Engineering Applications of Artificial Intelligence 29 (2014) 104–113

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

High-performance concrete compressive strength prediction using Genetic Weighted Pyramid Operation Tree (GWPOT)

Min-Yuan Cheng, Pratama Mahardika Firdausi [1], Doddy Prayogo [*,1]

Department of Civil and Construction Engineering, National Taiwan University of Science and Technology, #43, Section 4, Keelung Rd., Taipei 106, Taiwan, ROC

A R T I C L E   I N F O

Article history:
Received 11 July 2013
Received in revised form
2 October 2013
Accepted 24 November 2013
Available online 15 December 2013

Keywords:
Prediction
Concrete strength
Genetic Algorithm
Operation Tree
Weighted Pyramid Operation Tree

A B S T R A C T

This study uses the Genetic Weighted Pyramid Operation Tree (GWPOT) to build a model to solve the problem of predicting high-performance concrete compressive strength. GWPOT is a new improvement of the genetic operation tree that consists of the Genetic Algorithm, Weighted Operation Structure, and Pyramid Operation Tree. The developed model obtained better results in benchmark tests against several widely used artificial intelligence (AI) models, including the Artificial Neural Network (ANN), Support Vector Machine (SVM), and Evolutionary Support Vector Machine Inference Model (ESIM). Further, unlike competitor models that use "black-box" techniques, the proposed GWPOT model generates explicit formulas, which provide important advantages in practical application.

1. Introduction

High-performance concrete (HPC) is a new type of concrete used in the construction industry (Yeh, 1998). HPC works better in terms of performance characteristics and uniformity characteristics than high-strength concrete (Mousavi et al., 2012; Yeh and Lien, 2009). Apart from the 4 conventional cement ingredients, Portland Cement (PC), water, fine aggregates, and coarse aggregates, HPC further incorporates cementitious materials, fly ash, blast furnace slag, and a chemical admixture (Yeh, 1998). These additional ingredients make HPC mix proportion calculations and HPC behavior modeling significantly more complicated than corresponding processes for conventional cement.

Machine learning and AI are attracting increasing attention in academic and empirical fields for their potential application to civil engineering problems (Mousavi et al., 2012). In civil engineering, AI techniques have been categorized into two approaches, optimization and prediction, with numerous prediction applications including Artificial Neural Network (ANN), Support Vector Machine (SVM), and Linear Regression Analysis, among others. Optimization applications include the Genetic Algorithm (GA) and Particle Swarm Optimization (PSO).

In the field of civil engineering, much research has focused on hybridizing optimization techniques and prediction techniques. Many papers have reported on hybrid techniques that are able to predict HPC to a high degree of accuracy (Cheng et al., 2012; Peng et al., 2009; Yeh, 1999). The Evolutionary Support Vector Machine Inference Model (ESIM), one hybridization technique, uses a fast messy Genetic Algorithm (fmGA) and SVM to search simultaneously for the fittest SVM parameters within an optimized legal model (Cheng and Wu, 2009). However, the aforementioned techniques, especially ANN, SVM, and ESIM, are considered "black-box" models due to massive node sizes and internal connections. Because these models do not provide explicit formulae, they do not explain the substance of the associated model, which is a serious disadvantage in practical applications.

Yeh and Lien (2009) proposed the novel Genetic Operation Tree (GOT) to overcome this disadvantage. The GOT consists of a GA and an Operation Tree (OT). This model is a practical method for eliciting both an explicit formula and an accurate model from experimental data. Although many studies have used GOT to develop formulae to optimally fit experimental data (Chen et al., 2012; Peng et al., 2009; Yeh et al., 2010), this model has yet to achieve results comparable to other prediction techniques such as ANN and SVM. This suggests the potential to further improve the GOT.

This paper introduces a novel approach based on OT called Genetic Weighted Pyramid Operation Tree (GWPOT) to predict HPC compressive strength. The GWPOT integrates the Weighted Operation Structure (WOS) and Pyramid Operation Tree (POT) models to enhance the prediction capability and the fit with experimental data.

Remaining sections in this paper are organized as follows: Section 2 provides a brief explanation of OT, GA, and WOS; Section 3 describes the GWPOT model; Section 4 describes the case study

* Corresponding author. Tel.: +886 2 27336596; fax: +886 2 27301074.
E-mail addresses: myc@mail.ntust.edu.tw (M.-Y. Cheng), wowdikawow@yahoo.com (P.M. Firdausi), doddyprayogo@ymail.com (D. Prayogo).
[1] Tel.: +886 2 27336596; fax: +886 2 27301074.

0952-1976/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.engappai.2013.11.014

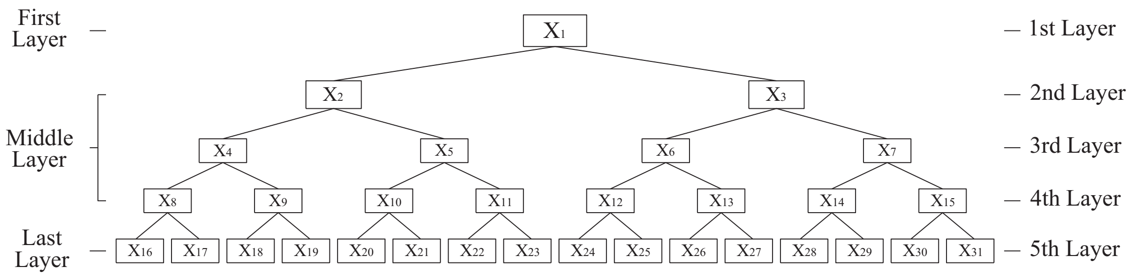

Fig. 1. Five-layer OT model.

Bits	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Code	1	5	2	10	2	3	9	2	5	3	5	7	3	7	4

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Code	6	10	9	8	8	9	6	10	7	7	8	6	6	7	8	9

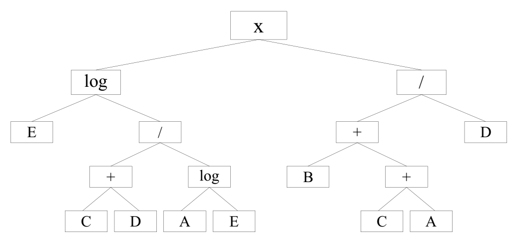

Fig. 2. Example of an OT model.

and configuration of GWPOT parameters, presents GWPOT model results, and compares those results with those of common prediction techniques; and, finally, Section 5 presents study conclusions.

2. Literature review

2.1. Operation Tree (OT)

Operation Tree (OT) is a hierarchical tree structure that represents the architecture of a mathematical formula. Fig. 1 illustrates an OT model with 31 nodes. In Fig. 1, the OT model consists of a root value and sub-trees of children, represented as a set of connected nodes. Each node on the OT model has either 0 or 2 child branches, with the former designated as “leaf nodes” and associated with either a variable or constant and the latter associated with a mathematical formula (+, −, ×, ÷, log, etc.) (Hsie et al., 2012). Fig. 2 shows an example of the OT model with a 31-bit-node code. Table 1 lists the bit codes for mathematical operations, variables, and constants. The OT in Fig. 2 may be expressed as

$$Output = \left(\left(\log_E \frac{C+D}{\log_E A}\right)\frac{A+B+C}{D}\right) \quad (1)$$

Its flexibility in expressing mathematical formulae allows OT to avoid a disadvantage common to other prediction techniques (Peng et al., 2009; Yeh and Lien, 2009). The branch-and-leaf configuration of OT facilitates the deduction of function values and formulae. Input values may thus be substituted into the formula to generate a predicted output value for each data point. OT performance may be evaluated by calculating the root-mean-squared error (RMSE) between predicted and actual output values (Yeh et al., 2010). The best OT formula is achieved when RMSE reaches the lowest possible value. Because searching the best combination formula to fit with the data is a discrete optimization problem, an optimization technique capable of solving a discrete problem must be integrated into the OT model (Peng et al., 2009).

Table 1
Genetic code of mathematical operations, variables, and constants.

Code	1	2	3	4	5	6	7	8	9	10
Meaning	×	÷	+	−	log	*A*	*B*	*C*	*D*	*E*

2.2. Genetic Algorithm (GA)

Genetic Algorithm (GA) is an optimization technique first proposed by Holland (1975). GA is based on Darwin's theory of evolution and mimics biological competition in which only comparatively strong chromosomes survive into the next generation. Each chromosome in a GA population represents a candidate solution for a given problem and is able to generate a result based on the objective function. Ability to handle various types of objective functions is another advantage of GA.

GA proceeds through progressive generations from an initial population. Each GA generation is subjected to genetic operation processes such as evaluation, selection, crossover, and mutation and generates a new result. A new-generation chromosome will replace the current-generation chromosome if it generates a better result.

2.3. Weighted operation structure (WOS)

The weighted operation structure (WOS) is an improvement of the OT model proposed by Tsai (2011). This study added a constant value to every variable to balance every input variable to help OT generate a better formula. Therefore, WOS assigns weights to every node connection in the OT model so that each WOS element produces node outputs conducted by 2 OT nodes and 2 undetermined weights.

The WOS is thus able to search for the best formula in a wider search space with more combinations over a longer time period than the original OT model. Fig. 3 shows a 5-layer weighted operation structure. The example of the WOS model in Fig. 4 may be expressed as

$$Output = \frac{0.1 \times (0.3P1 + 0.4P2)}{0.2 \times (0.5P4 \times 0.6P3)} = \frac{0.03P1 + 0.04P2}{0.012 \times P4 \times P3} \tag{2}$$

3. Genetic Weighted Pyramid Operation Tree (GWPOT)

3.1. GWPOT architecture

This study proposes a new operation tree algorithm to address the shortcomings of OT called the Genetic Weighted Pyramid Operation Tree (GWPOT). GWPOT applied a pyramid-shaped, 4 connected OT called Pyramid Operation Tree (POT). The significantly wider search area of GWPOT results from its use of multiple trees that allows coverage of a greater number of combination possibilities. The weighted concept of WOS was integrated into GWPOT due to the success of this concept in improving GOT performance. Fig. 5 illustrates a GWPOT model with 3 layers per tree.

Tuning parameters in the GWPOT model include: mutation rate, crossover rate, weight rate, layers per tree, and total number of trees. Mutation rates and crossover rates were retained from OT. The other parameters are new to GWPOT and explained as follows: (1) weight rate sets the probability of each node having a constant value in the WOS structure; (2) layers per tree sets the number of layers used for each tree; and (3) total number of trees sets the number of trees used to produce the formula in one model process. Four trees were used to assemble the pyramid shape in this study.

GWPOT operations start by generating a population of chromosomes for the first tree. Every chromosome represents the solution vector of formula components. Next, the evaluation step obtains the objective function (RMSE) for each chromosome. Afterward, the GA optimizer searches for the optimal parameter or best formula combination. GA optimization repeats until the stopping criterion is achieved. Fig. 6 shows the flowchart for GWPOT.

An explanation of the principal steps in GWPOT follows below:

(1) *Initialization and parameters setting*: This step sets GWPOT tuning parameters and randomly generates the initial population.

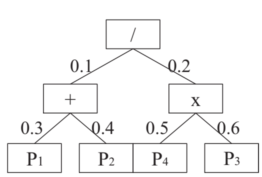

Fig. 3. Example of a WOS model.

(2) *Training dataset*: The dataset is divided into two parts, with the first used as training data and the second as testing data.
(3) *OT training model*: OT is a learning tool able to build an explicit formula for use as a prediction model. Each chromosome in OT represents one formula, as explained in the previous section of this paper.
(4) *Fitness evaluation*: A fitness function formula, RMSE in the current study, evaluates each chromosome. Smaller RMSE values indicate better chromosomes.
(5) *GA procedure*: GA begins with the selection process. This study uses a roulette wheel as the selection technique and uniform crossover as the crossover mechanism. After completing crossover, mutation is performed by generating random real value numbers based on the mutation rate. The termination criterion in the current study is the total number of generations. This GA procedure will repeat until either the stopping criterion or termination criterion is satisfied.
(6) *Checking the number of tree*: If the number of trees does not reach the maximum parameter value, the GWPOT process continues to the next tree structure. In each new tree structure, the top-half population is generated randomly and includes the best chromosomes from the previous tree structure. Moreover, the amount of bits in every chromosome is expanded in order to modify the current population. The newly added bits are provided to store the formula combination of the next tree structure. The top-half chromosomes from the previous tree are retained due to the possibility that the previous tree structure may not require further change to improve performance. Random numbers replace the bottom-half population in order to find new tree structure combinations. The process continues until the 4th tree is produced.
(7) *Optimal solution*: The chromosome with the lowest RMSE is considered to be the optimal solution. This solution is the formula that will be employed to map the input–output relationship of the target dataset.

3.2. GWPOT example

To further clarify the GWPOT procedure, this study applied GWPOT to an example problem. A 3-layer GWPOT was created. This example uses 5 types of operators and variables, respectively. The operators are ×, ÷, +, –, and \widehat and the variables are A, B, C, D and E. The weights are set between 0.01 and 10.00, and 4 tree-structure phases must be passed to establish the GWPOT model.

The first tree-structure phase begins by generating a population that is identified by node (variables and mathematical operations) and weight. The process continues through the OT training model, fitness evaluation, and GA search procedures. Fig. 7 shows the best chromosome in the first tree structure, calculated as

$$Tree\ \ 1 = \frac{0.3A + 0.04E}{0.1C \times 0.2C} \tag{3}$$

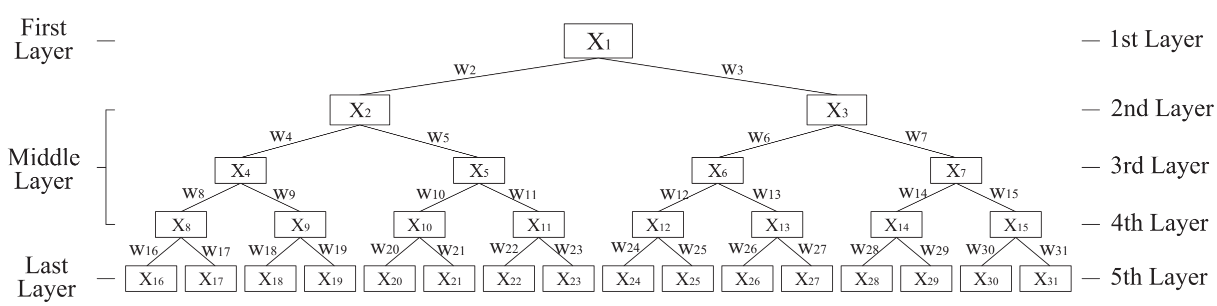

Fig. 4. Five-layer WOS model.

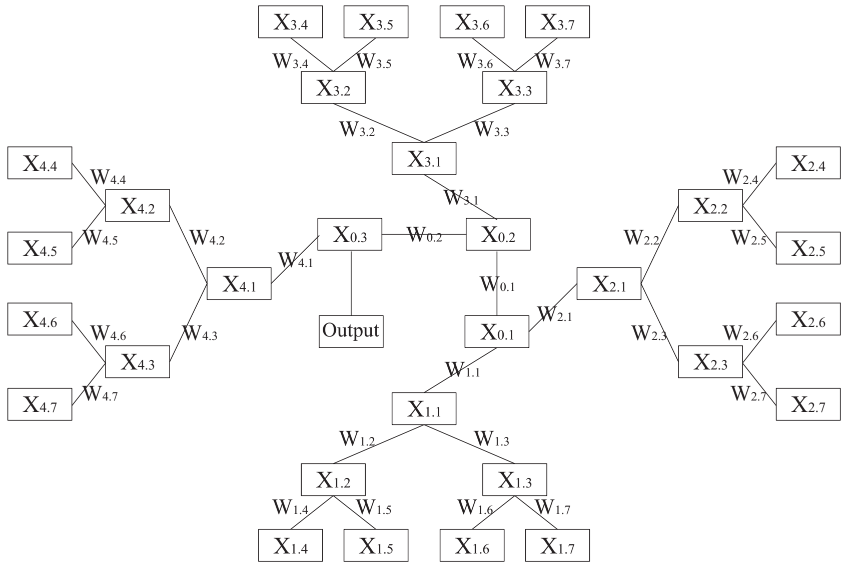

Fig. 5. Three-Layer GWPOT Model.

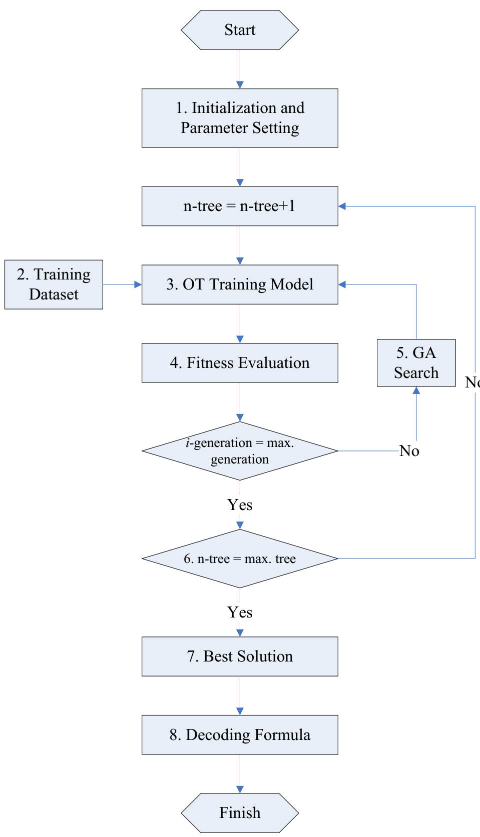

Fig. 6. GWPOT flowchart.

Node Bits	X1.1	X1.2	X1.3	X1.4	X1.5	X1.6	X1.7
Code	2	3	1	6	10	8	8

Weight Bits	W1.1	W1.2	W1.3	W1.4	W1.5	W1.6	W1.7
Code	1	1	0.2	0.3	0.4	0.5	1

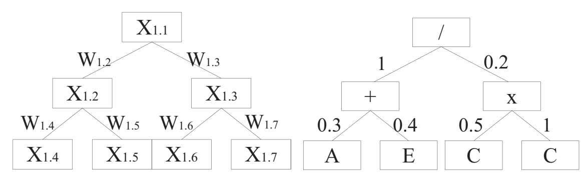

Fig. 7. First tree result example.

The second tree structure starts with a top-half population generated randomly from the best solutions from the first tree structure and augmented by additional new bits. The bottom-half population is tasked to identify new tree structure combinations between the first tree and second tree. One additional mathematical operator is used to connect the first tree and second tree. GA finds this operator concurrently using other nodes in the second tree-structure phase. Assuming the first tree solution in the second phase is the same as the first tree solution in first tree-structure phase, the best solution from second tree structure phase may be illustrated as in Fig. 8.

The corresponding formula for this tree (Fig. 8) is

$$Output = \frac{(0.3A + 0.4E/0.1C \times 0.2C)}{2.2A - 11.44E + 2.42B} \tag{4}$$

The GWPOT procedure continues to the next phase after completing the second tree-structure process. The second tree-structure procedures are repeated to find the best solutions in the 3rd and 4th tree structure. Fig. 9 shows the best result from the third tree structure, which incorporates the first, second, and third trees. The last tree structure contains the result for all 4 trees, as shown in Fig. 10, while Fig. 11 represents the pyramid-shape model.

The corresponding formula for the 3rd tree and 4th trees are:

$$Tree\quad 3 = (1.1C - 1.1D) \times 5.9A^{0.1A} \tag{5}$$

$$\text{Tree}\quad 4 = 2.4D + 0.3A \tag{6}$$

After combining all trees and simplification, the final formula may be expressed as

$$Output = \alpha + \beta \times \left(2 \times \left(2 \times \frac{Tree\ \ 1}{2.2Tree\ \ 2} + 0.5Tree\ \ 3\right) + 0.5Tree\ \ 4\right) \tag{7}$$

3.3. Modified predicted output value

An oblique phenomenon frequently occurs in OT-generated formula. This phenomenon reflects the concurrently high linear correlation and high RMSE in the relationship between actual output and predicted output (Yeh and Lien, 2009). To address this problem, some researchers have used single linear regression analysis to modify the OT result (Hsie et al., 2012; Mousavi et al., 2012; Yeh et al., 2010;

Node Bits	X1.1	X1.2	X1.3	X1.4	X1.5	X1.6	X1.7	X2.1	X2.2	X2.3	X2.4	X2.5	X2.6	X2.7	X0.1
Code	2	3	1	6	10	8	8	3	4	7	6	10	9	7	2

Weight Bits	W1.1	W1.2	W1.3	W1.4	W1.5	W1.6	W1.7	W2.1	W2.2	W2.3	W2.4	W2.5	W2.6	W2.7	W0.1
Code	1	1	0.2	0.3	0.4	0.5	1	2.2	1	1.1	1	5.2	4	0.8	2

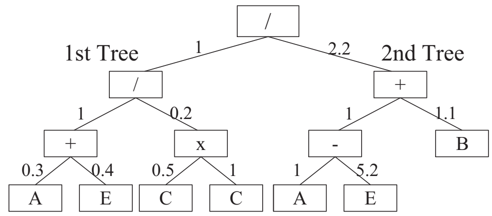

Fig. 8. First and second tree result example.

Node Bits	X1.1	X1.2	X1.3	X1.4	X1.5	X1.6	X1.7	X2.1	X2.2	X2.3	X2.4	X2.5	X2.6	X2.7
Code	2	3	1	6	10	8	8	3	4	7	6	10	9	7

Node Bits	X3.1	X3.2	X3.3	X3.4	X3.5	X3.6	X3.7	X0.1	X0.2
Code	1	4	5	8	9	6	6	2	3

Weight Bits	W1.1	W1.2	W1.3	W1.4	W1.5	W1.6	W1.7	W2.1	W2.2	W2.3	W2.4	W2.5	W2.6	W2.7
Code	1	1	0.2	0.3	0.4	0.5	1	2.2	1	1.1	1	5.2	4	0.8

Weight Bits	W3.1	W3.2	W3.3	W3.4	W3.5	W3.6	W3.7	W0.1	W0.2
Code	0.5	1.1	1	1	1	5.9	0.1	2	2

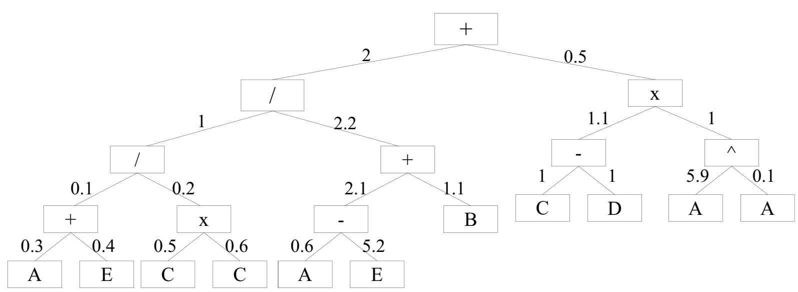

Fig. 9. First, second, and third tree result example.

Yeh and Lien, 2009). The practical result of this regression analysis is to assign the same value to the prediction output mean value and the actual output mean value. The equation for the single linear regression analysis is

$$y = \alpha + \beta f \quad (8)$$

where f is the predicted output value of the operation tree; y is the modified predicted value; α and β are the regression coefficients.

According to single linear regression analysis

$$\alpha = \bar{y} - \beta \cdot \bar{f} \quad (9)$$

$$\beta = \frac{\sum_{i=1}^{n} (f_i - \bar{f})(y_i - \bar{y})}{\sum_{i=1}^{n} (f_i - \bar{f})^2} \quad (10)$$

where $\bar{y}$ is the mean of actual output values in the dataset; $\bar{f}$ is the mean of predicted output values in the dataset; y_i is the actual output value of the ith data in the dataset; and f_i is the predicted output value of the ith data in the dataset.

4. Case study

4.1. The dataset

The dataset used was obtained from Yeh (1998) and published in the data repository of the University of California, Irvine (UCI). A total of 1030 concrete samples covering 9 variables were collected from the database. Eight of the 9 variables or influencing factors in the dataset, including cement, fly ash, slag, water, SP, coarse aggregate, fine aggregate, and age, were treated as input variables. The remaining variable, concrete compressive strength, was treated as the output variable. Table 2 shows descriptive statistics for these factors.

4.2. Tuning parameters

In this study, each parameter was set as: crossover rate = 0.8, mutation rate = 0.05, and weighted rate = 0.5. Moreover, the total tree was set at 4, with 3, 4, 5, and 6 layers. Furthermore, the total population size and the total number of generations for each tree selected for this study were 100 and 2,000, respectively.

4.3. k-Fold cross validation

k-Fold cross validation is a statistical technique that divides study data into k subsamples to determine the accuracy of a prediction model. The original data is divided randomly into k equally sized or approximately equally sized segments, with one subsample used as testing data and the remaining k–1 subsamples used as training data. The cross validation process recreates the

Node Bits	X1.1	X1.2	X1.3	X1.4	X1.5	X1.6	X1.7	X2.1	X2.2	X2.3	X2.4	X2.5	X2.6	X2.7
Code	2	3	1	6	10	8	8	3	4	7	6	10	9	7

Node Bits	X3.1	X3.2	X3.3	X3.4	X3.5	X3.6	X3.7	X4.1	X4.2	X4.3	X4.4	X4.5	X4.6	X4.7
Code	1	4	5	8	9	6	6	3	9	6	6	8	7	7

Weight Bits	W1.1	W1.2	W1.3	W1.4	W1.5	W1.6	W1.7	W2.1	W2.2	W2.3	W2.4	W2.5	W2.6	W2.7
Code	1	1	0.2	0.3	0.4	0.5	1	2.2	1	1.1	1	5.2	4	0.8

Weight Bits	W3.1	W3.2	W3.3	W3.4	W3.5	W3.6	W3.7	W4.1	W4.2	W4.3	W4.4	W4.5	W4.6	W4.7
Code	0.5	1.1	1	1	1	5.9	0.1	0.5	2.4	0.3	8	2.7	9.2	0.1

Node Bits	X0.1	X0.2	X0.3	Weight Bits	W0.1	W0.2	W0.3
Code	2	3	3	Code	2	2	2

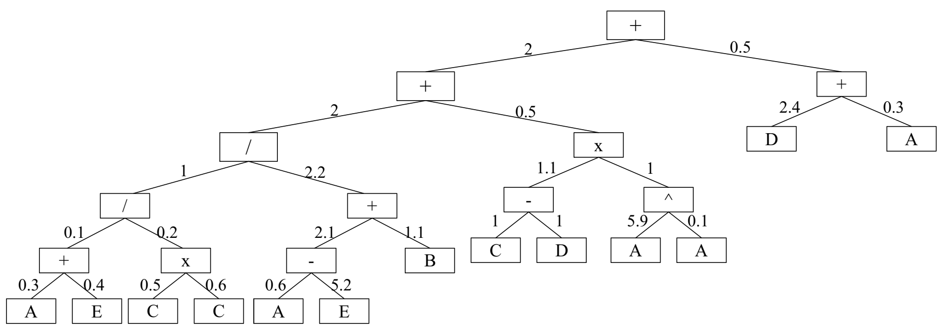

Fig. 10. Four-tree result example.

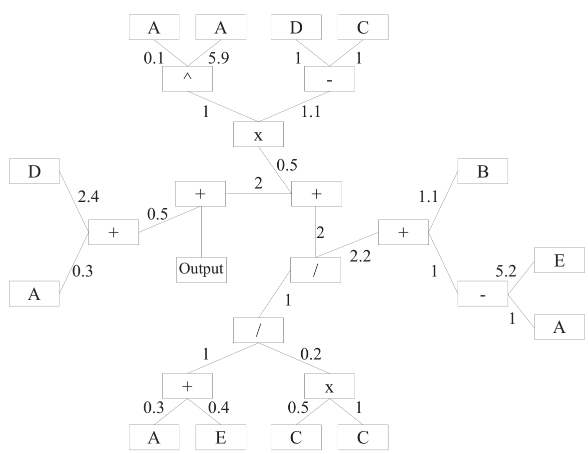

Fig. 11. GWPOT result example.

Table 2
HPC variables data.

Variable	Notation	Unit	Min	Max	Mean	Standard deviation
Cement	*A*	(kg/m^3)	102	540	281.2	104.51
Slag	*B*	(kg/m^3)	0	359.4	73.9	86.28
Fly ash	*C*	(kg/m^3)	0	200.1	54.2	64.00
Water	*D*	(kg/m^3)	121.8	247	181.6	21.36
Super plasticizer	*E*	(kg/m^3)	0	32.2	6.2	5.97
Coarse aggregate	*F*	(kg/m^3)	801	1145	972.9	77.75
Fine aggregate	*G*	(kg/m^3)	594	992.6	773.6	80.18
Age	*H*	Day	1	365	45.7	63.17
Compressive atrength	*Output*	MPa	2.3	82.6	35.8	16.71

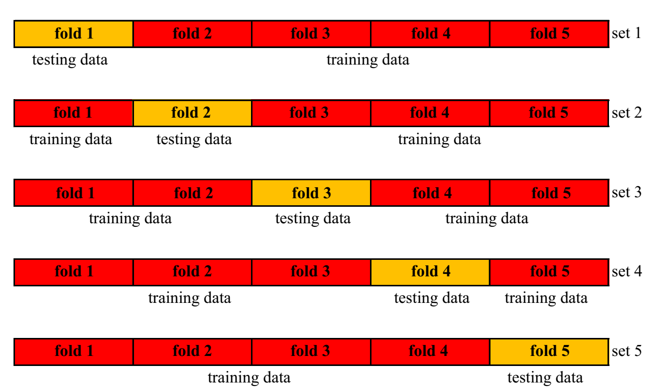

Fig. 12. Five-fold cross-validation model.

model *k* times, with each *k* subsample used exactly once as the validation data. Results from all subsamples or from all folds are then averaged to produce a single value estimation.

In general, because *k* remains an unfixed parameter, the value of *k* may be any suitable number. The current study set the value of *k* as 5 in order to limit total computational time. Five-fold means that each set uses 20% of the data (206 data points) as testing data and 80% (824 data points) as training data. The data contains 1030 HPC records is divided randomly into 5 equally sized (206 data), with one subsample used as testing data and the remaining 4 subsamples used as training data. Five-fold means that each set uses 20% of the data (206 data points) as testing data and 80% (824 data points) as training data. The cross validation process recreates the model 5 times, with each 5 subsample used exactly once as the validation data. Fig. 12 illustrates *k*-fold cross validation operations.

4.4. Performance measurement

To explore the accuracy of each model, this study used 3 performance-measurement equations: Root Mean Square Error

(RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE). All performance measures used in testing data were combined to create a normalized reference index (RI) in order to obtain an overall comparison (Chou et al., 2011). Each fitness function was normalized to a value of 1 for the best performance and 0 for the worst. The RI was obtained by calculating the average of every normalized performance measure as shown in Eq. (11). Eq. (12) shows the function used to normalize the data.

$$RI = \frac{RMSE + MAE + MAPE}{3} \tag{11}$$

$$x_{norm} = \frac{(x_{max} - x_i)}{(x_{max} - x_{min})} \tag{12}$$

4.4.1. Root mean square error

Root mean square error (RMSE) is the square root of the average squared distance between the model-predicted values and the observed values. RMSE may be used to calculate the variation of errors in a prediction model and is very useful when large errors are undesirable. RMSE is given by the following equation:

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2} \tag{13}$$

where y_j is the actual value; $\bar{y}_j$ is the predicted value; and n is the number of data samples.

4.4.2. Mean absolute error

Mean absolute error (MAE) is the average absolute value of the residual (error). MAE is a quantity used to measure how close a prediction is to the outcome. The MAE may be expressed as

$$MAE = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j| \tag{14}$$

where y_j is the actual value; $\hat{y}_j$ is the predicted value; and n is the number of data samples.

4.4.3. Mean absolute percentage error

Mean absolute percentage error (MAPE) calculates percentage error of prediction. Small denominators are problematic for MAPE because they generate high MAPE values that impact overall value. The MAPE may be expressed as

$$MAPE = \frac{1}{n}\sum_{j=1}^{n}\left|\frac{y_j - \hat{y}_j}{y_j}\right| \times 100\% \tag{15}$$

where y_j is actual value; $\hat{y}_j$ is predicted value; and n is number of data samples.

4.5. Results and discussion

This study compared the performance of the proposed model against two other OT-based tools, the Genetic Operation Tree (GOT) and the Weighted Operation Structure (WOS). After testing a variety of layer numbers, the number of layers needed to build the best GWPOT model formula was determined as between 3 and 6.

Table 3
Average of GOT performance measurements results.

No. of layers	Training data			Testing data			*RI*
	RMSE	*MAE*	*MAPE*	*RMSE*	*MAE*	*MAPE*	
3	10.226	8.250	30.735	10.415	8.400	31.282	0.136
4	8.105	6.394	22.741	8.066	6.429	23.092	0.600
5	7.117	5.537	18.661	7.120	5.509	18.611	0.817
6	7.005	5.465	18.573	7.357	5.681	19.049	0.779
7	7.270	5.637	19.425	7.253	5.615	19.033	0.792

Table 4
Average of WOS performance measurement results.

No. of layers	Training data			Testing data			*RI*
	RMSE	*MAE*	*MAPE*	*RMSE*	*MAE*	*MAPE*	
3	9.020	7.089	24.853	8.939	7.025	23.959	0.214
4	7.685	5.990	20.345	7.712	6.065	20.646	0.569
5	6.983	5.405	18.540	7.021	5.349	18.437	0.801
6	6.646	5.070	16.668	6.890	5.230	16.909	0.872
7	6.801	5.169	17.307	7.410	5.549	18.160	0.535

Table 5
Performance measurement results of GWPOT model.

No. of layers	Training			Testing			*RI*
	RMSE	*MAE*	*MAPE*	*RMSE*	*MAE*	*MAPE*	
3	6.825	5.273	17.490	7.050	5.442	18.333	0.263
4	6.352	4.877	16.335	6.780	5.174	17.253	0.448
5	5.864	4.440	14.986	6.379	4.787	16.095	0.695
6	5.689	4.307	14.597	6.446	4.750	16.118	0.690

Five-fold cross-validation techniques validated the performance of the models. The average 5-fold results for GOT, WOS, and GWPOT are summarized in Tables 3–5, respectively.

Table 3 shows that the 5-layer configuration generated the best RI result for the GOT model (0.817). Eq. (16) shows the formula associated with this result. On the other hand, the six-layer configuration generated the best overall and average for the WOS model (0.872). Eq. (17) shows the formula associated with this result.

$$y = 2.905 + 0.077 \times \left(\left(\frac{\log\,(AH + H^2)}{\log\,(\ln\,(D+H))}\right)^{(\log\,(A+B+\,\ln\,C)/\log\,D)}\right) \tag{16}$$

$$y = -35.74 + 2.842\left(\frac{\log\,9.9H}{(\log\,((3.173A + 3.188B + 1.033C - 2.922D) \times (2.467E + 0.262H - 0.725A - 0.125F))/\,\log\,(2.798D \times\,\ln\ln 83.025E))}\right) \tag{17}$$

The RI results in Table 5 for the 3-, 4-, 5-, and 6-layer configurations are: 0.263, 0.448, 0.695, and 0.690, respectively. This indicates that the 5-layer configuration generated the best RI result for the GWPOT model. The 6-layer configuration generated the worst result due to increased model complexity and large chromosome bit number. Further, larger layer numbers require more computational time, which increases the difficulty of finding the best combination.

Fig. 13 illustrates the best OT produced using GWPOT. The OT in Fig. 13 is generated from the best 5-layer GWPOT model and may

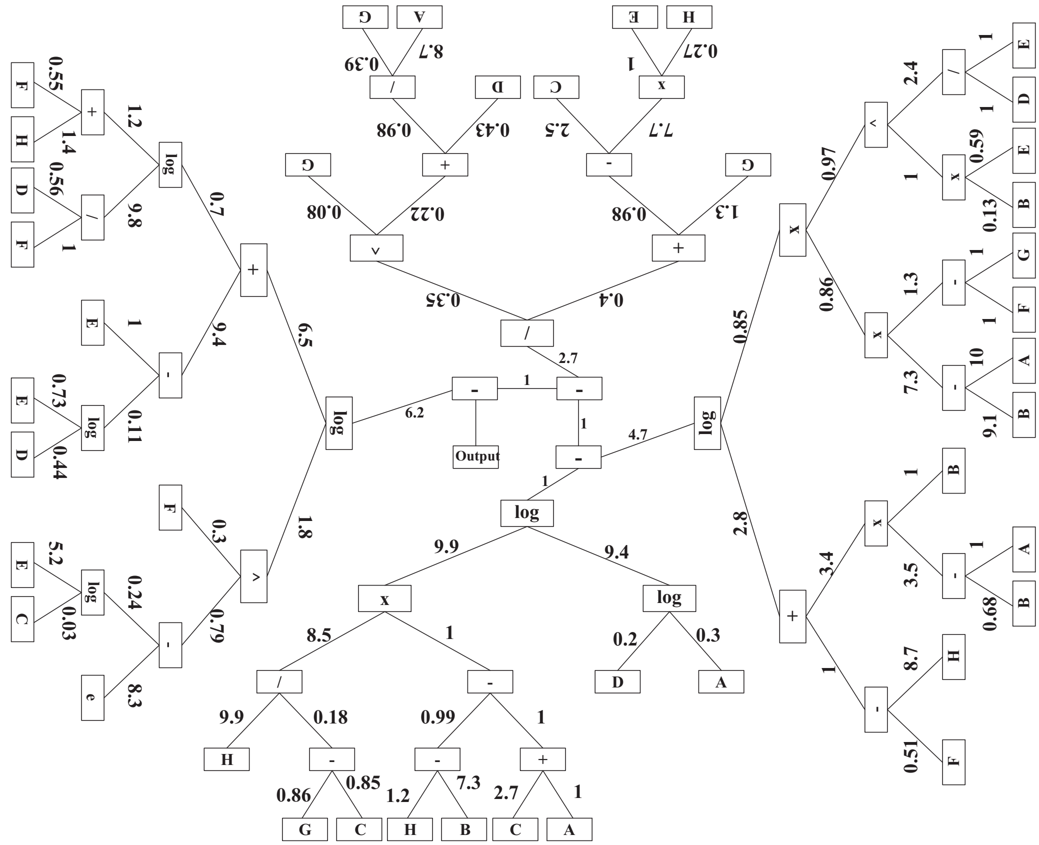

Fig. 13. Best solution GWPOT model.

be decoded into a complete formula using Eqs. (18) through (22). Figs. 14 and 15 show scatter diagrams for the actual values and predicted values of, respectively, training data and testing data.

$$x_1 = First\ tree\ output = \frac{\ln((833.085H/13.026G - 12.875C) \times (11.761H - 71.547B - 26.73C - 9.9A))}{\ln \log_{1.88A} 2.82D} \quad (18)$$

$$x_2 = Second\ tree\ output = \frac{\ln(1.428F - 2.436H + 215.704B^2 - 317.206AB)}{\ln((48.56B - 53.363A)(0.95F - 0.95G)((0.052BE)^{D/E}))} \quad (19)$$

$$x_3 = Third\ tree\ output = \frac{0.52G + 2.46EH - 0.98C}{(0.033D + (0.657A/0.029G))^{0.028G}} \quad (20)$$

$$x_4 = Fourth\ tree\ output = \frac{\ln((\log_{(24.97D/44.59F)}(3.003F + 7.644H)) + 6.11E - \log_{2.957D} 4.906E)}{\ln(0.54F^{(\log_{0.01C} 1.775E) - 11.803e})} \quad (21)$$

$$y = -90.089 + 17.628(x_1 - 4.7x_2 - 2.7x_3 - 6.2x_4) \quad (22)$$

Wider search area allows GWPOT to precisely identify the relationships between all the input variables (variables *A* through *H*) and the prediction output. Meanwhile, the result shows that GOT and WOS exclude some input variables. In the GOT formula example (Eq. (16)), super plasticizer, fine aggregate, and coarse aggregate are not included in the formula. Peng et al. stated that excluding some variables from a formula does not mean they do not impact the compressive strength (Peng et al., 2009). The GWPOT depicts the relationship between all input–output variables and shows that each variable has a distinct influence on HPC compressive strength.

This study compared 3 types of OT to verify that the proposed new OT outperforms the other two. The best-solution configuration for each OT was determined and used in this comparison. These configurations were: 5-layer for the GOT model, 6-layer for the WOS model, and 5-layer for the GWPOT model. Table 6 summarizes comparisons among these models.

GWPOT obtained the best result for every performance measure, with an RI value of 1.00. GOT and WOS obtained *RI* values of 0.00 and 0.458, respectively. Due to the superior performance of GWPOT over the two other OT models, the GWPOT model was tested against other prediction techniques.

4.6. Comparison

This section presents the results of comparing GWPOT to other prediction techniques including SVM, ANN, and ESIM. The GWPOT result was obtained as explained in the previous section using a 5-layer model in fold set one, and 5-fold cross validation was performed on all results. Table 7 presents model results for comparison.

GWPOT obtained a better *RI* value than SVM and ANN but worse than ESIM. ESIM demonstrated its superiority with a prediction *RI* value of 0.918 compared to *RI* values for GWPOT, SVM, and ANN of 0.085, 0.072, and 0.684, respectively. Although

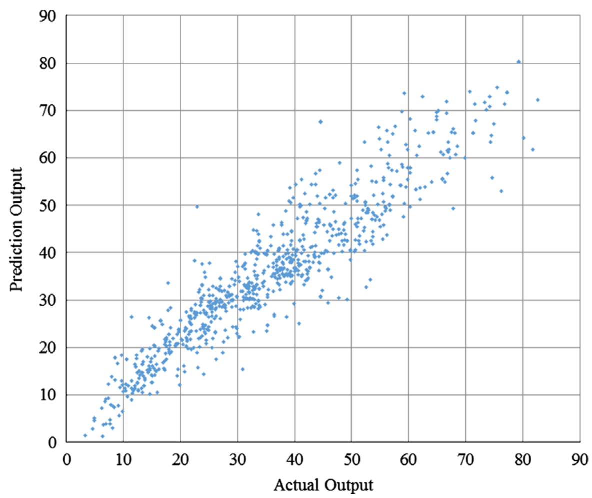

Fig. 14. Scatter diagram, GWPOT training data.

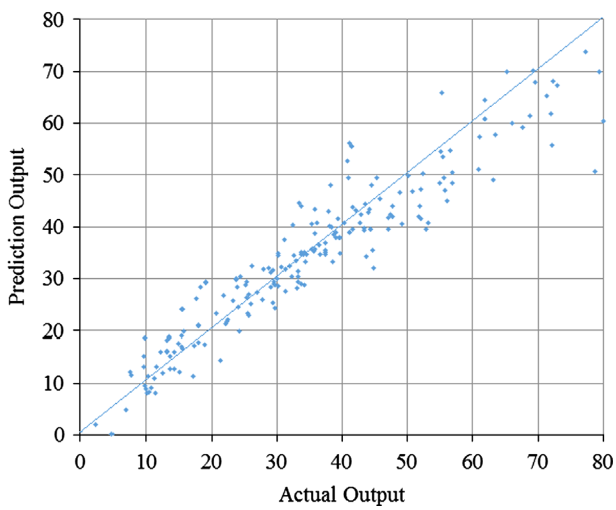

Fig. 15. Scatter diagram, GWPOT testing data.

Table 6
Performance measurement results of various OT techniques.

Prediction tools	Training			Testing			*RI*
	RMSE	*MAE*	*MAPE*	*RMSE*	*MAE*	*MAPE*	
GOT	7.117	5.537	18.661	7.120	5.509	18.611	0.000
WOS	6.646	5.070	16.668	6.890	5.230	16.909	0.458
GWPOT	5.864	4.440	14.986	6.379	4.787	16.095	1.000

the *RI* value for GWPOT fell short of the *RI* value for ESIM, GWPOT remains capable of competing with ESIM, as demonstrated by the superior RMSE value obtained by GWPOT (6.379) compared to ESIM (6.574).

4.7. Sensitivity analysis

OT uses a single-tree structure to build its model while GWPOT uses 4 trees to form a pyramid shape. Other OT model configurations such as 2-tree and 3-tree exist as well. To increase the validity of the GWPOT concept, this study conducted another comparative analysis that used various combinations of layer numbers and tree numbers to identify differences among these parameters. Each tree number and layer number used the 5-fold

Table 7
Performance measurement results of various prediction techniques.

Prediction technique	Training			Testing			*RI*
	RMSE	*MAE*	*MAPE*	*RMSE*	*MAE*	*MAPE*	
SVM	6.533	4.610	16.356	7.170	5.295	18.610	0.085
ANN	6.706	5.189	18.761	6.999	5.416	19.632	0.072
ESIM	3.294	1.597	5.191	6.574	4.160	13.220	0.918
GWPOT	5.864	4.440	14.986	6.379	4.787	16.095	0.684

Table 8
RI Values from various layer and tree number configurations.

Number of trees	Number of layers			
	3	4	5	6
1	0.2960	0.5341	0.6665	0.7283
2	0.5820	0.6766	0.7241	0.7752
3	0.6569	0.7128	0.7654	0.7858
4	0.6764	0.7354	0.7901	0.7693
5	0.7197	0.7419	0.7823	0.7537

cross validation technique to avoid potential bias. Table 8 shows average *RI* results for each fold set.

As shown in Table 8, the 5-tree structure with 4 layers generated the best result with an *RI* value of 0.792. The 5-tree, 5-layer model obtained a good result of 0.790, which differed only slightly from the best *RI* solution. The worst *RI* result was 0.296, produced by the 1-tree, 3-layer model.

The unexpected results and the flexibility of the solution indicate that another OT model may generate the ultimate best solution. However, the 4-tree, 5-layer model obtained the best result in this study.

5. Conclusions

This study develops a new GWPOT to predict HPC compressive strength. Accurately predicting HPC strength is critical to building a robust prediction model for HPC. The GWPOT model employs 4 hierarchical OT to form the pyramidal shape. Multiple trees widen the GA search area, creating a formula that is more complex and more flexible to fit with the data. Comparisons between GWPOT and other prediction techniques, including GOT, WOS, SVM, ANN, and ESIM, showed the highly competitive performance of GWPOT in predicting HPC compressive strength.

GWPOT performs comparably to the well-known ESIM in terms of prediction accuracy. However, while ESIM uses a black-box approach that does not depict input–output relationships in an explicit formula, GWPOT generates and shows corresponding formulae for these relationships. This comparative transparency gives GWPOT an important advantage in practical applications. In future research, another optimization technique may be developed to replace the GA technique used in this study for further comparison. Additionally, the efficacy of GWPOT may be further tested and verified on other construction management case studies.

References

Chen, K.-T., Kou, C.-H., Chen, L., Ma, S.-W., 2012. Application of genetic algorithm combining operation tree (GAOT) to stream-way transition. In: Proceedings of the 2012 International Conference on Machine Learning and Cybernetics, vol. 5. IEEE, Xian, China, pp. 1774–1778.

Cheng, M.-Y., Chou, J.-S., Roy, A.F.V., Wu, Y.-W., 2012. High-performance concrete compressive strength prediction using time-weighted evolutionary fuzzy support vector machines inference model. Automat. Constr. 28, 106–115.
Cheng, M.-Y., Wu, Y.-W., 2009. Evolutionary support vector machine inference system for construction management. Automat. Constr. 18, 597–604.
Chou, J.-S., Chiu, C.-K., Farfoura, M., Al-Taharwa, I., 2011. Optimizing the prediction accuracy of concrete compressive strength based on a comparison of data-mining techniques. J. Comput. Civ. Eng. 25, 242–253.
Holland, J.H., 1975. Adaptation in Natural and Artificial Systems. University of Michigan Press.
Hsie, M., Ho, Y.-F., Lin, C.-T., Yeh, I.C., 2012. Modeling asphalt pavement overlay transverse cracks using the genetic operation tree and Levenberg–Marquardt Method. Expert Syst. Appl. 39, 4874–4881.
Mousavi, S.M., Aminian, P., Gandomi, A.H., Alavi, A.H., Bolandi, H., 2012. A new predictive model for compressive strength of HPC using gene expression programming. Adv. Eng. Software 45, 105–114.
Peng, C.-H., Yeh, I.-C., Lien, L.-C., 2009. Building strength models for high-performance concrete at different ages using genetic operation trees, nonlinear regression, and neural networks. Eng. Comput. 26, 61–73.
Tsai, H.-C., 2011. Weighted operation structures to program strengths of concrete-typed specimens using genetic algorithm. Expert Syst. Appl. 38, 161–168.
Yeh, I.-C., 1998. Modeling of strength of high-performance concrete using artificial neural networks. Cem. Concr. Res. 28, 1797–1808.
Yeh, I.-C., 1999. Design of high-performance concrete mixture using neural networks and nonlinear programming. J. Comput. Civ. Eng. 13, 36–42.
Yeh, I.-C., Lien, C.-H., Peng, C.-H., Lien, L.-C., 2010. Modeling concrete strength using genetic operation trees. In: Proceedings of the 2010 International Conference on Machine Learning and Cybernetics, vol. 3. IEEE, Qindao, China, pp. 1572–1576.
Yeh, I.-C., Lien, L.-C., 2009. Knowledge discovery of concrete material using Genetic Operation Trees. Expert Syst. Appl. 36, 5807–5812.