Contents lists available at SCCE

Journal of Soft Computing in Civil Engineering

Journal homepage: www.jsoftcivil.com

Assessing Compressive Strength of Concrete with Extreme Learning Machine

S.C. Nayak[1*], S.K. Nayak[2], S. K. Panda[3]

1. Department of Computer Science and Engineering, CMR College of Engineering & Technology, Hyderabad - 501401, India
2. Department of Computer Application, VSS University of Technology, Odisha – 768018, India
3. Department of Computer Science and Engineering, National Institute of Technology, Warangal – 506004, India
Corresponding author: *saratnayak234@gmail.com*

https://doi.org/10.22115/SCCE.2021.286525.1320

ARTICLE INFO

Article history:
Received: 16 May 2021
Revised: 11 July 2021
Accepted: 12 July 2021

Keywords:
Compressive strength of cement;
ELM;
ANN;
Back propagation neural network;
SVM;
ARIMA.

ABSTRACT

Manual estimation of compressive strength of concrete (CSC) is time consuming and expensive. Soft computing techniques are found better to statistical methods applied to this problem. However, sophisticated prediction models are still lacking and need to be explored. Extreme learning machine (ELM) is a faster and better learning method for artificial neural networks (ANNs) with solitary hidden layer and has enhanced generalization capacity. This article presents an ELM-based forecast for efficient prediction of CSC. A publicly available dataset from UCI repository is used to develop and access the performance of the model. The prediction accuracy of ELM is compared with few machine learning methods such as back propagation neural network (BPNN), support vector machine (SVM), auto-regressive integrated moving average (ARIMA), and least squared estimation (LSE). A comparative study for the prediction of CSC at the curing ages of 28, 56, and 91 days has been carried out using all models. The experimental findings from ELM-based forecasting demonstrate its ability in predicting CSC in a robust manner. On an average, it achieves lowest MAPE of 0.048024, ARV of 0.052872, U of Theil's statistics (UT) of 0.038772, NMSE of 0.058522, and standard deviation (SD) of 0.256267. Comparative analysis of simulation results and statistical significance test suggests the superiority of ELM-based CSC prediction.

How to cite this article: Nayak SC, Nayak SK, Panda SK. Assessing compressive strength of concrete with extreme learning machine. J Soft Comput Civ Eng 2021;5(2):68–85. https://doi.org/10.22115/scce.2021.286525.1320.

1. Introduction

Concrete is the most important material for construction of civil structures. An ever-growing demand for structural concrete is found during last few decades with respect to swift urbanization. Being a vital component of the structures, it experiences some abnormal effects i.e., wearing, freezing, and chemical attacks etc. over the lifespan of the structures. The main objective of investigation of concrete is to determine the CSC for each area of the structure. Based on the CSC, other properties like modulus of elasticity, and ductile strength etc. can be determined. The rebound hammer test and ultrasonic pulse velocity method are the conventional procedures followed for the estimation of mechanical features of concrete. However, significant deviations of their estimations from the true CSC values are the drawbacks of these methods.

Data driven models are suitable alternatives of analytical models which works on establishing models according to available input-output data relationships to extrapolate and predict the output for unseen data. ANNs are systematically used for predicting the CSC in the available literature [1–4]. Components of concrete such as type of specimen, cement, water, fine, coarse masses, mixtures, and temperature play vital role in data driven model [5]. Forecasting accuracy of ANN is critically depending upon network scale and learning scheme [6–9]. An ANN-based predictive model for compressive strength assessment of mortar is proposed in [10]. The authors considered the mortar incorporated with metakaolin, revealed that the binder-sand ratio is an important parameter for this prediction, and ANN yields better prediction. Neural models for compression strength prediction were also proposed in [11] where ANN trained with gradient descent method is found generating superior results compared to multiple regression analysis. Gradient based learning methods are common approaches for ANN training. However, they are associated with few drawbacks such as lethargic convergence rate, imprecise learning and inclined to local minima which add computational overhead to the model [12]. To overcome these, ELM was proposed in [13,14]. It chooses the input-hidden weights at random. Unlike iterative fine-tuning, the hidden-output weights are determined analytically. Numerous researches and experimentations are done in the last few years for diverse applications using ELM which include time series prediction [15], sales forecasting [16], and stock forecasting [17]. It is a faster and better learning method for networks with solitary hidden layer and has enhanced generalization performance. Few articles used evolutionary optimization methods for the process of compressive strength prediction. The authors in [18] proposed a modified model using whale algorithm and response surface methodology for compressive strength of the confined square and rectangular concrete columns with fiber reinforced polymer (FRP). Their predictions are found accurate than comparative models. A stress prediction model using membrane hypothesis to formulate the confining behaviour of FRP confined rectangular columns is proposed in [19] that achieved good corelation with the experimental results. Three new models considering FRP strain efficiency factor as a function of strain ratio, confinement stiffness ratio, and combination of these ratios are proposed in [20]. The model with strain ration found better to others. The authors in [21] developed four machine learning models for design of concrete mix with and without plasticizer and revealed the superiority of decision tree regressor model.

The objective of current study is to design and access performance of a robust ELM-based forecast for prediction of CSC. The trained model is used to extrapolate the numerical concrete data available in UCI repository. Unlike conventional methods where the whole dataset is segregated into training and testing parts, a rolling window method is used to generate the train and test patterns. To access predictability of ELM-based forecast, four other models such as BPNN, SVM, ARIMA, and LSE are developed in similar fashion. Predictability of the procedures are accessed in terms of MAPE, ARV, UT, NMSE, SD, and computation periods.

The rest of the article are segregated into: methods and materials in Section 2, experimental outcome summarization and analysis in Section 3 followed by closing notes.

2. Methodologies and material

This section describes the background methods such as ANN, ELM, BPNN, SVM, ARIMA, and LSE in a nutshell. The base articles for these are cited at the appropriate places. The prospective readers are suggested to follow the base articles for more information. The proposed ELM-based forecasting is discussed in details in Subsection 2.7. The summary statistics of the material used are discussed in Subsection 2.8.

2.1. ANN

ANNs are the computer programs which simulate the way in which human brain works [22]. It consists of hundreds of processing units as artificial neurons, which are fully connected through neural links. Neural links are assigned with weights (also called as synaptic weights) which plays the vital role in neural network learning. A typical ANN architecture is shown in Figure 1. The adder node calculates a linear amalgamation of inputs and the synaptic weights, and incorporates a bias. The induced local field of the neuron passes through the activation link to produce output. The signal flows from layer to layer in forward direction. Once it reaches the output layer, an error signal is generated by comparing the computed output with the expected output, and this error signal moves backward layer by layer until it reaches the input layer. There is no exact rule to fix the optimal number of layer and size of each layer. The decision process relies on domain knowledge as well as trial and error method. A supervised learning approach is followed here for error correction.

The first layer corresponds to the input variable of the given problem. Each input variable corresponds to one neuron. The hidden layers are used to capture the non-linear relationships among variables. At each hidden neuron *j,* the weighted sum y_j is calculated as in Eq. 1.

$$y_j = f\left(b_j + \sum_{i=1}^{n} w_{ij} * x_i\right) \tag{1}$$

Where, x_i is the i^{th} constituent of input vector, w_{ij} is the i^{th} input-j^{th}hidden neuron weight, b_j is a bias and f is a nonlinear activation function. Suppose there are m numbers of nodes in this hidden layer, then for the next hidden layer these m outputs become the input. Then, for each neuron j of the next hidden layer, the output is as in Eq.2.

$$y_j = f\left(b_j + \sum_{i=1}^{m} w_{ij} * y_i\right) \quad (2)$$

This signal flows in the forward direction through each hidden layer until it reaches the output layer. Output y_{esst} for the single output neuron is calculated using Eq. 3.

$$y_{esst} = f\left(b_o + \sum_{j=1}^{m} v_j * y_j\right) \quad (3)$$

Where, v_j is a weight from j^{th} hidden to output neuron, y_j is the weighted sum, and b_o is the output bias. Given a set of training samples $S = \{x_i, y_i\}_{i=1}^{N}$ to train the ANN, let y_i be the output of i^{th} input sample, and y_{esst} is the computed output of the same i^{th}input, then the error is calculated by using Eq. 4.

$$Error_i = y_i - y_{esst} \quad (4)$$

The error value that is produced by n^{th} training sample at the output of neuron i is defined by

$$Error_i(n) = y_i(n) - y_{esst}(n) \quad (5)$$

Then the instantaneous error at neuron i is defined by:

$$\varepsilon_i(n) = \frac{1}{2} Error_i^{\,2}(n) \quad (6)$$

Hence the total instantaneous error of the whole network will be:

$$\varepsilon(n) = \sum_{i \epsilon C} \varepsilon_i(n) \quad (7)$$

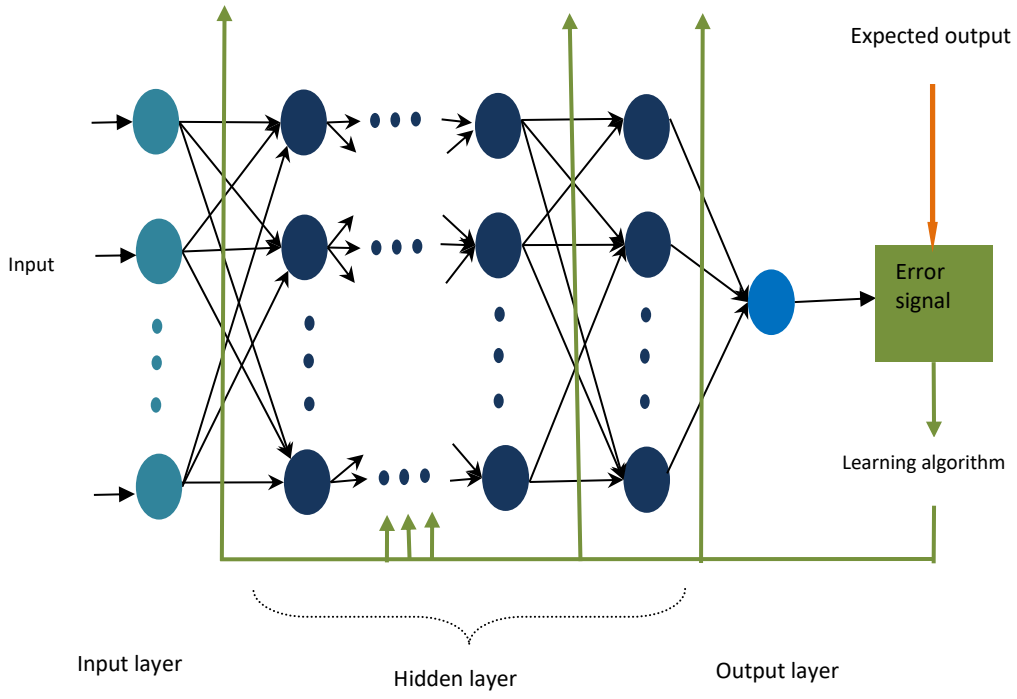

Fig. 1. ANN-based forecasting with multiple hidden layers.

2.2. ELM

As stated earlier, to overcome the limitations of gradient descent learning, ELM was proposed in [13,14]. It was proposed for training networks with one hidden layer. It considers random weight and bias for hidden nodes. As an alternative to iterative tuning the output weights are determined through generalized inverse function on the outputs from the hidden layer. For an input vector x_j, weight vector from input to i^{th} hidden neuron $w_i = [w_{i1}, w_{i2}, \cdots, w_{iN}]^T, (i = 1,2, \cdots, N_h)$ and weights from i^{th} hidden to output neurons $\beta_i = [\beta_{i1}, \beta_{i2}, \cdots, \beta_{iN}]^T$, the output vector O_j is computed as:

$$O_j = \sum_{i=1}^{N_h} \beta_i * f\left(w_i x_j + bias_i\right), j = 1,2, \cdots, N \tag{8}$$

Here N_h is the hidden layer size. The output weight vector β_i can be attained by solving $H\beta = Y$, Where,

$$H(w_i, x_i, b_i) = \begin{bmatrix} f(w_1 x_1 + b_1) & \cdots & f(w_{N_h} x_1 + b_{N_h}) \\ \vdots & \ddots & \vdots \\ f(w_1 x_N + b_1) & \cdots & f(w_{N_h} x_N + b_{N_h}) \end{bmatrix}_{N \times N_h} \tag{9}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{N_h}^T \end{bmatrix}_{N_h \times m} \qquad Y = \begin{bmatrix} y_1^T \\ \vdots \\ y_N^T \end{bmatrix}_{N \times m}$$

In real cases, the inequalities $N_h \ll N$ holds true. So, H is a non-square matrix and may be non-singular in the majority of cases. Therefore, a combination (w_i, b_i, β_i) satisfying Eq. (8) may not exist. So, the network can be trained by finding the least square minimum norm solution $\hat{\beta}$ of (9) as follows:

$$\left\|H\hat{\beta} - Y\right\| = min_\beta \|H\beta - Y\| \tag{10}$$

$$\hat{\beta} = H^+ Y \tag{11}$$

Here H^+ is the pseudo inverse or Moore-Penrose inverse of H.

2.3. BPNN

Backpropagation is the widely used learning method for ANN (shown in Figure 1) and the method is commonly referred as BPNN [23]. It adjusts the weights and bias of a neural network based on the error rate attained in the previous iteration. The modification is based on the gradient of a loss function with respect to the weights and bias set of the network. Proper tuning of these parameters (i.e., weights and bias) is the influential factor for error rate reduction and make the model reliable by increasing its generalization ability. The high-level steps of BPNN are described by the following Algorithm.

Algorithm: BPNN

While (termination criteria not meeting)

Step 1: Supply inputs through the preconnected path

Step 2: Model the inputs using randomly initialized weights and bias

Step 3: Compute the output for every neuron from the input layer, to the hidden layers, to the output layer.

Step 4: Compute the error in the outputs as: error = Model output – Desired output

Step 5: Travel back from the output layer to the hidden layer to adjust the weights such that the error is decreased.

End

2.4. ARIMA

ARIMA model is extensively used statistical models for time series forecasting [24]. The model is based on the hypothesis that the allied time series can be generated from a linear combination of predefined number of past observations and a random white noise term. Mathematically the model is represented as follows:

$$\emptyset(\mathrm{S})(1-\mathrm{S})^{\mathrm{d}}(\mathrm{y_t}) = \theta(\mathrm{S})\varepsilon_{\mathrm{t}} \tag{12}$$

Where, $\emptyset(S) = 1 - \sum_{i=1}^{p} \emptyset_i S^i$, $\theta(S) = 1 + \sum_{j=1}^{q} \theta_j S^j$. The parameter p is the number of autoregressive, q is the moving average terms, and d is the degree of differencing. The term ε_t is the random error term and y_t represents the actual observations of the time series. The random error term basically satisfies the i.i.d property. More generally these models are referred as ARIMA (p, d, q) model.

2.5. SVM

SVM is a popular supervised machine learning method, has demonstrated high performance in solving classification and regression problems [25,26]. It works on discriminating between two classes by generating a hyperplane that optimally separates classes after the input data are transformed mathematically into a high-dimensional space. The nonlinear problem in original space can be viewed as a linear one in high-dimensional space. Because the SVM approach is data-driven and model-free, it may have important discriminative power for cases where sample sizes are small and a large number of variables are involved. The special nonlinear functions called kernels are responsible to transform the input space into a multidimensional space, thus achieves high discriminative power. Different kernels such as linear, polynomial, Gaussian, and radial basis functions can be used in SVM. A SVM with kernel K, label yi, and feature set x can be presented as in Eq.13.

$$y = \beta_0 + \sum \alpha_i y_i K(x(i), x) \tag{13}$$

Then elements in one category will be such that the $y > 0$, while elements in the other category will have $y < 0$. The radial basis function is used as kernel in this work.

2.6. LSE

The LSE is based on the principle that provides a way of choosing the coefficients effectively by minimizing the sum of the squared errors between the estimation and desired output. it gives the least value for the sum of squared errors. Finding the best estimates of the coefficients is often called fitting the model to the data. The LSE can be represented as in Eq. 14. The prospective readers are suggested to refer the material in [27] for details.

$$\sum_{t=1}^{T} error_t^2 = \sum_{t=1}^{T} \left(y_t - \beta_0 - \beta_1 x_{1,t} - \beta_2 x_{2,t} - \cdots - \beta_k x_{k,t}\right)^2 \quad (14)$$

2.7. ELM-based compressive strength prediction

A pictorial view of ELM-based forecasting of cryptocurrency is depicted in Figure 2. Here, the base model is a neural network having single hidden layer. The network is trained by ELM. The input-hidden weights are assigned with random values. The output weights are computed as per Eq. 4. A rolling window method is used to generate the train and test patterns from the dataset. The method is depicted in Figure 2. The ELM-based compressive strength prediction model is depicted by Figure 3 and the overall steps of the forecasting process is shown in Figure 4.

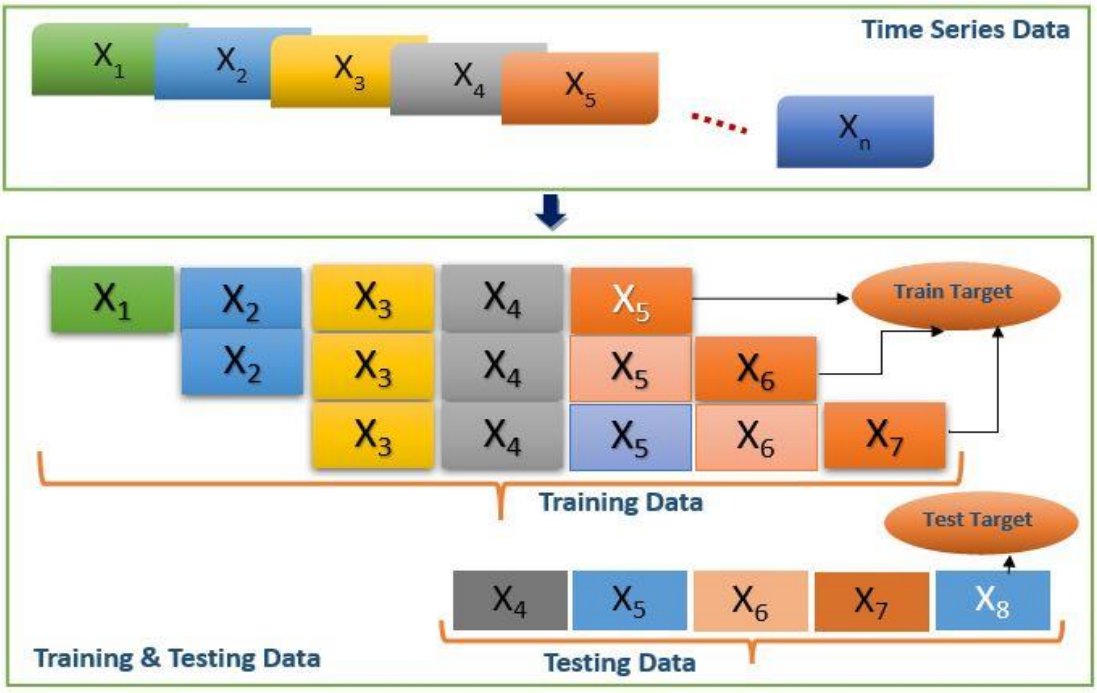

Fig. 2. Rolling window method for input pattern generation.

Each sample of the dataset is considered as a data point on the time series. A window of fixed size is rolled over the time series. On each movement, an old datapoint is dropped and a new datapoint is included. The number of datapoints included by the window at any instant of time is

considered as one training sample. The width of the window is determined experimentally. For example, one train/test set is formed by the rolling window of width three is as follows.

$$\begin{matrix} x(k) & x(k+1) & x(k+2) & \vdots x(k+3) \\ x(k+1) & x(k+2) & x(k+3) & \vdots x(k+4) \\ \underbrace{x(k+2)}_{} & \underbrace{x(k+3)}_{} & \underbrace{x(k+4)}_{Training\ data} & \vdots \underbrace{x(k+5)}_{Target} \end{matrix}$$

$$\underbrace{x(k+3)}_{} \quad \underbrace{x(k+4)}_{} \quad \underbrace{x(k+5)}_{Test\ data} \quad \vdots \underbrace{x(k+6)}_{Target}$$

The patterns are then normalized to scale the data into same range for each input feature so as to diminish the bias [28]. The $tanh$ normalization method as in Eq. 15 is used to standardize the input data. The mean and standard deviation of a training pattern are represented as μ and σ respectively.

$$\hat{x} = 0.5 * \left(tanh\left(\frac{0.01*(x-\mu)}{\sigma} \right) + 1 \right) \tag{15}$$

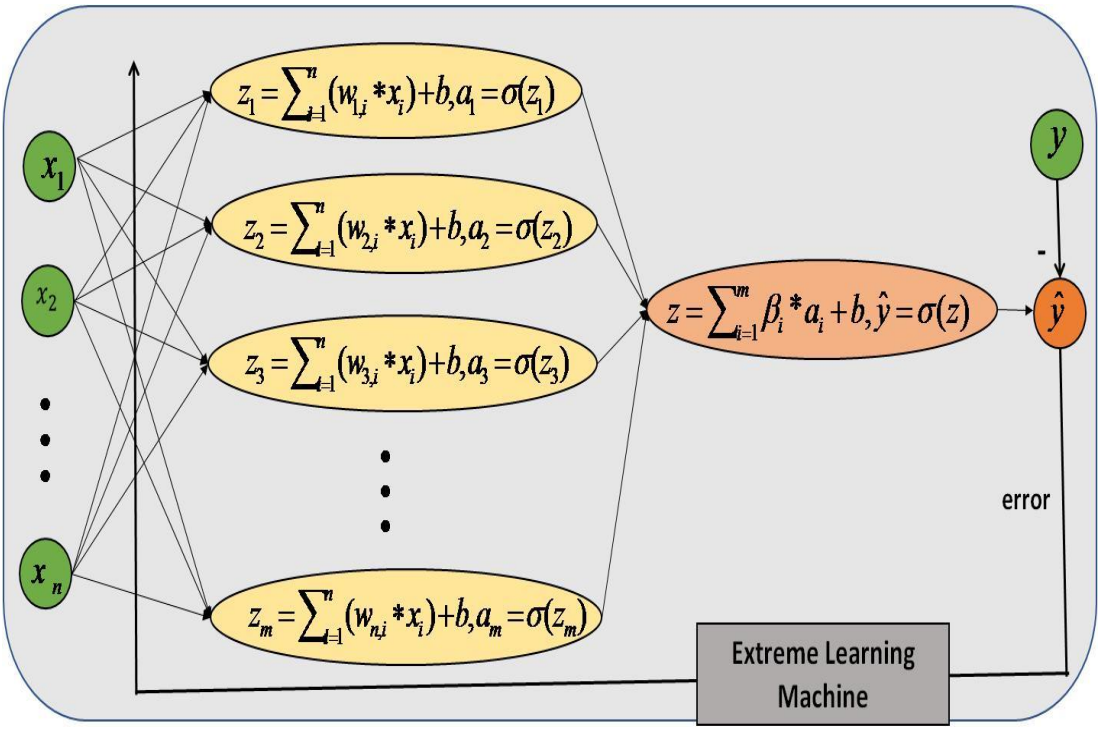

Fig. 3. ELM-based compressive strength prediction model.

After normalization the patterns are inputted sequentially to the network. A hidden neuron computes the weighted sum as well augments a bias. It then applies an activation to generate the net output. The output unit computes its net value in the same way. The model estimated an output ($\hat{y}$) at the output layer as in Eq.16. The amount of deviation of the estimation from the target (y) is calculated as error as in Eq.17. The minimum is the error; closer the model estimation towards the actual, hence better is the prediction accuracy of the model. The estimation is denormalized.

$$\hat{y} = \sum_{i=1}^{m} \sigma(\beta_i * a_i + b) \tag{16}$$

$$error = abs(y - \hat{y}) \tag{17}$$

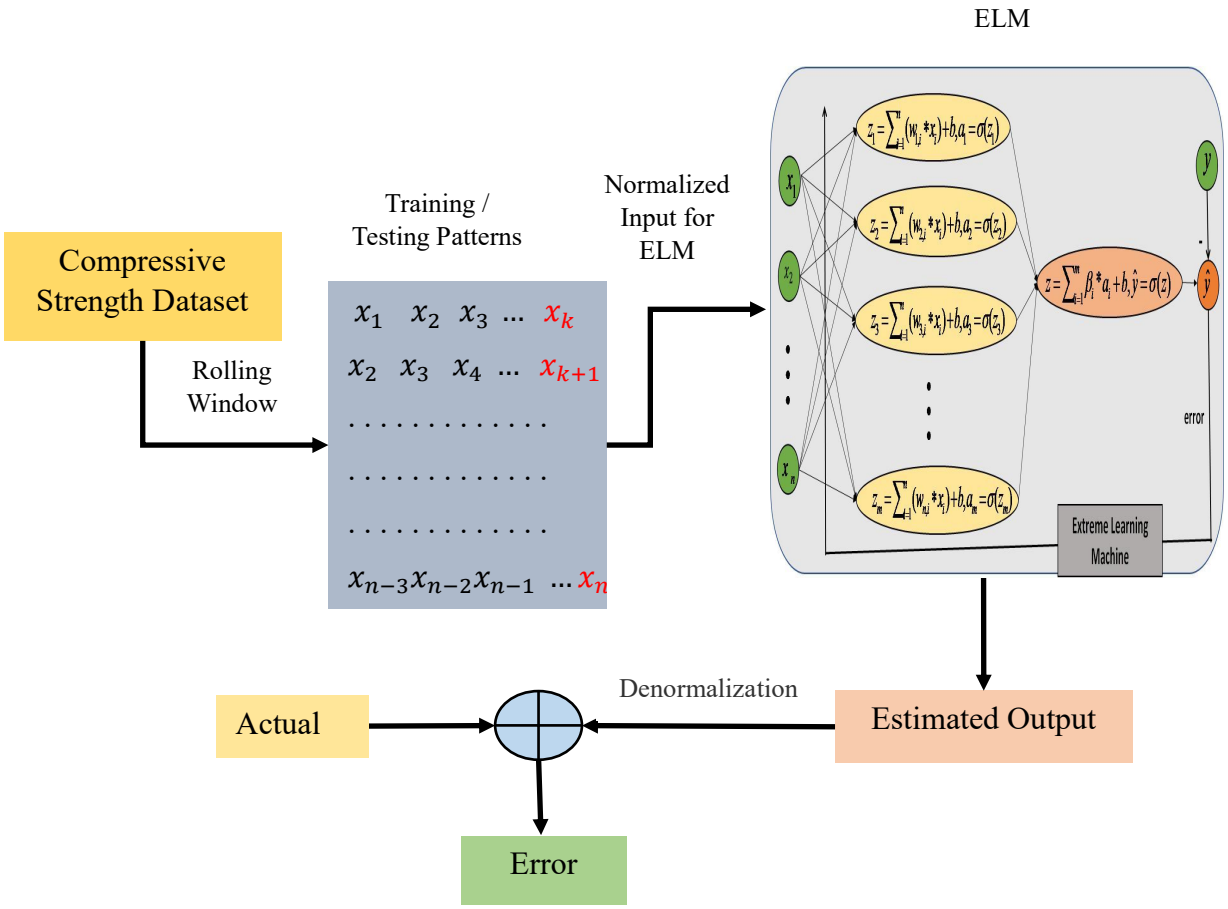

Fig. 4. Overall steps of compressive strength prediction.

2.8. Data

The numerical data for experimentation are collected from the open source of UCI repository [29]. The dataset contains 1030 samples each of has 9 real attributes as summarized in Table 1. The attributes 1 – 8 are used as input for the model and last one (i.e., Concrete compressive strength) is used as the target. All the data points are quantitative and have numeric values only. From the original dataset, samples with curing age at 28, 56, and 91 days only are considered for experimentation. The correlation between input features is shown in Table 2. It can be seen that the cement, super plasticizer, and curing age are the most influential components for the compressive strength. The visualization of distribution of different components are shown in Figure 5.

Table 1
Summary statistics from the dataset.

Component	Mean	Std	Min	25%	50%	75%	Max
Cement (kg/m3)	281.16786	104.50636	102.0000	192.37500	272.90000	350.00000	540.00000
Blast furnace slag (kg/m3)	73.895825	86.27934	0.000000	0.000000	22.000000	142.95000	359.40000
Fly ash (kg/m3)	54.188350	63.99700	0.000000	0.000000	0.000000	118.300000	200.10000
Water (kg/m3)	181.56728	21.35421	121.80000	164.90000	185.00000	192.00000	247.00000
Super plasticizer (kg/m3)	6.204660	5.973841	0.000000	0.000000	6.400000	10.200000	32.200000
Coarse Aggregate (kg/m3)	972.91893	77.75395	801.0000	932.00000	968.00000	1029.4000	1145.0000
Fine Agg. (kg/m3)	773.58048	80.175980	594.0000	730.95000	779.50000	824.00000	992.60000
Age (numeric)	45.662136	63.169912	1.000000	7.00000	28.000000	56.000000	365.00000
Compressive Strength of Concrete (MPa)	35.817961	16.705742	2.330000	23.710000	34.445000	46.135000	82.600000

Table 2
Correlation between input features.

	C	BFS	FA	W	SP	CA	FAG	AGE	CSC
Cement	1.000000	-0.275216	-0.397467	-0.081587	0.092386	-0.109349	-0.222718	0.081946	0.497832
Blast furnace slag	-0.27521	1.000000	-0.323580	0.107252	0.043270	-0.283999	-0.281603	-0.044246	0.134829
Fly ash	-0.39746	-0.323580	1.000000	-0.256984	0.377503	-0.009961	0.079108	-0.154371	-0.105755
Water	-0.08158	0.107252	-0.256984	1.000000	-0.657533	-0.182294	-0.450661	0.277618	-0.289633
Super plasticizer	0.092386	0.043270	0.377503	-0.657533	1.000000	-0.265999	0.222691	-0.192700	0.366079
Coarse Aggregate	-0.109349	-0.283999	-0.009961	-0.182294	-0.265999	1.000000	-0.178481	-0.003016	-0.164935
Fine Agg.	-0.222718	-0.281603	0.079108	-0.450661	0.222691	-0.178481	1.000000	-0.156095	-0.167241
Age	0.081946	-0.044246	-0.154371	0.277618	-0.192700	-0.003016	-0.156095	1.000000	0.328873
Compressive Strength of Concrete	0.497832	0.134829	-0.105755	-0.289633	0.366079	-0.164935	-0.167241	0.328873	1.000000

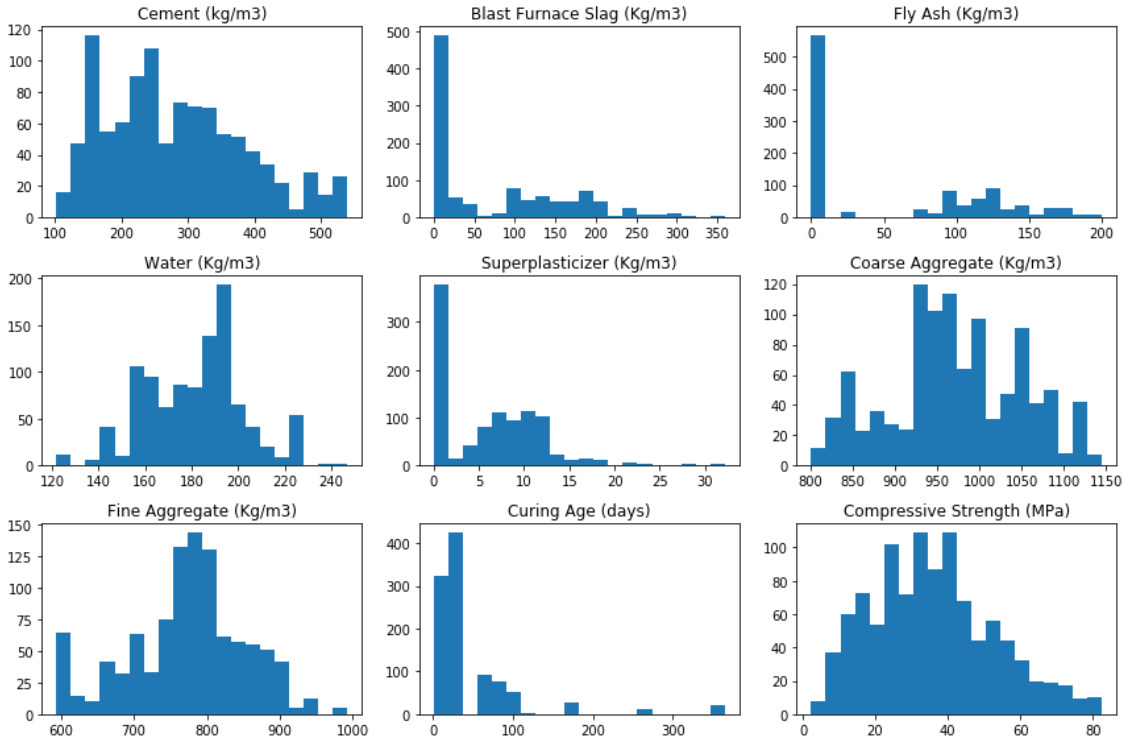

Fig. 5. Distribution of components.

3. Analysis of experimental outcomes and model evaluation

The developed forecasts are trained and tested separately under the same environmental setup. From the original dataset, samples with curing age at 28, 56 and 91-days are only considered for experimentation here. The number of samples are 425, 91, and 76 respectively. The *tanh* function as in Eq. 15 for normalization as mentioned in Section 2.3. The same training and test patterns are fed to all models to maintain the fairness in compression. Five evaluation metrics such as mean absolute percentage of error (MAPE), and average relative variance (ARV), U of Theil's statistics (UT), NMSE, and standard deviation (SD) are used to measure the predictability of the methods [18,30] as follows:

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\frac{|x_i - \hat{x}_i|}{x_i} \times 100\% \tag{18}$$

$$\text{ARV} = \frac{\sum_{i=1}^{N}(\hat{x}_i - x_i)^2}{\sum_{i=1}^{N}(\hat{x}_i - \bar{X})^2} \tag{19}$$

$$U\ of\ Theil = \frac{\sum_{i=1}^{N}(x_i - \hat{x}_i)^2}{\sum_{i=1}^{N}(x_i - x_{i+1})^2} \tag{20}$$

$$NMSE = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x}_i)^2 \tag{21}$$

$$SD = \sqrt{\frac{\sum_{1}^{N}\left(\frac{x_i}{\hat{x}_i} - \frac{Actual_{avg.}}{Esstd._{avg.}}\right)^2}{N-1}} \tag{22}$$

In the above equations, an original data point is represented as x_i, a model estimation is represented as $\hat{x}_i$, and N is the total number of samples.

To access the capacity of the ELM-based forecast, four other methods such as ARIMA, SVM, BPNN, and LSE are implemented in a similar way and a comparison is done through five performance metrics. To overcome the biasness of the models under consideration, each model is simulated twenty times and the mean error value is recorded for comparisons. An ANN with solitary hidden layer is used as the base network and ELM is used to train it. Here, the input layer has eight neurons as there are eight numbers of attributes in a sample. The output layer has only one neuron, as there is one target output. However, the numbers of neurons in the hidden layer are decided experimentally. Inadequate number of hidden neurons may produce poor accuracy, whereas excess number of such neurons add computational overhead. Therefore, the size of hidden layer impacts the model performance a lot and must be decided carefully. For the three datasets (i.e., 28-days series, 56-days series, and 91-days series) the hidden layer size are chosen as 24, 16, and 16 respectively. Therefore, the optimal ANN structure for the three datasets is *8-24-1*, *8-16-1*, and *8-16-1* respectively. Radial basis function is used as the kernel for SVM. A three-layer architecture is used for BPNN where the hidden layer size lies in the range of 25 – 28 and gradient descent-based backpropagation is used as the learning method.

Different MAPE, NMSE, UT, SD, and ARV values generated from all models considering three data samples are recorded in Table 3 - 5. The best statistic values are highlighted in bold face. From Table 3, it is noted that, ELM-based model always produced lowest error values compared to others. For example, the MAPE, ARV, UT, NMSE, and SD values of the proposed model are 0.033452, 0.051725, 0.024752, 0.032755, and 0.2901respectively. Similar observations are recorded from Table 4 and Table 5 data. The model performances can be further analyzed through computation time. All the experimentations are done with a system of Intel core i3 CPU, 2.27GHz and 8.0 GB memory and matlab-2015 program writing environment. The execution times (in seconds) are summarized in last column of Table 3 - 5. It can be found from Table 3 that the ELM-based prediction model required lesser running time than others, i.e., 185.015 seconds for 28-days sample data.

Table 3
Error statistics and execution times from sample series with curing age at 28-days.

Method	Error Statistics					Run time (Seconds)
	MAPE	ARV	UT	NMSE	SD	
ELM	**0.033452**	**0.051725**	**0.024752**	**0.032755**	0.2901	185.015
SVM	0.082636	0.274423	1.014201	0.364503	0.3327	215.720
BPNN	0.082368	0.297465	1.004255	0.047157	0.3604	274.325
ARIMA	1.022592	0.889472	1.083962	0.365490	0.4002	144.003
LSE	1.035935	0.897655	1.118757	0.656845	0.4512	190.945

Table 4
Error statistics and execution times from sample series with curing age at 56-days.

Method	Error Statistics					Run time (Seconds)
	MAPE	ARV	UT	NMSE	SD	
ELM	**0.053575**	**0.054265**	**0.048758**	**0.065275**	0.2275	105.531
SVM	0.085033	0.279245	0.084244	0.367377	0.2739	117.755
BPNN	0.223653	0.297883	1.104290	0.076357	0.3155	152.005
ARIMA	1.250905	1.007524	1.000364	0.406548	0.3411	106.362
LSE	1.130562	0.995035	1.301725	0.686450	0.4125	110.577

Table 5
Error statistics and execution times from sample series with curing age at 91-days.

Method	Error Statistics					Run time (Seconds)
	MAPE	ARV	UT	NMSE	SD	
ELM	**0.057045**	**0.052625**	**0.042805**	**0.077536**	0.2512	92.006
SVM	0.079382	0.298726	0.088479	0.364399	0.2825	105.704
BPNN	0.246527	0.299036	1.293300	0.095563	0.2988	112.533
ARIMA	1.200055	1.025262	1.030762	0.060045	0.3427	90.375
LSE	1.320443	1.100085	1.225383	0.565332	0.3992	83.505

In order to analyze the performance of the proposed approach further, the performance gain of it over comparative methods is calculated as in Eq. 23. The outcomes are summarized in Table 6. It is observed that the ELM approach attained 60% - 97% MAPE reduction, 81% - 94% ARV reduction, 98% UT reduction, 91% - 95% NMSE reduction, and 13% - 36% SD reduction in case of 28-days sample dataset. Considering 56-days sample set, it achieved 37% - 96% reduction in MAPE, 81% - 95% in ARV, 42% - 96% in UT, 82% - 90% in NMSE, and 17% - 45% in SD values. Similar observations are found in case of 91-days sample set. Therefore, almost all cases the ELM approach found performing better to comparative models.

$$Performance\ gain = \frac{(Error\ from\ comparative\ model - Error\ \text{from}\ proposed\ model)}{Error\ from\ \text{comparative}\ model} \times 100\% \qquad (23)$$

Table 6
Percentage (%) of error reduction on adopting ELM approach over comparative methods.

Error metric	28-days sample data				56-days sample data				91-days sample data			
	SVM	BPNN	ARIMA	LSE	SVM	BPNN	ARIMA	LSE	SVM	BPNN	ARIMA	LSE
MAPE	60	59	97	97	37	76	96	95	28	77	95	96
ARV	81	83	94	94	81	82	95	95	82	82	0.95	95
UT	98	98	98	98	42	96	95	96	52	97	0.96	97
NMSE	91	31	91	95	82	15	84	90	79	19	-29	86
SD	13	20	28	36	17	28	33	45	11	16	27	37

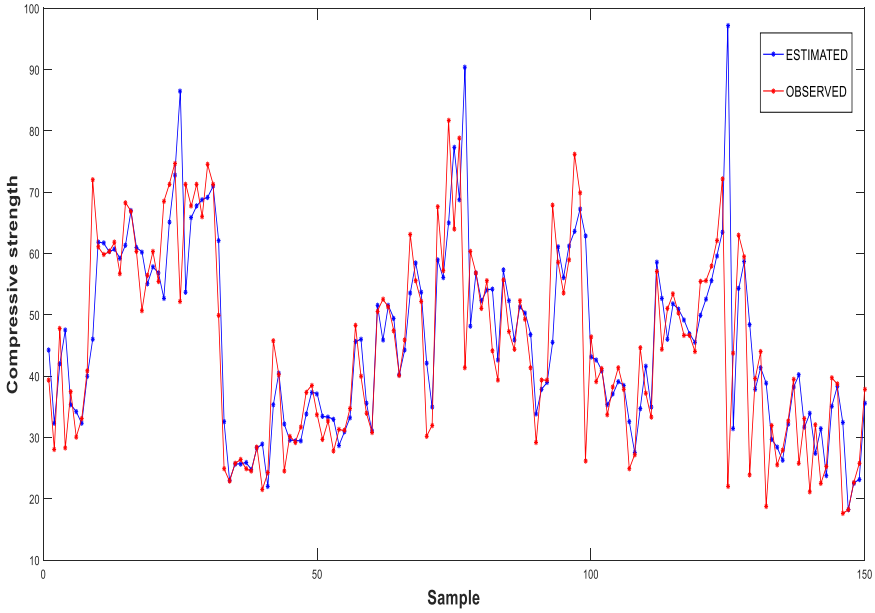

Fig. 6. Forecasting plots from sample series with 28 days curing age using ELM.

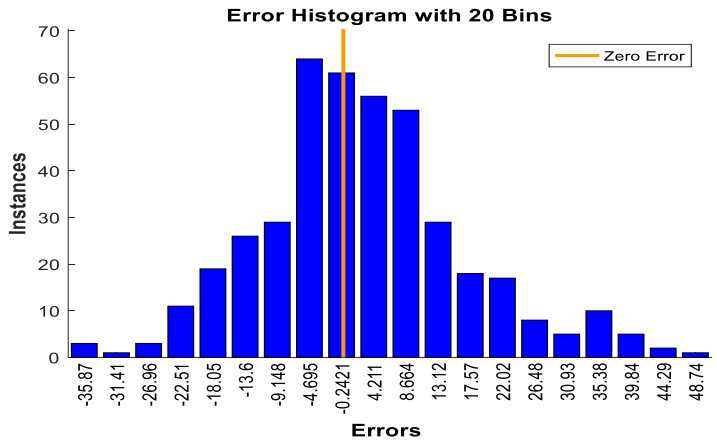

Fig. 7. Error distribution graph from 28-days curing age sample series using ELM.

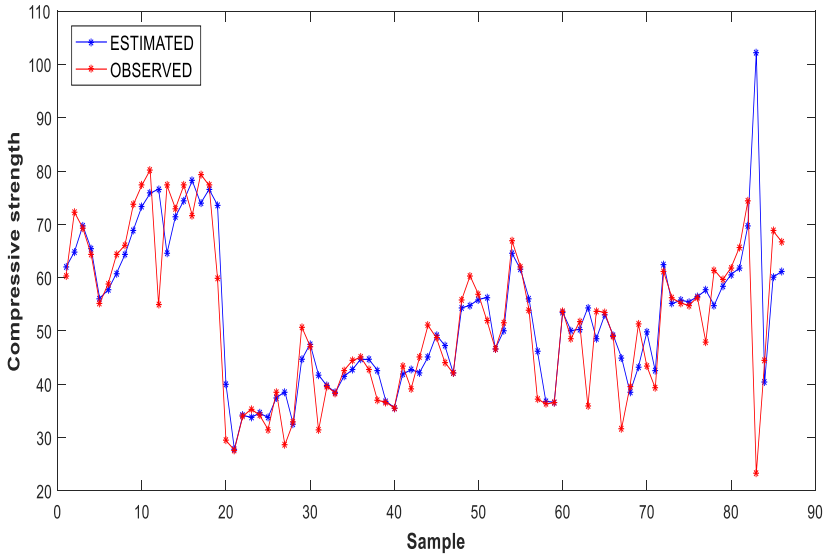

Fig. 8. Forecasting plots from sample series with 56 days curing age using ELM.

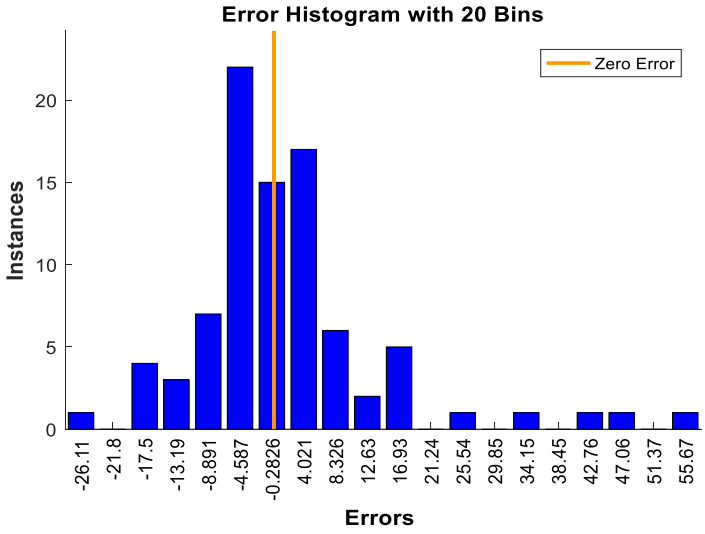

Fig. 9. Error distribution graph from 56-days curing age sample series using ELM.

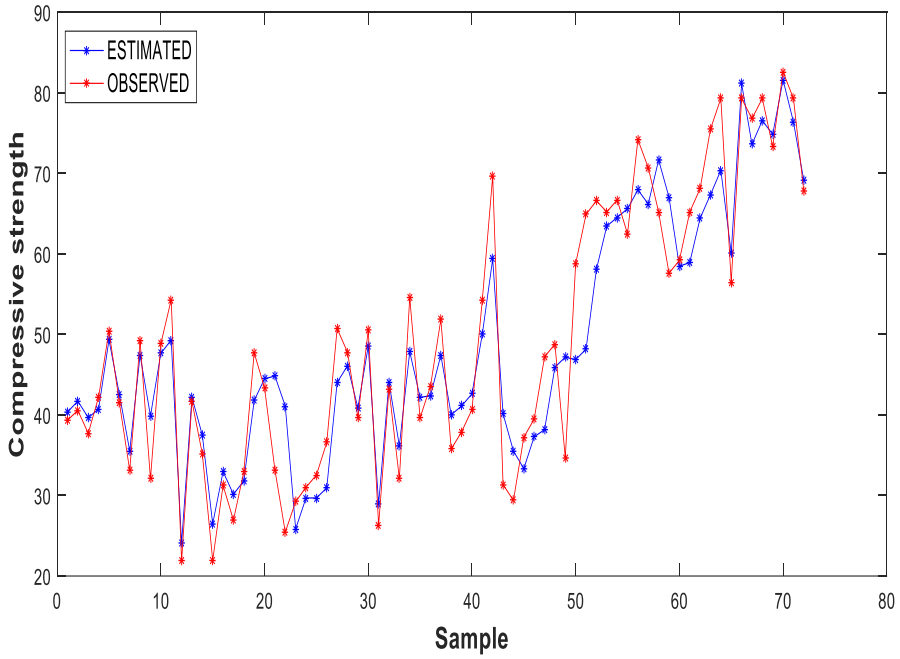

Fig. 10. Forecasting plots from sample series with 56 days curing age using ELM.

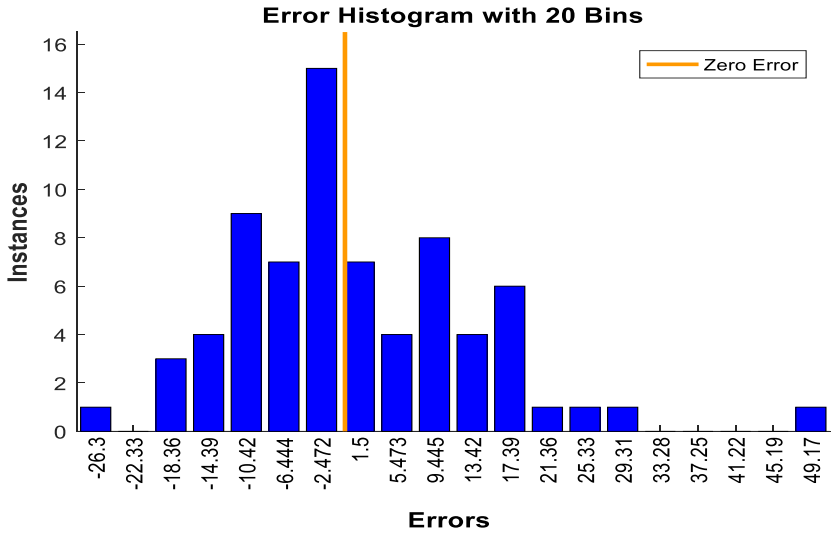

Fig. 11. Error distribution graph from 91-days curing age sample series using ELM.

To establish the better predictability of ELM-based model, the estimated model values against true compressive strength values are plotted in Figure 6, 8 and 10 for 28, 56, and 91-days sample sets respectively. For the sake of visibility, 150 observations only are plotted in Figure 6. It is visible that the model estimations are closer to the true compressive strength values and following the trend of actual data. The error distribution graphs from different datasets are depicted in Figure 7, 9, and 11. From these figures it is observed that for majority of the samples, the ELM generated error values closer to zero.

The Wilcoxon signed-rank test is then conducted for a significance check. Outcomes of the paired, two-sided test for the null hypothesis shows that the change between the proposed and

reasonable models comes from a zero-median distribution. Rejection of the null hypothesis indicated by the logical value $h = 1$. The Wilcoxon signed-rank test results are summarized in Table 7. These statistic values show that the ELM-based approach is appreciably different from others.

Table 7
Statistics from Wilcoxon signed-rank test.

Analyzed methods		*p*-value		
		28-days sample set	56-days sample set	91-days sample set
ELM	SVM	3.2527e-5 ($h = 1$)	3.12402e-4 ($h = 1$)	3.2452e-3 ($h = 1$)
	ARIMA	3.1417e-3 ($h = 1$)	3.2741e-5 ($h = 1$)	3.3244e-3 ($h = 1$)
	BPNN	3.3522e-4 ($h = 1$)	2.3604e-3 ($h = 1$)	2.5465e-2 ($h = 1$)
	LSE	3.5316e-3 ($h = 1$)	2.03632 ($h = 1$)	3.7367e-2 ($h = 1$)

From the above experimental result analysis, comparative studies, and statistical significance test the following points can be drawn.

- The ELM-based neural network model is quite capable in modelling the CSC data.
- It achieved better forecasting accuracy compared to other statistical models.
- Substantial reductions in error values are found on adopting ELM-based approach.
- The proposed approach is significantly different from comparative models.

4. Conclusions

Prediction of compressive strength of concrete is an active area of research in the domain of manufacturing engineering. For accurate prediction of compressive strength of concrete, this article proposed an ELM-based forecast. The forecast has a single hidden layer ANN as the base architecture, and ELM as the learning method hence, possess less structural and computational complexity. The data driven model is evaluated on a publicly available dataset. To establish the superiority of ELM-based forecast, a comparative study with few standard forecasting methods such as BPNN, SVM, ARIMA, and LSE is carried out. From the original dataset, samples with curing age at 28, 56, and 91-days are considered for experimentation. The training and test patterns are generated by the rolling window method from the original dataset. Considering the three datasets, the ELM on an average achieved lowest MAPE of 0.048024, ARV of 0.052872, U of Theil's statistics (UT) of 0.038772, NMSE of 0.058522, and standard deviation (SD) of 0.256267. Also, there is significant performance gains on adopting ELM over others. Analysis from different error statistics and computation times reveals the superiority of the proposed model. Finding optimal hidden layer size manually is a limitation of current study and need to be automated. Exploring other sophisticated ANNs and learning algorithms can be another future

direction. Further, the proposed approach can be applied to other data driven problems in material science domain.

Funding

Not applicable, No funding available.

Conflicts of interest

The author(s) declare that they have no conflict of interest.

Authors contribution statement

SCN carried out the problem formulation, model design and implementation, analysis of outcomes and article write up. SKN and SKP prepared the literature study, and writeup. All author(s) read and approved the final manuscript.

Availability of data and material

The datasets analyzed and experimented during the current study are available at http://archive.ics.uci.edu/ml/datasets/Concrete+Compressive+Strength, which openly available.

References

[1] Asteris PG, Mokos VG. Concrete compressive strength using artificial neural networks. Neural Comput Appl 2020;32:11807–26. doi:10.1007/s00521-019-04663-2.

[2] Asteris PG, Kolovos KG, Douvika MG, Roinos K. Prediction of self-compacting concrete strength using artificial neural networks. Eur J Environ Civ Eng 2016;20:s102–22. doi:10.1080/19648189.2016.1246693.

[3] Asteris P, Roussis P, Douvika M. Feed-Forward Neural Network Prediction of the Mechanical Properties of Sandcrete Materials. Sensors 2017;17:1344. doi:10.3390/s17061344.

[4] Asteris PG, Moropoulou A, Skentou AD, Apostolopoulou M, Mohebkhah A, Cavaleri L, et al. Stochastic Vulnerability Assessment of Masonry Structures: Concepts, Modeling and Restoration Aspects. Appl Sci 2019;9:243. doi:10.3390/app9020243.

[5] Bungey JH, Grantham MG. Testing of concrete in structures, 3rd edn. Blackie Academic & Professional, London. 1996.

[6] Nayak SC. Development and Performance Evaluation of Adaptive Hybrid Higher Order Neural Networks for Exchange Rate Prediction. Int J Intell Syst Appl 2017;9:71–85. doi:10.5815/ijisa.2017.08.08.

[7] Nayak SC, Misra BB, Behera HS. Fluctuation prediction of stock market index by adaptive evolutionary higher order neural networks. Int J Swarm Intell 2016;2:229. doi:10.1504/IJSI.2016.081152.

[8] Nayak SC, Misra BB, Behera HS. Adaptive hybrid higher order neural networks for prediction of stock market behavior. Nature-Inspired Comput. Concepts, Methodol. Tools, Appl., IGI Global; 2017, p. 553–70.

[9] Behera AK, Nayak SC, Dash CSK, Dehuri S, Panda M. Improving Software Reliability Prediction Accuracy Using CRO-Based FLANN, 2019, p. 213–20. doi:10.1007/978-981-10-8201-6_24.

[10] Sharifi Y, Hosainpoor M. A Predictive Model Based ANN for Compressive Strength Assessment of the Mortars Containing Metakaolin. J Soft Comput Civ Eng 2020;4:1–12.
[11] Priyadarshee A, Chandra S, Gupta D, Kumar V. Neural Models for Unconfined Compressive Strength of Kaolin clay mixed with pond ash, rice husk ash and cement. J Soft Comput Civ Eng 2020;4:85–102.
[12] Fernández-Navarro F, Hervás-Martínez C, Ruiz R, Riquelme JC. Evolutionary Generalized Radial Basis Function neural networks for improving prediction accuracy in gene classification using feature selection. Appl Soft Comput 2012;12:1787–800. doi:10.1016/j.asoc.2012.01.008.
[13] Huang G-B, Zhu Q-Y, Siew C-K. Extreme learning machine: Theory and applications. Neurocomputing 2006;70:489–501. doi:10.1016/j.neucom.2005.12.126.
[14] Huang G-B, Zhou H, Ding X, Zhang R. Extreme learning machine for regression and multiclass classification. IEEE Trans Syst Man, Cybern Part B 2011;42:513–29.
[15] Grigorievskiy A, Miche Y, Ventelä A-M, Séverin E, Lendasse A. Long-term time series prediction using OP-ELM. Neural Networks 2014;51:50–6. doi:10.1016/j.neunet.2013.12.002.
[16] Sun Z-L, Choi T-M, Au K-F, Yu Y. Sales forecasting using extreme learning machine with applications in fashion retailing. Decis Support Syst 2008;46:411–9. doi:10.1016/j.dss.2008.07.009.
[17] Nayak SC, Misra BB. Extreme learning with chemical reaction optimization for stock volatility prediction. Financ Innov 2020;6:16. doi:10.1186/s40854-020-00177-2.
[18] Moodi Y, Mousavi SR, Ghavidel A, Sohrabi MR, Rashki M. Using Response Surface Methodology and providing a modified model using whale algorithm for estimating the compressive strength of columns confined with FRP sheets. Constr Build Mater 2018;183:163–70. doi:10.1016/j.conbuildmat.2018.06.081.
[19] Pham TM, Hadi MNS. Stress Prediction Model for FRP Confined Rectangular Concrete Columns with Rounded Corners. J Compos Constr 2014;18:04013019. doi:10.1061/(ASCE)CC.1943-5614.0000407.
[20] Moodi Y, Mousavi SR, Sohrabi MR. New models for estimating compressive strength of concrete confined with FRP sheets in circular sections. J Reinf Plast Compos 2019;38:1014–28. doi:10.1177/0731684419858708.
[21] Pandey S, Kumar V, Kumar P. Application and Analysis of Machine Learning Algorithms for Design of Concrete Mix with Plasticizer and without Plasticizer. J Soft Comput Civ Eng 2021;5:19–37.
[22] Haykin S. Neural Networks and Learning Machines. 3/E. Pearson Education India 2010.
[23] Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. Nature 1986;323:533–6. doi:10.1038/323533a0.
[24] Box GEP, Jenkins GM. Time Series Analysis-Forecasting and Control, Holden-Day Inc., San Francisco 1976.
[25] Cortes C, Vapnik V. Support-vector networks. Mach Learn 1995;20:273–97. doi:10.1007/BF00994018.
[26] Halls-Moore M. “Support vector machines: A guide for beginners,” September 2014. [Online]. Available: http://www.quantstart.com/articles/Support-Vector-Machines-A-Guide-for-Beginners n.d.
[27] Hyndman RJ, Athanasopoulos G. Forecasting: principles and practice. OTexts; 2018.
[28] Nayak SC, Misra BB, Behera HS. Impact of data normalization on stock index forecasting. Int J Comput Inf Syst Ind Manag Appl 2014;6:257–69.
[29] http://archive.ics.uci.edu/ml/datasets/Concrete+Compressive+Strength. n.d.
[30] Nayak SC, Misra BB, Behera HS. ACFLN: artificial chemical functional link network for prediction of stock market index. Evol Syst 2019;10:567–92. doi:10.1007/s12530-018-9221-4.