Expert Systems with Applications 42 (2015) 885–893

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Modeling slump of ready mix concrete using genetic algorithms assisted training of Artificial Neural Networks

Vinay Chandwani *, Vinay Agrawal, Ravindra Nagar

Department of Civil Engineering, Malaviya National Institute of Technology Jaipur, Rajasthan, India

ARTICLE INFO

Article history:
Available online 6 September 2014

Keywords:
Artificial Neural Networks
Genetic algorithms
Back-propagation algorithm
Lavenberg Marquardt training algorithm
Concrete slump
Ready mix concrete

ABSTRACT

The paper explores the usefulness of hybridizing two distinct nature inspired computational intelligence techniques viz., Artificial Neural Networks (ANN) and Genetic Algorithms (GA) for modeling slump of Ready Mix Concrete (RMC) based on its design mix constituents viz., cement, fly ash, sand, coarse aggregates, admixture and water-binder ratio. The methodology utilizes the universal function approximation ability of ANN for imbibing the subtle relationships between the input and output variables and the stochastic search ability of GA for evolving the initial optimal weights and biases of the ANN to minimize the probability of neural network getting trapped at local minima and slowly converging to global optimum. The performance of hybrid model (ANN-GA) was compared with commonly used back-propagation neural network (BPNN) using six different statistical parameters. The study showed that by hybridizing ANN with GA, the convergence speed of ANN and its accuracy of prediction can be improved. The trained hybrid model can be used for predicting slump of concrete for a given concrete design mix in quick time without performing multiple trials with different design mix proportions.

1. Introduction

The mathematical relationships commonly used to describe the material behavior of concrete are available in the form of empirical formulae derived from experimental results. Although these empirical relationships in the form of regression equations are widely used and recommended for extracting knowledge about a particular property of concrete, yet these cannot be applied wherein the modeling problem involves a large number of independent variables or the interactions among the variables is either unknown or too complex to represent. In such cases the traditional technique of regression fails to yield the expected accuracy and predictability. Over the past few decades nature inspired computational tool, Artificial Neural Network (ANN) has been used for modeling the real world problems due to its immense ability to capture inter-relationships among input and output data pairs which are unknown, nonlinear or too difficult to formulate. This potential of ANN has been harnessed for wide applications in modeling the material behavior and properties of concrete. Notable among them are successful implementations in predicting and modeling compressive strength of self compacting concrete (Uysal & Tanyildizi, 2012), high performance concrete (Yeh, 1998), recycled aggregate concrete (Duan, Kou, & Poon, 2013), rubberized concrete (Abdollahzadeh, Masoudnia, & Aghababaei, 2011), fiber reinforced concrete (FRP)-confined concrete (Naderpour, Kheyroddin, & Ghodrati Amiri, 2010), durability of high performance concrete (Parichatprecha & Nimityonskul, 2009), predicting drying shrinkage of concrete (Bal & Buyle-Bodin, 2013), concrete mix design (Ji, Lin, & Lin, 2006) and prediction of elastic modulus of normal and high strength concrete (Demir, 2008).

One of the physical properties of concrete which plays an important role in the success of RMC industry is its workability. It signifies the ease, with which fresh concrete can be placed, compacted and finished at site with sufficient resistance to segregation. Being a quality assurance metric quantitatively measured as concrete slump value, it not only controls quality and uniformity of concrete from batch to batch but also acts a measure to ascertain the shelf life of the RMC during its transit course from manufacturing plant to subsequent placing at the construction site. Moreover it ensures that the RMC design mix is customized catering to the type of construction viz., heavily reinforced sections, lightly reinforced sections, road pavements, shallow sections or construction requiring intensive vibration, demanding high, medium, low, very low or extremely low workability concrete respectively. Recent applications of ANN modeling for concrete slump include prediction of slump and strength of ready mix concrete containing retarders and high strength concrete containing silica fume and plasticizers (Dias & Pooliyadda, 2001), predicting slump of fly ash

* Corresponding author.
E-mail addresses: chandwani2@yahoo.com (V. Chandwani), agrawal_vinay_2000@yahoo.com (V. Agrawal), ravindranagar@hotmail.com (R. Nagar).

http://dx.doi.org/10.1016/j.eswa.2014.08.048
0957-4174/

and slag concrete (Yeh, 2006), modeling slump of high strength concrete (Oztas et al., 2006), modeling slump of high performance concrete (Yeh, 2007) and modeling and analysis of concrete slump using laboratory test results (Jain, Jha, & Misra, 2008).

Back-propagation neural network (BPNN) due to its ability to map complex non-linear and unknown relationships is a preferred choice among researchers for modeling unstructured problems. BPNN is a multi-layer feed-forward neural network (MFNN) trained using back-propagation (BP) algorithm. The BP algorithm is a local search algorithm which employs gradient descent to iteratively update the weights and biases of the neural network, minimizing the performance function commonly measured in terms of a squared error between the actual and ANN predicted output. Despite its popularity as a universal function approximator and easy implementation, the BP algorithm is faced with inherent drawback of getting trapped in local minima and slow convergence. The reason for this drawback is attributed to random initialization of synaptic weights and biases prior to training a neural network. With every re-run of neural network during training phase, the BP algorithm evaluates a different set of final weights leading to trained neural network having different prediction performance and convergence speed. In order to minimize the BPNN's probability of inconsistency, it is necessary to develop an effective methodology for improving its prediction performance and convergence to global optima.

To overcome the inherent drawback of BP algorithm, genetic algorithms (GA) have been harnessed for evolving the optimal initial weights and biases for ANN. GA is a gradient free global optimization and search technique inspired by the evolutionary processes namely, natural selection and genetic variation, which allow simultaneous search for optimal solutions in different directions minimizing the chance of getting trapped in a local minimum and faster convergence. Successful implementations of this methodology can be found in Asadi, Shahrabi, Abbaszadeh, and Tabanmehr (2013), Irani and Nasimi (2011), Johari, Javadi, and Habibagahi (2011), Pendharkar (2009), Sedki, Ouazar, and El Mazoudi (2009), Tan, He, Nie, Zhang, and Hu (2014). Despite numerous applications of integrating GA with ANN in various fields of study, the methodology has not been explored so far for modeling slump of concrete. The study deals with amalgamating the universal function approximating ability of BPNN and the global search ability of GA for developing a robust computational tool for modeling slump of RMC.

The study has been organized into sections. Section 2 deals with data collection. Section 3 deals with the methodology, in which neural network modeling of concrete slump, its optimization using genetic algorithm assisted training and statistical performance measures have been discussed. Results, discussions and conclusions and future work have been dealt in Sections 4, 5, 6 respectively.

2. Data collection

The exemplar data for ANN were collected from the same RMC plant to mitigate any chance of change caused in the slump data due to change in composition of concrete mix constituents. The data comprised of concrete design mix constituents consisting of 560 mix proportions namely, cement, fly ash, sand (as fine aggregate), coarse aggregate 20 mm, coarse aggregate 10 mm, admixture, water-binder ratio and corresponding slump value.

3. Methodology

For conducting the study, the Neural Network Toolbox and Global Optimization Toolbox included in the commercially available software MATLAB R2011b (Version 7.13.0.564) were used to implement the BPNN and GA respectively.

3.1. ANN modeling of concrete slump

3.1.1. Preparing training, validation and test data sets

ANN is an information processing paradigm inspired by the learning ability of human brain. ANN therefore requires exemplar patterns to establish the underlying relationships between the input–output data pairs. Moreover, it is also necessary to assess the predictive power of the trained ANN when presented with examples not included in the neural network training. To facilitate training and testing of the neural networks, the collected data were randomized and split into training, validation and test data-sets. 70% of the data were used for training purpose and the remaining 30% data were equally divided and set aside for validation and testing of the trained ANN. The training data-set was used for training the ANN, enabling it to learn the relationships between the input and output data-pairs by systematic updating of the neural network weights and biases using BP algorithm. During the training phase, there is a tendency of the neural network to over-fit or over-learn the exemplar patterns presented during the training phase. This leads to poor generalization of the network when subjected to unseen data. Validation data-set is indirectly used during the training of ANN to monitor the over-fitting of the neural network and to act as a guide to stop the training of the neural network when the validation error begins to rise. Testing of the neural network is done after completion of the training phase. The test data set used during the testing phase evaluates the prediction performance of the trained neural network.

Efficient training of ANN requires that all representative patterns included in the exemplar data, should form a part of the training data-set. Hence, to allow the training data-set extend to the edges of modeling domain, it was ensured that extreme values (maximum and minimum values) of each constituent of total data-set were included in training data-set. Moreover data division should also reflect that training, validation and test data set is representative of the same population. Therefore, three ways split of data was done in such a way that the statistical parameters of Training, Validation and Test data sets viz., maximum value, minimum value, mean and standard deviation of each constituent are marginally different from each other. Table 1 shows the statistical parameters of the data used for training, validation and testing.

3.1.2. Preprocessing of data

The input data and output data generally comprise of different identities either having no or minimum similarities. Preprocessing or normalization of data eliminates the possibility of neural network bias towards the different identities and scales down all the input and output data preferably in a bound range [0, 1] or [−1, 1]. Scaling of inputs to the range [−1, 1] greatly improves the learning speed, as these values fall in the region of sigmoid transfer function where the output is most sensitive to the variations of the input values (Alshihri, Azmy, & El-Bisy, 2009). Linear scaling in the range [−1, 1] has been used in present study having function

$$x_{norm} = \frac{2 * (x - x_{\min})}{(x_{\max} - x_{\min})} - 1 \quad (1)$$

where x_{norm} is the normalized value of the variable x, $x_{\max}$ and $x_{\min}$ are the minimum and maximum values of variable x respectively.

3.1.3. Neural network architecture and training parameters

The architecture of an ANN consists of a number of artificial neurons connected through weighted connections. The artificial

Table 1
Statistical parameters of training, validation and test data-sets.

RMC data constituents	Training				Validation				Test			
	Max	Min	Mean	SD	Max	Min	Mean	SD	Max	Min	Mean	SD
Cement (kg/m^3)	425	100	254.33	53.61	350	120	254.87	57.09	350	120	254.21	52.86
Fly ash (PFA) (kg/m^3)	220	0	82.92	46.17	180	0	83.58	46.97	120	0	82.57	44.27
Sand (kg/m^3)	900	550	780.44	55.37	850	662	780.45	51.13	849	583	780.61	52.35
Coarse aggregate 20 mm (kg/m^3)	788	58	626.26	76.63	760	438	627.54	70.18	745	380	628.08	70.57
Coarse aggregate 10 mm (kg/m^3)	771	343	453.56	95.71	680	343	458.10	100.24	600	343	452.07	93.23
Admixture (kg/m^3)	5.50	1	3.25	0.70	4.70	2.00	3.33	0.65	5.50	1.30	3.34	0.72
Water-binder ratio	0.76	0.36	0.52	0.06	0.75	0.39	0.51	0.07	0.73	0.40	0.51	0.06
Slump (mm)	175	110	150.22	12.19	170	110	150.86	12.15	165	110	149.82	12.97

neurons are synonymous to biological neurons as these constitute the processing elements of an ANN. Each of these neurons has a number of weighted inputs, a transfer function and an output. The utility of transfer functions in neural networks is to introduce non-linearity into the network. A consequence of the non-linearity of this transfer function in the operation of the network, when so introduced, is that the network is thereby enabled to deal robustly with complex, undefined relations between the inputs and the output (Shamseldin, Nasr, & O'Connor, 2002).

Based on the sequence in which the neurons are connected and the way they process information, neural networks can be classified as multilayer perceptrons (MLP), radial basis function (RBF), wavelet neural networks, self organizing maps (SOM) and recurrent networks. Multi-layer feed forward neural networks (MFNNs) are a type of MLP wherein the inter layer neurons are connected in the forward direction only and no loops are allowed. MFNNs are ordered into layers comprising of an "input layer" and an "output layer" joined by a number of "hidden layer/s". By varying the number of hidden layers and hidden layer neurons, improvement in learning and generalization of neural networks can be achieved. As noticed by Sovil, Kvanicka, and Pospichal (1997) it critically depends on the number of training cases, the amount of noise and the degree of complexity of the function or the classification desired to be learnt. Hornik, Stinchcombe, and White (1989) concluded that a three layered feed-forward neural network with back-propagation algorithm can map any non-linear relationship with desired degree of accuracy. Some "rules of thumb" acting as initial guidelines for choosing neural network architecture have been suggested by Berry and Linoff (1997), Blum (1992), Boger and Guterman (1997), Swingler (2001). Nevertheless, the selection of hidden layers and hidden layer neurons is a trial and error process and generally started by choosing a network with minimum number of hidden layers and hidden neurons.

Modeling a particular phenomenon using ANN is started by presenting the information in the form of training data-set. The information provided by the training data-set, there is forward-propagated from the input layer to the output layer through hidden layer/s. The weights and biases of the neural network are adjusted and the predicted output is generated. The computed the error between the actual and predicted output is propagated backwards. Based on the computed error, the weights and biases are adjusted using steepest gradient descent principle employed by BP algorithm. A suitable learning rate and momentum coefficient is employed for efficient learning of the network. A higher learning rate leads to faster training but by doing so it produces large oscillations in the weight change which may force the ANN model to overshoot the optimal weight values. On the other hand, a lower learning rate makes convergence slower and increases the probability of ANN model to get trapped in local minima. The momentum term effectively filters out the high frequency variations of the error surface in the weight space, since it adds the effect of the past weight changes on the current direction of movement in the weight space (Rajasekaran & Pai, 2003). A combined use of these parameters helps the BP algorithm to overcome the effect of local minima. By incorporating these two parameters the change in weights is determined by:

$$\Delta w_n = \alpha \Delta w_{n-1} - \eta \frac{\partial E}{\partial w} \tag{2}$$

$$\text{where, } E = \frac{1}{N}\sum_{i=1}^{N}(T_i - P_i)^2 \tag{3}$$

where w represent the weight allocated to the connection between any two neurons; Δw and Δw_{n-1} are the changes in the neural network weights at n and $n-1$ iterations respectively; α is momentum coefficient; η is the learning rate; E is the computed error; T_i is the target or actual output and P_i is the neural network predicted output.

For the present study, RMC mix proportion ingredients, namely, cement, fly ash (PFA), sand, coarse aggregate (CA) 20 mm, coarse aggregate (CA) 10 mm, admixture and water-binder ratio, form the seven inputs or input neurons for the neural network. Correspondingly, the value of concrete slump forms the output or output neuron for the neural network. Tangent hyperbolic transfer function which maps the inputs between −1 and +1 has been used for hidden layers whereas linear transfer function is used for output layer for comparison of actual values and ANN predicted outputs. For arriving at optimal neural network architecture, seven single hidden layer feed-forward neural network architectures of different complexities having hidden layer neurons in the range 3–11 were trained and validated using training and validation data-set respectively. The neural network architecture having the minimum validation error is selected as the optimal neural network. Flow chart exhibited in Fig. 1 shows the training and validation of the neural networks using BP algorithm.

The systematic updating of weights and biases was performed by Lavenberg Marquardt back-propagation algorithm. Lavenberg–Marquardt back-propagation (LMBP) training algorithm is the fastest converging algorithm preferred for supervised learning. It can be regarded as a blend of steepest descent and Gauss–Newton method, combining the speed of Newton algorithm with the stability of the steepest descent method (Wilamowski, Chen, & Malinowski, 1999). A learning rate of 0.4 and momentum coefficient of 0.9 was used during training of neural networks. Using trial and error, the numbers of hidden layer neurons were established as 8 and neural network model with architecture 7-8-1 was selected for modeling the slump of concrete. The neural network architecture for modeling slump of concrete is shown in Fig. 2.

3.2. Genetic algorithm optimization of neural networks (ANN-GA)

Genetic algorithms are evolutionary optimization algorithms based on the Darwin's principle "Survival of the fittest". They employ computational models of evolutionary processes like

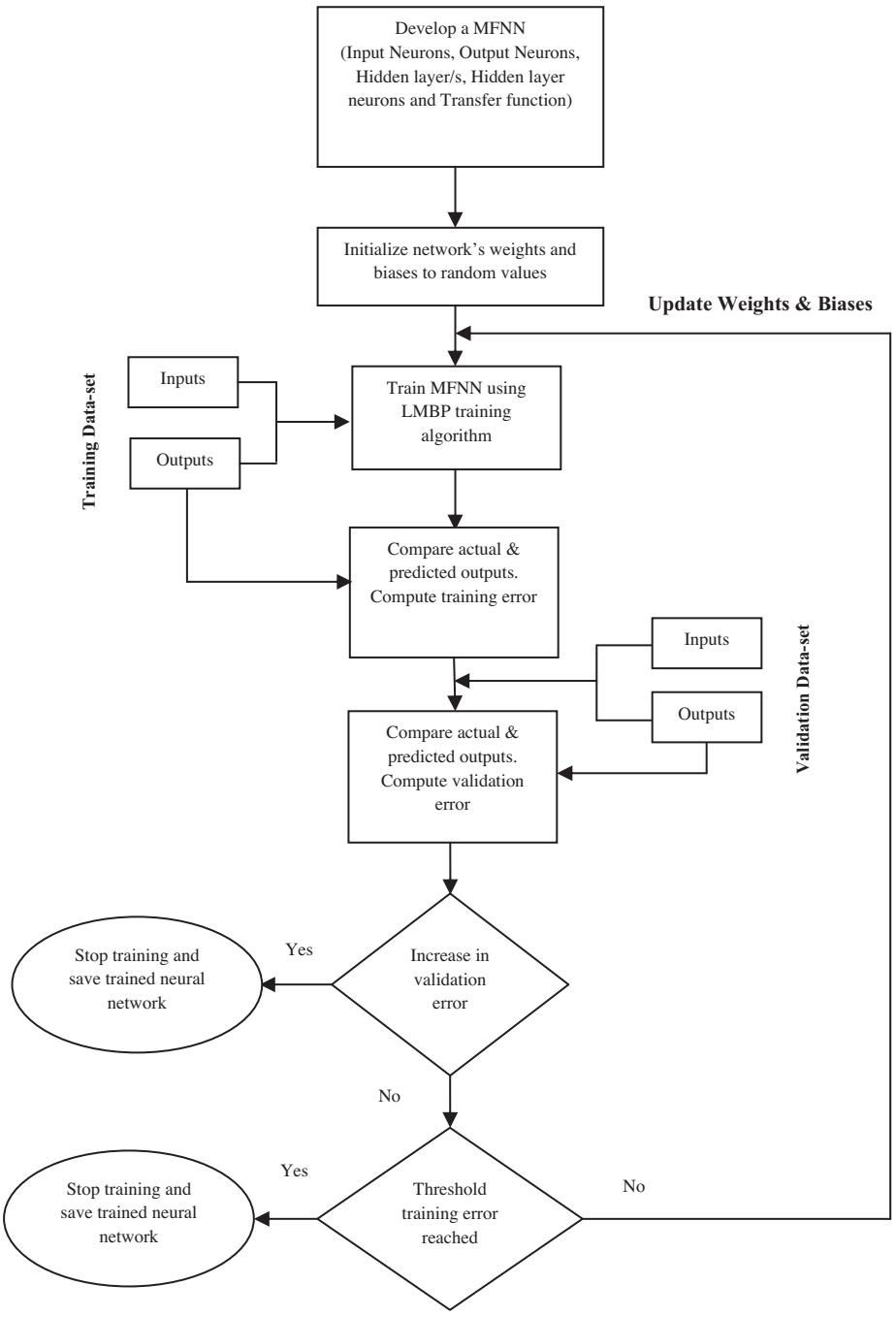

Fig. 1. Training and validation of neural networks using BP algorithm.

selection, crossover and mutation as stochastic search techniques for finding global minimum for complex non-linear problems having numerous sub-optimal solutions. GA's ability to extensively search the solution space and to intensively concentrate on the global optimum provides a perfect blend of exploration and exploitation of the search space. In contrast to BP algorithm that uses local gradient descent for finding the optimal set of neural network connection weights, the GAs parallel nature of global search, gradient free optimization and use of stochastic operators helps in evolving the initial weights for ANN, thereby minimizing the probability of the BP algorithm to get stuck in the local minima.

For optimizing the performance of ANN and to minimize the drawback of BP algorithm, GA is hybridized with ANN. This methodology involves two stages. In the first stage, ANN is trained using GA. GA is used for evolving the optimal set of initial weights and biases for training of the neural network. This is accomplished by simultaneous search performed by GA in all possible directions in the search space and narrowing down to the region where there is maximum probability of finding the optimal weights and biases. The second stage involves training of neural network using BP algorithm. The training is started by initializing the BP algorithm with set of initial weights and biases evolved using GA assisted training of ANN. This initialization of ANN with optimal weights and biases is harnessed by BP algorithm to carry forward the search for the global optima started by GA through fine tuning of neural network's weights and biases. The different steps of this methodology are presented in Fig. 3 and are summarized as under.

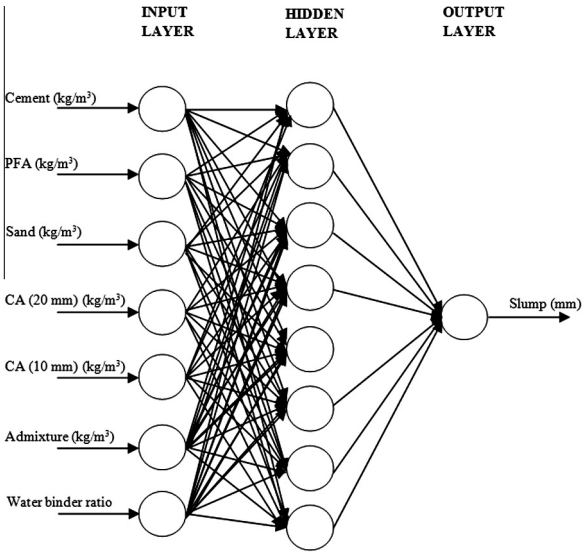

Fig. 2. Architecture of neural network (7-8-1).

3.2.1. Initialization of genetic algorithm

GA is a population based heuristic technique and requires initialization with an initial population. The solutions to the problem are encoded as genes and these form initial population of chromosomes. The chromosomes resemble initial guesses to the probable solutions. These probable solutions are distributed randomly in the search space. In the present study initial population comprises of neural network weights and biases. For 7-8-1 architecture of neural network, the number of weights and biases are 73. These 73 weights and biases are coded as genes of the chromosomes. Since each weight and bias value is a real number, hence it is expressed as a real number.

Since in GA every chromosome in the population represents a potential solution therefore, the initial population size should be chosen to promote the best solution in the search space leading to global optimization of the problem. A higher population size involves greater computational time whereas in case of small population size, the quality of solution is left to the vagaries of chance. In the present study an initial population size of 50 chromosomes is used.

3.2.2. Evaluating fitness of chromosomes

The fitness function for each probable solution or chromosome is evaluated. Fitness function forms a measure of distinguishing optimal solution from numerous sub-optimal solutions by evaluating the ability of the possible solutions to survive or biologically speaking, it test's the reproductive efficiency of chromosomes. The training data-set consisting of input–output data pairs are presented to the neural network. Each chromosome comprising of weights and biases is assigned to the ANN. The ANN through forward propagation of information computes the root mean square error (RMSE) between the actual and the predicted slump value. The fitness of each chromosome is computed using:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(T_i - P_i)^2} \quad (4)$$

where T_i and P_i denote the target or observed values and ANN predicted concrete slump values respectively.

3.2.3. Selecting the fitter chromosomes

GA uses the evolution operator selection, for selecting the fitter chromosomes. The selection procedure is synonymous to a filtering membrane, which allows chromosomes having high fitness to pass on their genes to next generation while prohibiting the entrance of low fitness chromosomes, thereby guiding the algorithm to search the promising regions of the solution space. The present study uses roulette wheel selection strategy which allows probability of selection proportional to the fitness of the chromosome. The basic advantage of roulette wheel selection is that it

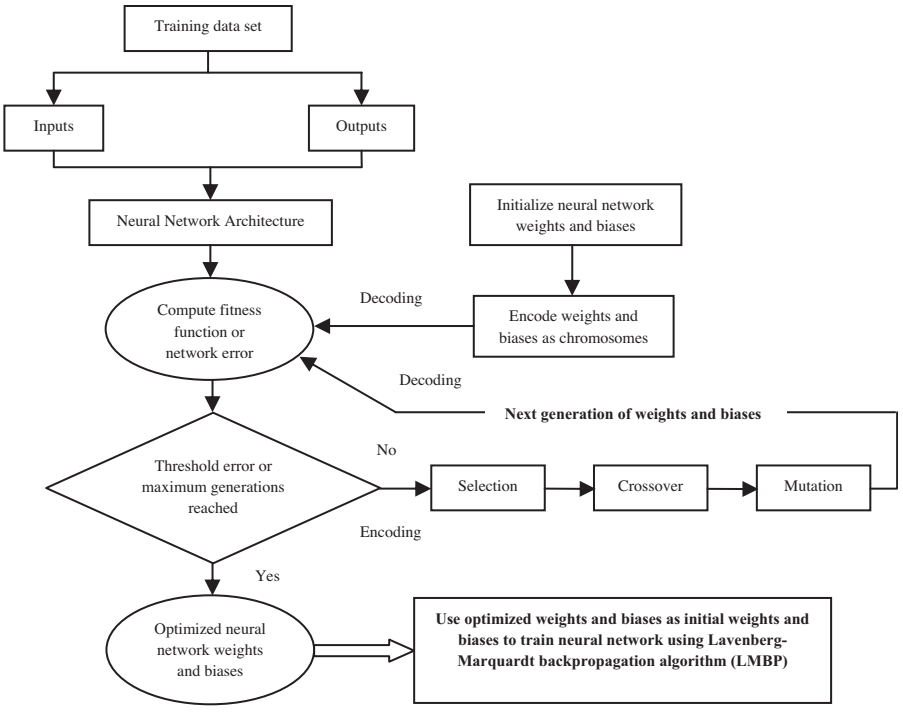

Fig. 3. Flow chart of genetic algorithm assisted training of neural networks.

discards none of the individuals in the population and gives a chance to all of them to be selected (Razali & Geragthy, 2011).

3.2.4. Creating new generation of population

The power of genetic algorithms arises primarily from crossover and mutation (Lin, Lee, & Hong, 2003). GA's stochastic operations in the form of crossover and mutation, allow GA to produce next generation of population. Crossover is a recombination operator that selects a random pair of two chromosomes for mating and swaps the genes between the chromosomes based on the cross site selected along the string length of the chromosome. Crossover operator thus generates new population by extracting the strengths of two individuals and produces new individuals in the hope that these individuals will be better than their parents. Probability of crossover is a parameter to describe how often the crossover will be performed. The present utilized the scattered crossover with probability 0.9 for recombining the two parent chromosomes for producing a fitter child.

The mutation operation adds diversity to the population helping the algorithm to attain a larger exploratory space thereby, preventing the search process to fall into local minima. Viewed as a background operator, mutation exploits the current population to find better individuals. It also plays an important part in recovering the genetic information lost inadvertently during the crossover operations. It also keeps the pool of chromosomes well stocked thus ensuring the dynamics of the creating new generation. The probability of mutation decides how often the parts of the chromosomes will be mutated. Too high mutation rate increases the search space to a level that convergence or finding global optima becomes a difficult issue. Whereas a lower mutation rate drastically reduces the search space and eventually leads genetic algorithm to get stuck in a local optima. The present study uses uniform mutation with mutation rate 0.01. The procedure for creating new population of chromosomes is continued till maximum generation limit is achieved or the fitness function reaches a saturation level. Maximum number of generations used for present study is 100.

3.2.5. Fine tuning of the initial weights and biases using BP algorithm

The initial weights and biases evolved using GA in step 1 to 4 is assigned to BP algorithm. The ANN is trained using these initial set of weights and biases. The BP algorithm through forward propagation of information and back-propagation of errors, allows fine tuning of weights and biases to render the RMSE error between the actual and predicted slump values a minimum.

The flow-chart of genetic algorithm training of ANN for evolving the optimal weights and biases and subsequent fine tuning of these weights and biases is shown in Fig. 3.

3.3. Performance evaluation of trained models

In the present study six different statistical parameters have been employed for judging the performance of the trained ANN models. The parameters include: root mean square error (RMSE), mean absolute percentage error (MAPE), coefficient of correlation (R), coefficient of efficiency (E), root mean square error to observation's standard deviation ratio (RSR) and normalized mean bias error (NMBE). The above performance statistics were evaluated using:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(T_i - P_i)^2} \tag{5}$$

$$MAPE\,(\%) = \frac{1}{N}\sum_{i=1}^{N}\frac{|T_i - P_i|}{T_i} \times 100 \tag{6}$$

$$R = \left(\frac{\sum_{i=1}^{N}((T_i - \overline{T})(P_i - \overline{P}))}{\sqrt{\sum_{i=1}^{N}(T_i - \overline{T})^2 \sum_{i=1}^{N}(P_i - \overline{P})^2}}\right) \tag{7}$$

$$E = 1 - \frac{\sum_{i=1}^{N}(T_i - P_i)^2}{\sum_{i=1}^{N}(T_i - \overline{T})^2} \tag{8}$$

$$RSR = \frac{RMSE}{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(T_i - \overline{T})^2}} \tag{9}$$

$$NMBE\,(\%) = \frac{1/N\sum_{i=1}^{N}(P_i - T_i)}{1/N\sum_{i=1}^{N}T_i} \times 100 \tag{10}$$

where T_i and P_i denote the target or observed values and ANN predicted values and $\overline{T}$ and $\overline{P}$ represent the mean observed and mean ANN predicted values, respectively. N represents the total number of data.

RMSE statistics compares the observed values to the predicted values and computes the square root of the average residual error. A lower value of RMSE indicates good prediction performance of the model. But RMSE gives more weightage to large errors (Kisi, Shiri, & Tombul, 2013). MAPE is a dimensionless statistics that provides an effective way of comparing the residual error for each data point with respect to the observed or target value. Smaller values of MAPE indicate better performance of the model and vice versa. Pearson's correlation coefficient (R) and coefficient of determination (R^2) measure the strength of association between the two variables. R and R^2 statistics are dependent on the linear relationships between the observed and predicted values and may sometimes give biased results when this relationship is not linear or when the values contain many outliers. For perfect association between the observed and predicted values, the value of R^2 is unity. The coefficient of efficiency (E) or Nash Sutcliffe efficiency (Nash & Sutcliffe, 1970) is a ratio of residual error variance to measured variance in observed data. A value close to unity indicates the accuracy of model. RSR statistics was formulated by Moriasi et al. (2007). RSR incorporates the benefits of error index statistics and includes a scaling/normalization factor, so that the resulting statistic and reported values can apply to various constituents (Chen, Xu, & Guo, 2012). The optimal value of RSR is zero. Hence a lower value of RSR indicates good prediction. NMBE measures the ability of the model to predict a value which is situated away from the mean value. A positive NMBE indicates over-prediction and a negative NMBE indicates under-prediction of the model (Srinivasulu & Jain, 2006). A combined use of the performance metrics narrated above can provide an unbiased estimate for prediction ability of the neural network models.

4. Results

As discussed in the previous sections, the ANN is trained using GA for evolving the optimal set of initial weights and biases for subsequent training of neural networks using BP algorithm. GA was able to search the optimal values of weights and biases in 32 generations (Fig. 4). The time taken by GA to reach the saturation RMSE (fitness function) 9.4308 mm was evaluated as 32.6822 s. During this period, GA performed 1600 function evaluations for arriving at an optimized value of fitness function.

The neural network architecture selected for the modeling slump of concrete (7-8-1) is trained using BP algorithm. The ANN-GA model was initialized with optimal weights and biases derived through GA assisted training of ANN. The hybrid model was able to achieve the desired performance goal 0.003 in 43 epochs taking 1.3728 s (Fig. 5(a)). The same neural network

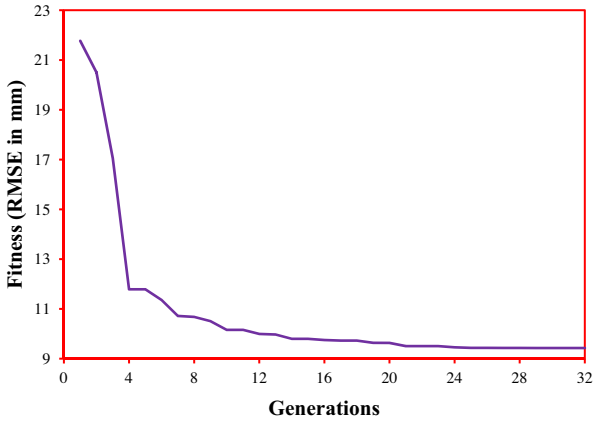

Fig. 4. Evolving optimal weights and biases using genetic algorithms.

architecture was trained using BP algorithm initialized with random draw of weights and biases. The BPNN algorithm took 2688 epochs and 68.604 s to reach the desired performance (Fig. 5(b)).

Both ANN-GA and BPNN models subsequent to training were validated and tested. The results in terms of the performance statistics are presented in Table 2. The results of GA assisted training of ANN have also been included in table.

The entire RMC data was also used for evaluating the prediction ability of the trained models viz., BPNN and ANN-GA. The regression plots showing the prediction of trained BPNN and ANN-GA models are exhibited at Fig. 6(a) and (b) respectively. The statistical performance for the entire data set is tabulated at Table 3.

5. Discussions

Analyzing the results it can be seen that by initializing the BP algorithm with optimal weights and biases, its drawback of getting stuck in local minima and slow convergence can be easily avoided. In comparison to BPNN learning which took 2688 epochs and 68.604 s to reach the desired level of performance, the ANN-GA model took merely 43 epochs and a total time of 34.055 s (including GA time) to achieve the same performance.

Table 2
Statistical performance of ANN models for training, validation and test data-sets.

Model	RMSE (mm)	MAPE (%)	R	E	RSR	NMBE (%)
Training						
GA	9.4308	4.8595	0.6322	0.3995	0.7749	0.1024
BPNN	3.0638	1.3714	0.9678	0.9366	0.2518	−0.0144
ANN-GA	1.8494	0.9298	0.9884	0.9769	0.1520	0.0048
Validation						
GA	19.1416	10.9884	0.6593	−1.5108	1.5846	9.8154
BPNN	3.2567	1.5151	0.9633	0.9273	0.2696	0.1064
ANN-GA	2.6895	1.2527	0.9754	0.9504	0.2226	0.0293
Testing						
GA	30.6507	19.6310	0.4194	−4.6479	2.3765	18.6721
BPNN	3.3409	1.4807	0.9667	0.9329	0.2590	0.2802
ANN-GA	3.0703	1.4805	0.9753	0.9436	0.2375	0.1012

The statistical performance metrics shows that GA alone cannot effectively train an ANN. This is proved by a high training RMSE, MAPE statistics of 9.4308 mm and 4.8595% respectively and lower correlation coefficient (R) statistics of 0.6322. Moreover, negative values of statistics E, −1.5108 and −4.6479 and very high value of NMBE, 9.8154% and 18.6721% during validation and testing respectively, indicates that training of ANN by GA alone leads to unacceptable performance. The second phase comprising of BP algorithm training ensures that the initial weights and biases evolved using GA, are further fine tuned to increase the prediction performance of the neural network. The ANN-GA model gave the best training RMSE, MAPE, R, E, RSR and NMBE statistics of 1.8494 mm, 0.9298%, 0.9884, 0.9769, 0.1520 and 0.0048% respectively. ANN-GA also provided the best performance statistics during validation and testing of the trained neural network. The NMBE statistics for training, validation and testing of BPNN model was evaluated as −0.0144%, 0.1064% and 0.2802% respectively. The negative value of this statistics during training and positive values during validation and testing indicate the prediction inconsistency of the BPNN model. A positive and lower value of statistics NMBE 0.0048%, 0.0293% and 0.1012% for ANN-GA model during training, validation and testing phases respectively shows, its consistency and improved prediction performance.

The performance statistics computed for the entire data-set using the trained ANN-GA model, shows a lower RMSE, MAPE

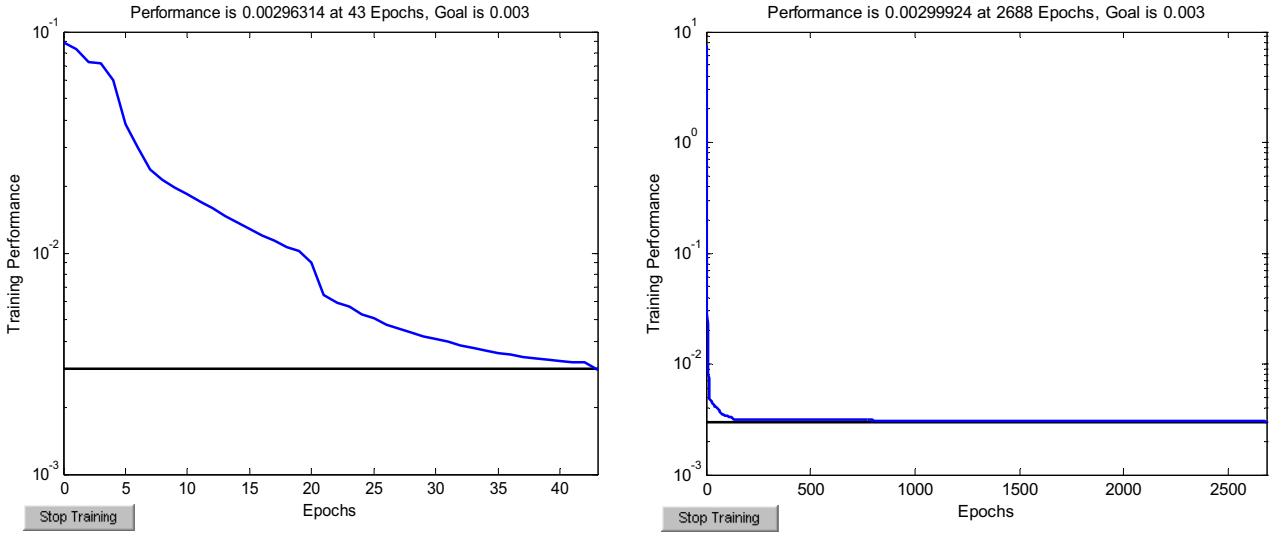

Fig. 5. Training of ANN-GA and BPNN models.

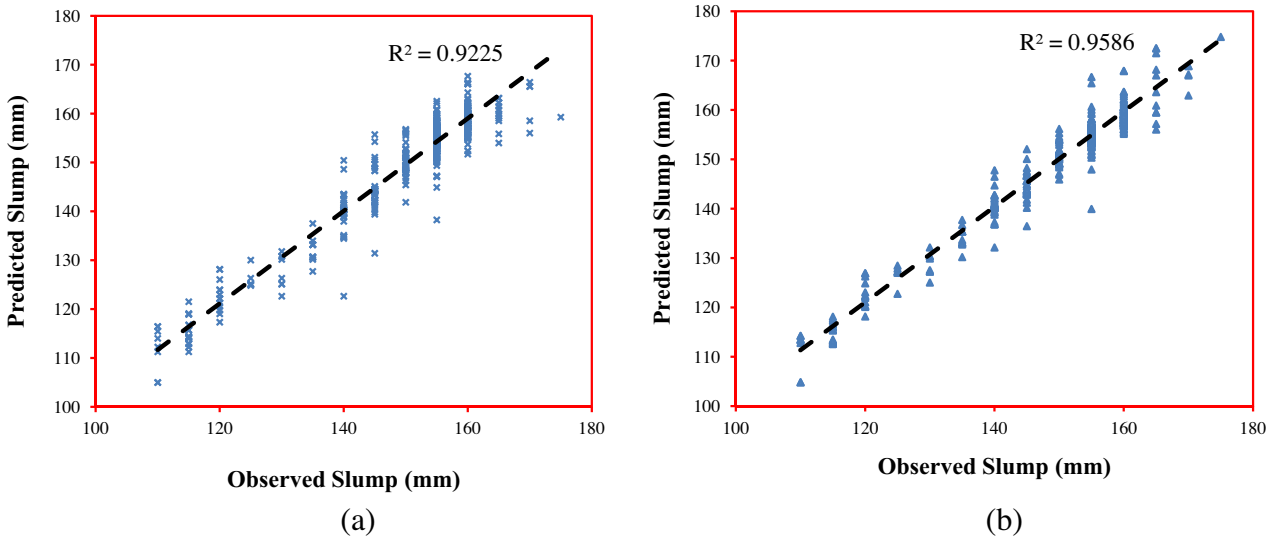

Fig. 6. Regression plot of BPNN and ANN-GA predicted slump versus observed slump.

Table 3
Statistical performance of the trained ANN models for the entire data-set.

Model	RMSE (mm)	MAPE (%)	R	E	RSR	NMBE (%)
BPNN	3.4634	1.6782	0.9605	0.9204	0.2822	−0.3163
ANN-GA	2.4994	1.1979	0.9791	0.9585	0.2037	0.0349

and RSR value of 2.4994 mm, 1.1979% and 0.2037 respectively and higher E and R value of 0.9585 and 0.9791 respectively, in comparison to trained BPNN model. Moreover, NMBE statistics value of −0.3163% and 0.0349% for BPNN and ANN-GA models shows that, BPNN model is under-predicting the slump data, whereas ANN-GA achieved near to optimal prediction accuracy. Overall, the performance metrics shows that, ANN-GA model has consistently outperformed the BPNN model.

6. Conclusions and future work

In this paper, the optimal initial weight and biases for ANN have been evolved using GA assisted training of ANN. The hybridization of two distinct nature inspired computational techniques has been proposed for covering up the drawback of BP algorithm to converge at suboptimal points and slow speed of convergence. The proposed hybrid technique harnessed GA to evolve the optimal set of initial neural network weights and biases which were further fine tuned using Lavenberg Marquardt back-propagation training algorithm. This two stage optimization of ANN helped in deriving the best from global search ability of GA and local search ability of BP algorithm. The study showed that in comparison to BPNN approach which uses gradient descent for updating the weights and biases, the hybrid ANN-GA model which utilized genetic algorithm derived weights and biases, gave consistent predictions during training, validation and testing phases, indicating the robustness of the hybrid modeling approach. Moreover, the ANN-GA model in comparison to BPNN model, took almost half the time in reaching the desired performance, indicating its fast convergence to global optimum. The proposed model based on past experimental data can be very handy for predicting the complex material behavior of concrete in quick time. It can be used as a decision support tool, aiding the technical staff to easily predict the slump value for a particular concrete design mix. This technique will considerably reduce the effort and time to design a concrete mix for a customized slump without undertaking multiple trials.

In the present study trial and error technique has been employed for determining the optimal architecture of the neural network. The future work will concentrate on evolving the optimal number of the hidden layers and hidden layer neurons, transfer function, learning rate and momentum coefficient using genetic algorithms. Another direction for future study will be the use of Extreme Learning Machines (ELM) for modeling concrete's material behavior. ELMs are single layer feed-forward neural networks (SLFN) which are known for their faster convergence and minimal human intervention. However, the weights and biases of ELM are randomly initialized, which in some cases may affect it's the generalization performance. GA can be hybridized with ELM for evolving optimal initial weights and biases, thereby improving its overall performance. GA can also be harnessed for evolving the number of hidden layer neurons for ELM to strike a balance between the generalization and convergence speed.

References

Abdollahzadeh, A., Masoudnia, R., & Aghababaei, S. (2011). Predict strength of rubberized concrete using artificial neural network. *WSEAS Transactions on Computers, 10*(2), 31–40.
Alshihri, M. M., Azmy, A. M., & El-Bisy, M. S. (2009). Neural Networks for predicting compressive strength of structural light weight concrete. *Construction and Building Materials, 23*(6), 2214–2219.
Asadi, S., Shahrabi, J., Abbaszadeh, P., & Tabanmehr, S. (2013). A new hybrid neural network for rainfall-runoff process modeling. *Neurocomputing, 121*, 470–480.
Bal, L., & Buyle-Bodin, F. (2013). Artificial neural network for predicting drying shrinkage of concrete. *Construction and Building Materials, 38*, 248–254.
Berry, M. J. A., & Linoff, G. (1997). *Data mining techniques*. New York: John Wiley & Sons.
Blum, A. (1992). *Neural networks in C++: An object-oriented framework for building connectionist systems*. New York: John Wiley & Sons.
Boger, Z., & Guterman, H. (1997). Knowledge extraction from artificial neural network models. *IEEE Systems, Man, and Cybernetics Conference*. Orlando, Florida.
Chen, H., Xu, C., & Guo, S. (2012). Comparison and evaluation of multiple GCM's, statistical downscaling and hydrological models in the study of climate change impacts on runoff. *Journal of Hydrology, 434–435*, 36–45.
Demir, F. (2008). Prediction of elastic modulus of normal and high strength concrete by artificial neural networks. *Construction and Building Materials, 22*(7), 1428–1435.
Dias, W. P. S., & Pooliyadda, S. P. (2001). Neural networks for predicting properties of concrete with admixtures. *Construction and Building Materials, 15*(7), 371–379.

Duan, Z. H., Kou, S. C., & Poon, C. S. (2013). Prediction of compressive strength of recycled aggregate concrete using artificial neural networks. *Construction and Building Materials, 40*, 1200–1206.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feed forward networks are universal approximators. *Neural Networks, 2*(5), 359–366.

Irani, R., & Nasimi, R. (2011). Evolving neural network using real coded genetic algorithm for permeability estimation of the reservoir. *Expert Systems with Applications, 38*(8), 9862–9866.

Jain, A., Jha, S. K., & Misra, S. (2008). Modeling and analysis of concrete slump using artificial neural networks. *Journal of Materials in Civil Engineering, 20*(9), 628–633.

Ji, T., Lin, T., & Lin, X. (2006). Concrete mix proportion design algorithm based on artificial neural networks. *Cement and Concrete Research, 36*(7), 1399–1408.

Johari, A., Javadi, A. A., & Habibagahi, G. (2011). Modelling the mechanical behaviour of unsaturated soils using a genetic algorithm-based neural network. *Computers and Geotechnics, 38*(1), 2–13.

Kisi, A., Shiri, J., & Tombul, M. (2013). Modeling rainfall-runoff process using soft computing techniques. *Computers and Geosciences, 51*, 108–117.

Lin, W. Y., Lee, W. Y., & Hong, T. P. (2003). Adapting crossover and mutation rates in genetic algorithms. *Journal of Information Science and Engineering, 19*, 889–903.

Moriasi, D. N., Arnold, J. G., Van Liew, M. W., Bingner, R. L., Harmel, R. D., & Veith, T. L. (2007). Model evaluation guidelines for systematic quantification of accuracy in watershed simulations. *Transactions of the ASABE, 50*(3), 885–900.

Naderpour, H., Kheyroddin, A., & Ghodrati Amiri, G. (2010). Prediction of FRP – Confined compressive concrete using artificial networks. *Composite Structures, 92*(12), 2817–2829.

Nash, J. E., & Sutcliffe, J. V. (1970). River flow forecasting through conceptual models Part I – A discussion of principles. *Journal of Hydrology, 10*(3), 282–290.

Oztas, A., Pala, M., Ozbay, E., Kanca, E., Caglar, N., & Bhatti, M. A. (2006). Predicting the compressive strength and slump of high strength concrete using neural network. *Construction and Building Materials, 20*(9), 769–775.

Parichatprecha, R., & Nimityonskul, P. (2009). Analysis of durability of high performance concrete using artificial neural networks. *Construction and Building Materials, 23*(2), 910–917.

Pendharkar, P. C. (2009). Genetic algorithm based neural network approaches for predicting churn in cellular wireless network services. *Expert Systems with Applications, 36*(3), 6714–6720. Part 2.

Rajasekaran, S., & Pai, G. A. V. (2003). *Neural networks, fuzzy logic and genetic algorithms: synthesis & applications*. New Delhi: Prentice-Hall of India Private Limited.

Razali, N. M., & Geragthy, J. (2011). Genetic Algorithm performance with different Selection strategies in solving TSP. In *Proceedings of the World Congress on Engineering 2011* (Vol. II, pp. 1134–1139). .

Sedki, A., Ouazar, D., & El Mazoudi, E. (2009). Evolving neural networks using real coded genetic algorithm for daily rainfall-runoff forecasting. *Expert Systems with Applications, 36*(3), 4523–4527. Part 1.

Shamseldin, A. Y., Nasr, A. E., & O'Connor, K. M. O. (2002). Comparison of different forms of multi-layer feed-forward neural network method used for river flow forecasting. *Hydrology and Earth System Sciences, 6*(4), 671–684.

Sovil, D., Kvanicka, V., & Pospichal, J. (1997). Introduction to multilayer feed forward neural networks. *Chemo metrics and Intelligent Laboratory Systems, 39*(1), 43–62.

Srinivasulu, S., & Jain, A. (2006). A comparative analysis of training methods for artificial neural network rainfall-runoff models. *Applied Soft Computing, 6*, 295–306.

Swingler, K. (2001). *Applying neural networks: A practical guide*. San Francisco: Morgan Kaufman Publishers Inc.

Tan, M., He, G., Nie, F., Zhang, L., & Hu, L. (2014). Optimization of ultrafiltration membrane fabrication using backpropagation neural network and genetic algorithm. *Journal of the Taiwan Institute of Chemical Engineers, 45*(1), 68–75.

Uysal, M., & Tanyildizi, H. (2012). Estimation of compressive strength of self compacting concrete containing polypropylene fibre and mineral additives exposed to high temperature using artificial neural network. *Construction and Building Materials, 27*(1), 404–414.

Wilamowski, B. M., Chen, Y., & Malinowski, A. (1999). *Efficient algorithm for training neural networks with one hidden layer. Proceedings of the International Joint Conference on Neural Networks (IJCNN'99)*. IEEE (pp. 1725–1728). IEEE.

Yeh, I.-C. (1998). Modeling of strength of high-performance concrete using artificial neural networks. *Cement and Concrete Research, 28*(12), 1797–1808.

Yeh, I.-C. (2006). Exploring concrete slump model using artificial neural networks. *Journal of Computing in Civil Engineering, 20*(3), 217–221.

Yeh, I.-C. (2007). Modeling slump flow of concrete using second-order regressions and artificial neural networks. *Cement and Concrete Composites, 29*, 474–480.