Engineering with Computers (2022) 38:757–779
https://doi.org/10.1007/s00366-020-01081-0

ORIGINAL ARTICLE

A novel hybrid extreme learning machine–grey wolf optimizer (ELM-GWO) model to predict compressive strength of concrete with partial replacements for cement

Mahdi Shariati[1] · Mohammad Saeed Mafipour[2] · Behzad Ghahremani[2] · Fazel Azarhomayun[2] · Masoud Ahmadi[3] · Nguyen Thoi Trung[4,5] · Ali Shariati[4,5]

Received: 8 December 2019 / Accepted: 2 June 2020 / Published online: 20 June 2020

Abstract
Compressive strength of concrete is one of the most determinant parameters in the design of engineering structures. This parameter is generally determined by conducting several tests at different ages of concrete in spite of the fact that such tests are not only costly but also time-consuming. As an alternative to these tests, machine learning (ML) techniques can be used to estimate experimental results. However, the dependence of compressive strength on different parameters in the fabrication of concrete makes the prediction problem challenging, especially in the case of concrete with partial replacements for cement. In this investigation, an extreme learning machine (ELM) is combined with a metaheuristic algorithm known as grey wolf optimizer (GWO) and a novel hybrid ELM-GWO model is proposed to predict the compressive strength of concrete with partial replacements for cement. To evaluate the performance of the ELM-GWO model, five of the most well-known ML models including an artificial neural network (ANN), an adaptive neuro-fuzzy inference system (ANFIS), an extreme learning machine, a support vector regression with radial basis function (RBF) kernel (SVR-RBF), and another SVR with a polynomial function (Poly) kernel (SVR-Poly) are developed. Finally, the performance of the models is compared with each other. The results of the paper show that combining the ELM model with GWO can efficiently improve the performance of this model. Also, it is deducted that the ELM-GWO model is capable of reaching superior performance indices in comparison with those of the other models.

Keywords Grey wolf optimizer · Extreme learning machine · Hybrid ELM-GWO · Compressive strength prediction · Partial replacement of cement

1 Introduction

Since the advent of concrete as a human-made artificial stone, several construction projects have been carried out successfully by using this profitable material. However, estimating the properties of hardened concrete has always been one of the principal challenges in concrete technology. This is mainly due to the fact that many predictable or unpredictable factors may significantly affect the properties of concrete [1, 2]. Compressive strength is one of the properties of concrete which plays a prominent role in the design of engineering structures. In current practice, to determine the compressive strength of concrete, several cubic or cylindrical samples are fabricated and tested at different ages of the samples. However, these tests are not only costly but also time-consuming [3, 4]. Also, changes in the mix design of concrete can result in concrete with completely different

✉ Ali Shariati
alishariati@tdtu.edu.vn

1 Institute of Research and Development, Duy Tan University, Da Nang 550000, Vietnam

2 School of Civil Engineering, College of Engineering, University of Tehran, Tehran, Iran

3 Department of Civil Engineering, Ayatollah Boroujerdi University, Boroujerd, Iran

4 Division of Computational Mathematics and Engineering, Institute for Computational Science, Ton Duc Thang University, Ho Chi Minh City 758307, Vietnam

5 Faculty of Civil Engineering, Ton Duc Thang University, Ho Chi Minh City 758307, Vietnam

properties; thus, the tests need to be repeated even if the content of an ingredient varies [5]. This issue can be better felt in the case of concrete in which cement has been partially replaced by pozzolan powders.

Fly ash (FA) and furnace slag (FS), as sorts of pozzolan powders, are commonly used as partial replacements for cement [6, 7]. FA is generally gained as residue from the combustion of pulverized coal in the furnaces of thermal power plants. Properties of FA are dependent on the combustion process through which it is obtained [8]. Dry processing leads to the FA which is homogenous in particle size, while wet processing can result in the FA with highly separated aggregates [9]. Adding FA to concrete as a partial replacement for cement reduces the values of compressive strength and slump; however, it increases the workability and integrity of the concrete [10]. FS is the by-product of iron and steel production in blast furnaces whose chemical composition is dependent on the raw materials from which it is produced. The slag floats on top of the iron in blast furnaces, and it is decanted for separation [11]. Quick cooling of the molten slag converts it to non-crystalline ingredients with hydraulic properties [12]. Partial replacement of cement with FS results in concrete with higher compressive strength and durability. However, a higher dosage of FS can cause thermos-hygral (TH) damages and cracks, which adversely affect the strength and mechanical properties of concrete [13, 14]. The use of FA and FS in concrete not only addresses a mean for disposal of these waste materials, but also provides an alternative for cement whose production results in the emission of a large amount of carbon dioxide (CO_2) and subsequently, global warming [7, 15, 16]. However, due to the dependence of compressive strength on several parameters and high sensitivity to the mixture proportions, it seems that more advanced methods should be employed to not only eliminate the need for conducting experiments as much as possible but also provide a simpler tool for engineers to predicting experimental results.

Soft computing (SC) can be mentioned as an efficient approach that can be used in this regard. The most significant advantage of SC is generating solutions for nonlinear or linear problems where mathematical models cannot easily express the relation among the involving parameters in the problem [17]. Moreover, SC methods use human-based knowledge, recognition, understanding, and learning in computation [18]. In recent years, several researchers have employed artificial intelligence (AI) methods and machine learning (ML) techniques, as sub-branches of SC methods, in properties prediction of different types of concrete.

Oztas et al. [19] investigated the potential of using an artificial neural network (ANN) in predicting the slump value and compressive strength of high strength concrete (HSC). This study revealed the high capability of ANN in the properties prediction of HSC. Alshihri et al. [20] used an ANN model in the compressive strength prediction of lightweight concrete and concluded that using ANN can reduce cost and save time. In another study, the satisfying performance of ANN was reported in compressive strength prediction of self-compacting concrete (SCC) and high-performance concrete (HPC) [21]. Khademi et al. [22] employed an ANN model, an adaptive neuro-fuzzy inference system (ANFIS) model, and a multiple linear regression (MLR) model to predict compressive strength of normal strength concrete. They concluded that the MLR model is not feasible enough in the case of compressive strength prediction as the problem seems nonlinear, while the ANN and ANFIS models are reliable enough to be used in this area. Yaseen et al. [23] compared the performance of an extreme learning machine (ELM), a support vector regression (SVR), a multivariable adaptive regression spline (MARS), and M5 tree model in compressive strength prediction of lightweight foamed concrete. Superior performance of the ELM was reported in this investigation. Alshamiri et al. [24] also examined the capability of ELM in the compressive strength prediction of HSC and compared it with an ANN model. In this case, they could observe better performance from the ELM model than that of the ANN model. Han et al. [25] combined an ANN with particle swarm optimization (PSO) algorithm to estimate the compressive strength of ground granulated blast furnace slag (GGBFS) concrete and compared the result of the hybrid ANN-PSO model with an ANN model. They reported improvements in the performance of ANN after combining with PSO. Golafshani et al. [26] combined an ANN and an ANFIS model with grey wolf optimizer (GWO) and showed that hybridization of the models with GWO improves the training and generalization capability of both ANN and ANFIS models. Sun et al. [27] also predicted and optimized the influencing factors on the compressive strength of concrete containing silica fume (SF) and fly ash (FA) by a hybrid ANN-ABC (artificial bee colony) model. In another interesting research, Ashrafian et al. [28] combined a typical MARS model with a water cycle algorithm (WCA) and suggested a hybrid MARS-WCA model for predicting the compressive strength of foamed cellular lightweight concrete. This research also revealed that the hybridization of standard models with metaheuristic algorithms can improve the performance of the models.

The main objective of this study is to predict compressive strength of concrete in which cement has been partially replaced with fly ash (FA) and furnace slag (FS). To achieve this goal, a SC approach is adopted. An extreme learning machine (ELM) is combined with a metaheuristic algorithm known as grey wolf optimizer (GWO) and a hybrid ELM-GWO model is proposed. Next, the capability of this model is investigated in the case of compressive strength prediction. For this purpose, the most well-known and powerful machine learning (ML) models including an artificial neural network (ANN), an adaptive neuro-fuzzy inference system (ANFIS), an extreme learning machine (ELM), a support vector regression (SVR) with a radial basis function (RBF) kernel (SVR-RBF), and another SVR with a polynomial (Poly) kernel (SVR-Poly) are developed. Design mixture proportion of concrete together with the age of samples is considered as the inputs of the models, and compressive strength of concrete is predicted as the output. Finally, the results of the developed ML models will be compared with each other in terms of different statistical performance indices and the required time for training.

2 Methodologies

2.1 Artificial neural network (ANN)

Artificial neural network (ANN) can be mentioned as the most well-known and implemented methodology in the case of function approximation. ANN is an intelligence tool that has been inspired by the biological neural network of humans or animals [29, 30]. This model with a layer-to-layer structure is able to learn patterns and predict results in the high-dimensional space of the problem [31, 32]. Multilayer perceptron (MLP) is a simple and reliable class of feed-forward ANN. A typical MLP contains an input layer, at least one hidden layer, and an output layer [33, 34]. The input layer takes the values of predictors and sends them to the available neurons in the next layer. A typical neuron in the structure of an ANN is shown in Fig. 1. Inside each neuron, a weighted sum of inputs is computed. Then, this value, plus a value of bias, called *Net*, is calculated. In the next step, the net value passes through an activation function. The most popular and well-known activation functions are tanh and sigmoid. The concept behind using the activation function in the neuron structures is the fact that without any activation function, the output of a neural network seems to be a linear combination of input values. Accordingly, activation functions are used to make a nonlinear relation between the inputs and outputs of a neural network. The number of neurons, the number of layers, activation functions, and definition of error and evaluation criteria play a vital role in the performance of a neural network. Therefore, these free parameters should be selected in the way that the network's results reach an acceptable outcome. This mathematical process can be formulated as follows [21, 35]:

$$\text{Net} = \sum_{i=1}^{n} w_{ij}x_i + b_j \tag{1}$$

where x_i is the nodal values in the previous. n is the total number of the nodal values received from the previous layer. w_{ij} and b_j are also weights and biases of the network in the current layer.

Finally, the *Net* value is transformed by an activation function and the output signal is transferred to the neurons in the next layer. The tangent hyperbolic function is an activation function that generally leads to more accurate results [36]; thus, this activation function is used in this study. This function varies between − 1 and 1 and is defined as follows:

$$y = f(Net) = \frac{2}{1 + e^{-2\cdot Net}} - 1 \tag{2}$$

where y is the output signal; f is the activation function in terms of calculated network value (Net).

This process is performed on each of the layers of an MLP until the output signals or predicted values in the last layer are determined. Then, the error value of the neural network is calculated and this value of error is minimized by changing the used weights and biases throughout the MLP. This process which is defined as training can be conducted by different optimization algorithms. However, the fast convergence rate and appropriate precision of back-propagation (BP) algorithms have caused these algorithms to be basically employed in the training phase of standard ANNs [35].

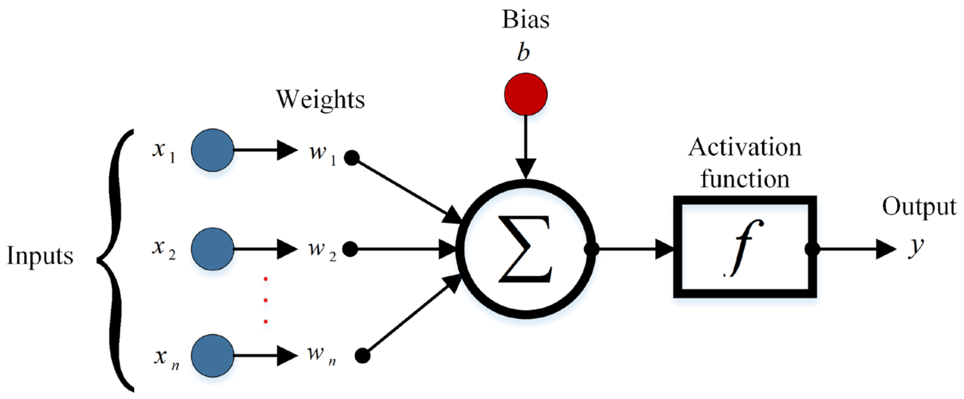


Fig. 1 A typical neuron in an ANN

2.2 Adaptive neuro-fuzzy inference system (ANFIS)

An adaptive neuro-fuzzy inference system (ANFIS) is a specific sub-branch of ANN which benefits from the combined features of neural networks and fuzzy logic principles [37–40]. ANFIS was developed by Jang [41] in 1993 to model nonlinear functions, identify nonlinear components, and predict chaotic time series. ANFIS is capable of constructing an input–output mapping, based on the Takagi–Sugeno fuzzy inference system (in the form of fuzzy IF-THEN rules) [42, 43]. Many advantages of ANFIS such as the ability to capture the nonlinear structure of a process, adaptation capability, and rapid learning have made it very popular among engineers [44–46].

ANFIS architecture has five layers, as shown in Fig. 2. The central core of the ANFIS is a fuzzy inference system (FIS). The first layer receives inputs (x and y in Fig. 2) and converts them to fuzzy values by membership functions (MFs). The rule base contains two fuzzy IF-THEN rules of Takagi's and Sugeno's type:

Rule 1: if x is A_1 and y is B_1, then $f_1 = p_1x + q_1y + r_1$,

Rule 2: if x is A_2 and y is B_2, then $f_2 = p_2x + q_2y + r_2$.

Every node in this layer (i.e., the first layer) is selected as an adaptive node with a node function,

$$O_i^1 = \mu A_i(x) \tag{3}$$

where A_i is a linguistic label and O_i^1 is the membership function of A_i.

Bell-shaped membership functions (or Gaussian functions) are usually used in ANFIS as they have a higher capacity in the regression of nonlinear data [30, 47–53]. A bell-shaped membership function with the maximum value of one and minimum value of zero is defined as follows:

$$\mu(x) = \text{bell}\left(x; a_i, b_i, c_i\right) = \frac{1}{1 + \left[\left(\frac{x-c_i}{a_i}\right)^2\right]^{b_i}} \tag{4}$$

where $\{a_i, b_i, c_i, d_i\}$ are the parameters set and x is the input. The parameters of this layer are known as *premise parameters*.

The second layer multiplies the incoming signals and sends their product to the next layer. For instance:

$$w_i = \mu A_i(x) \times \mu B_i(y), \quad i = 1, 2. \tag{5}$$

Every output of the nodes exhibits the firing strength of a rule.

The third layer is the rule layer. In this layer, the ratio of the ith node firing strength of rule to those of the other nodes is calculated. This means that:

$$w_i^* = \frac{w_i}{w_1 + w_2} \quad i = 1, 2. \tag{6}$$

The outcomes w_i^* are known as normalized firing strength.

The fourth layer is the defuzzification layer in which every node has a node function as follows:

$$O_i^4 = w_i^* f_i = w_i^* \left(p_i x + q_i y + r_i\right) \tag{7}$$

where w_i^* is the output of the third layer and $\{p_i, q_i, r_i\}$ are the parameters of this layer known as *consequent parameters*.

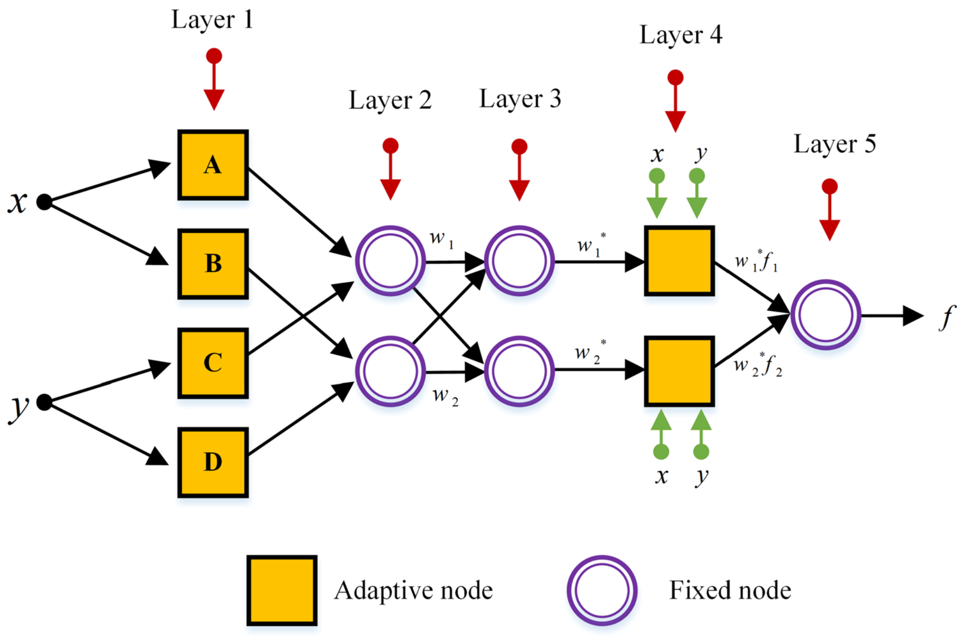


Fig. 2 Layers of a typical ANFIS model

The output layer is the fifth layer. In this layer, the overall output is computed by summing all the incoming signals. This means that:

$$O_1^5 = f = \sum_i w_i^* f_i \tag{8}$$

In this process, a threshold value between the actual value and the output is set. Then, the consequent parameters are obtained by the least-squares method and an error for each of data is obtained. If this value is larger than the considered threshold, the premise parameters are updated by the use of a gradient descent algorithm. This process continues until the error becomes less than the threshold. Since the parameters are determined by two algorithms (i.e., least-squares and gradient descent algorithms) simultaneously, the used algorithm in this process is known as a hybrid algorithm.

2.3 Support vector regression (SVR)

The fundamental principle of support vector regression (SVR) is to map input data to multidimensional feature space and perform a linear regression in this space so that the empirical risk is minimized [54–57]. The flexible nature of SVR is attributed to the kernel functions (k) which implicitly chart data to the feature space. This process is known as the kernel trick because the linear regression in the feature space represents a nonlinear regression in the original space of the problem [58]. Several kernel functions have been defined for this purpose. However, the radial basis function (RBF) kernel and polynomial (Poly) function kernel usually lead to better results in comparison with other kernels [59]. These kernel functions are defined as follows:

$$\begin{aligned}\text{RBF kernel} \rightarrow k &= k(x, x_i) \\ &= \exp\left(-\frac{\|x - x_i\|^2}{2\sigma^2}\right) \\ &= \exp\left(-\gamma\|x - x_i\|^2\right), \quad \gamma = \frac{1}{2\sigma^2}\end{aligned} \tag{9}$$

$$\text{Poly kernel} \rightarrow k = k(x, x_i) = (x.x_i + 1)^d \tag{10}$$

where x and x_i are vectors in the input space; γ and σ are the parameters which define how far the influence of a single sample reaches; d is the degree of the polynomial kernel function.

The SVR theory can be defined as follows:

Assume that there is a training set of N samples in the d -dimensional space of the problem; these samples can be represented as follows:

$$(x_1, y_1), \ldots, (x_i, y_i), \ldots, (x_N, y_N) \quad x_i, y_i \in R^d \tag{11}$$

where x_i is a sample value of input vector x containing N training points. y_i is the corresponding output value of the sample.

As mentioned before, SVR does perform a linear regression in the feature space of the problem. Thus, if the data points are transferred to such space by the kernel functions (k), the SVR model can be defined by the following linear equation:

$$\hat{y}_i = f(x) = \omega^T \phi(x) + b \tag{12}$$

where $\hat{y}_i$ is the output predicted vector; ω is the weight vector; $\phi(x)$ is the mapping functions applied for feature extraction; and b is the term of bias.

As described previously, SVR minimizes the empirical risk in the problem. Hence, if the variable of empirical risk is defined by R_{emp}, it can be written that:

$$R_{\text{emp}} = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}_i|_\varepsilon \tag{13}$$

where $|y_i - \hat{y}_i|_\varepsilon$. is Vapnik's ε-intensive loss function, defined by:

$$\begin{cases} 0 & \text{if} |y_i - \hat{y}_i| \le \varepsilon \\ |y_i - \hat{y}_i| - \varepsilon |y_i - \hat{y}_i| - \varepsilon & \text{otherwise} \end{cases} \tag{14}$$

The weight vector ω and the bias term b can then be determined by minimizing the cost function $J(\omega, \zeta, \zeta_i^*)$, described as the following:

$$J(\omega, \zeta, \zeta_i^*) = \frac{1}{2}\omega^T\omega + C\sum_{i=1}^{N}(\zeta + \zeta_i^*) \tag{15}$$

The constraints of this function are also as follows:

$$\begin{cases} y_i - \hat{y}_i \le \varepsilon + \zeta_i \\ -y_i + \hat{y}_i \le \varepsilon + \zeta_i^* \\ \zeta_i \ge 0 \\ \zeta_i^* \ge 0 \end{cases} \quad i = 1, 2, 3, \ldots, N \tag{16}$$

where ζ and ζ^* are positive slack variables, and C is a positive real cost value.

2.4 Extreme learning machine (ELM)

Extreme learning machine (ELM) was proposed by Huang et al. [60] in 2006 for single-layer feed-forward neural network (SLFN) architectures. ELM was originated from the observations which proved that an SLFN with random weights and biases could approximate any continuous function on any compact input set [61]. Hence, it was realized that input weights and biases of an SLFN could be randomly selected, and based on them, the output weights of the SLFN would be analytically calculated. Employing this idea in finding the weights and biases of the SLFN resulted in an algorithm with extremely fast learning speed. Also, ELM systematically determines all the network factors, thus preventing unnecessary human interferences [35, 62–67].

A three-step procedure is involved in developing the ELM model as follows: (I) an SLFN is created; (II) weights and biases of the network are randomly selected; (III) the output weights are estimated by inverting the hidden layer output matrix [24, 68].

For a dataset containing N training samples with n -dimensional input vectors and m-dimensional target vectors, the SLFN with L hidden nodes can mathematically be defined as follows:

$$\sum_{i=1}^{L} \beta_i G\left(w_i.x_j + b_i\right) = o_j \quad j = 1,2,3,\ldots,N \tag{17}$$

where G is the activation function, all the neural network-based activation functions can be used herein too; $w_i = \left[w_{i1}, w_{i2}, \ldots, w_{in}\right]^T$ is the weight vector connecting the ith hidden neuron to the input neurons; $x_j = \left[x_{j1}, x_{j1}, \ldots, x_{jm}\right]^T$ is the input vector; $\beta_i = \left[\beta_{i1}, \beta_{i2}, \ldots, \beta_{im}\right]^T$ is the weight vector connecting the hidden neurons to the output neurons; $b_i = \left[b_{i1}, b_{i2}, \ldots, b_{im}\right]^T$ is the bias vector; $o_j = \left[o_{j1}, o_{j1}, \ldots, o_{jm}\right]^T$ is the output vector.

If it is assumed that an SLFN with L hidden neurons and activation function G can approximate the targets (t_j) with zero error, i.e., $\sum_{j=1}^{L} o_j - t_j = 0$, Eq. (17) can be transformed to:

$$\sum_{i=1}^{L} \beta_i G\left(w_i.x_j + b_i\right) = t_j \quad j = 1,2,3,\ldots,N \tag{18}$$

where $t_j = \left[t_{j1}, t_{j2}, \ldots, t_{jm}\right]^T$ is the target vector. Also, the above N equations can be compactly written as:

$$H\beta = T \tag{19}$$

in which:

$$H = \begin{bmatrix} G\left(w_1 + x_1 + b_1\right) & \ldots & G\left(w_L.x_1 + b_L\right) \\ \vdots & \ldots & \vdots \\ G\left(w_1 + x_N + b_1\right) & \ldots & G\left(w_L.x_N + b_L\right) \end{bmatrix}_{N\times L} \tag{20}$$

and

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{bmatrix}_{L\times m} \text{ and } T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N\times m} \tag{21}$$

The output weights will be obtained if the minimum difference between the left side (predicted values) and the right side (target values) of Eq. (19) occurs, i.e., min min $\|H\beta - T\|$. Although backpropagation (BP) algorithms can minimize this fitness function, similar to what occurs in an ANN, ELM uses mathematical theories and proves that the minimum error between the predicted and target values occurs when the output weights vector is determined as follows [60]:

$$\hat{\beta} = H^{\dagger} T \tag{22}$$

where $\hat{\beta}$ is the output weight vector; $H^{\dagger}$ is Moore–Penrose generalized inverse matrix; and T is the target vector.

As it was illustrated theoretically, in contrast to other models that obtain weights and biases of the models through minimization of errors, no minimization and iteration process are involved in a standard ELM model and the SLFN is tuned by calculating the output weights through the Moore–Penrose generalized inverse matrix.

2.5 Grey wolf optimizer (GWO)

Grey wolf optimizer (GWO) is a metaheuristic algorithm which was proposed by Mirjalili et al. [69]. This algorithm has been inspired by the leadership hierarchy and the hunting mechanism of grey wolves. Grey wolves live in a pack and have a very strict social dominant as illustrated in Fig. 3. Leaders of the wolves are called alpha (α) as they are responsible for making decisions. The second level wolves are beta (β) that help alpha wolves in their responsibilities. The last one in this hierarchy is known as omega (ω) that plays the role of scapegoat. If a wolf is categorized in none of the mentioned levels, it is known as a delta (δ) wolf as well [67, 70]. According to this well-defined leadership hierarchy, grey wolves try to encircle a prey, attack, hunt, and search for other prey as depicted in Fig. 4.

The encircling behavior of grey wolves in hunting can be mathematically expressed as follows [69]:

$$\vec{D} = \left|\vec{C} \cdot \overrightarrow{X_p}(t) - \vec{X}(t)\right| \tag{23}$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} \cdot \vec{D} \tag{24}$$

where $\vec{X}$ describes the position of a grey wolf in a circular configuration; $\vec{X}_p$ is the location vector of a prey; t is the current moment; $\vec{A}$ and $\vec{D}$ are the coefficient vectors which can be defined as follows:

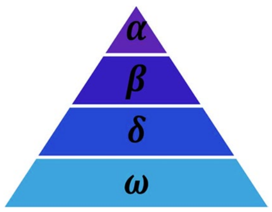


Fig. 3 Hierarchy of grey wolves [69]

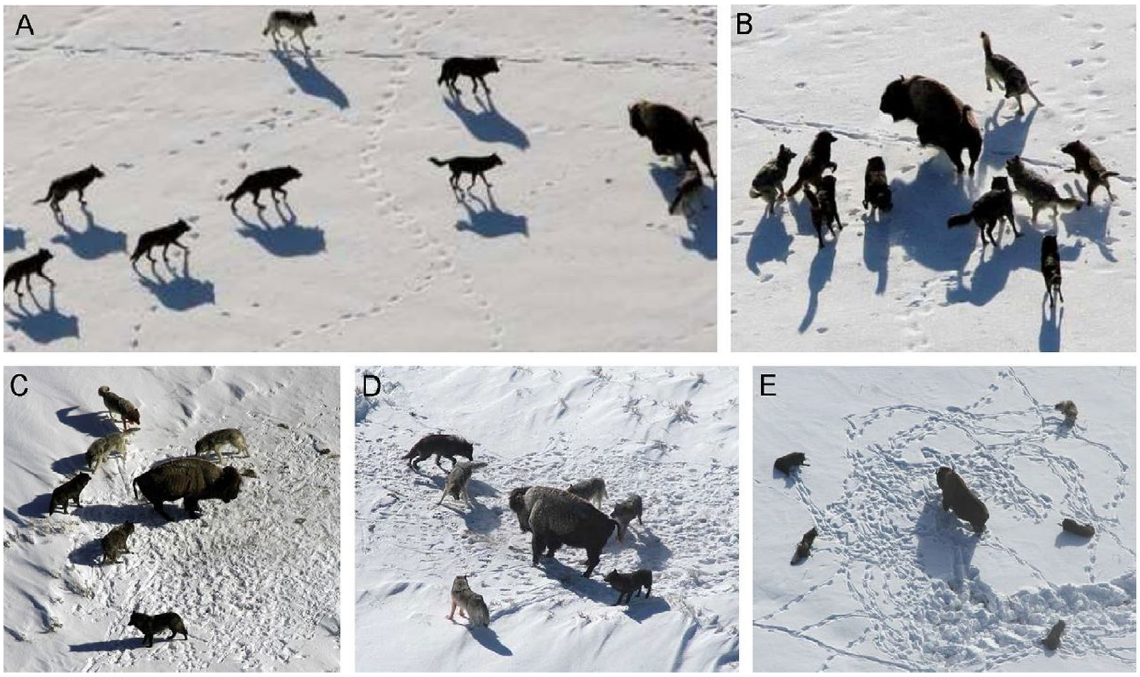


Fig. 4 Hunting strategy of grey wolves: **a** searching and tracking, **b–d** pursuit, harass, and encircling, **e** final stationary configuration at the end of the hunt [70]

$$\vec{A} = 2\vec{a} \cdot \vec{r_1} - \vec{a} \tag{25}$$

$$\vec{C} = 2 \cdot \vec{r_2} \tag{26}$$

in which the component a is linearly decreased from 2 to 0; $\vec{r_1}$ and $\vec{r_2}$ are also random vectors uniformly distributed between 0 and 1.

Since the location of the prey (the optimum location) is not obvious in advance, it is assumed that the α, β, and δ wolves have better knowledge about it [69]. Therefore, the average location of these wolves is used in order to determine the location of the prey. It can thus be written that:

$$\overrightarrow{D_\alpha} = \left|\overrightarrow{C_1} \cdot \overrightarrow{X_\alpha} - \vec{X}\right|, \quad \overrightarrow{D_\beta} = \left|\overrightarrow{C_2} \cdot \overrightarrow{X_\beta} - \vec{X}\right|, \quad \overrightarrow{D_\delta} = \left|\overrightarrow{C_3} \cdot \overrightarrow{X_\delta} - \vec{X}\right| \tag{27}$$

$$\vec{X}_1 = \vec{X}_\alpha - \overrightarrow{A_1} \cdot \overrightarrow{D_\alpha}, \quad \vec{X}_2 = \vec{X}_\beta - \overrightarrow{A_2} \cdot \overrightarrow{D_\beta}, \quad \vec{X}_3 = \vec{X}_\delta - \overrightarrow{A_3} \cdot \overrightarrow{D_\delta} \tag{28}$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{29}$$

After approximating the location of the prey, the next step is to hunt it (exploitation). This purpose can be achieved by the vector $\vec{A}$ because when the value of a in Eq. (25) decreases from 2 to 0, the position of wolves approaches the location of the prey according to Eq. (24). Moreover, to maintain the searching capability (exploration) of this algorithm and avoidance from local minimums, both of the parameters C and A contribute. On the one hand, the parameter C may change the location of prey and the hardness of hunting and, on the other hand, the A values greater than 1, i.e.,$|A| > 1$, force the grey wolves to diverge from the prey and find a fitter prey [69]. If this process is repeated for a population of grey wolves and a specific number of iterations, finally, Eq. (29) will show the location of the prey or global optimum point.

2.6 Hybrid ELM-GWO

As mentioned in Sect. 2.4., ELM calculates the output weights of the model based on mathematical theories and without using any optimization algorithm. This approach causes that ELM has the following advantages over the other proposed models [60, 65, 71]:

1. In spite of the backpropagation algorithms which cannot generally reach the exact values of weights and biases, ELM is capable of computing these values precisely since it uses mathematical theories in the calculation.
2. In contrast to backpropagation algorithms that do not consider the magnitude of weights and biases and only try to minimize the value of the fitness function, ELM employs small values for output weights of the SLFN.
3. This approach makes ELM an extremely fast algorithm in calculating output weights and tuning the SLFN.

However, the most considerable deficiency of ELM lies in the initial input weights and biases which are randomly assigned to the SLFN. Although it has been proposed that the initial input weights do not significantly affect the efficiency of the ELM model, it is highly possible that these weights and biases cannot result in the best output weights. Moreover, the input weights and biases cannot find any opportunity to be updated throughout a standard ELM algorithm and this, in turn, can adversely reduce the capability of the model.

To address these issues and improve the efficiency of ELM, this algorithm can be combined with other optimization algorithms. In this study, the ELM algorithm will be combined with GWO since this algorithm benefits from an appropriate convergence rate and does not have many parameters for tuning. However, ELM can be combined with other optimization algorithms too. To have a hybrid ELM-GWO algorithm, these steps can be followed:

1. Considering a number of neurons in the hidden layer, develop an SLFN.
2. Initially assign random weights and biases to the SLFN, these values can be in the range of [0, 1] or [− 1, 1].
3. Reform the weights and biases so that they can represent the location of a wolf in the *D*-dimensional space of the problem, where *D* is the total number of weights and biases.
4. For each of the wolves in every iteration, calculate the output weights by employing the ELM algorithm.
5. By having the output weights, the output values can be predicted and the ELM-GWO model is trained. For this purpose, define a fitness function to minimize the error of the model. Herein, the fitness function (E), in terms of root mean squared error ($RMSE$), is proposed as follows:

$$E\left(\vec{w},\vec{b}\right)=\sqrt{\left(\frac{\sum_{i=1}^{n}\left(O_i-P_i\right)^2}{n}\right)} \tag{30}$$

   where $\vec{w}$ and $\vec{b}$ are the vectors of initial weights and biases; n is the total number of training samples; O_i and P_i are the observed value (actual value) and the predicted value in the sample i, respectively.
6. Repeat steps 4 to 6 for a specific number of wolves and iteration until the stopping criteria are satisfied. In this study, the maximum number of iterations and acceptable performance were considered as the stopping criteria.

As can be realized, this process provides the opportunity to evaluate the performance of the ELM model for different initial weights and biases. As a result, input weights and biases can be updated and subsequently, output weights are determined in a way they can lead to a more robust model.

3 Data and preparation

The used data in this investigation were obtained from the literature [72–77]. A dataset containing 798 data points was collected totally. The contents of cement (C), water (W), fly ash (FA), furnace slag (FS), fine aggregates (FAG), coarse aggregates (CAG), superplasticizer (SP), and age (A) have been considered as the inputs of the models, and compressive strength of concrete (f_c') is predicted as the output. Table 1 shows the variation range, average, and standard deviation of each input variable, while the distribution of the variables is illustrated in Fig. 5.

Although the current data can be used in all the proposed models, the performance of the models can be improved if the data are normalized in the range of [0, 1] or [− 1, 1]. This preprocessing on the data can be efficient because, as it was observed previously, the variation range of activation functions or membership functions are usually in these ranges of variation. Therefore, these functions are more sensitive to the input variables which have been normalized prior to training [78]. In order to normalize the inputs in the range of [− 1, 1], the following formulas can be used [21]:

$$X_i=\frac{X_{\text{io}}-X_{\min}}{X_{\max}-X_{\min}}\times 2-1 \tag{31}$$

$$Y_i=\frac{Y_{\text{io}}-Y_{\min}}{Y_{\max}-Y_{\min}}\times 2-1 \tag{32}$$

where X_{io} and X_i are the ith component of each input vector before and after normalization, respectively, and Y_{io} and Y_i are the ith component of the output vector before and after

Table 1 Details of the data

Variable	Minimum	Maximum	Average	St.D
C (kg/m^3)	102.00	505.00	256.60	94.72
FS (kg/m^3)	0.00	359.40	95.38	86.94
FA (kg/m^3)	0.00	200.10	69.94	64.69
W (kg/m^3)	121.75	247.00	179.27	22.33
SP (kg/m^3)	0.00	32.20	7.69	5.53
CAG (kg/m^3)	801.00	1145.00	961.78	74.24
FAG (kg/m^3)	594.00	992.60	773.36	79.67
A (day)	3.00	365.00	41.84	53.90
f_c' (MPa)	2.33	82.60	36.93	16.74

C cement, *FS* furnace slag, *FA* fly ash, *W* water, *SP* superplasticizer, *CAG* coarse aggregate, *FAG* fine aggregate, *A* age, f_c' compressive strength of concrete

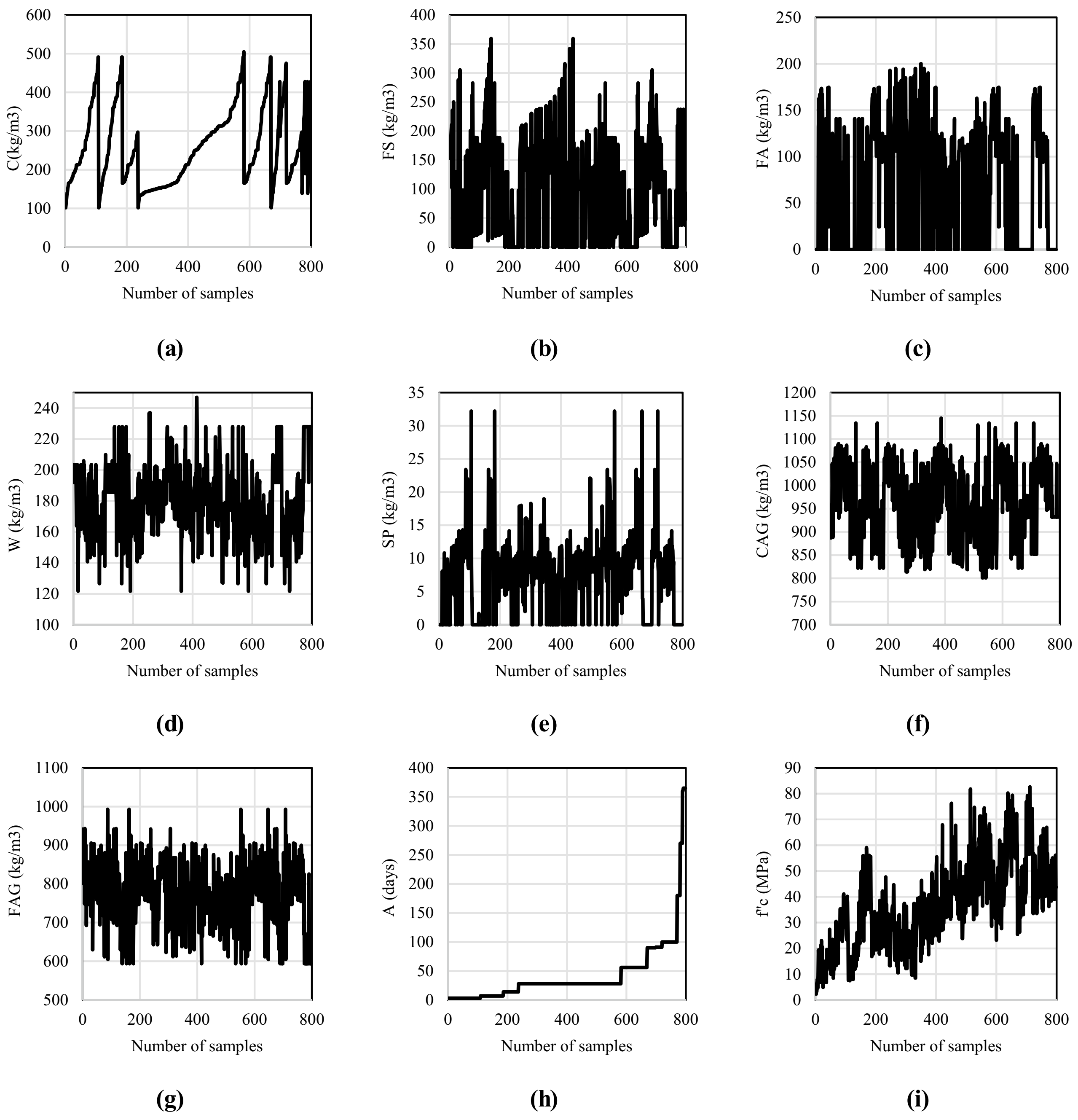


Fig. 5 Distribution pattern of: **a** cement content (C), **b** furnace slag (FS), **c** fly ash (FA), **d** water content (W), **e** superplasticizer (SP), **f** coarse aggregate (CAG), **g** fine aggregate (FAG), **h** age (A), **i** compressive strength (f_c')

normalization, respectively. $X_{\min}$, $X_{\max}$, $Y_{\min}$, and $Y_{\max}$ are the minimum and maximum values of each input and output vector, respectively.

This is also important to be mentioned that since actual values are more tangible, postprocessing has been also conducted in this study and results have been denormalized after the training process and before reporting the final results.

4 Model performance indicators

To evaluate the performance of the models, 70% of the data have been randomly devoted to the training phase and the remained 30% have been assigned to the testing phase. As a primary criterion, statistical model performance indicators including Pearson correlation coefficient (r), determination coefficient (R^2), root mean squared error ($RMSE$), mean absolute error (MAE) are used. These indicators are defined as follows:

$$r=\frac{N\left(\sum_{i=1}^{N}O_i\cdot P_i\right)-\left(\sum_{i=1}^{N}O_i\right)\cdot\left(\sum_{i=1}^{N}P_i\right)}{\sqrt{\left(N\sum_{i=1}^{N}O_i^2-\left(\sum_{i=1}^{N}O_i\right)^2\right)\cdot\left(N\sum_{i=1}^{N}P_i^2-\left(\sum_{i=1}^{N}P_i\right)^2\right)}} \tag{33}$$

$$R^2=\frac{\left[\sum_{i=1}^{N}\left(O_i-\overline{O}\right)\cdot\left(P_i-\overline{P}\right)\right]^2}{\sum_{i=1}^{N}\left(O_i-\overline{O}\right)\cdot\sum_{i=1}^{N}\left(P_i-\overline{P}\right)} \tag{34}$$

$$\mathrm{RMSE}=\sqrt{\sum_{i=1}^{N}\frac{1}{N}\left(O_i-P_i\right)^2} \tag{35}$$

$$\mathrm{MAE}=\frac{1}{N}\sum_{i=1}^{N}\left|O_i-P_i\right| \tag{36}$$

where N is the number of training or testing samples; O_i and P_i are the observed and predicted values in the sample i, respectively; $\overline{O}$ and $\overline{P}$ are also the mean observed and predicted values, respectively.

In addition to the aforementioned formulations, to show the difference between observed and predicted values as the percentage of the mean observed values, relative root mean squared error (RRMSE) and relative mean absolute error (RMAE) are also described in percentage as the following:

$$\mathrm{RRMSE}=\frac{\mathrm{RMSE}}{\overline{O}}\cdot 100 \tag{37}$$

$$\mathrm{RMAE}=\frac{\mathrm{MAE}}{\overline{O}}\cdot 100 \tag{38}$$

Since Eqs. (33–38) are based on the linear relations between observed values (O) and predicted values (P), they can be only sensitive to outliers or extreme values and cannot take into account the additive or proportional differences between the observed (O) and predicted (P) values [79]. To address this issue, Willmott's index [80] ($0 \le WI \le 1.0$) and Nash–Sutcliffe coefficient [81] ($-\infty \le E_{NS} \le 1$) are also defined as follows:

$$E_{NS}=1-\left[\frac{\sum_{i=1}^{N}\left(O_i-P_i\right)^2}{\sum_{i=1}^{N}\left(O_i-\overline{P}\right)^2}\right] \tag{39}$$

$$\mathrm{WI}=1-\left[\frac{\sum_{i=1}^{N}\left(O_i-P_i\right)^2}{\sum_{i=1}^{N}\left(\left|P_i-\overline{O}\right|+\left|O_i-\overline{O}\right|\right)^2}\right] \tag{40}$$

To have a reasonable comparison among the models, all the codes were developed in the MATLAB environment and no external toolbox or compiler was used. Also, the codes were run in a computer system with a processor of the type Intel(R) Core(TM) i7-6700 HQ CPU @ 2.60 GHz 2.59 GHz and 16.0 GB RAM.

5 Results and discussion

Six different models have been considered in this investigation. These models include an ANN, an ANFIS, an ELM, an SVR-Poly, an SVR-RBF, and a hybrid ELM-GWO. Figure 6 briefly shows a flowchart of the models with a focus on the ELM-GWO model. In what follows, the development of the models and results of each model are presented and discussed comprehensively.

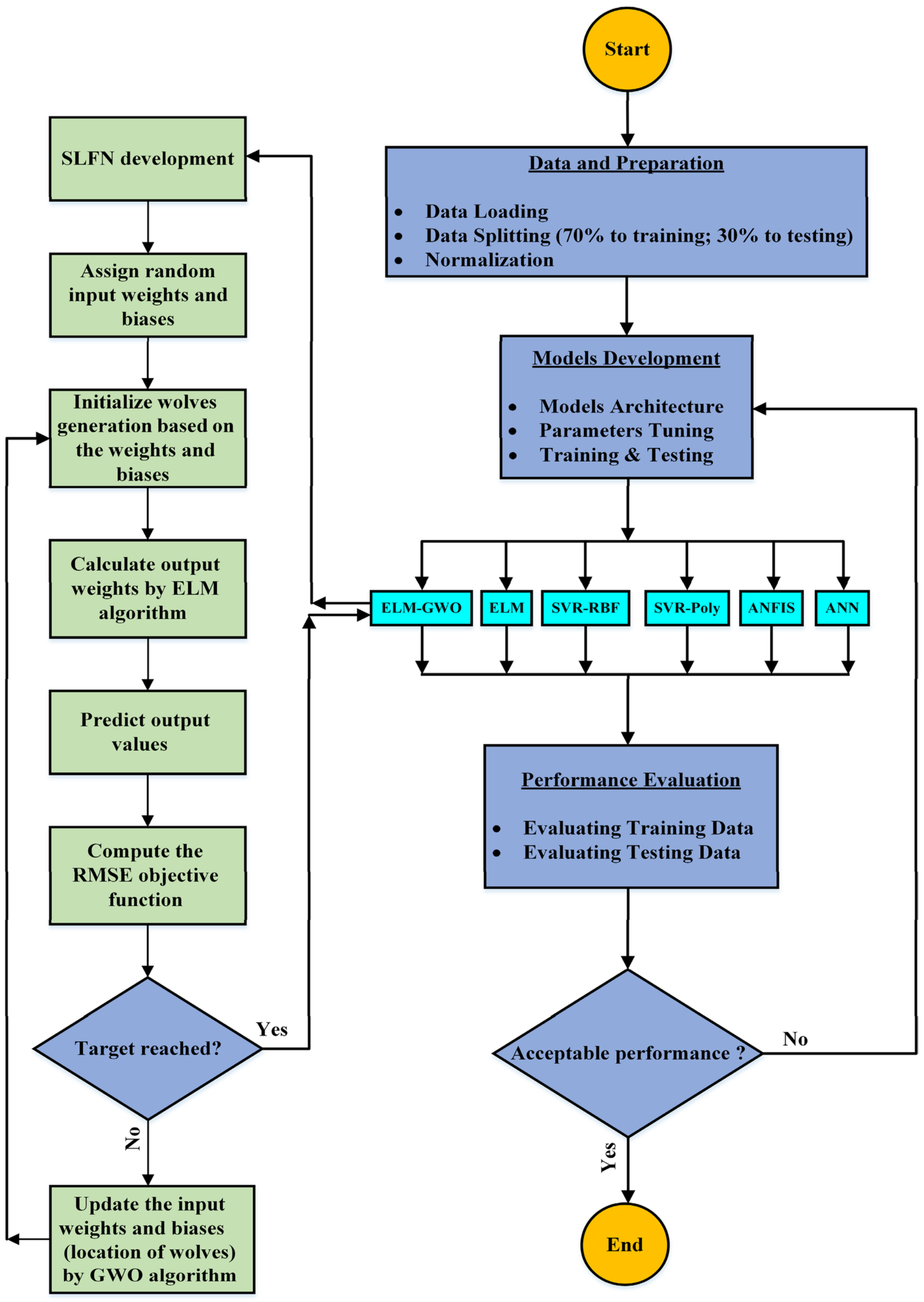
5.1 Models development

5.1.1 ANN development

The performance of an ANN model significantly depends on the architecture of the model, i.e., the number of hidden layers and the number of neurons in each of the layers. To determine the architecture of ANN, a trial and error process was conducted. Different architectures with various number of hidden layers and neurons were developed, and each model was run three times with 1000 epochs. In order to tune the weights and biases of ANN models, the Levenberg–Marquardt algorithm (LMA) was used as it is often the fastest BP algorithm in training [65, 82, 83]. Finally, the mean value of RMSE was obtained as a statistical indicator to show the performance of the models. Table 2 represents a summary of the recorded values of RMSE throughout the trial and error process.

As can be seen in this table, architecture number 3, i.e., a single hidden layer with 8 neurons has shown the lowest value of RMSE in the testing phase. Note that although some of the models could reach lower RMSE values in the training phase, they had not been able to achieve such performance in the testing phase too. Therefore, the architecture with the lowest difference of RMSE in the training phase and testing phase is selected as the most reliable model. Considering these observations, architecture number 3 was adopted as the final architecture as depicted in Fig. 7.

Fig. 6 Flowchart of the models



5.1.2 ANFIS development

To develop an ANFIS model, an initial fuzzy inference system (FIS) should initially be created and then be trained. For this purpose, membership functions (MFs) of the type Takagi–Sugeno were adopted as they generally culminate in more robust results [47, 84–86]. In addition, the default hybrid algorithm of ANFIS was employed in the training phase. To select the number of MFs for each input variable (number of clusters), a trial and error process was carried out and the *RMSE* values were recorded, as shown in Table 3. It was also seen in this procedure that an initial FIS with a degree of 3 and 100 iterations lead to establishing a model with better performance. According to Table 3, in model number 4, not only the lowest value of *RMSE* in the testing phase has been obtained, but also the lowest difference between the *RMSE* values in the training and testing phases has been recorded. Therefore, an ANFIS architecture with 8 MFs for each input variable was selected, as illustrated in Fig. 8.

Table 2 Results of the trial and error process to find the optimum ANN architecture

Architecture No	Number of neurons		RMSE	
	Hidden layer 1	Hidden layer 2	Training phase	Testing phase
1	4	–	5.6	6.23
2	6	–	4.79	5.29
3	**8**	–	**3.77**	**5.14**
4	10	–	4.16	5.49
5	12	–	3.39	5.9
6	14	–	3.098	6.01
7	4	2	4.91	5.64
8	6	2	4.56	5.96
9	8	2	3.68	5.68
10	8	4	3.6	6.05
11	8	6	3.09	6.92
12	10	2	3.23	5.49
13	10	4	3.04	5.96
14	10	6	2.78	6.82
15	10	8	2.41	7.29

*Bold is the best

5.1.3 SVR development

The performance of an SVR model is dependent on the value of involving parameters in the problem. Thus, these parameters need to be determined precisely. As it was mentioned, two SVR models have been considered in this study. The first model is an SVR model with an RBF kernel in which the parameters including C, γ, and ϵ are unknown, and the second one is another SVR model with a polynomial kernel whose unknown parameters are C, d, and ϵ. To determine these parameters, the grid search technique was used in both of the cases and for constant values of ϵ, the impact of different values of the two other parameters on the performance of the models was evaluated in terms of RMSE. Finally, the grid was selected as optimum in which the lowest value of RMSE had been occurred. The results of the grid search algorithm showed that the best performance of the SVR-RBF model is obtained for $C = 1024$ and $\gamma = 0.25$, while the SVR-Poly model shows its best performance when $C = 0.125$ and $d = 4$. The 3D surfaces of the grid search algorithm are also depicted in Fig. 9.

5.1.4 ELM development

ELM algorithm only deals with single-layer feed-forward neural network (SLFN) architectures; thus, the number of hidden layers does not need to be determined and the only unknown variable is the number of neurons in the SLFN. To specify the number of neurons, a trial and error process was conducted and the performance of the ELM model was evaluated in the training and testing phases by the RMSE indicator. Table 4 shows the considered models in the trial and error process with their corresponding RMSE values. As can be seen in this table, the lowest RMSE value has been obtained in model number 11 in which 110 neurons have been considered in the hidden layer. Therefore, an SLFN

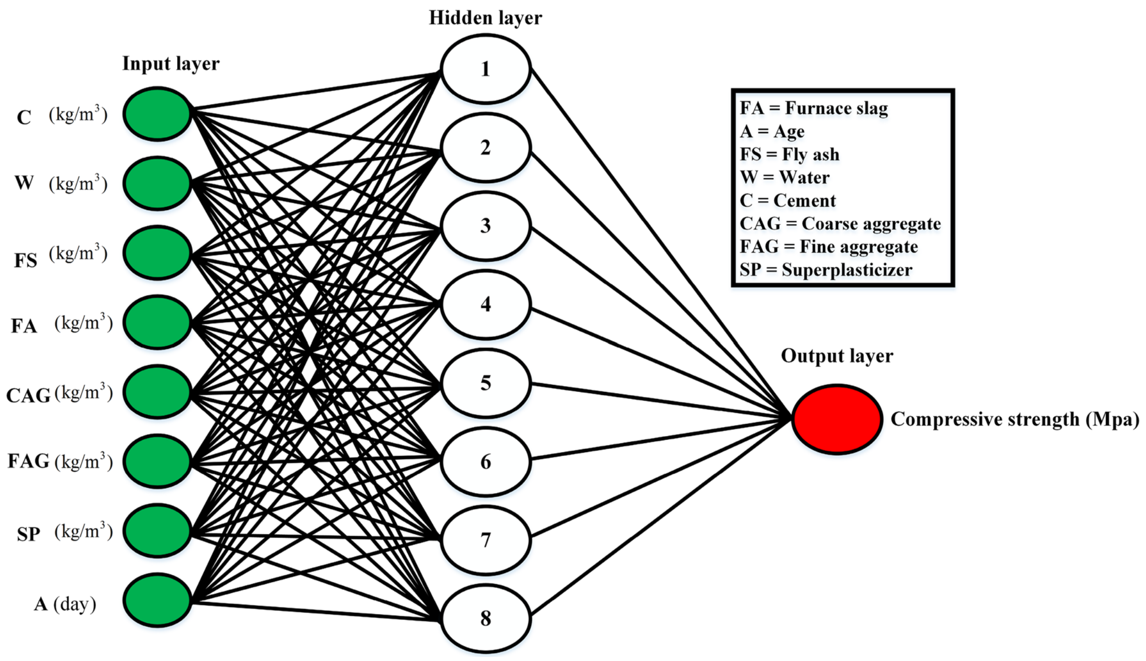


Fig. 7 Considered ANN architecture

Table 3 Performance evaluation of ANFIS with different numbers of MFs

Model no	Number of MFs	RMSE	
		Training phase	Testing phase
1	2	6.28	7.15
2	4	4.89	6.42
3	6	3.92	5.68
4	**8**	**3.42**	**5.48**
5	10	3.01	5.99
6	12	2.91	7.32
7	14	2.68	8.32

*Bold is the best

with 110 neurons in the SLFN was developed, as shown in Fig. 10.

5.1.5 ELM-GWO development

The architecture of the ELM-GWO was considered the same as the ELM (i.e., 110 hidden neurons in the SLFN) so that a reasonable comparison between the ELM and the hybrid ELM-GWO could be conducted. One of the most considerable advantages of the GWO algorithm is that it is not dependent on many parameters and the only parameter which should be tuned is the number of the grey wolves' population. For this purpose, the ELM-GWO algorithm was run for a different number of populations and the convergence rate of the algorithm and the required time for convergence were recorded. Figure 11a illustrates the convergence curves in different populations of wolves for 1000 iterations. As can be seen in this diagram, the number of the population has not significantly affected the fitness value (*RMSE*). Figure 11b also shows the variation of the best fitness value (i.e., the fitness value after 1000 iterations) and the time required to reach this value in the different number of populations. As can be realized, in population number 75, not only the lowest value of *RMSE* has been obtained, but also the required time for convergence is less than those of the higher populations. These observations caused the number of 75 wolves were considered in the ELM-GWO model.

5.2 Comparison of the results and discussion

After tuning the involving parameters in all the models, each model was run and its performance in terms of previously mentioned performance metrics was evaluated. Table 5 shows the obtained performance indices in the training phase of all the six models. As can be seen in this table, all the models have been capable of reaching satisfying performance indices by resulting in high values (close to one) of *r*, R^2, NSE, and WI, and low values (close to zero) of RMSE, MAE, RRMSE, and RMAE. However, the best performance metrics have resulted in the case of the ELM-GWO model and this reveals that the ELM-GWO model has been more successful in the training phase in comparison with the other models. After the ELM-GWO model, other models have had a very close competition. If the other 5 models (standard models) are compared in the terms of *r*, R^2, RMSE, RRMSE,

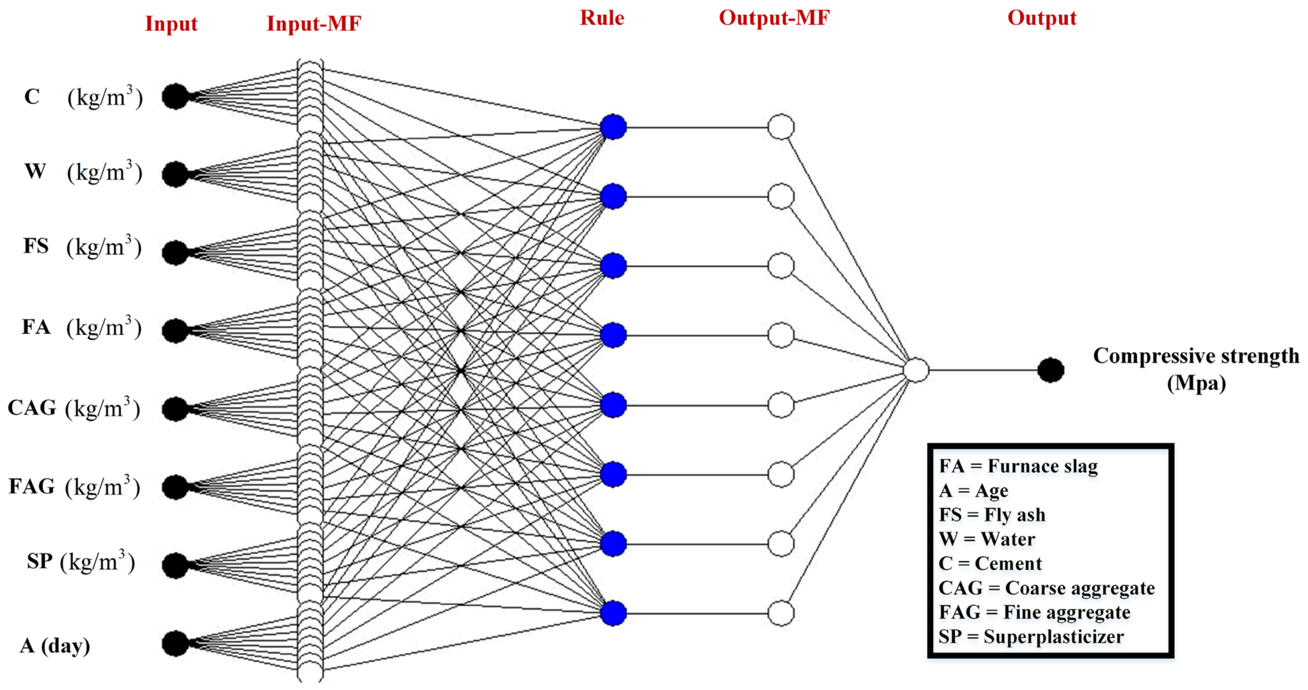


Fig. 8 Considered ANFIS architecture

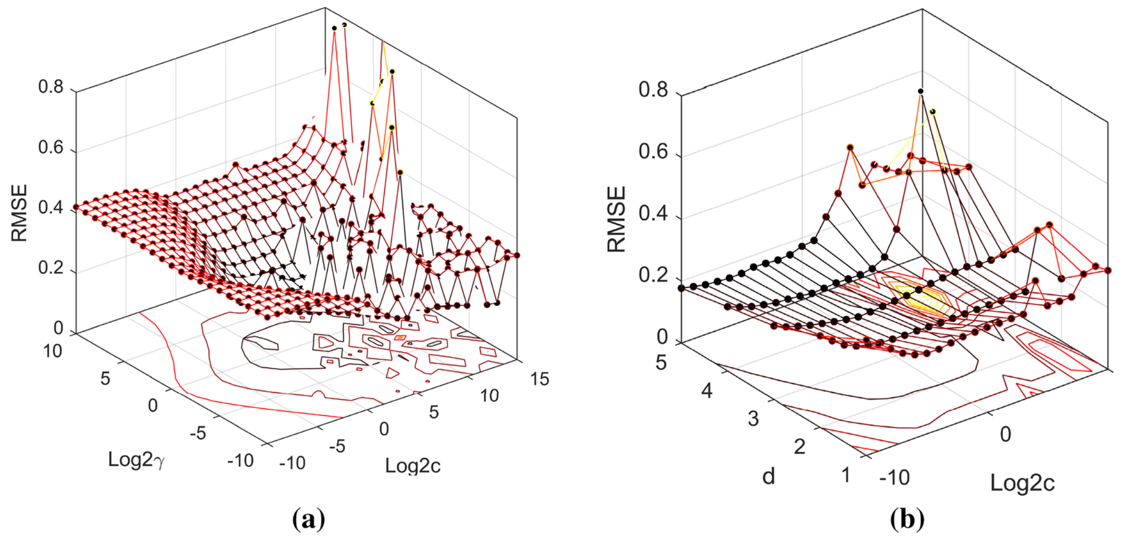


Fig. 9 3D graphical diagrams for tuning the parameters of SVR with: **a** RBF kernel, **b** polynomial kernel

Table 4 Results of the trial and error process to determine the number of neurons in the ELM model

Model no	Number of neurons	RMSE	
		Training phase	Testing phase
1	10	10.96	10.71
2	20	10.15	10.29
3	30	8.05	9.24
4	40	7.00	8.49
5	50	7.26	8.85
6	60	6.66	8.09
7	70	6.03	7.95
8	80	5.72	7.67
9	90	5.68	7.54
10	100	5.37	7.31
11	**110**	**4.92**	**6.92**
12	120	4.73	7.54
13	130	4.68	7.95
14	140	4.51	8.78
15	150	4.34	7.81

*Bold is the best

and WI, it can be concluded that the SVR-RBF model has been able to achieve the best performance. However, if the 5 models are compared in the terms of performance indices such as MAE, RMAE, and NSE, it can be concluded the ANFIS model has been more successful in the training phase. These cases imply that although the magnitude of errors in the SVR-RBF model is lower, the mean of errors in the ANFIS model is closer to zero. The same case can be also seen in the comparison of ANN with SVR-Poly. The other point which can be mentioned is related to the performance metrics of the ELM model which vindicate the worse performance of the ELM model compared to other models. If the performance of the ELM model (i.e., the worst model) is compared to that of the ELM-GWO model (i.e., the best model), it can be concluded that tuning the initial weights and biases of an SLFN in the ELM algorithm can be highly effective in improving the performance of the model.

Figure 12 demonstrates the regression diagrams of all the models in the training phase. The horizontal axis of each diagram represents the observed values in the training samples, while the vertical axis shows the predicted values by the model. The blue line in each diagram is also the line with 100% agreement between the observed and predicted values, and other radial lines are the lines with 15% and 30% difference from the blue line. Accordingly, if all the points are placed on the blue line with the equation of $y = x$, it means that the model has been able to predict the actual values without any error. As can be observed in this figure, the ELM-GWO model not only has the highest value of R^2 but it also has the most similar equation to $y = x$. After this model, ANFIS, SVR-RBF, ANN, SVR-Poly, and ELM could reach the best values of R^2.

Although having a proper performance in the training phase will help models in predicting the targets, it does not guarantee the performance of the models in the testing phase too. In other words, models might not be able to repeat their luminous results in the testing phase too; thus, their performance needs to be tested. Table 6 illustrates the performance indices of the models in the testing phase. As can be observed in this table, the ELM-GWO model has been capable of reaching the best performance indices among the other models by resulting in higher values of r, R^2, NSE,

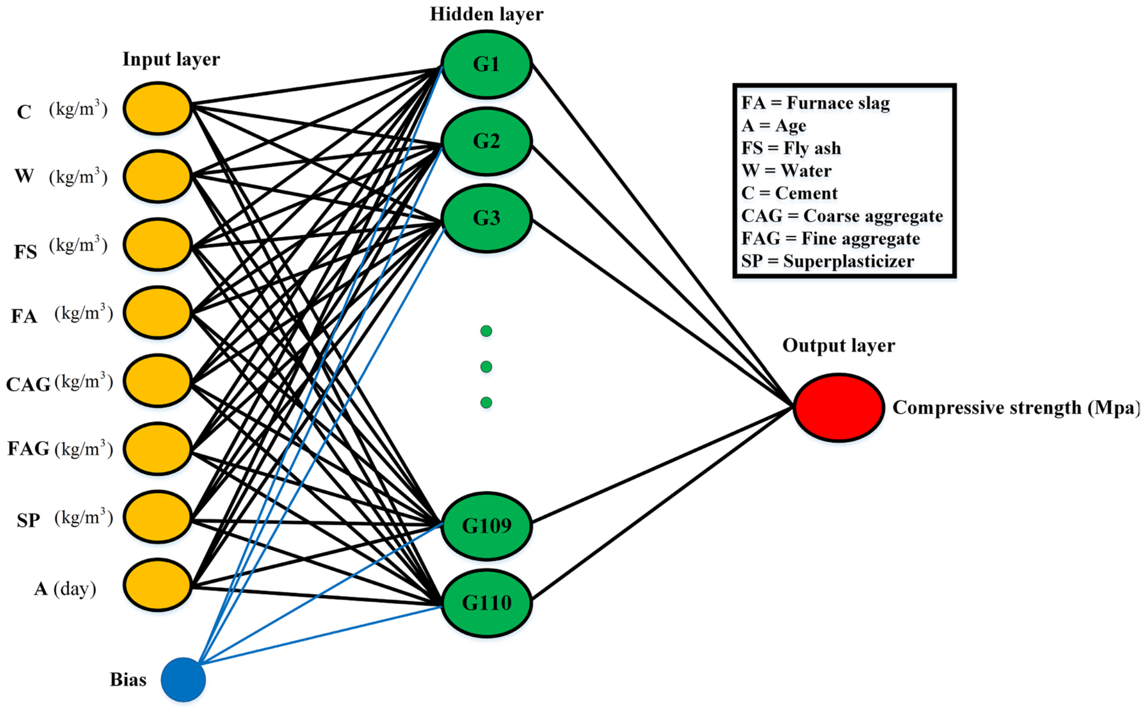


Fig. 10 Considered ELM architecture

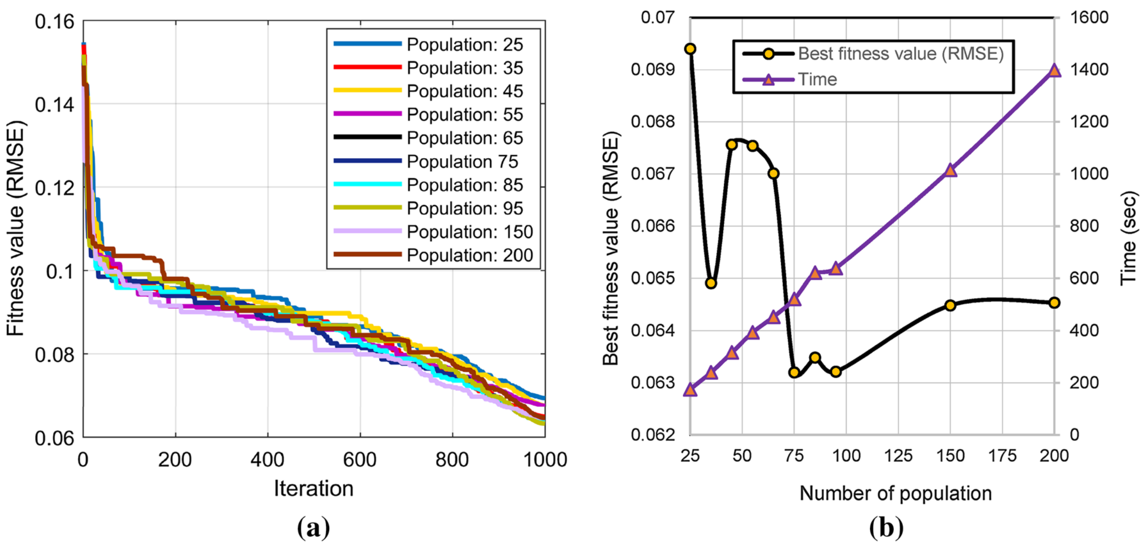


Fig. 11 ELM-GWO development: **a** the impact of number of population on the fitness value, **b** comparison of the required time for convergence and the best fitness value

and WI, and lower values of RMSE, MAE, RRMSE, and RMAE. After this model, in contrast to the training phase, the ANN model has shown the best performance by resulting in superior indices among the other models. This means that after the ELM-GWO model, the ANN model is the most reliable model for predicting new targets. This is important to be mentioned that the ELM-GWO is almost an ANN with a single hidden layer structure whose weights and biases have been determined by another optimization algorithm (i.e., GWO instead of a commonly used backpropagation (BP) algorithm) combined with Moore–Penrose generalized inverse matrix. Therefore, an ELM-GWO model not only incorporates an ANN in its structure but also uses mathematical theories to better arrange the value of weights and biases.

Table 5 Performance evaluation of the ML models in the training phase

ML models	*r*	R^2	RMSE	MAE	RRMSE (%)	RMAE (%)	NSE	WI
ANN	0.9679	0.9369	4.1887	3.1602	11.3904	8.5935	0.9369	0.9834
ANFIS	0.9754	0.9513	3.6785	2.7544	10.0029	7.4901	0.9513	0.9874
SVR-RBF	0.9757	0.9519	3.6568	3.0916	9.9432	8.4062	0.9502	0.9876
SVR-Poly	0.9692	0.9393	4.1110	3.3576	11.1348	9.0943	0.9341	0.9840
ELM	0.9583	0.9183	4.7659	3.6913	12.9599	10.0377	0.9110	0.9783
ELM-GWO	**0.9876**	**0.9753**	**2.6223**	**1.8204**	**7.1307**	**4.9502**	**0.9746**	**0.9937**

*Bold is the best

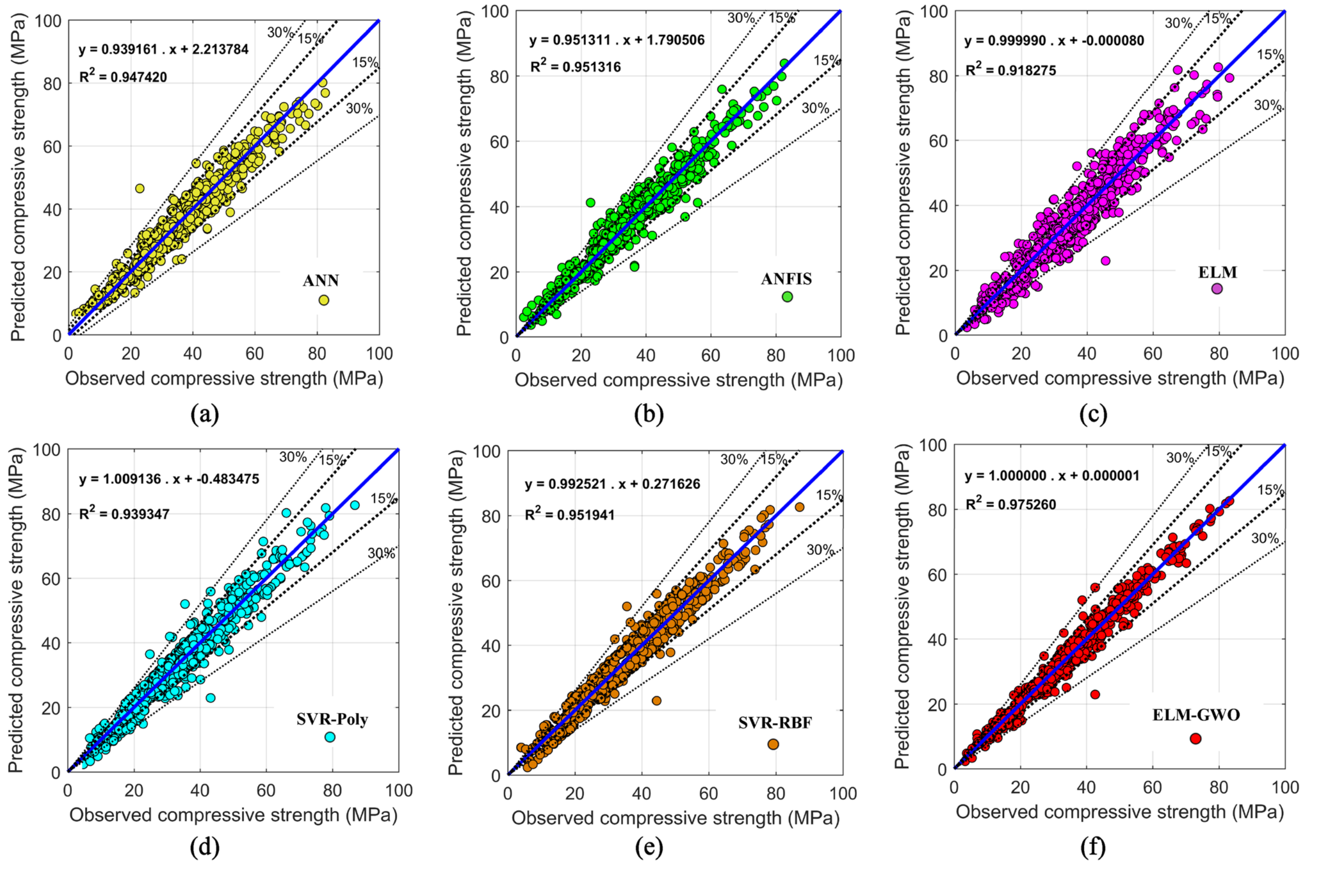


Fig. 12 Regression diagrams of the models in the training phase: **a** ANN model, **b** ANFIS model, **c** ELM model, **d** SVR-Poly model; **e** SVR-RBF model, **f** ELM-GWO model

Table 6 Performance evaluation of the ML models in the testing phase

ML Models	*r*	R^2	RMSE	MAE	RRMSE (%)	RMAE (%)	NSE	WI
ANN	0.9434	0.8900	5.6069	4.3300	15.0331	11.6095	0.8895	0.9704
ANFIS	0.9409	0.8854	5.7541	4.3023	15.4278	11.5351	0.8838	0.9690
SVR-RBF	0.9335	0.8714	6.1206	4.7612	16.6566	12.9571	0.8627	0.9654
SVR-Poly	0.9315	0.8677	6.1609	4.9026	16.7600	13.3369	0.8494	0.9636
ELM	0.9195	0.8456	6.6961	5.0419	17.9534	13.5182	0.8426	0.9578
ELM-GWO	**0.9686**	**0.9381**	**4.1963**	**3.3502**	**11.2510**	**8.9825**	**0.9340**	**0.9838**

*Bold is the best

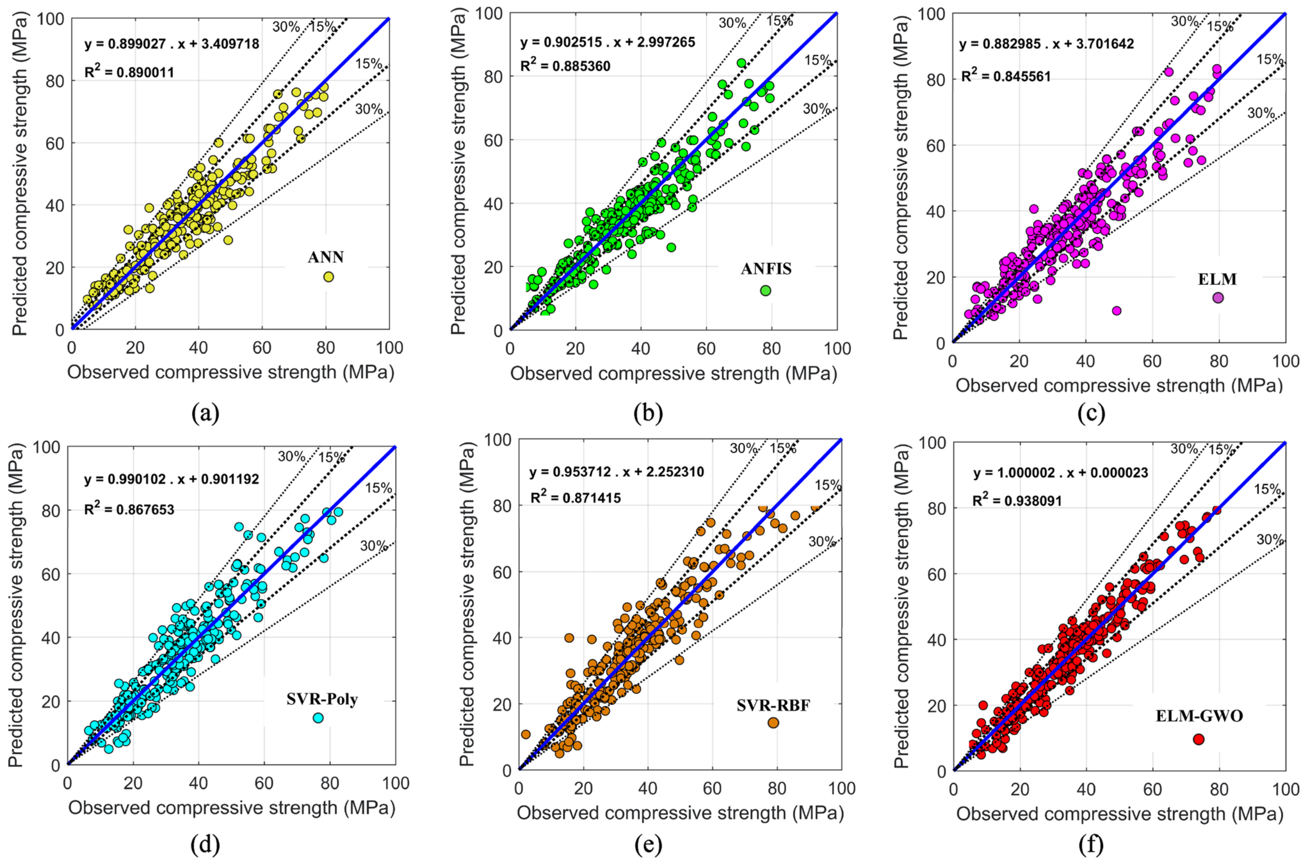


Fig. 13 Regression diagrams of the models in the testing phase: **a** ANN model, **b** ANFIS model, **c** ELM model, **d** SVR-Poly model, **e** SVR-RBF model, **f** ELM-GWO model

Figure 13 shows the regression diagrams of the models in the testing phase. The better performance of the ELM-GWO model can be realized as it has the highest value of R^2 and its data points in the diagram have been gathered along the blue line (the line with 100% agreement) more compactly. After the ELM-GWO model, the better performance in terms of R^2 can be found in the diagram of ANN, ANFIS, SVR-RBF, SVR-Poly, and ELM model, respectively.

To clearly show the performance of the models in the testing phase, Fig. 14 has also been drawn in which the capability of the models in predicting each of the testing samples has been shown. As can be seen in this figure, all the models have been able to predict most of the testing samples closely; however, the better performance of the ELM-GWO and the worse performance of the ELM are almost obvious. To evaluate the performance of the models more comprehensively, graphical Taylor diagrams are also presented in Fig. 15. The horizontal and vertical axes of these diagrams show the values of standard deviation which are connected to each other by circular lines. The drawn blue radial lines from the origin of coordinates show the value of the correlation coefficient as a performance indicator, and the green circular lines show the value of RMSE as another performance indicator. In this diagram, the observed data are assumed as a base model with zero error (i.e., RMSE = 0), the highest correlation coefficient (i.e., $r = 1$), and a calculated value of standard deviation. Then, the performance of other models in terms of standard deviation, *RMSE*, and r is compared with those of the observed data. Accordingly, the best model will be the model with the highest similarity to the base model of the observed data. As can be seen in this figure, the ELM-GWO model has been able to take a closer position to the observed data in both of the training and testing phases and this, in turn, illustrates the more successful performance of this model. Also, it can be observed that, on the one hand, the performance of the SVR-RBF model is highly similar to that of the SVR-Poly and, on the other hand, the performance of the ANN and ANFIS models is almost alike. This can be related to the origin of the models as the SVR-RBF and SVR-Poly models are from the SVR family and the ANFIS model is an ANN model which has been combined with fuzzy rule principles. However, a significant difference

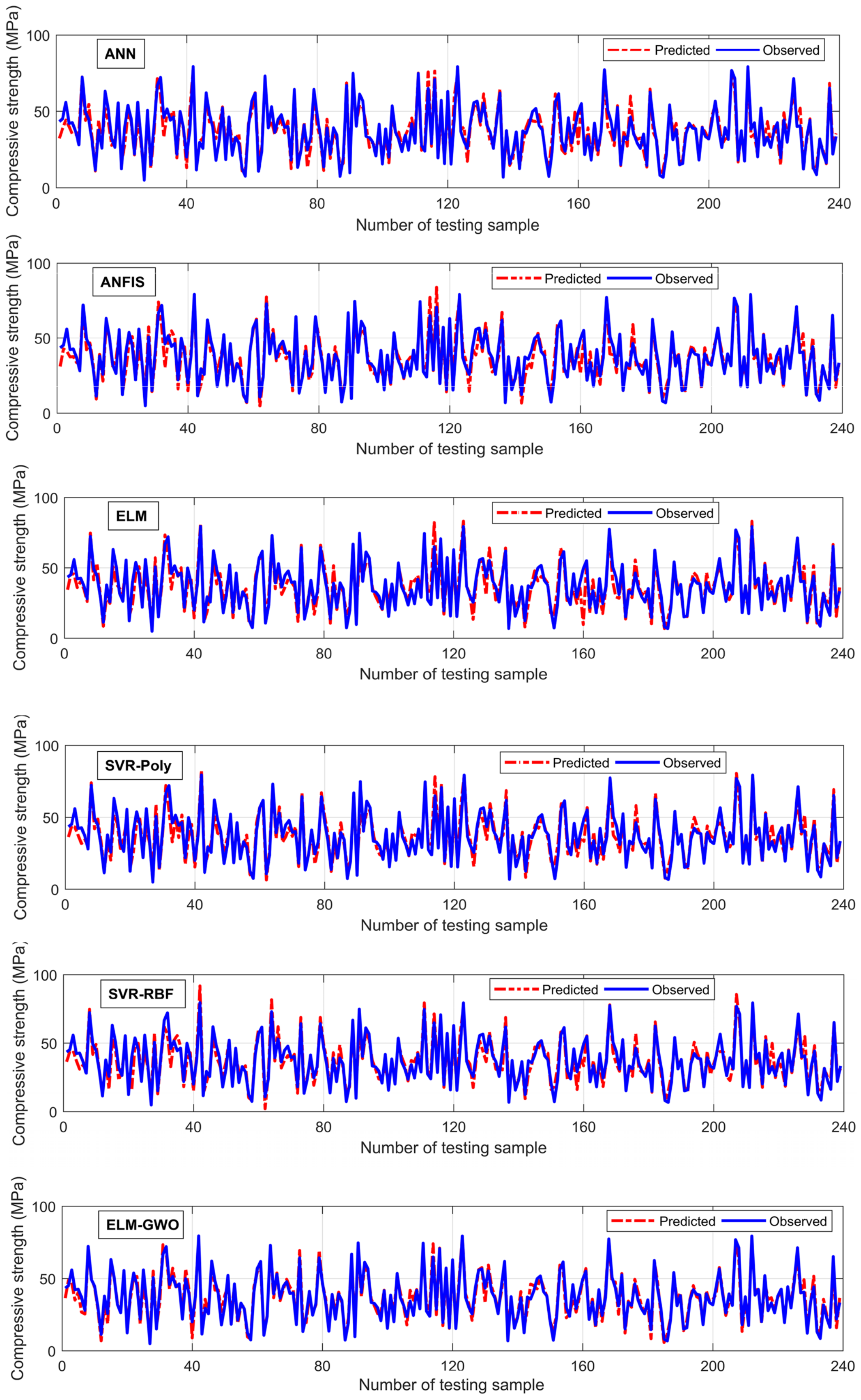
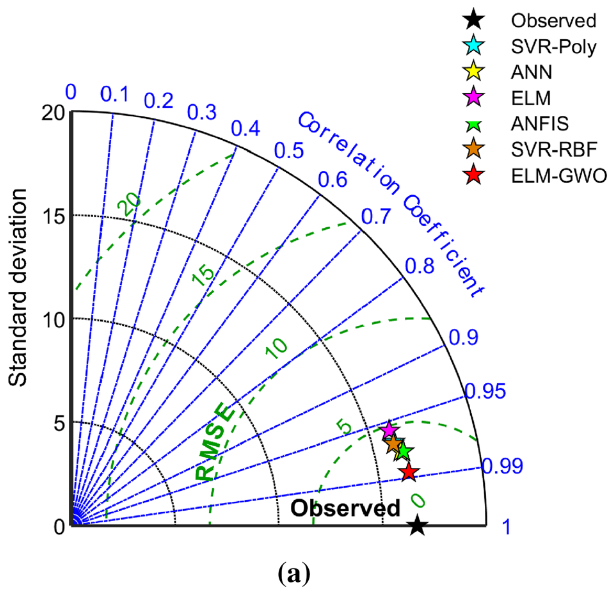


Fig. 14 Compressive strength prediction in the testing phase by different ML models



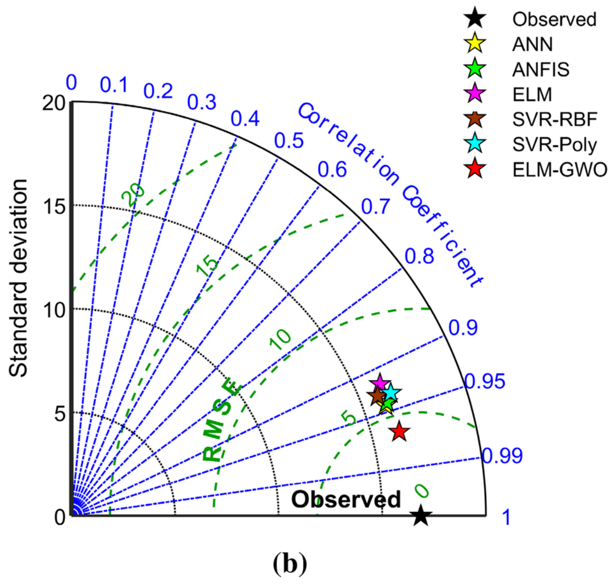


Fig. 15 Graphical Taylor diagrams for comparison of the ML models: **a** training phase, **b** testing phase

can be found between the ELM and ELM-GWO models. To exhibit this difference, the ratio of the predicted values by both of the models to the observed values in each of training and testing samples was calculated, and then, the results were prepared in the form of diagrams as shown in Fig. 16. The vertical axis of these diagrams is the ratio of the predicted to the output values, and the horizontal axis shows the training or testing samples. The more the data points in these diagrams concentrate on the blue line with the ratio of one, the more precise the corresponding model is. As can be seen in this figure, the data points of the ELM-GWO model have concentrated more compactly around the blue line, while less concentration can be seen in the ELM model. These, all take together, reveal that the performance of an ELM algorithm can be efficiently improved if the initial weights and biases of SLFN are tuned by other algorithms.

The other important parameter of the models which can be compared is the time that models required to be trained. As mentioned in the previous sections, no external compiler or toolbox was used in the developing of the models and the same computer system was used to running them. Therefore, the required time of the models for training can be compared in an equal condition. Table 7 shows the recorded time values from the start of the program to the end. As can be seen, the fastest performance in the training phase belongs to the ELM, while the slowest one is related to the ELM-GWO model. In other words, the required time for the ELM-GWO model is reluctantly 300 times more than that of the ELM model. This shows that using an evolutionary algorithm in the structure of ELM can severely increase the training time.

6 Conclusion

Concrete is a profitable material that plays a significant role in the construction industry. Partial replacement of cement in concrete with other pozzolans such as fly ash (FA) and furnace slag (FS) addresses not only a way to dispose of waste materials but also a mean to reduce the adverse by-products of cement production. However, estimating the properties of hardened concrete, on top of that compressive strength, is not easy at all and needs more advanced techniques. The main motivation of the current paper was to employ a soft computing approach to predict the compressive strength of hardened concrete in which cement has been partially replaced with FA and FS. For this purpose, an extreme learning machine (ELM) was combined with an inspired metaheuristic algorithm by the social behavior of grey wolves, known as grey wolf optimizer (GWO), and a novel ELM-GWO model was proposed. The predictability of the proposed model was validated against well-established nonlinear predictive models such as an artificial neural network (ANN), an adaptive neuro-fuzzy inference system (ANFIS), an standard ELM, and two support vector regression models with different kernel functions (i.e., radial basis function (RBF) and polynomial (Poly)). Different performance indices were defined and calculated for each of the models, and then, the models were assessed statistically. The results indicated that all the proposed models can predict the compressive strength of concrete satisfactory, thus eliminating the need for conducting costly experiments and saving time. In addition, it was seen that although the ELM model had the worst performance among the models, the

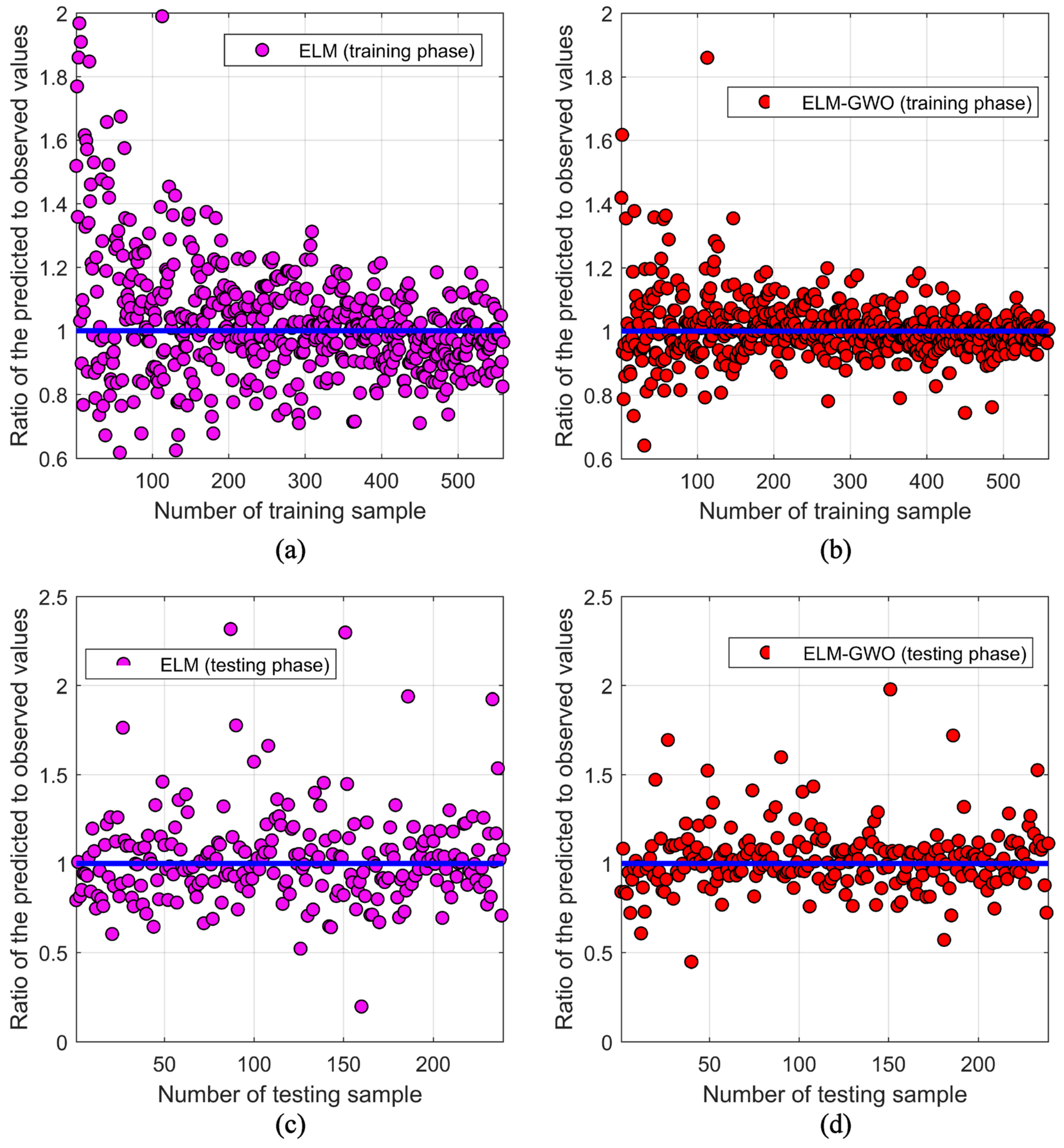


Fig. 16 Comparison of the ELM model with the ELM-GWO model: **a** training phase of ELM, **b** training phase of ELM-GWO, **c** testing phase of ELM, **d** testing phase of ELM-GWO

Table 7 Required time for training

Models	ANN	ANFIS	SVR-Poly	SVR-RBF	ELM	ELM-GWO
Time (s)	6.127	16.787	2.827	3.035	**1.716**	522.501

*Bold is the best

ELM-GWO model could reach the best performance indices. Therefore, the hybridization of ELM with GWO can be very efficient in improving the performance of the ELM model. It was also observed that ANN could provide a more reliable model in comparison with other standard ML models as it showed a better performance in the testing phase.

Although in this research, the application of a hybrid ELM-GWO model was evaluated for the first time and considerable improvements in the accuracy of the ELM model were seen, the required time for the training of the model increased severely. This enhancement in time was principally due to employing an evolutionary algorithm (i.e., GWO) in the structure of ELM and as is known, evaluation requires time to achieve. Therefore, time enhancement is inevitable in evolutionary algorithms. In the future study, the hybridization of the ELM algorithm with more advanced algorithms will be investigated so that the associated problem with the time requirement of the model can be addressed. Also, the application of this hybrid model in the behavior prediction of other structural components will be examined.

References

1. Topcu IB, Sarıdemir M (2008) Prediction of compressive strength of concrete containing fly ash using artificial neural networks and fuzzy logic. Comput Mater Sci 41(3):305–311
2. Shariati M et al (2019) Experimental investigation on the effect of cementitious materials on fresh and mechanical properties of self-consolidating concrete. Adv Concr Constr 8(3):225–237
3. Chopra P, Sharma RK, Kumar M (2016) Prediction of compressive strength of concrete using artificial neural network and genetic programming. Adv Mater Sci Eng
4. Toghroli A et al (2014) Prediction of shear capacity of channel shear connectors using the ANFIS model. Steel Compos Struct 17(5):623–639
5. Naderpour H, Rafiean AH, Fakharian P (2018) Compressive strength prediction of environmentally friendly concrete using artificial neural networks. J Build Eng 16:213–219
6. Antiohos S et al (2007) Improving the performance of ternary blended cements by mixing different types of fly ashes. Cem Concr Res 37(6):877–885
7. Abdelkader B, El-Hadj K, Karim E (2010) Efficiency of granulated blast furnace slag replacement of cement according to the equivalent binder concept. Cem Concr Compos 32(3):226–231
8. ASTMC (2012) Standard specification for coal fly ash and raw or calcined natural pozzolan for use in concrete. ASTM Int, West Conshohocken
9. Faleschini F et al (2015) Valorization of co-combustion fly ash in concrete production. Mater Des 85:687–694
10. Li G, Zhao X (2003) Properties of concrete incorporating fly ash and ground granulated blast-furnace slag. Cem Concr Compos 25(3):293–299
11. Suresh D, Nagaraju K (2015) Ground granulated blast slag (GGBS) in concrete–a review. IOSR J Mech Civil Eng 12(4):76–82
12. Aziz MA-E, Aleem SAE, Heikal M (2012) Physico-chemical and mechanical characteristics of pozzolanic cement pastes and mortars hydrated at different curing temperatures. Constr Build Mater 26(1):310–316
13. Özbay E, Erdemir M, Durmuş Hİ (2016) Utilization and efficiency of ground granulated blast furnace slag on concrete properties—a review. Constr Build Mater 105:423–434
14. Bınıci H et al (2012) Investigation of durability properties of concrete pipes incorporating blast furnace slag and ground basaltic pumice as fine aggregates. Scientia Iranica 19(3):366–372
15. Douglas E, Pouskouleli G (1991) Prediction of compressive strength of mortars made with portland cement-blast-furnace slag-fly ash blends. Cem Concr Res 21(4):523–534
16. Toghroli A et al (2020) Evaluating the use of recycled concrete aggregate and pozzolanic additives in fiber-reinforced pervious concrete with industrial and recycled fibers. Constr Build Mater 252:118997
17. Gao W et al (2019) Development of a novel soft-computing framework for the simulation aims: a case study. Eng Comput 35(1):315–322
18. Singh J et al (2016) A study of soft computing models for prediction of longitudinal wave velocity. Arab J Geosci 9(3):1–11
19. Öztaş A et al (2006) Predicting the compressive strength and slump of high strength concrete using neural network. Constr Build Mater 20(9):769–775
20. Alshihri MM, Azmy AM, El-Bisy MS (2009) Neural networks for predicting compressive strength of structural light weight concrete. Constr Build Mater 23(6):2214–2219
21. Prasad BR, Eskandari H, Reddy BV (2009) Prediction of compressive strength of SCC and HPC with high volume fly ash using ANN. Constr Build Mater 23(1):117–128
22. Khademi F et al (2017) Multiple linear regression, artificial neural network, and fuzzy logic prediction of 28 days compressive strength of concrete. Front Struct Civ Eng 11(1):90–99
23. Yaseen ZM et al (2018) Predicting compressive strength of lightweight foamed concrete using extreme learning machine model. Adv Eng Softw 115:112–125
24. Al-Shamiri AK et al (2019) Modeling the compressive strength of high-strength concrete: an extreme learning approach. Constr Build Mater 208:204–219
25. Han I-J et al (2019) Learned prediction of compressive strength of GGBFS concrete using hybrid artificial neural network models. Materials 12(22):3708
26. Golafshani EM, Behnood A, Arashpour M (2020) Predicting the compressive strength of normal and high-performance concretes using ANN and ANFIS hybridized with Grey Wolf Optimizer. Constr Build Mater 232:117266
27. Sun L et al (2019) Applying a meta-heuristic algorithm to predict and optimize compressive strength of concrete samples. Eng Comput 35:1–13
28. Ashrafian A et al (2020) Compressive strength of foamed cellular lightweight concrete simulation: new development of hybrid artificial intelligence model. Constr Build Mater 230:117048
29. Priddy KL, Keller PE (2005) Artificial neural networks: an introduction, vol 68. SPIE Press, Bellingham
30. Shariati A et al (2020) On transient hygrothermal vibration of embedded viscoelastic flexoelectric/piezoelectric nanobeams under magnetic loading. Adv Nano Res 8(1):49–58
31. Amiri M et al (2016) A new combination of artificial neural network and K-nearest neighbors models to predict blast-induced ground vibration and air-overpressure. Eng Comput 32(4):631–644
32. Shariati M et al (2020) Prediction of concrete strength in presence of furnace slag and fly ash using Hybrid ANN-GA (Artificial Neural Network-Genetic Algorithm). Smart Struct Syst 25(2):183
33. Jain AK, Mao J, Mohiuddin KM (1996) Artificial neural networks: a tutorial. Computer 29(3):31–44
34. Moosazadeh S et al (2018) Prediction of building damage induced by tunnelling through an optimized artificial neural network. Eng Comput 2:579–591
35. Shariati M et al (2019) Application of a hybrid artificial neural network-particle swarm optimization (ANN-PSO) model in behavior prediction of channel shear connectors embedded in normal and high-strength concrete. Appl Sci 9(24):5534
36. Karlik B, Olgac AV (2011) Performance analysis of various activation functions in generalized MLP architectures of neural networks. Int J Artif Intell Expert Syst 1(4):111–122

37. Mansouri I et al (2017) Analysis of influential factors for predicting the shear strength of a V-shaped angle shear connector in composite beams using an adaptive neuro-fuzzy technique. J Intell Manuf, 1–11
38. Safa M et al (2016) Potential of adaptive neuro fuzzy inference system for evaluating the factors affecting steel-concrete composite beam's shear strength. Steel Compos Struct 21(3):679–688
39. Shariati A et al (2020) Effect of residual surface stress on parametrically excited nonlinear dynamics and instability of viscoelastic piezoelectric nanoelectromechanical resonators. Eng Comput 36:1–16
40. Shariati A et al On the nonlinear dynamics of viscoelastic graphene sheets conveying nanoflow: parametric excitation analysis. Mech Des Struct Mach, 1–18
41. Jang J-S (1993) ANFIS: adaptive-network-based fuzzy inference system. IEEE Trans Syst Man Cybern 23(3):665–685
42. Petković D et al (2014) Adaptive neuro-fuzzy maximal power extraction of wind turbine with continuously variable transmission. Energy 64:868–874
43. Petković D et al (2012) Adaptive neuro-fuzzy estimation of conductive silicone rubber mechanical properties. Expert Syst Appl 39(10):9477–9482
44. Jovic S et al (2019) Potential of adaptive neuro-fuzzy methodology for investigation of heat transfer enhancement of a minichannel heat sink. Phys A Stat Mech Appl 523:516–524
45. Jovic S et al (2019) Analysing of exchange rate and gross domestic product (GDP) by adaptive neuro fuzzy inference system (ANFIS). Phys A Stat Mech Appl 513:333–338
46. Khajeh A, Modarress H, Rezaee B (2009) Application of adaptive neuro-fuzzy inference system for solubility prediction of carbon dioxide in polymers. Expert Syst Appl 36(3):5728–5732
47. Mayilvaganan MK, Naidu K (2011) Comparison of membership functions in adaptive-network-based fuzzy inference system (ANFIS) for the prediction of groundwater level of a watershed. J Comput Appl Res Dev 1:35–42
48. Shariati M et al (2020) Identification of the most influencing parameters on the properties of corroded concrete beams using an adaptive neuro-fuzzy inference system (ANFIS). Steel Compos Struct 34(1):155
49. Naghipour M et al (2020) Effect of progressive shear punch of a foundation on a reinforced concrete building behavior. Steel Compos Struct 35(2):279–294
50. Shariati M et al (2020) Monotonic behavior of C and L shaped angle shear connectors within steel-concrete composite beams: an experimental investigation. Steel Compos Struct 35(2):237–247
51. Shariati A et al (2020) Investigation of microstructure and surface effects on vibrational characteristics of nanobeams based on nonlocal couple stress theory. Adv Nano Res 8(3):191–202
52. Shariati A et al (2020) Nonlinear dynamics and vibration of reinforced piezoelectric scale-dependent plates as a class of nonlinear Mathieu-Hill systems: parametric excitation analysis. Eng Comput 36:1–17
53. Toghroli A et al (2018) Evaluation of the parameters affecting the Schmidt rebound hammer reading using ANFIS method. Comput Concrete 21(5):525–530
54. Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20(3):273–297
55. Vapnik V, Golowich SE, Smola A (1996) Support vector method for function approximation, regression estimation, and signal processing. In: Advances in neural information processing systems, vol 9. Citeseer
56. Safa M et al (2020) Development of neuro-fuzzy and neuro-bee predictive models for prediction of the safety factor of eco-protection slopes. Stat Mech Appl Phys A 2:124046
57. Sedghi Y et al (2018) Application of ANFIS technique on performance of C and L shaped angle shear connectors. Smart Struct Syst 22(3):335–340
58. Sadeghipour Chahnasir E et al (2018) Application of support vector machine with firefly algorithm for investigation of the factors affecting the shear strength of angle shear connectors. Smart Struct Syst 22(4):413–424
59. Hussain M et al (2011) A comparison of SVM kernel functions for breast cancer detection. In: 2011 eighth international conference computer graphics, imaging and visualization. IEEE
60. Huang G-B, Zhu Q-Y, Siew C-K (2006) Extreme learning machine: theory and applications. Neurocomputing 70(1):489–501
61. Huang G-B (2003) Learning capability and storage capacity of two-hidden-layer feedforward networks. IEEE Trans Neural Netw 14(2):274–281
62. Shariati M et al (2019) Application of extreme learning machine (ELM) and genetic programming (GP) to design steel-concrete composite floor systems at elevated temperatures. Steel Compos Struct 33(3):319–332
63. Shariati M et al (2020) A novel approach to predict shear strength of tilted angle connectors using artificial intelligence techniques. Eng Comput 36:1–21
64. Mansouri I et al (2019) Analysis of influential factors for predicting the shear strength of a V-shaped angle shear connector in composite beams using an adaptive neuro-fuzzy technique. J Intell Manuf 30(3):1247–1257
65. Toghroli A (2015) Applications of the ANFIS and LR Models in the prediction of shear connection in composite beams. Jabatan Kejuruteraan Awam, Fakulti Kejuruteraan, Universiti Malaya
66. Katebi J et al (2019) Developed comparative analysis of metaheuristic optimization algorithms for optimal active control of structures. Eng Comput 35:1–20
67. Armaghani DJ et al (2020) Hybrid ANN-based techniques in predicting cohesion of sandy-soil combined with fiber. Geomech Eng 20(3):191
68. Shariati M et al (2019) Moment-rotation estimation of steel rack connection using extreme learning machine. Steel Compos Struct 31(5):427–435
69. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
70. Muro C et al (2011) Wolf-pack (Canis lupus) hunting strategies emerge from simple rules in computational simulations. Behav Proc 88(3):192–197
71. Mohammadhassani M et al (2014) An evolutionary fuzzy modelling approach and comparison of different methods for shear strength prediction of high-strength concrete beams without stirrups. Smart Struct Syst Int J 14(5):785–809
72. Yeh I-C (2003) Prediction of strength of fly ash and slag concrete by the use of artificial neural networks. J Chin Inst Civil Hydraul Eng 15(4):659–663
73. Yeh I-C (2003) A mix proportioning methodology for fly ash and slag concrete using artificial neural networks. Chung Hua J Sci Eng 1(1):77–84
74. Yeh I-C (1998) Modeling of strength of high-performance concrete using artificial neural networks. Cem Concr Res 28(12):1797–1808
75. Yeh I-C (1998) Modeling concrete strength with augment-neuron networks. J Mater Civ Eng 10(4):263–268
76. Yeh I-C (1999) Design of high-performance concrete mixture using neural networks and nonlinear programming. J Comput Civ Eng 13(1):36–42
77. Yeh I-C (2006) Analysis of strength of concrete using design of experiments and neural networks. J Mater Civ Eng 18(4):597–604
78. Sola J, Sevilla J (1997) Importance of input data normalization for the application of neural networks to complex industrial problems. IEEE Trans Nucl Sci 44(3):1464–1468
79. Legates DR, McCabe GJ Jr (1999) Evaluating the use of "goodness-of-fit" measures in hydrologic and hydroclimatic model validation. Water Resour Res 35(1):233–241
80. Willmott CJ (1981) On the validation of models. Phys Geogr 2(2):184–194

81. Nash JE, Sutcliffe JV (1970) River flow forecasting through conceptual models part I—a discussion of principles. J Hydrol 10(3):282–290
82. Lourakis MI (2005) A brief description of the Levenberg-Marquardt algorithm implemented by levmar. Found Res Technol 4(1):1–6
83. Moré JJ (1978) The Levenberg–Marquardt algorithm: implementation and theory. In: Numerical analysis, pp 105–116. Springer
84. Xu C, Zhang X, Haido JH, Mehrabi P, Shariati A, Mohamad ET, Hoang N, Wakil K (2019) Using genetic algorithms method for the paramount design of reinforced concrete structures. Struct Eng Mech 71(5):503–513
85. Mohammadhassani M et al (2013) Identification of a suitable ANN architecture in predicting strain in tie section of concrete deep beams. Struct Eng Mech 46(6):853–868
86. Khorami M et al (2017) Seismic performance evaluation of buckling restrained braced frames (BRBF) using incremental nonlinear dynamic analysis method