS AND METHODS

2.1 Experimental Dataset

The experimental study (Ke, 2008) comprises lightweight aggregate concrete 16 32 cm cylindrical specimens with two types of lightweight aggregate (shale and expanded clay), five levels of lightweight aggregate volume (0%, 12.5%, 25%, 37.5% and 45%) and three types of mortars (a, b, and c) as shown in Table 1. The elasticity modulus (E) and the compressive force (f_c) were measured for each sample (averaged over three measures). A total of of 75 samples were collected and used to assess the performance of the approach proposed in this paper.

2.2 Extreme Learning Machines

The Extreme Learning Machine Regressor (ELM) is feedforward single hidden layer artificial neural network with three levels of randomness (Huang, 2015), described as follows: (1) fully connected, hidden node parameters are randomly generated, (2) the connection can be randomly generated, not all input nodes are connected to a particular hidden node, and (3) a hidden node itself can be a subnetwork formed by several nodes resulting in learning local features.

Table 1: Mix proportions of the lightweight aggregate concretes.

Mortar	Cement (kg/m^3)	Water/cement fraction	Aggregate density (kg/m^3)	Aggregate Volume (%)
Variable	x_2	x_0	x_3	x_1
a	336.24	0.446	1055.46	0.0
	294.21		923.53	12.5
	252.18		791.59	25.0
	210.15		659.66	37.5
	184.93		580.50	45.0
b	283.98	0.350	1135.94	0.0
	248.49		993.95	12.5
	212.99		851.95	25.0
	177.49		709.96	37.5
	156.19		624.77	45.0
c	263.64	0.290	1074.38	0.0
	230.68		940.08	12.5
	197.73		805.79	25.0
	164.77		671.49	37.5
	145.00		590.91	45.0

The output function of ELM is

$$\hat{y}(\mathbf{x}) = \sum_{i=1}^{L} \beta_i G(\mathbf{w}_i, b_i, \mathbf{x}) = \sum_{i=1}^{L} \beta_i G\left(\sum_{k=1}^{D} w_{kj} x_k + b_i\right)$$

where $\hat{y}$ is the ELM prediction associated to the input vector $\mathbf{x}$, $\mathbf{w}_i$ is the weight vector of the i-th hidden layer, b_i are the biases of the neurons in the hidden layer, β_i are output weights, D is the number of input variables, $G(\cdot)$ is the nonlinear activation function and L is the the number of neurons in the hidden layer. The parameters $(\mathbf{w}, b)$ are randomly generated (normally distributed with zero mean and standard deviation equals to one), and weights β_i of the output layer are determined analytically. The activation functions $G(\mathbf{w}, b, \mathbf{x})$ with the hidden nodes weights $(\mathbf{w}, b)$ are presented in Table 2.

The output weight vector $[\beta_1, ..., \beta_L]$ can be determined by minimizing the approximation error (Huang et al., 2015)

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^L} \|\mathbf{H}\boldsymbol{\beta} - \mathbf{y}\|$$

Table 2: Activation functions used in ELM. Independent of the training data, the hidden node parameters $(\mathbf{w}, b)$ are randomly generated using a normal distribution $N(0,1)$ instead of being explicitly trained.

#	Name	Activation Function G
0	Identity	$G(\mathbf{w}, b, \mathbf{x}) = \mathbf{x}$
1	ReLU	$G(\mathbf{w}, b, \mathbf{x}) = \max(0, x_i; i = 1, \cdots, D)$
2	Multiquadric	$G(\mathbf{w}, b, \mathbf{x}) = \sqrt{\|\|\mathbf{w} - \mathbf{x}\|\|^2 + b^2}$
3	Sigmoid	$G(\mathbf{w}, b, \mathbf{x}) = \dfrac{1}{1 + exp(-\mathbf{w} \cdot \mathbf{x} + b)}$
4	Gaussian	$G(\mathbf{w}, b, \mathbf{x}) = exp(-(\mathbf{w} \cdot \mathbf{x} + b)^2)$
5	Inverse Multiquadric	$G(\mathbf{w}, b, \mathbf{x}) = \dfrac{1}{(\|\|\mathbf{w} - \mathbf{x}\|\|^2 + b^2)^{1/2}}$
6	Hyperbolic Tangent	$G(\mathbf{w}, b, \mathbf{x}) = \dfrac{1 - exp(\mathbf{w} \cdot \mathbf{x} + b)}{1 + exp(\mathbf{w} \cdot \mathbf{x} + b)}$

where $\mathbf{y}$ is the output data vector, $\mathbf{H}$ is the hidden layer output matrix

$$\mathbf{H} = \begin{bmatrix} G_1(\mathbf{w}_1, b_1, \mathbf{x}_1) & \cdots & G_L(\mathbf{w}_L, b_L, \mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ G_1(\mathbf{w}_1, b_1, \mathbf{x}_N) & \cdots & G_L(\mathbf{w}_L, b_L, \mathbf{x}_N) \end{bmatrix} \text{ and } \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}$$

is the output data vector with N the number of data points. The optimal solution is given by

$$\boldsymbol{\beta} = (\mathbf{H^T H})^{-1} \mathbf{H^T y} = \mathbf{H}^\dagger \mathbf{y}$$

where $\mathbf{H}^\dagger$ is the pseudoinverse of $\mathbf{H}$.

Figure 1 shows an example of a 4-8-2 ELM with four inputs, one hidden layer (8 neurons in each) and two outputs. The input are water/cement fraction (x_0), lightweight aggregate volume (x_1), cement quantity (x_2) and the lightweight aggregate density (x_3). The outputs are the Young Modulus (E) and the compressive strength (f_c).

2.3 Multi-layer Perceptron Neural Network

The Multi-Layer Perceptron (MLP) (Haykin, 2008; Nissen, 2005) have been applied in various areas, performing pattern recognition functions, control, and signal processing. This architecture has one or more hidden layers, which are composed of computational neurons, also called as hidden neurons. The activity of hidden neurons is involved between the external and output layers of the network. Including one or more hidden layers, the network becomes capable of capturing non-linear relationships between inputs and outputs. This algorithm uses the number of hidden layers and neurons, the training algorithm, the connectivity and the normalization flag as parameters. If the renormalization flag is set to true, then the data are renormalized. The number of hidden layers is represented as a list of values. For instance, the configuration [5,5] indicates 2 hidden layers of 5 neurons each. The truncated Newton algorithm is used to to optimize the weights. The algorithm addresses the following methods to optimize the weights:

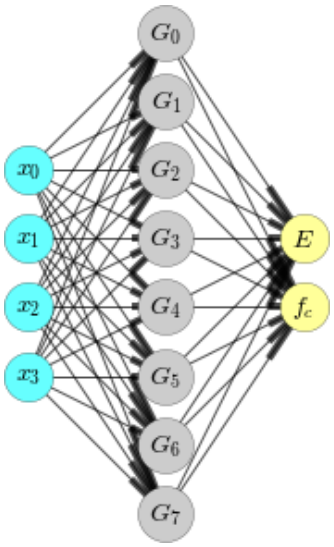

Figure 1: Connectivities for a 4-8-2 extreme learning machine which have four inputs, one hidden layer (8 neurons in each) and two outputs.

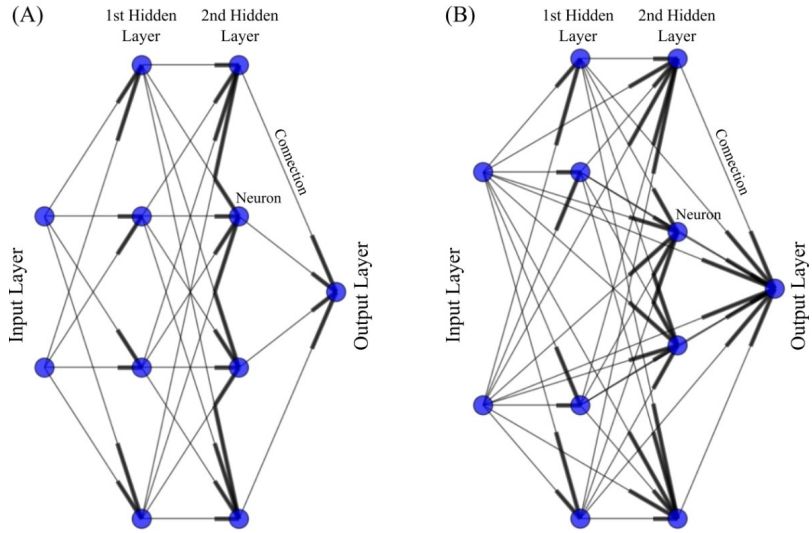

Figure 2: Connectivities for a 2-[4-4]-1 neural network which have two inputs, two hidden layers (4 neurons in each) and one output. The scheme of the simply connected scheme is shown in (A), where the neurons are connected only to the neurons of the previous layer, and in (B) there is the fully connected scheme, where a neuron is connected to its all predecessors.

l-bfgs refers to the Quasi-Newton approach, sgd refers to the Stochastic Gradient The connectivity can be simply feedforward connected or fully connected. Figure 2 exemplifies the types of connectivity.

2.4 Evolutionary settings of parameters

Setting the parameters of an estimator is usually a difficult task. Often, these parameters are defined empirically, by testing different settings by hand. In this paper, we employ an evolutionary algorithm to find the best set of parameters. The approach uses a simple Particle Swarm Optimization where each particle is a representation of Extreme Learning Machine parameters.

Particle swarm optimization (PSO) (Kennedy, 2011) is a stochastic population-based search method inspired by the social behavior of animals such as birds and fish. In PSO, each individual, called a particle, flies through the problem space and adjusts its position according to its own experience and the experience of its neighbors. A particle can fly either fast and far from the best positions to explore unknown areas (global search), or very slowly and close to a particular position (fine tune) to find better results. Each particle has a virtual position that represents a possible solution to some minimization problem. PSO is quite simple to implement and has few control parameters. Equations (1) and (2) are the two fundamental update rules of standard PSO.

$$v_i(t+1) = \omega v_i(t) + c_1 r_1(p_i(t) - x_i(t)) + c_2 r_2(p_g(t) - x_i(t) \tag{1}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{2}$$

where v_i and x_i are velocity and position vectors of the particle i, respectively, p_i is the best local position found by the particle i, and p_g is the best global position found in the whole population. The two parameters c_1 and c_2 are positive constants, called learning factors; c_1 presents how much a particle is attracted to its best position, and c_2 is the same for the global position. Values of these two parameters may vary depending on the nature of the problem but they are usually considered to be equal to 2.0; ω is the inertia weight; r_1 and r_2 are uniform random variables providing the stochastic aspect of the algorithm.

In this paper, each particle encodes a network candidate solution. For ELM, a particle's position represents the number of neurons in the hidden layers and the activation function, as shown in Tables 3. For MLP, each particle encodes the topology, the number of neurons and the number of neurons in each layer, as described in Table 4. For example, independent of the topology, the largest MLP network is [30, 30, 30, 30, 30] representing a 5-layer network with 30 neurons in each one. Considering the PSO approach, the goal is to find a position/weights so that the network generates computed outputs that match the outputs of the training data.

Table 3: Encoding of ELM candidate solutions.

Position	description	Range
0	Number of neurons in the hidden layer (see Fig. 1)	1–100
1	Coding representing the activation function according to Table 2	0: Identity, 1: ReLu; 2: Multiquadric, 3: Sigmoid, 4:Gaussian, 5: Inverse Multiquadric, 6: Hyperbolic Tangent

2.5 Cross-validation

Cross-validation is a sampling statistical technique to evaluate the ability of generalization of a model from a dataset. Among the cross-validation techniques, k-fold (Hastie et al., 2009) is one of the most used. k-fold uses a part of the data available to fit the model, and another different part to test it. The dataset is randomly divided into $k > 1$ subsets; from the k subsets,

Table 4: Encoding of MLP candidate solutions.

Position	description	Range
0	Number of layers (see Fig. 2)	1–5
1	Coding representing the activation function according to Fig. 2	0: feedforward connected, 1: fully connected;
2	Number of neurons in each hidden layer (see Fig. 2)	1–30

$k-1$ are used for training and the remaining set is used for testing. This process is repeated k times, using a different test set in each iteration. Figure 3 shows an example of 5-fold cross validation scheme.

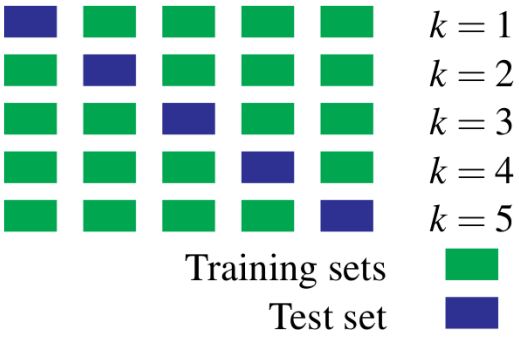

Figure 3: k-fold cross-validation method diagram ($k = 5$).

2.6 Performance Metrics

In this paper we have used the following metrics: R^2 score, Root Mean Squared Error ($RMSE$) and the Mean Absolute Percentage Error ($MAPE$). These criteria can be written as

$$R^2 = 1 - \frac{\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{N-1}(y_i - \bar{y})^2} \tag{3}$$

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2} \tag{4}$$

$$\text{MAPE} = 100 \times \frac{1}{N}\sum_{i=0}^{N-1}\frac{|y_i - \hat{y}_i|}{|y_i|} \tag{5}$$

where $\hat{y}_i$ is the estimated target output, y_i is the corresponding target output, N is the number of samples, p is the number of model parameters, and $\bar{y}$ is the mean of the vector $[y_1, ..., y_N]^T$.

3 COMPUTATIONAL EXPERIMENTS

In this section, we present the results obtained for ELM+PSO and MLP+PSO, both described in Section 2. We ran each computational experiment 25 times using 5-fold cross-validation with shuffled data generated by different random seeds. In the model selection step, PSO was set with the following parameters: 20 individuals in the population evolving under 35 generations; parameters c_1, c_2 were set to 2, respectively; $\omega = 0.6$ for all generations; the objective function (to be minimized) is the Root Mean Squared Error (RMSE) given in Eq. (4). The range of parameters are shown in Tables 3 and 4.

The computational experiments were conducted based in scikit-learn framework (Pedregosa et al., 2011) and implementations adapted from Friedman (1991), Wojciechowski (2012) and pys (2017). All codes and data are made available by the authors upon request. Computer specifications are given as follows: CPU AMD Opteron Processor 6272 (64 cores of 2.1GHz and cache memory of 2MB), RAM of 250GB and operational system Linux Ubuntu 14.04.4 LTS.

Table 5 shows the results produced by ELM. Each one of the 25 independent runs were executed with different random seeds. The first column of Table 5 shows the run number while the activation function G and number of hidden neurons (HL) are displayed in the second and third columns, respectively. The fourth, fifth and sixth columns present the performance metrics for Young Modulus (E) and the last three columns show the performance metrics for compressive strength (f_c). Observing the parameter settings for ELM in Table 5 and considering the evolutionary model selection procedure, we can observe that activation functions Identity and ReLu did not produce accurate models for Young Modulus and compressive strength.

Table 6 shows the results produced MLP. The second column shows the topology while the third column shows the arrangement of the hidden layers. The fourth, fifth and sixth columns present the performance metrics for Young Modulus (E) and the last three columns show the performance metrics for compressive strength (f_c). Although up to five layers were allowed in the model selection, in almost all runs the networks converged to a single layer with 5-121 neurons in each layer; only one run produced a two-layer MLP network ([12,1] topology), with 12 neurons in the first hidden layer and 1 neuron in the second hidden layer.

Figure 4 displays the MAPE for the Young Modulus (E) and the compressive strength (f_c). From this figure, we can observe MLP+PSO produced slightly better results when compared to ELM+PSO for both mechanical properties, probably due to the flexibility of MLP topology, connectivities and its related wider search space.

Figure 5 depicts the boxplots for the R^2 score and RMSE obtained in 25 runs. From this figure, we observe ELM+PSO model produced a better R^2 and RMSE for Young Modulus (E) when compared with MLP+PSO. However, MLP+PSO performs better then ELM+PSO for Compressive Strength (f_c). Compressive Strength values are obtained from destructive laboratory tests; the results depend on the experimental setup and expertise of the human operator. In addition, the mechanism of failure of specimens depends on the relative positioning of lightweight aggregates within the cement paste matrix. Those characteristics increase the noise in the f_c values resulting in an input/output relationship harder to learn during the training procedure.

Figure 6 shows the distribution of activation functions (G) and number of neurons in the hidden layer (HL) for ELM+PSO and the distribution of connectivity and number of neurons

Table 5: Results for ELM – 25 runs with different seeds for random number generator.

	Parameters		E			f_c		
Run	G	HL	MAPE	R^2	RMSE	MAPE	R^2	RMSE
1	Inv. Multiquadric	21	2.515	0.980	798.423	5.797	0.955	3.596
2	Multiquadric	23	2.197	0.986	676.280	5.718	0.961	3.338
3	Sigmoid	21	2.107	0.987	648.468	5.254	0.968	3.048
4	Inv. Multiquadric	22	2.248	0.983	744.848	5.621	0.954	3.628
5	Sigmoid	19	2.095	0.987	640.884	5.197	0.967	3.088
6	Sigmoid	19	2.201	0.986	670.614	6.392	0.951	3.743
7	Sigmoid	18	2.084	0.987	639.776	5.260	0.968	3.031
8	Sigmoid	17	2.235	0.986	667.501	6.304	0.952	3.705
9	Sigmoid	22	1.975	0.988	614.724	4.914	0.970	2.927
10	Hyp. Tangent	19	2.535	0.982	759.790	6.146	0.951	3.748
11	Sigmoid	23	2.274	0.985	693.532	5.608	0.961	3.321
12	Hyp. Tangent	22	2.223	0.985	702.739	5.672	0.958	3.473
13	Sigmoid	18	2.088	0.987	649.019	5.908	0.957	3.498
14	Sigmoid	21	2.431	0.982	764.226	5.432	0.960	3.384
15	Inv. Multiquadric	19	2.286	0.985	689.539	5.705	0.958	3.487
16	Inv. Multiquadric	19	2.538	0.981	773.937	6.382	0.944	3.995
17	Multiquadric	14	2.393	0.981	773.367	6.310	0.953	3.678
18	Hyp. Tangent	16	2.221	0.985	696.905	5.706	0.964	3.216
19	Sigmoid	20	2.133	0.987	649.315	5.705	0.962	3.295
20	Multiquadric	19	2.198	0.984	708.555	5.291	0.968	3.032
21	Hyp. Tangent	21	2.315	0.982	748.595	5.622	0.962	3.313
22	Multiquadric	18	2.398	0.985	698.171	6.510	0.917	4.885
23	Sigmoid	21	2.114	0.987	654.115	5.550	0.962	3.314
24	Hyp. Tangent	18	2.040	0.988	624.911	5.413	0.964	3.190
25	Sigmoid	18	2.129	0.987	644.341	5.044	0.970	2.929

Table 6: Results for ELM – 25 runs with different seeds for random number generator.

	Parameters		E			f_c		
Run	Topology	Hidden Layers	MAPE	R^2	RMSE	MAPE	R^2	RMSE
1	Feedforward	[7]	2.052	0.988	628.995	5.637	0.959	3.445
2	Fully connected	[11]	2.249	0.983	736.099	6.667	0.949	3.805
3	Feedforward	[9]	1.866	0.989	592.077	4.684	0.974	2.750
4	Fully connected	[6]	2.218	0.985	698.623	6.193	0.950	3.773
5	Fully connected	[20]	2.304	0.983	738.742	5.507	0.954	3.612
6	Fully connected	[4]	2.324	0.984	714.240	5.514	0.959	3.416
7	Feedforward	[10]	2.740	0.975	888.227	5.822	0.951	3.735
8	Feedforward	[5]	2.244	0.985	683.582	5.359	0.967	3.076
9	Feedforward	[6]	2.320	0.984	723.136	5.061	0.966	3.123
10	Feedforward	[8]	2.369	0.984	718.219	5.231	0.965	3.178
11	Feedforward	[14]	2.163	0.981	782.643	5.421	0.961	3.360
12	Feedforward	[9]	2.071	0.986	664.285	4.809	0.970	2.952
13	Feedforward	[11]	2.104	0.987	646.234	5.002	0.971	2.885
14	Fully connected	[14]	2.338	0.983	735.502	6.030	0.957	3.527
15	Feedforward	[12]	2.257	0.983	742.923	4.710	0.971	2.884
16	Feedforward	[21]	2.010	0.986	672.059	5.434	0.966	3.131
17	Fully connected	[14]	2.181	0.984	718.680	6.288	0.955	3.572
18	Feedforward	[11]	1.881	0.989	606.219	5.750	0.956	3.564
19	Feedforward	[9]	2.125	0.986	679.712	5.122	0.965	3.173
20	Fully connected	[10]	2.424	0.981	772.947	5.684	0.955	3.609
21	Feedforward	[11]	1.954	0.987	639.241	5.503	0.962	3.296
22	Fully connected	[12, 1]	1.918	0.986	675.749	6.497	0.945	3.986
23	Feedforward	[6]	2.345	0.985	699.533	4.677	0.973	2.805
24	Feedforward	[10]	2.191	0.984	711.121	4.756	0.968	3.037
25	Fully connected	[8]	2.293	0.984	711.434	6.498	0.943	4.027

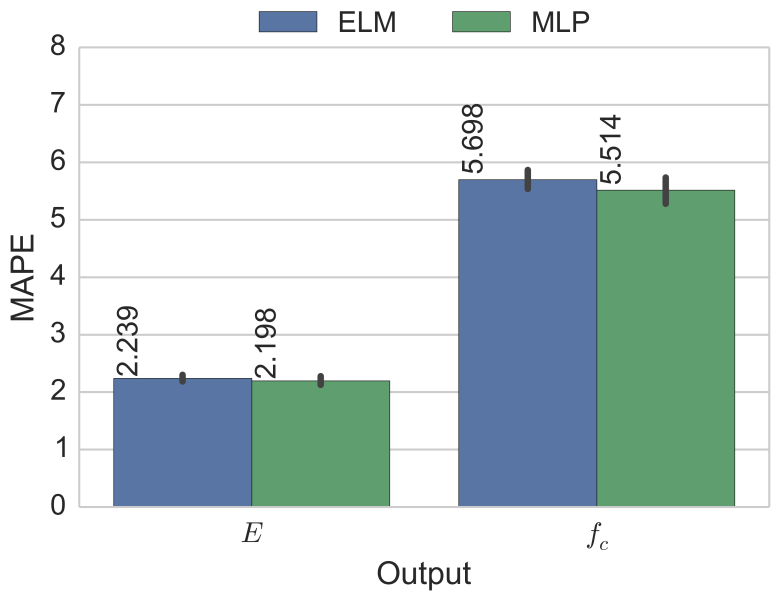

Figure 4: MAPE for ELM and MLP (averaged over 25 runs).

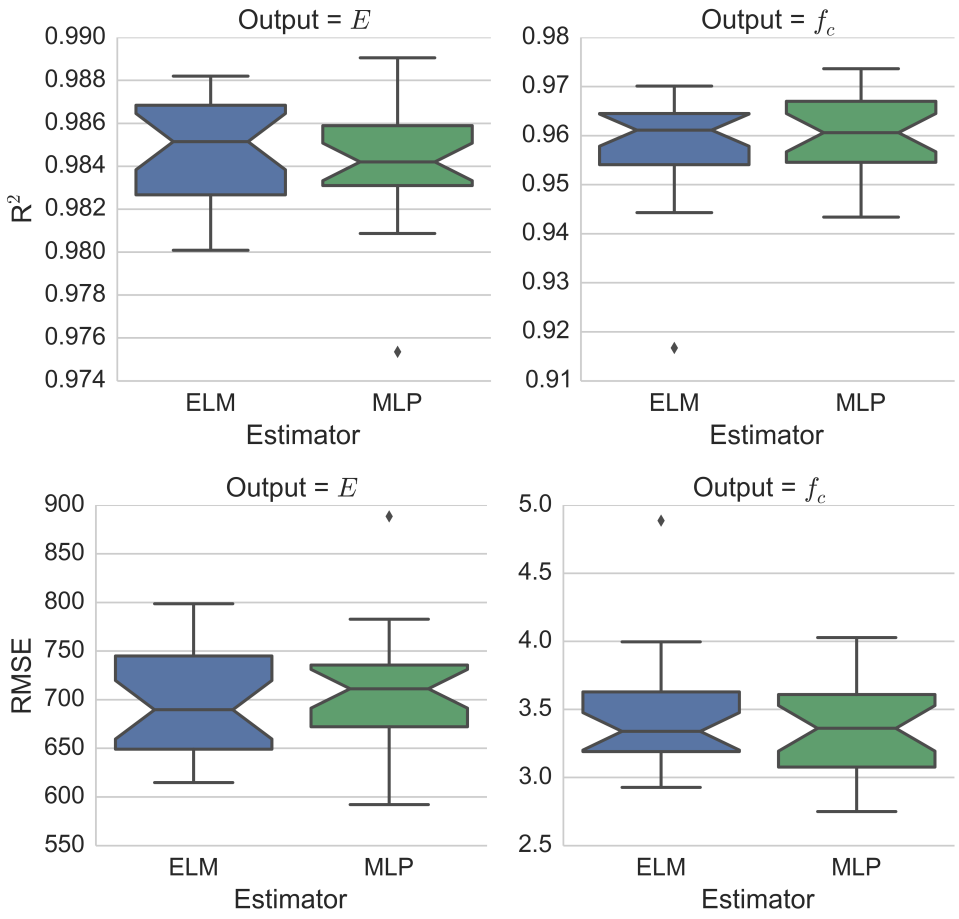

Figure 5: R^2 and RMSE metrics for the ELM

in the hidden layers (HL) for MLP+PSO. We remark the parameters were selected based on the minimization of the RMSE, used as objective function in the evolutionary process in PSO. It can be seen that HL ranges from 14 to 23, whereas 19 is the most frequent value, chosen 6 out 25 runs; we also observe that HL values between 18 and 21 appear 17 out 25 times, corresponding to 68% of the runs. For the activation function, Sigmoid was the most common choice, chosen 12 out 25 runs. Summarizing, in more than half of the cases an ELM network with 17-21 hidden neurons and implementing Sigmoid activation function was suggested by the model selection guided by the PSO algorithm. It can be seen that the connectivity feedforward was the most common choice. For HL, most of the time only one hidden layer was chosen, whereas 11 is the most frequent value. Thus, in most cases, the feedforward MLP network with a single hidden layer, from 6 to 14 neurons, was suggested by the PSO selection.

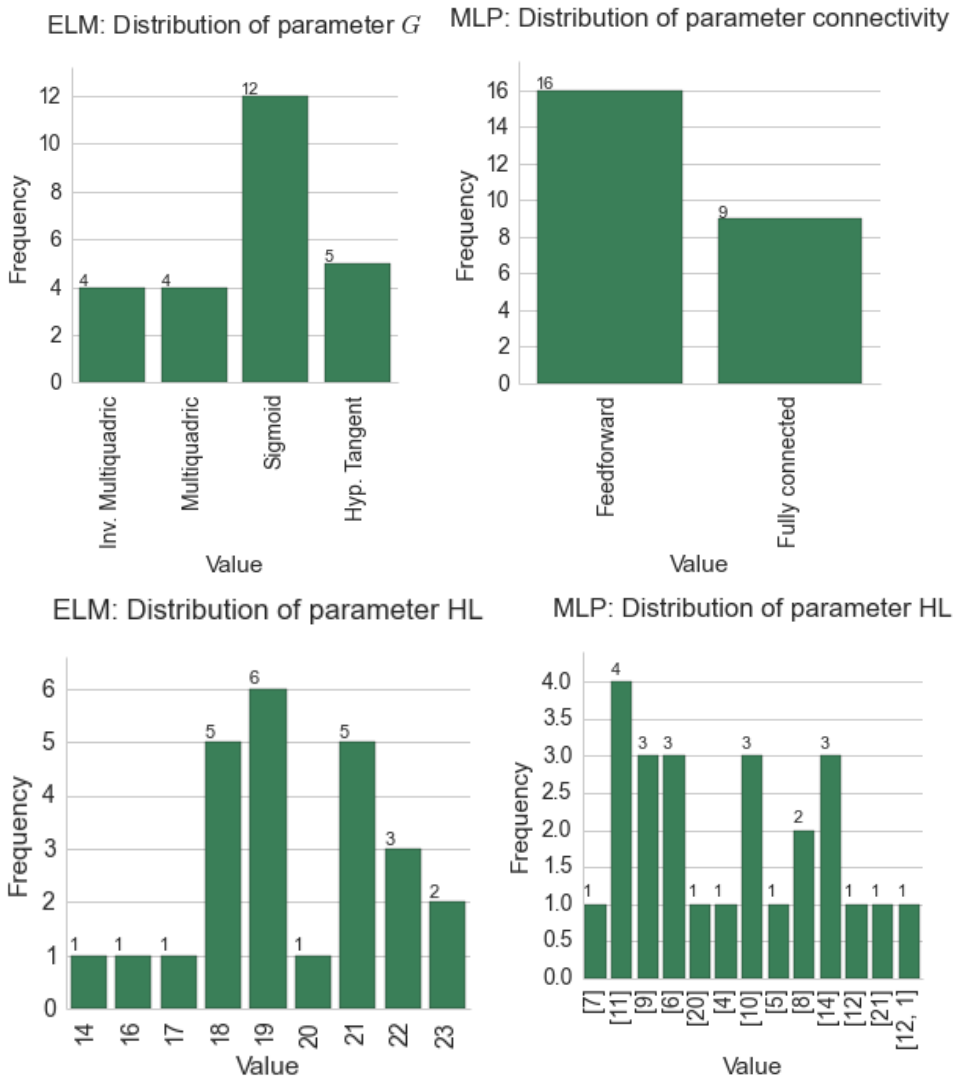

Figure 6: Distribution of parameters along the 25 runs.

Tables 7 and 8 provide a comparison of ELM+PSO and MLP+PSO results with the same networks considering exhaustive search with cross-validation on a set of user defined parameters from Andrade et al. (2015). It can be seen that ELM+PSO outperformed ELM and MLP+PSO outperformed MLP. We highlight the ELM+PSO and MLP+PSO estimate both E and f_c si-

Table 7: Elasticity modulus: comparison with other methods (without evolutionary parametric search) considering MAPE, RMSE, R^2 metrics.

Estimator	MAPE	RMSE	R^2
ELM	2.369 ($\pm$0.206)	5.5e+05 ($\pm$1.0e+05)	0.983 ($\pm$0.003)
ELM+PSO	2.239 ($\pm$0.153)	4.8e+05 ($\pm$7.3e+04)	0.985 ($\pm$0.002)
MLP	2.200 ($\pm$0.155)	5e+05 ($\pm$7.9e+04)	0.984 ($\pm$0.002)
MLP+PSO	2.198 ($\pm$0.191)	5e+05 ($\pm$8.5e+04)	0.984 ($\pm$0.003)

Table 8: Compressive strength: comparison with other methods (without evolutionary parametric search) considering MAPE, RMSE, R^2 metrics.

Estimator	MAPE	RMSE	R^2
ELM	6.883 ($\pm$0.764)	16.370 ($\pm$5.261)	0.943 ($\pm$0.018)
ELM+PSO	5.698 ($\pm$0.431)	11.959 ($\pm$3.075)	0.958 ($\pm$0.011)
MLP	5.577 ($\pm$0.563)	11.579 ($\pm$3.560)	0.959 ($\pm$0.012)
MLP+PSO	5.514 ($\pm$0.588)	11.345 ($\pm$2.437)	0.960 ($\pm$0.009)

multaneously. This feature may turn the training procedure harder when compared with single models built for each output.

It is also interesting to compare the computational cost of the evolutionary model selection with the exhaustive search procedure Pedregosa et al. (2011). For a grid search procedure considering the range of 7 activations functions and 100 hidden neurons in ELM as suggested in this paper, a total of $7 \times 100 = 700$ model evaluations are required to obtain the best model. This is the same budget of PSO that was implemented using a population size of 20 individuals evolving along 35 generations, resulting in a total of $20 \times 35 = 700$ model evaluations. In the case of the MLP suggested in this article in the grid search, considering the range of 5 layers, 2 types of connections and 30 neurons, it would be necessary to evaluate 4.86×10^7 models to find a better combination of parameters.

The grid search can become a very time-consuming step when the number of parameters increases or when we have a large set of user-defined options of each parameter. For example, a model with 5 parameters to be tuned where each set has 10 different parameter values requires 10^5 model evaluations. In such a case, the evolutionary approach for model selection arises as an interesting alternative to save computing time while producing optimal or near-optimal solutions for the model selection procedure.

4 CONCLUSIONS

In this paper, an Extreme Learning Machine and a Multilayer Perceptron were combined with a Particle Swarm Optimization algorithm to simultaneously predicting the elastic modulus and the compressive strength of lightweight aggregate concretes. The proposed approach

performs an intelligent search on the parameter space and arises as an interesting alternative to exhaustive search allowing for computational savings in large sets of parameters. The results show good agreement with experimental results and competitive performance when compared with other machine learning approaches.

Future works include exploring the relative importance of each attribute in mechanical properties estimation and evaluate other methods with a wide range of parameters and different estimation capabilities such as Support Vector Machines and Gradient Boosting Regression.

ACKNOWLEDGMENT

The authors would like to thank the Federal University of Juiz de Fora, FAPEMIG (grant APQ 01606/15) and CAPES (Procad 071/2013) for supporting this work.

REFERENCES

(2017). Particle swarm optimization (PSO) with constraint support. `https://github.com/tisimst/pyswarm/`. Accessed: 2017-03-01.

ACI (1999). *Guide for structural lightweight aggregate concrete. ACI 213R-87.* American Concrete Institute.

Alshihri, M. M., Azmy, A. M., and El-Bisy, M. S. (2009). Neural networks for predicting compressive strength of structural light weight concrete. *Construction and Building Materials*, 23(6):2214 – 2219.

Altun, F., zgr Kii, and Aydin, K. (2008). Predicting the compressive strength of steel fiber added lightweight concrete using neural network. *Computational Materials Science*, 42(2):259 – 265.

Andrade, J. J., de Souza Barbosa, F., Farage, M. C. R., Beaucour, A.-L., Ortola, S., and da Fonseca, L. G. (2015). Aplicação de métodos de inteligência computacional para a previsão de propriedades mecânicas do concreto de agregado leve. In *XXXVI Ibero-Latin American Congress on Computational Methods in Engineering*.

Bingl, A. F., Tortum, A., and Gl, R. (2013). Neural networks analysis of compressive strength of lightweight concrete after high temperatures. *Materials & Design (1980-2015)*, 52:258 – 264.

Bogas, J. A. and Gomes, A. (2013). Compressive behavior and failure modes of structural lightweight aggregate concrete characterization and strength prediction. *Materials & Design*, 46:832 – 841.

Chandra, S. and Berntsson, L. (2003). *Lightweight aggregate concrete*. Science, technology and applications. Noyes Publications - William Andrew Publishing, USA.

Duan, Z., Kou, S., and Poon, C. (2013). Using artificial neural networks for predicting the elastic modulus of recycled aggregate concrete. *Construction and Building Materials*, 44(0):524 – 532.

Friedman, J. H. (1991). Multivariate adaptive regression splines. *The annals of statistics*, pages 1–67.

Gao, T., Shen, L., Shen, M., Liu, L., Chen, F., and Gao, L. (2017). Evolution and projection of {CO2} emissions for china's cement industry from 1980 to 2020. *Renewable and Sustainable Energy Reviews*, 74:522 – 537.

Garciaa, A., Anjos, O., Iglesias, C., Pereira, H., Martnez, J., and Taboada, J. (2015). Prediction of mechanical strength of cork under compression using machine learning techniques. *Materials & Design*, 82:304 – 311.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning - Data Mining, Inference, and Prediction*. Springer, Verlag, New York, 2 edition.

Haykin, S. O. (2008). *Neural Networks and Learning Machines*. Prentice Hall, 3rd edition edition.

Huang, G., Huang, G.-B., Song, S., and You, K. (2015). Trends in extreme learning machines: A review. *Neural Networks*, 61:32 – 48.

Huang, G.-B. (2015). What are extreme learning machines? filling the gap between frank rosenblatts dream and john von neumanns puzzle. *Cognitive Computation*, 7(3):263–278.

Igliski, B. and Buczkowski, R. (2017). Development of cement industry in poland history, current state, ecological aspects. a review. *Journal of Cleaner Production*, 141:702 – 720.

Ke, Y. (2008). *Characterization of the mechanical behavior of lightweight aggregate concretes: Experiment and modelling*. PhD thesis, PhD. Thesis, Université de Cergy-Pontoise.

Kennedy, J. (2011). Particle swarm optimization. In *Encyclopedia of machine learning*, pages 760–766. Springer.

Kockal, N. U. and Ozturan, T. (2011). Strength and elastic properties of structural lightweight concretes. *Materials and Design*, 32(4):2396 – 2403.

Nilson, A. H., Martinez, S., et al. (1986). Mechanical properties of high-strength lightweight concrete. In *Journal Proceedings*, volume 83, pages 606–613.

Nissen, S. (2005). Neural networks made simple. *Software 2.0*, 2:14–19.

Papadakis, V. and Tsimas, S. (2002). Supplementary cementing materials in concrete: Part i: efficiency and design. *Cement and Concrete Research*, 32(10):1525 – 1532.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research*, 12:2825–2830.

Shen, W., Liu, Y., Yan, B., Wang, J., He, P., Zhou, C., Huo, X., Zhang, W., Xu, G., and Ding, Q. (2017). Cement industry of china: Driving force, environment impact and sustainable development. *Renewable and Sustainable Energy Reviews*, 75:618 – 628.

Wojciechowski, M. (2012). Solving differential equations by means of feed-forward artificial neural networks. In *Proceedings of the 11th International Conference on Artificial Intelligence and Soft Computing - Volume Part I*, ICAISC'12, pages 187–195, Berlin, Heidelberg. Springer-Verlag.