An optimized instance based learning algorithm for estimation of compressive strength of concrete

Behrouz Ahmadi-Nedushan*

Civil Engineering Department, Engineering Faculty, Yazd University, Pajoohesh Street, Safa-ieh, Yazd, P.O. Box 89195-741, Iran

**A R T I C L E  I N F O**

Article history:
Received 28 April 2011
Received in revised form 7 January 2012
Accepted 15 January 2012
Available online 3 February 2012

Keywords:
k nearest neighbor algorithm
Instance based learning
Differential evolution
Data mining
Optimization
Compressive strength

**A B S T R A C T**

This article proposes an optimized instance-based learning approach for prediction of the compressive strength of high performance concrete based on mix data, such as water to binder ratio, water content, super-plasticizer content, fly ash content, etc. The base algorithm used in this study is the k nearest neighbor algorithm, which is an instance-based machine learning algorithm. Five different models were developed and analyzed to investigate the effects of the number of neighbors, the distance function and the attribute weights on the performance of the models. For each model a modified version of the differential evolution algorithm was used to find the optimal model parameters. Moreover, two different models based on generalized regression neural network and stepwise regressions were also developed. The performances of the models were evaluated using a set of high strength concrete mix data. The results of this study indicate that the optimized models outperform those derived from the standard k nearest neighbor algorithm, and that the proposed models have a better performance in comparison to generalized regression neural network, stepwise regression and modular neural networks models.

© 2012 Elsevier Ltd. All rights reserved.

1. Introduction

High-performance concrete has higher strength properties and superior constructability as compared to normal concrete. One of the major differences between the conventional normal strength concrete and high performance concrete is the use of chemical and mineral admixtures (Lim et al., 2004). Chemical admixtures reduce the water content and hence the porosity within the hydrated cement paste. Mineral admixtures act as pozzolanic materials as well as fine fillers; and as a result, the microstructure of hardened cement matrix becomes denser and stronger (Kosmatka et al., 2003).

Among various properties of concrete, the compressive strength is generally regarded to be its most important property. Many other physical properties of concrete such as elastic modulus, water tightness or impermeability appear to have a direct relationship with concrete strength (Gamibhir, 2004). Therefore, the compressive strength is commonly used as the main criterion in defining the required quality of concrete. The compressive strength is usually determined based on a standard uniaxial compression test performed 28 day after casting the concrete. If the test results do not satisfy the required strength, costly remediation efforts must be undertaken. Therefore, an accurate estimation of the compressive strength before the placement of concrete is very important.

In recent years, prediction of the compressive strength of concrete has been an active area of research and different approaches have been proposed to estimate the compressive strength based on the mix proportions of different ingredients. Various authors have applied a multilayer feed-forward artificial neural network (ANN) trained by a back propagation algorithm to predict the compressive strength of concrete (Lai and Serra, 1997; Yeh, 1998; Ni and Wang, 2000; Lee, 2003; Kim et al., 2004; 2005; Jain et al., 2005). Kim et al. (2004) used a feed-forwarded ANN for the prediction. Kim et al. (2005) further enhanced the previously reported ANN results in Kim et al. (2004) by using a probabilistic ANN method to handle uncertainty. They concluded that a probabilistic ANN requires less time for training, as compared to a multilayer feed-forward ANN. Jain et al. (2005) commented on the work reported by Kim et al. (2004) and provided further insight into the implementation of ANN models for concrete mixes. Tesfamariam and Najjaran (2007) used an adaptive neuro fuzzy inference system to predict the compressive strength. Tsai and Lin (2011) used a modular neural network (MNN) to predict the compressive strength. The parameters of MNN were optimized by genetic algorithm.

The research on the prediction of the compressive strength has been focused mainly on application of different variants of ANN,
and other machine learning methods have received little attention in this context.

In this article, an optimized instance-based learning approach is proposed to evaluate the compressive strength of concrete based on mix proportions. This approach combines the $k$ nearest neighbor algorithm (kNNA) with the differential evolution (DE) to develop optimal predictive models. The kNNA is an instance-based machine learning algorithm and the DE is a very recent evolutionary algorithm. Five different models were implemented to quantify the improvements associated with optimization of various effective factors: the number of neighbors, the distance function and the attribute weights. The performance of these models are compared with a generalized regression neural network (GRNN) model, a stepwise regression model, and an optimized MNN.

The kNNA, DE and GRNN are described in details in the next sections followed by a brief description of the concrete mix data set used for the development and evaluation of the proposed methods. Finally, the results of the analysis of the data and performances of five models are presented and discussed.

2. $k$ Nearest neighbor algorithm

The kNNA is an instance-based learning algorithm, in which the data set are stored, so that a prediction for a new record may be found by comparing it to the most similar records in the data set (Larose, 2005). In the kNNA it is assumed that observations (i.e., data points) that are close in the space of the data attributes (i.e., mixture ingredients) will be also close to each other in the space of the response variable (i.e., the compressive strength of concrete). The response value is predicted by considering only $k$ closest neighbors in the space of the data attributes and using a pre-defined function of the response values of the $k$ nearest neighbors. In the standard kNNA the average function is generally used (Myatt, 2007).

In any kNNA a meaure of closeness between observations should be specified using one of the many different metrics. In the standard kNNA, the Euclidean distance function is used for calculating the distance of continuous variables. The Euclidean distance between observations $X_i$ and $X_j$ in the data set, in the space of the data attributes, is calculated as

$$d_{E}(X_i, X_j) = \sqrt{\sum_{l=1}^{n}(x_{il} - x_{jl})^2} \quad j \neq i, \quad j = 1, 2, \ldots, n$$

(1)

where $D$ represents the number of attributes, $x_{il}$ and $x_{jl}$ are components of vectors $X_i$ and $X_j$ (i.e., corresponding attributes of observations $i$ and $j$, respectively) and $n$ is the number of observations. The Euclidean distance is a special case of the Minkowski metric

$$d(X_i, X_j) = \left(\sum_{l=1}^{n}|x_{il} - x_{jl}|^p\right)^{1/p}$$

(2)

where $p$ is a real number greater than or equal to one. The Minkowski distance function becomes the Manhattan or City Block distance function if $p=1$, and the Euclidean distance function if $p=2$.

For calculation of the unknown response value of $y_i$ associated with attributes of $X_0$, the distances calculated based on Eqs. (1) or (2) are sorted and the $k$ nearest neighbors of $X_0 (X_{1,0}, X_{2,0}, \ldots, X_{k,0})$ are then selected. In the simplest form of kNNA, the average of the response values of the $k$ nearest neighbors, is used to estimate the unknown response value of $y_i$

$$y_i = f(X_0) = \frac{\sum_{l=1}^{k}f(X_l)}{k}$$

(3)

In Eq. (3) the weights of all data points are assumed to be equal regardless of the proximity of a data point to the target $X_0$. Alternatively, to include a higher influence for data closer to the target $X_0$, a weight ($w_l$) which is inversely proportional to its distance from $X_0$ may be applied

$$y_i = f(X_0) = \frac{\sum_{l=1}^{k}w_lf(X_l)}{k} \quad w_l = \frac{1}{d_{E}(X_l, X_0)} \quad l = 1, \ldots, k$$

(4)

Before calculating the distances and selecting the $k$ nearest neighbors, it is important to normalize the attributes since otherwise attributes with large values, can greatly suppress the influence of other attributes with smaller values. For continuous variables, either the min-max normalization or the z-score standardization may be used (Larose, 2005).

The standard kNNA has the following properties: (1) all neighbors are given equal importance and the average function is used for calculating the response value of an unknown observation, (2) all normalized attributes are assumed to be equally important and therefore equal weights are assigned to all attributes and (3) the Euclidean distance function is used to calculate the distances. Obviously, the performance of kNNA depends on a number of factors including the number of neighbors considered, the distance function used, and the weights assigned to attributes.

In this study, a series of analyses is performed to investigate the influence of the above factors on the performance of the algorithm. An evolutionary algorithm is also used to select the optimal setting of these factors so that the best performance can be achieved.

In order to apply the kNNA, the number of the nearest neighbors, $k$, should be selected. The best strategy is to limit the values of $k$ within a range and to select the value that gives the best performance. In a standard kNNA, the Euclidean metric is used as the measure of the closeness. In this study the more general Minkowski distance function is used where the unknown parameter $p$ is obtained by optimization.

In the standard kNNA, it is also assumed that all attributes have equal importance. Usually, not all of the input variables will be equally informative since some may have no significant relationship with the output variable being modeled. The existence of irrelevant or less relevant attributes decreases the accuracy of a kNNA since they affect the distances between the observations. The importance of attributes can be considered by assigning an appropriate numerical weight to each attribute. The numerical weight is an indicator of the relevance of the attribute for the prediction of output variable and higher weights are assigned to the more important attributes.

The process of selection of attributes can be regarded as a particular case of determination of these weights, where the assigned weights are binary, taken as zero to ignore the presence of a particular attribute or as one to include a particular attribute.

Development of the best predictive model with optimal attribute weights or attribute selection can be defined as an optimization problem where the goal is to find the optimal weights that provide the best performance. The performance of models can be evaluated by measures of differences between the predicted responses ($P$) and the observed responses ($O$). The coefficient of determination ($r^2$) and the root mean squared error (RMSE) are used to evaluate the performance of the models

$$r^2 = \left[\frac{\sum_{i=1}^{n}(O_i-P_i)(P_i-P)}{\sqrt{\sum_{i=1}^{n}(O_i-P)^2} \sqrt{\sum_{i=1}^{n}(P_i-P)^2}}\right]^2$$

(5)

$$RMSE = \left(\frac{\sum_{i=1}^{n}(P_i-O_i)^2}{n}\right)^{0.5}$$

(6)
where $\overline{O}$ and $P$ are the mean values of the observed and predicted responses respectively.

The RMSE is defined as the objective function to be minimized for different developed kNNa based models described in the next sections. These models are optimized by DE algorithm, a description of which is given in the following section.

### 3. Differential evolution algorithm

DE has recently been emerged as an efficient algorithm for global optimization over continuous spaces. DE has been proven to be robust and easy to implement in optimization of various problems (Price et al., 2005; Storn, 2008). DE optimizes a problem by maintaining a population of candidate solutions and creating new candidate solutions using the operators of mutation, crossover, and selection. Each candidate solution is a vector that contains as many variables as the dimensions of the problem.

The current population, symbolized by $P_{og}$ is composed of vectors, $X_{ig}$, that have already been found to be acceptable either as initial points, or by comparison with other vectors (Price et al., 2005). The population is defined as

$$P_{Xg} = (X_{ig})_{i=1,2,...,N_P}, g = 1,...,g_{\text{max}}$$

$X_{ig} = (X_{i1},...,X_{id})_{j=1,2,...,D}$.

In the Eq. (7), $N_P$ denotes the number of population vectors, the index $g$ represents the generation counter, $g_{\text{max}}$ is maximum number of generations and $i$ is the population index parameters within each vector (Storn, 2008).

The process of optimization with DE starts with an initial population of $N_P$ with vectors of $D$ dimensions. The initial value $(g=1)$ of the $j$th parameter of $ith$ vector is determined as

$$X_{i1} = b_{li} + rand(0,1) \times (b_{ui} - b_{li})$$

where the vectors $b_l$ and $b_u$ are the lower and upper bounds of the parameter vectors $X$ and $rand(0,1)$ is a uniformly distributed random number between 0 and 1 (Storn, 2008).

After initialization, DE mutates randomly chosen vectors to form an intermediary population, $P_{ig}$ of $N_P$ mutant vectors, $V_{ig}$

$$P_{ig} = (V_{i1},...,V_{id})_{j=1,2,...,D}$$

Different mutation strategies are defined in DE, depending on the choice of the three individuals used to build the mutated vectors (Feoktistov, 2006). A few of these strategies can be expressed as

$$V_{ig} = X_{ag} + F(X_{ig} - X_{bg})$$

where $a,b,c \in \{1,...,N_P\}$ and $a \neq b \neq c$.

$F$ is a control parameter, which manages the trade-off between exploitation and exploration of the design space (Feoktistov, 2006). A larger value of $F$ generally results in a more diverse population. Zaharie modified the standard DE algorithm by multiplying $F$ by a standardized random normal variable (Zaharie, 2002).

To enhance the diversity of population, each vector in the current population is then recombined with a mutant to produce the trial population, $P_{tg}$ of $N_P$ trial vectors, $V_{tg}$

$$P_{ig} = (U_{i1},...,U_{id})_{j=1,2,...,D}$$

This is performed using the crossover operation. A binomial crossover operator is expressed as

$$U_{ig} = u_{ig} = \begin{cases} 
    v_{ij} & \text{if } rand(0,1) \leq C_r 	ext{ or } j = j_{\text{rand}}, \\
    X_{ij} & \text{otherwise}
\end{cases}$$

where $C_r \in [0,1]$ is the crossover rate, that controls the fraction of parameter values that are copied from mutant. $j = 1,2,...,D$; $rand(0,1)$ is a uniformly distributed random number generated for the $j$th parameter. Moreover, the trial parameter with randomly chosen index, $j_{\text{rand}}$ is taken from the mutant vector $V_{ig}$ to ensure that trial vector does not duplicate $X_{ig}$.

The standard notation used for defining DE variants is given by DE/ab/c where a denotes the base vector, b denotes the number of difference vectors and c represents the crossover.

Although DE has only a few control parameters, the proper setting of these parameters is often critical for its performance (Noman and Iba, 2008). A good value of population size generally results in a more diverse population. DE optimizes a problem by maintaining a population of candidate solutions and creating new candidate solutions using the operators of mutation, crossover, and selection. Each candidate solution is a vector that contains as many variables as the dimensions of the problem.

The current population, symbolized by $P_{og}$ is composed of vectors, $X_{ig}$, that have already been found to be acceptable either as initial points, or by comparison with other vectors (Price et al., 2005). The population is defined as

$$P_{Xg} = (X_{ig})_{i=1,2,...,N_P}, g = 1,...,g_{\text{max}}$$

$X_{ig} = (X_{i1},...,X_{id})_{j=1,2,...,D}$.

In the Eq. (7), $N_P$ denotes the number of population vectors, the index $g$ represents the generation counter, $g_{\text{max}}$ is maximum number of generations and $i$ is the population index parameters within each vector (Storn, 2008).

The process of optimization with DE starts with an initial population of $N_P$ with vectors of $D$ dimensions. The initial value $(g=1)$ of the $j$th parameter of $ith$ vector is determined as

$$X_{i1} = b_{li} + rand(0,1) \times (b_{ui} - b_{li})$$

where the vectors $b_l$ and $b_u$ are the lower and upper bounds of the parameter vectors $X$ and $rand(0,1)$ is a uniformly distributed random number between 0 and 1 (Storn, 2008).

After initialization, DE mutates randomly chosen vectors to form an intermediary population, $P_{ig}$ of $N_P$ mutant vectors, $V_{ig}$

$$P_{ig} = (V_{i1},...,V_{id})_{j=1,2,...,D}$$

Different mutation strategies are defined in DE, depending on the choice of the three individuals used to build the mutated vectors (Feoktistov, 2006). A few of these strategies can be expressed as

$$V_{ig} = X_{ag} + F(X_{ig} - X_{bg})$$

where $a,b,c \in \{1,...,N_P\}$ and $a \neq b \neq c$.

$F$ is a control parameter, which manages the trade-off between exploitation and exploration of the design space (Feoktistov, 2006). A larger value of $F$ generally results in a more diverse population. Zaharie modified the standard DE algorithm by multiplying $F$ by a standardized random normal variable (Zaharie, 2002).

To enhance the diversity of population, each vector in the current population is then recombined with a mutant to produce the trial population, $P_{tg}$ of $N_P$ trial vectors, $V_{tg}$

$$P_{ig} = (U_{i1},...,U_{id})_{j=1,2,...,D}$$

This is performed using the crossover operation. A binomial crossover operator is expressed as

$$U_{ig} = u_{ig} = \begin{cases} 
    v_{ij} & \text{if } rand(0,1) \leq C_r 	ext{ or } j = j_{\text{rand}}, \\
    X_{ij} & \text{otherwise}
\end{cases}$$

where $C_r \in [0,1]$ is the crossover rate, that controls the fraction of parameter values that are copied from mutant. $j = 1,2,...,D$; $rand(0,1)$ is a uniformly distributed random number generated for the $j$th parameter. Moreover, the trial parameter with randomly chosen index, $j_{\text{rand}}$ is taken from the mutant vector $V_{ig}$ to ensure that trial vector does not duplicate $X_{ig}$.

The standard notation used for defining DE variants is given by DE/ab/c where a denotes the base vector, b denotes the number of difference vectors and c represents the crossover.

In this study, the approach for comparison of infeasible points in the standard DE algorithm is modified using pareto-dominance rules. The comparison of target and trial vectors is based on the following rules: (1) a feasible solution is better than an infeasible solution, (2) if both solutions $U_{ig}$ and $X_{ig}$ are feasible, the one with a minimum value of the objective function is selected, (3) if both solutions are infeasible, the preference is given to the less infeasible solution. In this study, in order to implement the third rule, all constraint violations of the target and the trial vectors are computed and the norms of the constraint violations are calculated for both the target and the trial vectors. These norms are then compared and the vector with smaller norm is selected to proceed to the next generation:

$$X_{ig+1} = \begin{cases} 
    X_{ig} & \text{if } \sum_{k=me+1}^{m} (\text{Max}(0,g_k(X_{ig})))^2 \leq \sum_{k=me+1}^{m} (\text{Max}(0,g_k(U_{ig})))^2 \\
    U_{ig} & \text{if } \sum_{k=me+1}^{m} (\text{Max}(0,g_k(X_{ig})))^2 > \sum_{k=me+1}^{m} (\text{Max}(0,g_k(U_{ig})))^2
\end{cases}$$

where $\text{Max}(0,g_k(X_{ig}))$ and $\text{Max}(0,g_k(U_{ig}))$ represent the constraint violations of the target and the trial vectors for constraint function $g_k$, respectively. It should be noted that in the standard DE, the preference is given to the target vector, $X_{ig}$ even if one element of the constraint violation for trial vector $U_{ig}$ is larger than the corresponding element of the target vector (Storn, 2008).

### 4. Parameter setting of the differential evolution algorithm

Although DE has only a few control parameters, the proper setting of these parameters is often critical for its performance (Noman and Iba, 2008). A good value of population size can be found by considering the dimensionality of the problem similar to what is commonly used for the other evolutionary algorithms. For DE, the number of population is generally selected to be ten times of the number of variables (Price et al., 2005).

The performance of the DE is dependent on the settings of the scale factor $F$ and the crossover rate $C_r$. Several studies have been dealt with this matter. Liu and Lampinen (2002a) recommended $F \in [0.5,1]$ and $C_r \in [0.8,1]$ while Liu and Lampinen (2002b) and Rönkkönen et al. (2005) suggested $F=CR=0.9$. The empirical analysis reported by Zielinski et al. (2006) demonstrated that in
many cases, values of $F \geq 0.6$ and $CR \geq 0.6$ lead to an acceptable performance of DE. Based on above studies, values of $F=0.9$ and $CR=0.9$ were selected for the five different models developed and used in this study.

5. Generalized regression neural networks (GRNN)

The GRNN proposed by Specht (1991), is a normalized radial basis function (RBF) network in which there is a hidden unit centered at every training example (Pal et al., 2011). These RBF units are called “kernels” and are usually Gaussian probability density functions. In contrast to a back propagation neural networks that requires an iterative training procedure, a GRNN is trained by a single pass through the training data.

A schematic diagram of a GRNN architecture is presented in Fig. 1. A GRNN consists of four layers. The input units are in the first layer, the second layer has the pattern units, the outputs of this layer are passed on to the summation units in the third layer, and the final layer covers the output units.

The number of neurons in the first layer is equal to number of attributes. The first-layer weights are set to transpose of input vector, and the bias $b$ is set to a column vector of $0.8326/\sigma$. The user chooses the smoothing parameter, $\sigma$. The second layer also has as many neurons as input vectors. The hidden-to-output weights are just the response values, therefore the output is a weighted average of the target values of training cases close to the given input case, wherein the weights are taken proportional to the Euclidean distances between the training input vectors and the test input vector

$$y(x) = E(y|x) = \sum_{i=1}^{n} y_i h_i, \quad h_i = \exp \left(-\frac{d_i^2}{2\sigma^2}\right), \quad d_i^2 = (x-x_i)^T(x-x_i)$$

where $h_i$ denotes the Gaussian radial basis function, $\sigma$ is the smoothing parameter and $d_i^2$ represents the squared Euclidean distance between a test input vector $x$ and $x_i$.

GRNNs training algorithm uses only one adjustable parameter namely the smoothing parameter ($\sigma$) of Gaussian RBF. The optimal value of $\sigma$ can be obtained by leave-one-out cross-validation. For further details of the GRNN readers are referred to Specht (1991).

6. Model training and evaluation

The performance of a model relates to its predictive capability over independent test data. Cross-validation is the evaluation method of choice in most practical limited-data situations (Witten and Frank, 2005). $N$-fold cross-validation uses part of the available data to train the model, and a different part to test the model. The data are split into $N$ roughly equal sized parts. At each instance of cross-validation, the model is trained using $N-1$ parts of the data, and the prediction error of the model is calculated when it is used to predict the left out part of the data. In this way the cross-validation can be performed $N$ times and a combined estimate of prediction error can be found (Hastie et al., 2009). This procedure is attractive since the greatest possible amount of data is used for training (Witten and Frank, 2005).

For not very large data sets, leave-one-out strategy is often used. Leave-one-out cross-validation is simply $N$-fold cross-validation. Each instance in turn is left out, and the learning method is trained on all the remaining instances and tested on the left out instance. The results of all $n$ tests, one for each member of the data set, are averaged, and that average represents the final error estimate of the predictive model. This procedure is attractive since the greatest possible amount of data is used for training in each case (Witten and Frank, 2005). For all models proposed in this article, the leave-one-out strategy is used.

7. Description of concrete mixture data

The concrete mix data used in this article is similar to those used and reported by Lim et al. (2004). The coarse aggregates used were crushed granite with a specific gravity of 2.7, a fineness modulus of 7.2, and a maximum particle size of 19 mm. The volumetric ratio of coarse aggregates varied between 32% and 36%. The fine aggregates were quartz sand with a fineness modulus of 2.94 and a specific gravity of 2.61. In order to limit the water to binder ($W/B$) ratio to a very low value, a naphthalene super-plasticizer was used. An air-entraining agent (AE), a class F fly ash (FA) and a silica fume were also utilized in the mix. The detailed properties of these materials are given in Lim et al. (2004).

The data set contains the test results of 104 mix proportions with compressive strengths ranging from 38 to 76 MPa. The $W/B$ varies between 0.30 and 0.45, and the amount of FA varies from 0% to 20% of the total binder. The range of water content ($W$) is 160–180 kg/m$^3$, the content of super-plasticizer (SP) and AE, expressed as the percentage of mass of dry solids to the binder content, are 0–2% and 0.010–0.013%, respectively.

In this study six attributes are considered to form the attributes vector $X$. They are water to binder ratio, $W/B$, water content, $W$, the ratio of the weight of fine aggregate to the weight of all aggregate, $S/A$, fly ash content, $FA$, air-entraining agent content, $AE$ and super-plasticizer content, $SP$. The corresponding response (i.e. output) is the compressive strength of the mix, $f$. The input and output variables are defined in Table 1.

Compressive strength is highly dependent on the $W/B$ ratio, as this ratio affects the porosity of both the cement paste matrix and the transition zone between the matrix and the coarse aggregate.

![Fig. 1. Schematic diagram of the GRNN architecture.](image-url)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output variable</td>
<td>28-day concrete strength (MPa)</td>
<td>$f$</td>
</tr>
<tr>
<td>Input variables</td>
<td>Water–binder ratio (%)</td>
<td>$W/B$</td>
</tr>
<tr>
<td></td>
<td>Water content (kg/m$^3$)</td>
<td>$W$</td>
</tr>
<tr>
<td></td>
<td>Ratio of the weight of fine aggregate to the weight of all aggregate (%)</td>
<td>$S/A$</td>
</tr>
<tr>
<td></td>
<td>The percent of fly ash to the total binder (%)</td>
<td>$FA$</td>
</tr>
<tr>
<td></td>
<td>The content of air-entraining agent (kg/m$^3$)</td>
<td>$AE$</td>
</tr>
<tr>
<td></td>
<td>The content of super-plasticizer (kg/m$^3$)</td>
<td>$SP$</td>
</tr>
</tbody>
</table>
An increase of the W/B ratio results in weakening of the matrix and therefore tends to reduce compressive strength.

In a laboratory experiment, with a constant W/B ratio of 0.60, when the coarse/fine aggregate proportion and the cement content of a concrete mixture were progressively raised to increase the slump from 50 to 150 mm, a 12% decrease in the average 7-day compressive strength was observed (Lim et al., 2004).

Highly active pozzolans are capable of producing high-strength concrete at both early and late ages, especially when a water-reducing agent is used to reduce the water requirement. The W/B ratio is the most important factor that determines the porosity of the cement paste matrix at a given degree of hydration. However, when air voids are incorporated into the concrete, either as a result of inadequate compaction or through the use of an air-entraining admixture, they also have the effect of increasing the porosity and thereby decreasing the strength of the concrete. It has been reported that the extent of the strength loss as a result of entrained air depends not only on the W/B ratio of concrete mixture but also on the cement content. Silica fume has an effect on enhancing the strength of concrete and has been widely used in producing high strength concrete (Lim et al., 2004).

8. Results and discussion

Five different kNNA models, were developed and applied to the concrete mixture data. The models are set up in a systematic manner so that the improvements associated with optimization of various effective factors can be quantified. A detailed description of the models and the results obtained by different kNNA models, the GRNN model and stepwise regression model are presented in the following sections.

8.1. Standard kNNA (kNNA1)

The first model, kNNA1, is the standard kNNA. In this model, the Euclidean distance function was used as a measure of closeness and the average function was used to calculate the response value of the unknown observations (i.e. using Eq. (3)). The optimal value of k, number of neighbors, was obtained by DE. The minimum RMSE of 1.7466 MPa was obtained for \( k = 2 \). The value of coefficient of determination \( r^2 \) is 0.9654. A plot of the observed values versus the predicted responses is presented in Fig. 2.

8.2. kNNA with inverse distance weighting (kNNA2)

In the second model, kNNA2, the Euclidean distance function was used as the measure of closeness and the inverse distance function was used to calculate the response value of the unknown observations. Using the inverse distance function results in assigning higher weights for closer and more similar neighbors (i.e. using Eq. (4)).

Like kNNA1, DE was used for model optimization to obtain k, the number of neighbors. The minimum RMSE of 1.6446 MPa was obtained for \( k = 2 \). The value of coefficient of determination \( r^2 \) is 0.9679. The scatter plot of the observed versus predicted values is presented in Fig. 3.

8.3. kNNA with optimized Minkowski metric (kNNA3)

This model was developed to investigate the effect of application of a different metric, in particular the Minkowski distance function with an optimal parameter \( p \), on the performance of the algorithm. The power \( p \) of the Minkowski function and the number of neighbors, \( k \), were taken as parameters to be optimized and their optimal values were found using DE. The minimum RMSE of 1.3087 MPa was obtained for \( k = 3 \) and \( p = 1.11 \). The value of coefficient of determination, \( r^2 \) for this model is 0.9806. Fig. 4 shows a plot of the observed versus the predicted responses. The convergence history of RMSE for a typical run is presented in Fig. 5.

8.4. kNNA with attribute selection (kNNA4)

In the fourth model, kNNA4, the attribute selection was also considered. The number of neighbors, six binary weight coefficients, taking a value of either 0 to indicate absence or
1 to indicate existence of a particular attribute, and the power of distance function, $p$, were used as optimization variables. The optimal variables, derived from DE, are presented in Table 2. As stated earlier, attribute subset selection may in some cases improve the performance of the $k$NNA since attribute selection is not only concerned with reducing the number of attributes but also eliminating the attributes that are correlated with other already selected attributes. As can be seen in Table 2, for this data set, the optimal weights of all attributes for $k$NNA4 are equal to one. This indicates that all the attributes are important and should be present in the model. As can be seen in Table 2, the results are similar to $k$NNA3, and optimal values of $k=3$ and $p=1.11$ resulted in a RMSE of 1.3087 MPa and a coefficient of determination of 0.9806. The scatter plot of the observed versus predicted values is presented in Fig. 6. Convergence history of the RMSE for a typical analysis is shown in Fig. 7.

To determine the relative significance of each attribute on compressive strength (output), a sensitivity analysis was performed. Six models, each with the reduced subsets of five attributes, are obtained by removing one of six attributes at a time. The RMSE obtained for these six models are presented in Table 3. The first row represents the full model using all the six attributes and the models with one removed attribute are shown in rows 2–7.

Results presented in Table 3 indicate that the model with all attributes provides the smallest RMSE (1.3087) and removing each attribute results in an increase of RMSE. This finding confirms that all six attributes are important. The RMSE values also indicate that by removing AE and SP, the RMSE increases slightly from 1.3087 to 1.42 and 1.47, respectively. Removing the other four attributes significantly deteriorates the model performance with a RMSE values of 2.05, 2.288, 1.998 and 1.7076 associated with removing $W/B$, $W$, $S/A$ and FA respectively. These values are at least 40% higher than 1.3087, the RMSE of the full model. Therefore, $W$, $W/B$, $S/A$ and FA can be considered the most influential attributes of the model.

8.5. $k$NNA with attribute weighting ($k$NNA5)

In the final model, $k$NNA5, the relevance of different attributes is taken into account by assigning optimal weights to the attributes. In this model, the number of neighbors, $k$, six weight coefficients corresponding to six attributes, real numbers ranging between zero and one, and the power $p$ of the Minkowski distance...
function are defined as parameters to be optimized. Table 4 displays the optimal values of these parameters. The minimum value of RMSE (1.1739 MPa) was obtained for $k = 3$. The value of coefficient of determination, $r^2$ for this model is 0.9844. The scatter plot of the observed versus the predicted values are presented in Fig. 8. The convergence history plot of the RMSE for a typical run is shown in Fig. 9.

### 8.6. GRNN results

Leave-one-out cross validation approach was used to find the optimal value of smoothing parameter. The RMSE values, obtained for various values of this parameter, are shown in Fig. 10. The minimum value of RMSE (1.7725) was obtained at a smoothing parameter of 0.31.

### 8.7. Stepwise regression results

Lim et al. (2004) developed a stepwise regression for this data set and obtained a coefficient of determination of 0.955. However, the corresponding value of RMSE was not reported. In order to calculate the RMSE, we developed a stepwise regression model and obtained a value of 2.02 MPa for RMSE and a value of 0.956 for $r^2$. The stepwise regression results also indicate that all six attributes are significant. This fact is consistent with the conclusion obtained from the results of kNNA4 model.

### 8.8. Discussion

In order to evaluate and compare the performance of different models, the RMSEs of the five different kNNA models and the GRNN model developed in this article along with results of a modular neural network model (Tsai and Lin, 2011) and a
Table 5. The following points can be noted:

1. Comparison of the stepwise regression model (Lim et al., 2004) are presented in Table 5.

2. The RMSE of the stepwise regression model and the modular neural network model were developed for estimation of the compressive strength of concrete and their performances were compared with the stepwise regression model and the modular neural network models. This indicates that a proper consideration of the relevance of attributes is very important and results in a more accurate predictive model.

3. Results of the stepwise regression model (Lim et al., 2004) are presented in Table 5. The following points can be noted:

1. Comparison of $k$NNA2 and $k$NNA1 reveals that consideration of the effect of more similar neighbors results in a reduction of prediction error, where by assigning higher weights to closer observations, the RMSE is reduced from 1.7466 to 1.6435 MPa.

2. The RMSE of $k$NNA3 (1.3087) is lower than the corresponding error values of both $k$NNA1 and $k$NNA2 (1.7466 and 1.6435, respectively). Therefore it can be concluded that using the Minkowski metric with an optimized parameter $p$ results in a better performance as compared to application of the Euclidean metric, the most commonly used metric in $k$NNA.

3. Results of $k$NNA4 indicate that all the attributes are relevant. Therefore, the $k$NNA4, the model with attribute selection, and $k$NNA3 result in the similar RMSE of 1.3087 MPa.

4. The $k$NNA5, the model with the assigned optimal attribute weights, results in the lowest value of the RMSE (1.1739 MPa) among all $k$NNA models. This indicates that a proper consideration of the relevance of attributes is very important and results in a more accurate predictive model.

5. The $k$NNA4, the model with attribute selection, has a higher RMSE than $k$NNA5, the model with the assigned optimal attribute weights. This is expected as attribute selection can be regarded as a special case of the attribute weighting, where the weight coefficients are constrained to take a binary value of either 0 or 1.

6. Enhanced $k$NNAs ($k$NNA3, $k$NNA4 and $k$NNA5) outperform the GRNN, the stepwise regression and the modular neural network models.

8.9. Sensitivity of the DE algorithm to the control parameters

In order to investigate the effect of control parameters of DE, a sensitivity analysis of the best model ($k$NNA5) was performed to evaluate the effect of parameter settings on success rates. Success rate (SR) or probability of convergence is a commonly used metric to quantify the robustness of optimization algorithms. SR is defined as the percentage of successful to total trials (Price et al., 2005).

In order to calculate the success rates corresponding to different sets of parameters, a termination criterion was defined as finding the value-to-reach (VTR), before reaching the maximum number of iterations. A trial is classified as a success when the best vector's objective function (i.e. RMSE) value falls below VTR. Trials that do not reach the VTR within a predetermined maximum number of iterations are counted as failures.

In all experiments, the maximum number of iterations and the VTR were set to 300 and 1.1739 MPa, respectively. For each parameter set of $[F, CR]$ with values of $F \in [0.5, 0.7, 0.9]$ and $CR \in (0.5, 0.7, 0.9)$, the DE algorithm was executed ten times. The success rates of these ten independent runs are presented in Table 6. The results indicate that for all combinations of $F$ and $CR$, the DE algorithm found the global optima in at least one of the runs. The maximum success rate of 0.9 (i.e. the DE converged to VTR nine out of ten times) may be obtained for different parameter settings of $[0.9, 0.9]$, $[0.9, 0.7]$, $[0.7, 0.7]$ and $[0.5, 0.5]$ for $F$ and $CR$. These results demonstrate that selected parameter setting of $F=0.9$ and $CR=0.9$ is appropriate.

9. Summary and conclusions

Generally, concrete testing procedures are time consuming and experimental errors are inevitable. A typical compression test is usually performed about 28 day after casting the concrete. Should the test results fall short of the required strength, costly remediation efforts must be undertaken. Therefore, it is important to be able to estimate the compressive strength of concrete prior to casting.

In this article, five different optimal $k$NN models and a GRNN model were developed for estimation of the compressive strength of concrete and their performances were compared with the stepwise regression model and the modular neural network model in terms of RMSE and $r^2$. An analysis of the performance of the models demonstrated that assigning the optimal attributes weights, and using the optimized Minkowski distance function results in the best predictive $k$NNA model.

A primary assumption and perhaps weakness of standard $k$NNA is that it is sensitive to the presence of irrelevant or less relevant attributes. This is because the Euclidian distance function, assumes that all the attributes are equally important.
Attribute selection and attribute weighting may in some cases improve the performance of the kNN4.

Results of the model with attribute selection (kNN4A) indicate that for this data set all six attributes are important and should be present in the model. For certain data sets like the one analyzed in this article, using all attributes is necessary to predict the output. In such scenarios, feature selection algorithms such as kNN4A cannot add significant value.

In the attribute weighting algorithm (kNN5), optimal weights are calculated for the attributes, so that the highest weights are assigned to the most relevant attributes. The results of the analyses indicate that the model with optimal attribute weighting performs very well in estimating the compressive strength of concrete from mix ingredient data and the kNN5 outperforms all other kNNAs, GRNN, modular neural network and stepwise regression models. The root mean squared error of the best proposed model with optimal attribute weighting (kNN5) is 1.1739 MPa for compressive strengths ranging from 38 to 76 MPa.

Typically, concrete manufacturing companies have extensive data sets of their past mix proportions and the corresponding compressive strength of the mixes. Hence, the concrete industry can benefit from their data sets in conjunctions with the proposed optimal instance-based learning models discussed in this article to obtain a reliable estimate for the compressive strength of the final product.

Acknowledgment

The support from the research deputy of Yazd University is gratefully acknowledged. The author is grateful to editor and anonymous reviewers for their helpful and constructive comments on an earlier draft of this article.

References


