Prediction of Compressive Strength of Concrete: Critical Comparison of Performance of a Hybrid Machine Learning Model with Standalone Models

Rachel Cook[1]; Jonathan Lapeyre[2]; Hongyan Ma[3]; and Aditya Kumar, A.M.ASCE[4]

Abstract: The use of machine learning (ML) techniques to model quantitative composition–property relationships in concrete has received substantial attention in the past few years. This paper presents a novel hybrid ML model (RF-FFA) for prediction of compressive strength of concrete by combining the random forests (RF) model with the firefly algorithm (FFA). The firefly algorithm is utilized to determine optimum values of two hyper-parameters (i.e., number of trees and number of leaves per tree in the forest) of the RF model in relation to the nature and volume of the dataset. The RF-FFA model was trained to develop correlations between input variables and output of two different categories of datasets; such correlations were subsequently leveraged by the model to make predictions in previously untrained data domains. The first category included two separate datasets featuring highly nonlinear and periodic relationship between input variables and output, as given by trigonometric functions. The second category included two real-world datasets, composed of mixture design variables of concretes as inputs and their age-dependent compressive strengths as outputs. The prediction performance of the hybrid RF-FFA model was benchmarked against commonly used standalone ML models—support vector machine (SVM), multilayer perceptron artificial neural network (MLP-ANN), M5Prime model tree algorithm (M5P), and RF. The metrics used for evaluation of prediction accuracy included five different statistical parameters as well as a composite performance index (CPI). Results show that the hybrid RF-FFA model consistently outperforms the standalone ML models in terms of prediction accuracy—regardless of the nature and volume of datasets. **DOI: 10.1061/(ASCE)MT.1943-5533.0002902.** © *2019 American Society of Civil Engineers.*

Author keywords: Machine learning; Concrete; Compressive strength; Random forests; Firefly algorithm.

Introduction

The idea of using data-driven methods—such as supervised machine learning (ML)—for prediction and optimization of materials' performance forms the premise of the United States Materials Genome Initiative (Jain et al. 2013). In pursuit of the idea, researchers from various scientific domains have compiled extensive datasets of materials and subsequently employed ML models to better understand the underlying quantitative composition–performance correlations (Carrasquilla and Melko 2017; Pilania et al. 2013; Ward et al. 2016; Zdeborová 2017). Knowledge of such correlations, for any given material, can be leveraged to mitigate the cost and time involved in an Edisonian approach—involving rigorous and iterative synthesis-testing/analyses cycles (Liu et al. 2017)—to predict a material's properties or to design a new material that meets a desired set of performance criteria.

Concrete—the most produced and used material in the world—has garnered the interest of several researchers working the area of ML. The focus of various research articles (Akande et al. 2014; Behnood et al. 2017; Chou et al. 2010, 2014; Duan et al. 2013; Gupta et al. 2006; Kasperkiewicz et al. 1995; Nagwani and Deo 2014; Omran et al. 2016; Veloso de Melo and Banzhaf 2017; Yeh 1998a, b; Yeh and Lien 2009; Young et al. 2019; Zarandi et al. 2008) has been to employ ML models to predict concretes' properties (e.g., compressive strength and rheological parameters) using their mixture design variables [e.g., contents of cement, water, mineral additive(s), and admixture(s)] and age as inputs. Among concrete's properties, compressive strength is deemed relatively more important for quality control, and it is widely used as the primary specification criterion for construction of structures. Furthermore, compressive strength of concrete is very well correlated with other mechanical properties (e.g., elastic modulus, flexural strength) and, therefore, can be used to qualitatively estimate the overall mechanical stability and survivability of the structure (Manning and Hope 1971; Oluokun et al. 1991). Because of these reasons, most of the aforementioned studies have focused on predicting the compressive strength of concrete. The utilization of ML models—as opposed to Edisonian approaches, or simple linear regression models—is to overcome complexities pertaining to (1) the staggeringly large compositional degrees of freedom in concrete (i.e., mixture design variables, permutations and combinations of which can vary within wide ranges and exert significant influence on properties) and (2) the inherent nonlinear relationships between mixture design variables and properties of concrete. To be more specific on the latter point, the properties of concrete

[1]Graduate Student, Dept. of Materials Science and Engineering, Missouri Univ. of Science and Technology, Rolla, MO 65409. Email: recwx7@mst.edu

[2]Graduate Student, Dept. of Materials Science and Engineering, Missouri Univ. of Science and Technology, Rolla, MO 65409. Email: jllgg3@mst.edu

[3]Assistant Professor, Dept. of Civil, Architectural, and Environmental Engineering, Missouri Univ. of Science and Technology, Rolla, MO 65409. Email: mahon@mst.edu

[4]Assistant Professor, Dept. of Materials Science and Engineering, Missouri Univ. of Science and Technology, B49 McNutt Hall, 1400 N Bishop, Rolla, MO 65409 (corresponding author). ORCID: https://orcid.org/0000-0001-7550-8034. Email: kumarad@mst.edu

Note. This manuscript was submitted on January 9, 2019; approved on May 29, 2019; published online on August 19, 2019. Discussion period open until January 19, 2020; separate discussions must be submitted for individual papers. This paper is part of the ***Journal of Materials in Civil Engineering***, © ASCE, ISSN 0899-1561.

are complex—highly nonlinear, and, often, non-monotonous—functions of mixture design variables. For example, within the cement paste component of concrete, compressive strength decreases with increasing water-to-cement ratio (w/c, mass basis); however, this relationship exhibits nondifferentiability at the critical w/c of 0.42, below which the paste becomes water-deficient (Jennings et al. 2015; Powers and Brownyard 1946). This relationship between the w/c and compressive strength is further convoluted when a fraction of the cement is replaced with a reactive mineral additive (e.g., fly ash and blast furnace slag) (Li and Zhao 2003; Poon et al. 2000). Therefore, sophisticated approaches, such as ML, are required to reveal the hidden, and complex, semiempirical rules that govern the correlation between mixture design and properties of concrete.

The majority of past studies, focused on prediction of compressive strength of concrete, have used nonlinear regression based ML models [i.e., artificial neural network (ANN) (Schalkoff 1997) or support vector machine (SVM) (Hearst et al. 1998)]—presumably because of nonlinear correlations between mixture design variables and properties of concrete. Notable among these are studies conducted by Akande et al. (2014), Chopra et al. (2016, 2018), Chou et al. (2010, 2014), Duan et al. (2013), Omran et al. (2016), Yeh (1998a, b), and Young et al. (2019). The studies have shown that both ANN and SVM models, when trained using a sufficiently large dataset, can predict compressive strength of concrete with reasonable accuracy (i.e., coefficient of determination, $R^2 \geq 0.90$). However, it has been reported that ANN and SVM models are unreliable in making predictions in data domains that feature a highly nonlinear, periodic functional relationship between one or more input variables and the output (Cunningham et al. 2000; Yao 1999; Zhang et al. 1998). This is because both ANN and SVN models employ local search or optimization algorithms (e.g., back-propagation algorithm used in ANN), which are faced with an inherent drawback of getting trapped in local minima—especially when the functional relationship between input variables and output is composed of multiple local minima (e.g., periodic trigonometric functions)—rather than converging to the global minima. Because of this drawback, with every rerun of ANN or SVM during the training period (i.e., when different subsets of the same dataset are used to train the models) the convergence could occur at different local minima, thus resulting in disparate prediction performances (i.e., the ability to reliably predict outputs using previously unseen input variables) during the testing (Cunningham et al. 2000; Yao 1999; Zhang et al. 1998). This deficiency can be amended by using algorithms based on genetic programing (Chopra et al. 2016; Veloso de Melo and Banzhaf 2017). Alternatively, bootstrap aggregation of outputs of several models developed from various subsets of the training dataset—for example, by using bagging, voting, or stacking approaches (Chou et al. 2014; Polikar 2006)—can also be used. However, such techniques could slow down the rate of convergence or, in the worst case, result in overfitting (Dietterich 2000). This issue of ANN and SVM models, pertaining to their inferior performance on datasets featuring periodic input–output relationship, is further examined in the "Results and Discussion" section.

ANN and SVM models have another drawback—the objective functions produced by them are difficult to interpret and, therefore, difficult (albeit not impossible) to employ for optimization purposes. This is because the functional relationship between input variables and output is not yielded as a transparent mathematical formula. To overcome this limitation, in some recent studies (Behnood et al. 2017; Chou et al. 2014; Omran et al. 2016; Veloso de Melo and Banzhaf 2017; Young et al. 2019) *decision trees*–based ML models have been employed. Decision trees are essentially rules-based models, wherein the training dataset is classified into multiple independent subsets and subsequently processed using a collection of linear or nonlinear regression methods to develop multiple (transparent) functional relationships between input variables and output. Notable among these are application of the M5Prime (M5P) (Quinlan 1992; Wang and Witten 1997; Behnood et al. 2017; Deepa et al. 2010; Omran et al. 2016) and random forests (RF) (Breiman 2001; Chopra et al. 2018; Young et al. 2019) models. Deepa et al. (2010) and Behnood et al. (2017) showed that prediction accuracy of the M5P model was superior compared to the ANN model. Similarly, as per the R^2 values reported by Young et al. (2019), predictions of concrete compressive strength using the RF model were more accurate (albeit not comprehensively) compared to those using ANN and SVM models.

To the best of the authors' knowledge, the prediction performances of M5P and RF models (i.e., in terms of predicting the compressive strength of concrete) have not been compared. Notwithstanding, the RF model is hypothesized to be superior compared to the M5P model; this hypothesis will be corroborated in the "Results and Discussion" section. The premise of this hypothesis is that the M5P model assumes a multivariate linear relationship between input variables and output in each subset of the training data (Quinlan 1992; Wang and Witten 1997). More specifically, the training dataset is split into multiple subsets until in each subset a simple, linear input–output correlation can be established. Because of this limitation—like in the ANN and SVM models—predictions made by the M5P model in domains that feature highly nonlinear and/or periodic relationships between input variables and output are expected to be inaccurate. The RF model, on the other hand, does not suffer from this limitation because of its ability to handle continuous and discrete variables over both monotonous and non-monotonous domains (Breiman 2001). Notwithstanding, in the RF model, it is important to fine-tune two hyper-parameters—that is, the number of trees in the forest and the number of leaves per tree—to ensure that input–output correlations are identified and captured, and predictions are accurate. In the absence of an optimization algorithm, the two hyper-parameters need to be adjusted through trial and error [e.g., by using multifold cross-validation (Schaffer 1993)], which can be time-consuming and difficult. In a recent study (Ibrahim and Khatib 2017), it was shown that the firefly algorithm (FFA)—a metaheuristic optimization algorithm (Yang 2009)—can be used to determine optimum values of the two aforementioned hyper-parameters in relation to the volume and nature of the training dataset. The authors showed that by combining RF with FFA predictions (of global solar radiation) were rendered more accurate compared to those made by various standalone and ensemble ML models. In another study (Chou and Pham 2015), FFA was used in conjunction with SVM to predict concretes' compressive strength. Although the combined [SVM + FFA] model had better prediction performance than the standalone SVM model, it is expected that inherent limitations of SVM—as described previously—could have compromised the prediction performance of the combined model. To the best of the authors' knowledge, the combination of RF and FFA has never been used to process concrete datasets. Given the superior prediction performance of RF [compared to SVM as well as other ML models (Young et al. 2019)], it is deemed important to examine whether combining RF with FFA would result in further improvements in accuracy of predictions of concretes' compressive strength.

This study presents the first application of the hybrid RF-FFA model—developed by combining the RF model with FFA—to predict compressive strength of concretes in relation to their mixture design and age. The performance of the hybrid model is benchmarked against commonly used standalone ML models (i.e., SVM,

multilayer perceptron ANN, M5P, and RF). Six different statistical parameters are used for comprehensive evaluation of the models' prediction performance. Firstly, highly nonlinear, non-monotonous, and periodic trigonometric functions are used to evaluate the performance of the ML models. Focus is given to determine if the hybrid model is able to reveal the complex relationship between input variables and output of the trigonometric functions and, more importantly, reliably predict the function outputs in blank (i.e., previously untrained) data domains. Secondly, the prediction performance of the ML models is tested using two real-world concrete datasets, composed of concretes' mixture designs and their corresponding age-dependent compressive strengths. Based on comparisons of prediction performances—of the hybrid RF-FFA model vis-à-vis the standalone ML models—it is shown that the hybrid ML model consistently outperforms the standalone ML models.

The paper is organized as follows. "Machine Learning Models" describes the five ML models implemented in this study. "Data Collection and Performance Evaluation of Machine Learning Models" describes the datasets used for training and testing of the ML models. A brief description of statistical parameters used for evaluation of prediction performance of the models is also included. "Results and Discussion" reports the results and comparison of prediction performances of various ML models. "Conclusions" presents a summary of the main findings of the study.

Machine Learning Models

This section provides a brief overview of the machine learning (ML) models implemented in this study. In each of the following subsections, one or more original references are provided that describe the formulation and implementation of the ML model in more detail.

Multilayer Perceptron Artificial Neural Network

An artificial neural network (ANN) consists of several computational elements (termed as neurons) arranged in layers, resembling the network of neurons in the human brain responsible for processing information in a hierarchical fashion (Schalkoff 1997). The multilayer perceptron artificial neural network (MLP-ANN) is a subclass of ANN with strong self-learning capabilities (Gardner and Dorling 1998). The hierarchical structure of MLP-ANN is composed of (1) one input layer, which contains a set of neurons representing the input variables (e.g., concrete mixture design variables); (2) one or more hierarchical hidden layers, which contain computational neurons to process the information received from the previous layer so that it can be refined and passed on to the next layer; and (3) one output layer, which contains a computation node to produce the final prediction (e.g., compressive strength of concrete). Each neuron in any given hidden layer is functionally related—as shown in Eq. (1)—to all neurons in the previous layer

$$N_j = \sum w_{ji} o_i \tag{1}$$

$$y_j = f(N_j) = \frac{1}{1 + e^{w_{ji} \cdot N_j}} \tag{2}$$

here, N_j = activation of the jth neuron; i = set of all neurons present in the previous layer; w_{ji} = weight of connection between neurons j and i; and o_i = output of the neuron. Each neuron uses activation functions (while using all neurons from the previous layer) to calculate intermediate-output values, which are subsequently passed on as input values to the next neuron layer. This process proceeds throughout the network until reaching the final neuron layer that produces the final output. In the MLP-ANN model, activation functions are represented as sigmoidal or logistic-transfer functions (Gardner and Dorling 1998)—as shown in Eq. (2), wherein $y_j = f(N_j)$ is the activation function of the jth neuron. During training of the MLP-ANN model, a back-propagation algorithm (Goh 1995; Mueller et al. 2016) is used to minimize deviation [i.e., root-mean squared error (RMSE) or mean absolute error (MAE)] between actual and predicted values (of the activation functions). This is accomplished by iteratively adjusting and finally determining the optimal connection weights (i.e., w_{ji})—pertaining to each activation function—by using the gradient descent approach or the Levenberg–Marquardt algorithm (Gardner and Dorling 1998; Moré 1978).

In the MLP-ANN model used in this study, the neural network architecture is composed of five hidden layers, wherein each layer is composed of $(2m + 1)$ neurons (Hegazy et al. 1994); m is the number of input variables of the training dataset. The choice of five hidden layers was made based on comparison of prediction performances of the model—whilst varying the number of hidden layers between unity and higher values—against experimental datasets (i.e., datasets pertaining to concrete and those generated using trigonometric functions) described in the "Data Collection" section. For fine-tuning of the parameters (i.e., connection weights of each of the activation functions), the Levenberg–Marquart algorithm, as described in Moré (1978), was chosen because it resulted in superior prediction performance compared to the gradient descent approach. It is clarified that for determination of optimal parametric values (i.e., number of hidden layers) and for selection of the Levenberg–Marquart algorithm, a 10-fold cross-validation (CV) method (Chou et al. 2014; Dietterich 2000; Schaffer 1993) was used as the primary performance-assessment method. The same method has also been used to optimize parameters and functions of other ML models described subsequently in this section. In general, a p-fold CV method (where p is the number of folds, for example, 10 as used in this study) involves randomized stratification of the original dataset into p folds such that each fold contains $100p/N\%$ of the total number (i.e., N) of data-records of the original dataset. It is also important to ensure each of the p folds encompasses an approximately similar proportion of predictor labels as in the original dataset. Next, the model is trained using data-records in $p - 1$ folds out of the p folds and tested against the last fold. This process is repeated $p - 1$ times, in an iterative manner, such that each of the p folds are used once for testing and $p - 1$ times for training the model. With each training-followed-by-testing iteration, the CV error (usually expressed in the form of root-mean squared error) is estimated, and on such basis the optimum functions (e.g., for determination of functional forms of neuron activation) are chosen and the parameters of the ML model (e.g., number of hidden layers in MLP-ANN model) are progressively fine-tuned. Ultimately, the optimum function(s) and values of parameters of the ML model correspond to those that result in the smallest overall value of CV error when accumulated over all experimental datasets.

Support Vector Machine

Support vector machine (SVM) is an ML methodology for approximating the nonlinear relationship between input variables and output of a dataset by using an optimization approach—rather than a regression approach—to minimize a cost (i.e., ε-insensitive loss) function (Smola and Schölkopf 2004; Vapnik 2000). During training, the SVM, firstly, maps the input dataset from a lower-dimensional to a higher-dimensional feature space by using a

mapping procedure. Toward this, a nonlinear kernel function [e.g., polynomial function, sigmoidal function, Gaussian radial basis kernel function, or hyperbolic tangent function (Clarke et al. 2005; Garg and Verma 2006)] is used to fit the input data into a higher-dimensional feature space, wherein the data is distributed in a more sparse form compared to the original one. Next, the SVM attempts to determine a linear objective function $f_{SVM}(x, \omega)$ [see Eq. (3)]—such that its output has a maximum deviation of ε with respect to the actual (measured) value in the training dataset

$$f_{SVM}(x, \omega) = \sum_{i=1}^{n} \omega_i K_i(x) + b \tag{3}$$

In Eq. (3), K_i = set of n nonlinear kernel (i.e., mapping) functions used for transforming the original input data (x) into higher-dimensional feature space; b = a bias term; and ω represents the weight vector consisting of n choice coefficients. In order to derive the optimum objective function [$f_{SVM}(x, \omega)$]—and the associated parameters (i.e., b and ω)—the task of regression is approached as an optimization problem [Eq. (4)], within the constraints shown in Eq. (5). The objective of the optimization effort is to determine the global minimum of Eq. (4), such that for each input value (x), the output of $f_{SVM}(x, \omega)$ is within a Euclidian distance of ε from the actual value (Fang et al. 2008)

$$\text{minimize: } \frac{1}{2}\|\omega^2\| + C\sum_{i=1}^{n} \xi_i + \xi_i^* \tag{4}$$

$$\text{subject to} \begin{cases} y_i - f_{SVM}(x_i, \omega) - b \le \varepsilon + \xi_i \\ f_{SVM}(x_i, \omega) + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i \text{ and } \xi_i^* \ge 0, \text{ and } i = 1, 2, 3 \ldots n \end{cases} \tag{5}$$

$$K(x, y) = e^{(-\gamma\|x-y\|^2)} \tag{6}$$

In Eq. (4), the constant C is called the regularization term and represents the degree of penalty of the sample with error exceeding ε (i.e., when the prediction is farther than ε from the actual value). The parameters, ξ_i and ξ_i^*, shown in Eq. (5) are positive slack variables that represent the Euclidian distance of the predicted value from the corresponding boundary values of the ε-tube (i.e., a tube representing the actual training dataset, wherein each data-record is bounded by the maximum allowable error of ε). Therefore, based on the formulation described here, to derive the optimum objective function, $f_{SVM}(x, \omega)$ (and optimum values of the bias, b, and choice coefficients, ω)—that reliably links input variables with the output—the parameters that need to be optimized are ε, C, and any parameter associated with the kernel function (e.g., the parameter γ, which is associated with the Gaussian radial basis kernel function, as shown in Eq. (6) (Clarke et al. 2005; Garg and Verma 2006).

In the SVM model used in this study, three different kernel functions (i.e., Gaussian radial basis kernel function, 3rd-to-5th order polynomial functions, and sigmoidal function) were used. Based on comparisons of prediction performances of the model—ascertained using the 10-fold CV method (James et al. 2013; Schaffer 1993)—on experimental datasets (described in the "Data Collection" section), the Gaussian radial basis kernel function was chosen. Within the kernel function [see Eq. (6)], the optimum value of γ was determined (as per the CV method) as 0.10. The optimum values of ε (representing the radius of the ε-tube) and C (the regularization term) were determined as 1.5 MPa and 5.0, respectively.

M5Prime Model Tree Algorithm

The M5Prime model tree algorithm—often abbreviated as the M5P model—is a modification (Wang and Witten 1997) of the M5Rules algorithm introduced by Quinlan (1992). The M5P model is, essentially, a decision tree model that performs logical splits in the training dataset so that the input variables can be linked with the output using multivariate linear functions. During training of the M5P model, the input space is split in several subspaces, while ensuring that data in each subspace share at least one common attribute (e.g., similar range of one or more input variables). More specifically, the splitting of data is performed in a manner that data that are alike—in terms of one or more of their attributes—are clustered and contained within the same subspace. By repeating this procedure iteratively, several subspaces, each consisting of harmonious data, are obtained. Standard deviation is typically used as the criterion for determining the specific attribute or attributes on the bases of which optimal splitting of the dataset can be achieved. Such reduction in standard deviation allows determination, and subsequent creation, of several nodes (i.e., data clusters) on the bases of attributes. By creating such nodes, the model enables the building of an upside-down tree-like structure, wherein the root is at the top and the leaves are at the bottom. Once the tree is built, a new data-record propagates hierarchically from the root and down through the nodes until it reaches a leaf. Each node consists of a mathematical logic, which compares the new data-record with that of the split value and helps the data-record propagate down through the nodes until it reaches the appropriate leaf. In the M5P model, a linear regression model is used as the aforementioned mathematical logic. The regression model is developed in each of the subspaces, linking input variables and output of the data-records contained within it. To ensure that the tree-structure is optimal, a pruning technique is used to overcome the issue of overtraining—that is, when the chosen attribute, for any given subspace, is not the optimal one with the maximum expectation to reduce error (i.e., standard deviation, in this study). Use of the pruning technique, however, can result in discontinuities between adjacent linear models. To mitigate this issue, a smoothing operation is employed in the final step. Such iterative pruning and smoothening operations ultimately unify all linear regression models—across all the nodes between the root and leaf of the tree—into a singular, continuous model.

The M5P model used in this study was implemented using the Waikato Environment for Knowledge Analysis (WEKA) workbench toolbox (Frank et al. 2004; Holmes et al. 1994). The only variable parameter in the M5P model is the *minimum number of splits* in the dataset. This value was varied widely from unity to higher values, and the prediction performance was measured—using the 10-fold CV method—using experimental datasets (described in the "Data Collection" section) for both training and testing. Based on such evaluations, the optimum value of *minimum number of splits* was selected as 4.

Random Forests

Random forests (RF) is a modified version of the bagging (i.e., bootstrap aggregation) decision tree algorithm, which assimilates the concepts of *adaptive nearest neighbors* and *bagging* to achieve effective data-adaptive inference (Breiman 2001; Chen and Ishwaran 2012). Like in conventional decision tree models, the RF model utilizes a set (albeit large) number of independent trees, and within those trees are encompassed subsets of the homogenized training data (Breiman 1996). The elementary unit of RF is a binary tree constructed using the same foundational principle as that used in the classification and regression tree (CART) model—wherein binary splits partition the tree, in recursive fashion, into

near-homogeneous terminal nodes. When the binary split is done optimally, the propagation of data from a parent node to its two children nodes occurs in a way that ensures that homogeneity in the data between the children nodes is improved compared to the parent node. A typical RF model is composed of hundreds or thousands of trees, wherein each tree is grown using a unique bootstrap sample of the training data. Compared to the CART model, the RF model is different in the sense that trees in the latter model are grown nondeterministically using a two-stage randomization procedure. Here, the first stage of randomization involves growing the tree using a randomly chosen bootstrap sample of the original data. The second stage of randomization features at the node level—wherein, as opposed to splitting the tree node using all variables (as done in the CART model or the M5P model described in the previous subsection), a random subset of variables is selected and only those variables are used for ascertaining the best split of the node. On account of the two-stage randomization, the trees grown in the RF model are de-correlated and the ensemble has low variance. Based on this description, construction of the RF model can be summarized in the following steps:

- "n_t" bootstrap samples are drawn randomly from the original training dataset. At this point, the number of bootstrap samples (i.e., n_t) is equal to the number of trees. Based on prior studies (Chen and Ishwaran 2012; Ibrahim and Khatib 2017; Svetnik et al. 2003), the optimum value of n_t is ≈66.66% of the overall training dataset. The remaining ≈33.33% of the dataset are labelled as *out-of-bag* (OOB) data.
- From each of the n_t bootstrap datasets, a tree is grown. Unlike conventional decision tree models (e.g., the M5P and CART models), each tree in the RF model is an unpruned regression tree, wherein at each node, rather than choosing all variables of the training dataset, a random sample of m_{try} variables is chosen. As the tree is grown, the number of leaves per tree (n_{LV}) is held constant across the entire ensemble. Based on previous studies (Chen and Ishwaran 2012; Ibrahim and Khatib 2017), the optimum value of n_{LV} is between 3 and 10. In this study, the optimum value—assessed by performing predictions on different experimental datasets (described in the "Data Collection" section)—was determined as 5.
- Next, each of the n_t trees is utilized to predict a data point outside of the selected bootstrap space. Output of the prediction is designated as out of bag (OOB) prediction (Breiman 1996). These OOB predictions, for a given input vector (x), are designated as $f_{\text{RF}}(x)$ for each of the n_t decision trees; all OOB predictions are subsequently aggregated and averaged to produce the overall OOB prediction $\hat{f}_{RF}^{n_t}(x)$ and OOB error rate [see Eq. (7)]. Simply put, $\hat{f}_{RF}^{n_t}(x)$ is the arithmetic mean of the predicted values collated from all of the n_t trees. The OOB predictions—especially the OOB error rate—provide a good measure of influence of each variable on the output, which can be quantitatively estimated as variable importance (VI), thus eliminating the need for a test set or cross-validation (Schaffer 1993). The VI of a given variable is, essentially, a measure of increment in OOB error rate when OOB prediction for a given variable is permuted while all others are left unchanged (Svetnik et al. 2003).
- Once the OBB predictions, OOB error rate, and VI are calculated, outliers in the training dataset are detected using cluster analysis (e.g., K-means cluster analysis, density model, and mean-shift clustering) (Sarstedt and Mooi 2014). In the study, the K-means cluster analysis (Hartigan and Wong 1979) was used to determine data-records that do and do not belong in clusters. The data-records that do not belong in clusters are removed and subsequently replaced in an iterative manner through the training process.
- Lastly, in the testing phase, new input dataset—which is not part of the training dataset—is used to perform predictions. For each input data-record, the predicted value corresponds to the average of predictions from all of the n_t trees

$$\hat{f}_{RF}^{n_t}(x) = \frac{1}{n_t}\sum_{j=1}^{n_t} f_{RF_j}(x) \tag{7}$$

The RF model has a number of unique advantages. In the RF model, a large number of trees are grown (as opposed to other decision tree models)—on a one-node-at-a-time basis; as such, errors resulting from generalization are minimized and, therefore, the likelihood of overfitting the training data is negligible (Biau et al. 2008). Minimization of generalization, enabled by the large number of trees, entails that the RF model is able to proficiently deal with complex interactions and correlations among variables of the training dataset. By allowing each of n_t trees to grow to its maximum size (i.e., by allowing *deep* trees), without any pruning, and selecting only the best splits among a random subset at each node, the RF model is able to concurrently maintain diversity among trees and prediction performance. The two-stage randomization—as described previously—diminishes correlation among unpruned trees, keeps the bias low, and reduces variance. Lastly, the RF model is easy to implement because the number of trees (n_t) and the number of leave per tree (n_{LV}) are the only two hyper-parameters that need to be optimized by the user. Both of these hyper-parameters were optimized by the 10-fold CV method (James et al. 2013; Schaffer 1993), whilst training and testing the model using experimental datasets described in the Data Collection section. Through such assessments, the optimum values of n_t and n_{LV} for the standalone RF model were determined as 450 and 5, respectively.

Hybrid Random Forests: Firefly Algorithm Model

Firefly Algorithm

The FFA—originally conceived by Yang (2009)—is an optimization algorithm (Lukasik and Żak 2009; Yang and He 2013) based on idealized behavior of the flashing characteristics of fireflies. The rules of idealized flashing characteristics are (1) each firefly is attracted to all other fireflies; (2) the magnitude of attractiveness between any two fireflies is proportional to the difference in brightness between them; (3) the movement of a firefly is always toward a firefly with greater brightness; and (4) the brightness of the firefly is determined by the landscape [i.e., the objective function, $f(x)$, that is to be optimized]. The brightness, I, of a firefly at a particular location, x, can be chosen as $I(x)$, which is directly proportional to the objective function, $f(x)$. The attractiveness, β, between a pair of fireflies is a function of the distance, r_{ij}, between firefly i and firefly j. Likewise, the brightness, I_i, of firefly i varies with the distance r_i from the source in a monotonic and exponential manner [Eq. (8*a*)]

$$I_i = I_0 e^{-\gamma r_i} \tag{8a}$$

where,

$$\gamma = \frac{\gamma_0}{r_{\max}} \tag{8b}$$

here, I_0 = brightness at the source (typically set at 1.0); and γ = light absorption coefficient (representing the potential of fireflies to absorb light from the source). The value of γ can range from

J. Mater. Civ. Eng., 2019, 31(11): 04019255

0 to 10 (Yang 2009). In this study, however, γ was calculated using Eq. (8*b*), wherein $\gamma_0 = 1.0$, and $r_{\max}$ is the maximum of distances between all pairs of fireflies (Lukasik and Żak 2009) in the landscape. The mathematical formulation of the magnitude of attractiveness (β_{ij}) between two fireflies (i and j), in relation to the distance between them (r_{ij}), is given by Eq. (9):

$$\beta_{ij} = \beta_0 e^{-\gamma r_{ij}^m} \tag{9}$$

where β_0 = maximum attractiveness between a pair of fireflies (i.e., at $r = 0$); and m = a positive coefficient (ranging between 2 and 4). The values of β_0 can range from 0 to 1, wherein the upper bound represents cooperative local search with the brightest firefly dictating positions of most fireflies in the swarm. As the value of β_0 digresses from 1, dominance of the brightest firefly decreases, thus rendering the local search progressively more noncooperative. In this study, a value of 0.80 was used for β_0 and m was set at 2.0. The movement of a firefly i, as it is attracted to a brighter firefly j, is determined by Eq. (10). Here, $x_{i,\text{old}}$ and x_j are locations of the fireflies i and j, respectively, and the last term is randomization with the vector of random variables (ε_i) drawn from a Gaussian distribution; $x_{i,\text{new}}$ is the new position of firefly i as its moves because of its attraction toward firefly j

$$x_{i,\text{new}} = x_{i,\text{old}} + \beta_0 e^{-\gamma r_{ij}^m}(x_j - x_i) + \alpha\varepsilon_i \tag{10}$$

Based on the previously mentioned criteria and definitions, the FFA can be used to minimize a continuous, constrained cost function $f(x)$. Firstly, it is assumed that there exists a swarm of m fireflies—distributed randomly over the landscape. The fireflies are tasked to find x^*, wherein the value of the cost function $f(x^*)$—or, in other words, the overall brightness of the landscape—is minimum. Next, the FFA is implemented in the following steps: (1) all fireflies of the swarm are allowed to move, in a sequential manner, such that each firefly moves toward another in the neighborhood on the basis of its attractiveness toward the other firefly (which is a function of difference in brightness and the distance between the two fireflies); (2) once all fireflies have been allowed to move to their new locations, based on the new configuration of fireflies, the overall brightness of the landscape is updated, and assessed if it is lower than the original one; and (3) Steps 1 and 2 are repeated iteratively until convergence is reached, wherein the overall brightness of the landscape reaches the global minimum (i.e., the value does not change by more than 10^{-6} units between three successive iterations).

Hybrid Model

In the RF model—as described in the previous subsection [i.e., Random Forests (RF)]—it is important to fine-tune the two hyper-parameters—that is, the number of trees in the forest (n_t) and the number of leaves per tree (n_{LV})—to ensure that predictions are accurate. Typically, the two parameters are adjusted through trial and error or by cross-validation, which can be time-consuming and difficult. In a recent study (Ibrahim and Khatib 2017), it was shown that the firefly algorithm (FFA)—described in the section on the firefly algorithm (FFA)—can be used to determine optimum values of the two aforementioned hyper-parameters in relation to the nature and volume of the dataset. The authors showed that by combining RF with FFA, predictions were rendered more accurate compared to those made by various standalone and hybrid ML models—including the RF model.

In this study, the structure of the hybrid model has been drawn from the work of Ibrahim and Khatib (2017). In Stage 1, the RF model is implemented, wherein the values of n_t and n_{LV} are set at 450 and 5, respectively. In Stage 2, the FFA is implemented in the following steps:

- An objective function, $f(x)$, is defined, which corresponds to the total root-mean squared error (RMSE: described in the "Evaluation of Prediction Performance of Machine Learning Models" section) of predictions of the RF model with respect to actual values of the training dataset used in Stage 1.
- The FFA is implemented—by following the steps detailed in the firefly algorithm (FFA) section—to optimize the values of n_t and n_{LV} such that the objective function [i.e., $f(x) = \text{RMSE}$ of the RF model] continually decreases. Toward this, at the end of every iteration of the FFA, the RF model is implemented to update predictions based on new values of the two hyper-parameters; based on the predictions, the RMSE is also updated to be used in the next iteration.
- The values of n_t and n_{LV}, at which the objective function reaches a global minimum (changes by less than 10^{-6} units between three successive iterations), are selected as the final, optimum values.

Lastly, in Stage 3, the RF model is implemented to make predictions against the test dataset using the FFA-determined optimum values of the hyper-parameters.

Data Collection and Performance Evaluation of Machine Learning Models

Data Collection

Experimental datasets, consolidated from published studies (Chopra et al. 2014; Yeh 1998a, b), were used to train the ML models (described in the "Machine Learning Models" section) and to assess their prediction performance in previously untrained data domains.

The first dataset—subsequently referred to as Dataset 1—was first published by Yeh (1998a, b), and subsequently used by several researchers (Akande et al. 2014; Behnood et al. 2017; Chou et al. 2010, 2014; Chou and Pham 2015; Duan et al. 2013; Gupta et al. 2006; Kasperkiewicz et al. 1995; Nagwani and Deo 2014; Omran et al. 2016; Veloso de Melo and Banzhaf 2017; Yeh and Lien 2009; Young et al. 2019; Zarandi et al. 2008) for training, testing, and validation of statistical and ML models. Dataset 1 consists of 1,030 data-records, featuring 278 unique concrete mixture designs and their age-dependent compressive strengths. In the context of ML, in each data record, there are eight input variables [contents of cement ($\text{kg} \cdot \text{m}^{-3}$), blast furnace slag ($\text{kg} \cdot \text{m}^{-3}$), fly ash ($\text{kg} \cdot \text{m}^{-3}$), superplasticizer ($\text{kg} \cdot \text{m}^{-3}$), water ($\text{kg} \cdot \text{m}^{-3}$), fine aggregate ($\text{kg} \cdot \text{m}^{-3}$) and coarse aggregate ($\text{kg} \cdot \text{m}^{-3}$), and age (days)] and one output—compressive strength (MPa). Statistical parameters pertaining to Dataset 1 are summarized in Table 1.

The second dataset—subsequently referred to as Dataset 2—was first published by Chopra et al. (2014) and utilized in several later studies (Chopra et al. 2015, 2016, 2018). Dataset 2 consists of 76 data-records, featuring different concrete mixture designs and their compressive strengths at 28 days. In the context of ML, in each data record, there are five input variables [contents of cement ($\text{kg} \cdot \text{m}^{-3}$), fly ash ($\text{kg} \cdot \text{m}^{-3}$), water ($\text{kg} \cdot \text{m}^{-3}$), fine aggregate ($\text{kg} \cdot \text{m}^{-3}$) and coarse aggregate ($\text{kg} \cdot \text{m}^{-3}$)] and one output—compressive strength (MPa) at 28 days. Statistical parameters pertaining to Dataset 2 are summarized in Table 2.

Evaluation of Prediction Performance of Machine Learning Models

For training, and assessment of the prediction performance of ML models, the dataset (i.e., Dataset 1 or Dataset 2, as described in the Data Collection section) was randomly partitioned into two sets:

Table 1. Summary of statistical parameters pertaining to each of the 9 attributes (8 input and 1 output) of Dataset 1. The dataset consists of 1,030 unique data-records

Attribute	Unit	Minimum	Maximum	Mean	Standard deviation
Cement	$kg \cdot m^{-3}$	102.00	540.00	281.27	104.51
Blast furnace slag	$kg \cdot m^{-3}$	0.0000	359.40	73.896	86.279
Fly ash	$kg \cdot m^{-3}$	0.0000	200.10	54.188	63.997
Water	$kg \cdot m^{-3}$	121.80	247.00	181.57	21.354
Superplasticizer	$kg \cdot m^{-3}$	0.0000	32.200	6.2050	5.9740
Coarse aggregate	$kg \cdot m^{-3}$	801.00	1,145.0	972.92	77.754
Fine aggregate	$kg \cdot m^{-3}$	594.00	992.60	773.58	80.176
Age	Days	1.0000	365.00	45.662	63.170
Compressive strength	MPa	2.3300	82.600	35.818	16.706

Table 2. Summary of statistical parameters pertaining to each of the 6 attributes (5 input and 1 output) of Dataset 2. The dataset consists of 76 unique data-records

Attribute	Unit	Minimum	Maximum	Mean	Standard deviation
Cement	$kg \cdot m^{-3}$	350.00	475.00	433.88	34.810
Fly ash	$kg \cdot m^{-3}$	0.0000	71.250	24.030	32.641
Water	$kg \cdot m^{-3}$	178.50	229.50	202.81	12.821
Coarse aggregate	$kg \cdot m^{-3}$	798.00	1,253.8	1,050.9	134.52
Fine aggregate	$kg \cdot m^{-3}$	175.95	641.75	524.31	69.378
28-day compressive strength	MPa	31.660	54.490	44.374	5.2120

a training set and a testing set. Among the data-records, 75% of the parent dataset were used for training of the ML models (i.e., for fine-tuning and, ultimately, finalizing the optimum model parameters), and the remaining 25% were used for testing (i.e., for determination of cumulative error between predicted and actual values). Such a split of 75%–25% between the training and test sets—or a ratio close to that—has been used in various past studies (Chou et al. 2010, 2014; Young et al. 2019). Although the splitting was done randomly, special care was taken to guarantee that the training dataset was representative of the parent dataset. Toward this, it was ensured that the training dataset was composed of input attributes (i.e., concrete mixture design variables) with widespread values encompassing the entire range between the two extrema.

For quantitative measure of prediction performance of the ML models (against the test set), five different statistical parameters were used. The parameters, essentially, estimate the cumulative error in predictions—of compressive strength of concretes in the test dataset—with respect to the actual measurements. The statistical parameters are the Pearson correlation coefficient (R), coefficient of determination (R^2), mean absolute percentage error (MAPE), mean absolute error (MAE), and root-mean squared error (RMSE). The mathematical formulations to estimate these errors are shown in Eqs. (11)–(15); here, y' and y are predicted and actual values, and n is the total number of data-records in the test dataset

$$R = \frac{n\sum y.y' - (\sum y)(\sum y')}{\sqrt{n(\sum y^2) - (\sum y)^2}\sqrt{n(\sum y'^2) - (\sum y')^2}} \tag{11}$$

$$R^2 = \left[\frac{n\sum y \cdot y' - (\sum y)(\sum y')}{\sqrt{n(\sum y^2) - (\sum y)^2}\sqrt{n(\sum y'^2) - (\sum y')^2}}\right]^2 \tag{12}$$

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{i=n}\frac{|y - y'|}{y} \tag{13}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{i=n}|y - y'| \tag{14}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{i=n}|y - y'|^2} \tag{15}$$

$$CPI = \frac{1}{N}\sum_{j=1}^{j=N}\frac{P_j - P_{\min,j}}{P_{\max,j} - P_{\min,j}} \tag{16}$$

To obtain a comprehensive measure of prediction performance of the ML models—and to compare them—the five statistical parameters described in Eqs. (11)–(15) were unified into a composite performance index [CPI, see Eq. (16)] (Chandwani et al. 2015; Chou et al. 2014). In Eq. (16), N is the total number of performance measures (= 5 because five statistical parameters were used in this study), P_j is the value of the jth statistical parameter, and $P_{j,\min}$ and $P_{j,\max}$ are the minimum (i.e., worst) and maximum (i.e., best) values of the jth statistical parameter across the five values generated by the same number of ML models. Based on the formulation shown in Eq. (16), the values of CPI would range from 0 to 1, wherein 0 (or the lowest value) would represent the best ML model and 1 (or the maximum value) would represent the worst ML model in terms of overall prediction performance. In this study, the different ML models were ranked—from worst to best in terms of prediction performance—based on their CPI values.

Results and Discussion

Highly Nonlinear and Periodic Trigonometric Functions

In conventional regression-based machine learning (ML), the quality of an ML model is measured by its capability to learn from a training set and apply the knowledge to forecast in previously unseen data domains from the same distribution. Simply put, the prediction performance of an ML model boils down to its ability to identify trends in the dataset and subsequently use such trends for interpolation. When trained properly (e.g., by training with an adequately large dataset and by avoiding overfitting), nonlinear ML models are often able to perform interpolations with sufficient

accuracy. However, it has been reported that the interpolation accuracy of many ML models (e.g., ANN and SVM) becomes unreliable in data domains that feature complex, highly nonlinear, and periodic functional relationships between one or more input variables and the output (Cunningham et al. 2000; Martius and Lampert 2016; Yao 1999; Zhang et al. 1998). In the context of concrete, the ability to interpolate in such highly nonlinear domains could be the difference between reliable and unreliable predictions; this is because the relationships between mixture design variables and properties of concrete are also expected to be highly nonlinear and non-monotonous.

To test the ability of ML models—described in the "Machine Learning Models" section—to interpolate in highly nonlinear and periodic data domains, datasets generated from trigonometric functions shown in Eqs. (17) and (18) were used. Similar functions were originally suggested by Martius and Lampert (2016) to test the ability of various ML models to interpolate (and extrapolate) within periodic data domains. In Eq. (17), x is an input vector consisting of x_1, x_2, x_3, and x_4 variables, and $y_1 = F_1(x)$ is the output. For this function, $x_1 = x_2 = x_3 = 2x_4$. In Eq. (18), x is an input vector consisting of the same x_1, x_2, x_3, and x_4 variables, and $y_2 = F_2(x)$ is the output. Here, $x_1 = x_2 = x_3 = -5x_4$. Two separate datasets were generated by varying x_1 (and, on account of the aforementioned equality, x_2, x_3, and x_4 as well) between -4.0 and 4.0, and calculating $F_1(x)$ and $F_2(x)$ as functions of all four variables. The increment in x_1 was set at 0.01; as such, each of the two datasets consisted of 800 data-records with four input variables and a single output. Next, each dataset was split randomly into a training set (75%, or 600 data-records) and a test set (25%, or 200 data-records), using the procedure described in the Data Collection section. All five ML models implemented in this study (i.e., MLP-ANN, SVM, M5P, RF, and RF-FFA) were then trained using the training dataset; subsequently, their prediction performances were assessed using the corresponding test dataset

$$y_1 = F_1(x) = \left[\frac{1}{3}\sin(\pi x_1)\right] + \left[x_2 \cos\left(2\pi x_1 + \frac{\pi}{4}\right)\right] + [x_3] - [x_4^2] \tag{17}$$

$$y_2 = F_2(x) = \frac{1}{3}[(1 + x_2)(\sin(\pi x_1))] + [x_2 x_3 x_4] \tag{18}$$

Figs. 1 and 2 show predictions made by the five ML models plotted against the actual values, calculated using Eq. (17) [i.e., for $F_1(x)$] and Eq. (18) [i.e., for $F_2(x)$], respectively. Tables 3 and 4 summarize the statistical parameters (i.e., cumulative errors) pertaining to predictions made by the ML models, and the composite performance index [CPI, Eq. (16)] calculated using the five statistical parameters.

As can be seen in Figs. 1 and 2, the MLP-ANN and SVM models are unable to capture the periodic nature of the dataset. As stated previously in the Introduction, this is because both models employ local search or optimization algorithms, which are faced with an inherent drawback of getting trapped in local minima—especially when the functional relationship between the input variables and output is composed of multiple local minima [e.g., datasets generated using Eqs. (17) and (18)]—rather than converging to the global minima. The poor prediction performance of MLP-ANN and SVM models is also reflected in the values of statistical parameters listed in Tables 1 and 2 [e.g., RMSE of 2.8552 and 1.5245 for MLP-ANN and SVM models, respectively, when used for prediction of $F_1(x)$ and RMSE of 1.2602 and 0.9570 for MLP-ANN and SVM models, respectively, when used for prediction of $F_2(x)$]. It is indeed possible to improve prediction performance of the models by incorporating algorithms based on genetic programing (Chopra et al. 2016; Veloso de Melo and Banzhaf 2017), or by using ensemble techniques [e.g., bagging, voting, or stacking approaches (Chou et al. 2014; Polikar 2006)]. However, as stated previously in the Introduction, such techniques could result in slower convergence and/or overfitting.

The M5P model—which attempts to split data logically and then apply linear regression models in each data-split—performed better at predictions compared to MLP-ANN and SVM models [i.e., CPI of 0.2212 of M5P vis-à-vis 1.0000 and 0.6334 of MLP-ANN and SVM models, respectively, when used for prediction of $F_1(x)$]. Although the M5P model captures the periodic nature of the dataset because of the application of linear models and limited size of the decision tree, the actual intensities of the local minima and maxima are not well captured (Fig. 1). As such, the RMSE of the model's predictions are still high, that is, 0.7891 for $F_1(x)$ and 0.1648 for $F_2(x)$.

The RF model outperformed all the aforementioned models (i.e., MLP-ANN, SVM, and M5P) in terms of prediction accuracy. This is expected because, in the RF model, a large number of trees are grown [i.e., 450 trees, with 5 leaves per tree, as described in the Random Forests (RF) section] without pruning or smoothening (as opposed to the M5P model, wherein the number of trees is restricted and pruning and smoothening are required). On account of having a large number of the trees, splits in data are more logical, and, therefore, errors resulting from generalization are minimized and overfitting of the training data is mitigated (Biau et al. 2008; Breiman 1996). Furthermore, because of the two-stage randomization employed in the RF model—as described earlier and in Biau et al. (2008)—correlation among unpruned trees is minimized (diversity among trees is high), the bias is kept low, and variance is significantly reduced. The prediction of the RF model further improved when it was combined with the firefly algorithm (FFA). As shown in Figs. 1 and 2, and Tables 3 and 4, the hybrid RF-FFA model was not only able to capture the periodic nature of the dataset but also able to reliably interpolate the local minima and maxima (and the intermediate values) across the entire -4.0-to-4.0 range of the input variable x_1. This enhancement in prediction performance of the hybrid model, with respect to the standalone RF model, can be attributed to the FFA, which is able to optimize the two hyperparameters (i.e., number of trees and number of leaves per tree) of the RF model based on the nature and volume of the dataset. Based on overall prediction performance—as estimated using the CPI, which takes in account all of the statistical parameters [see Eq. (16)]—the ranking of the ML models is as follows: RF-FFA > RF > M5P > SVM > MLP-ANN.

Compressive Strength of Concrete: Dataset 1

Based on results shown in the previous subsection, it was established that the hybrid RF-FFA model outperformed the standalone MLP-ANN, SVM, M5P, and RF models in terms of prediction accuracy. The standalone RF model came as a close second. Notwithstanding, these results pertain to user-created trigonometric functions, wherein the relationship between input variables and output could be far more complex than real-world datasets. Therefore, to get a better understanding of prediction performance of the ML models, a real-world dataset of concrete—that is, Dataset 1, described in the Data Collection section—was used. Each data-record in the dataset consists of eight input variables—representing contents of cementitious materials and admixture, and age—and one output (i.e., compressive strength). Predictions of compressive strength of concretes from the test set of Dataset 1, as produced by the ML models, are shown in Fig. 3; statistical errors pertaining to predictions are summarized in Table 5.

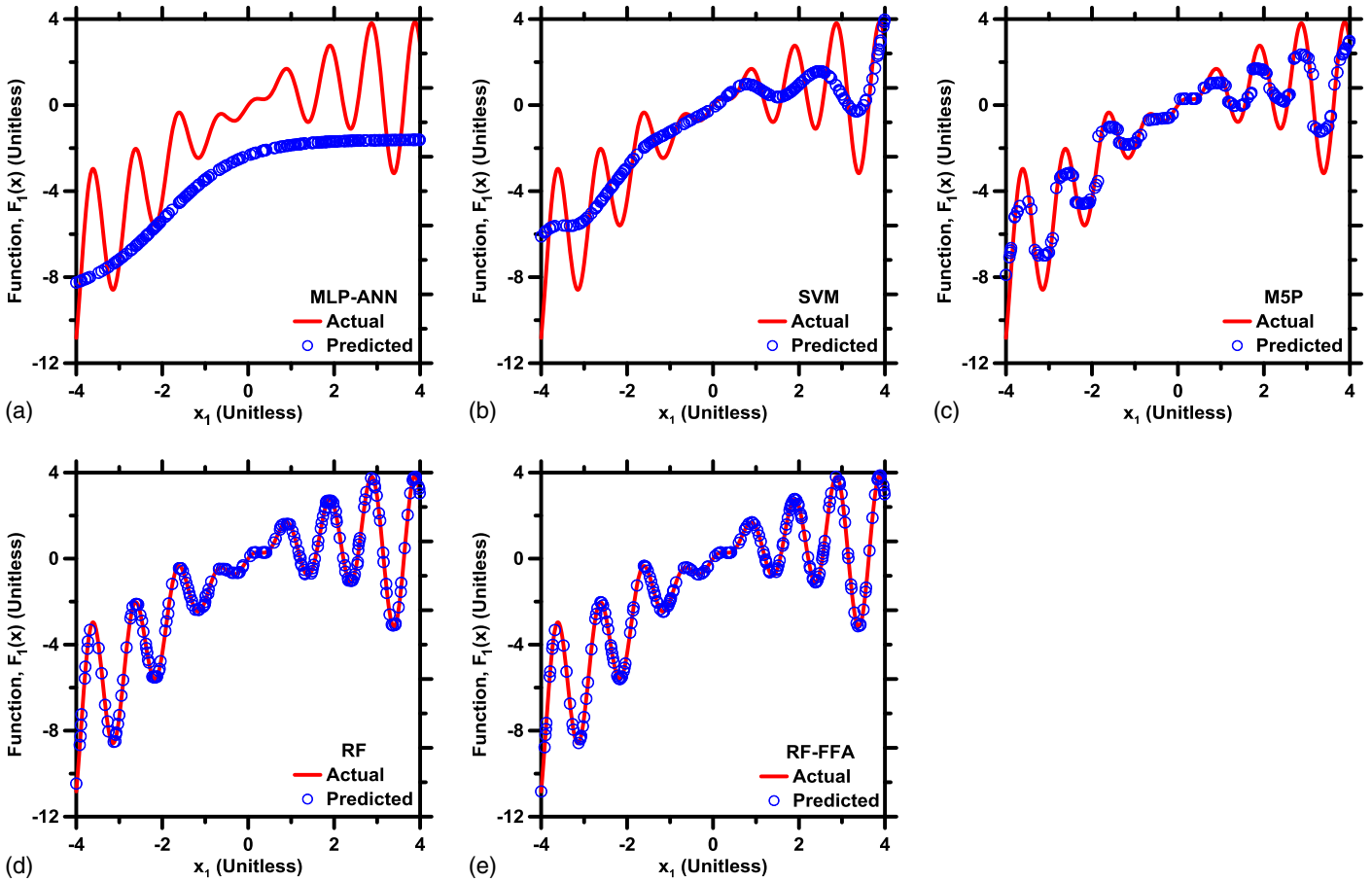

Fig. 1. Predictions made by ML models: (a) MLP-ANN; (b) SVM; (c) M5P; (d) RF; and (e) RF-FFA compared against actual values of the trigonometric function, $F_1(x)$ [Eq. (17)]. Both the actual values and predictions are plotted against x_1, an input variable ranging from −4 to 4. Here, $x_1 = x_2 = x_3 = 2x_4$.

As shown in Fig. 3 and Table 5, all ML models presented in this study were able to predict the age-dependent compressive strength of concrete with reasonable accuracy. This is evidenced by the relatively low and high values of RMSE (ranging between 4.0098 and 6.3300 MPa) and R^2 (ranging between 0.8664 and 0.9448), respectively, of predictions made by the ML models. It must be pointed out that the differences in statistical parameters among the different ML models are not as significant as in the case of Highly Nonlinear and Periodic Trigonometric Functions, wherein datasets developed from periodic trigonometric functions [Eqs. (17) and (18)] were used. This is hypothesized to be on account of the relatively simpler input–output relationship in the concrete dataset compared to the ones dictated by trigonometric functions. Several other studies—that have used the same dataset (i.e., published originally in Yeh (1998a, b) and applied different ML models for predictions—have reported RMSE and/or R^2 values similar to those shown in Table 5. Selected examples of prediction performance of various ML models (on Dataset 1) reported in literature are provided in the following; a comprehensive review, with additional examples, can be found in another study (Omran et al. 2016).

In the study conducted by Young et al. (2019), linear regression, ANN, RF, boosted tree, and SVM models were used, and the RMSE of predictions made by the models ranged between 4.4 and 5.0 MPa. In another study conducted by Veloso de Melo and Banzhaf (2017), kaizen programming with simulated annealing was used, and the RMSE was ≈6.8 MPa. In the study by Chou et al. (2014), several standalone and ensemble ML models were implemented to forecast compressive strengths of concretes listed in Dataset 1. Among the standalone models, the RMSE was between 5.59 and 10.11 MPa; and, among the ensemble models, the RMSE was between 5.51 and 38.41 MPa. In the study by Behnood et al. (2017), the M5P model was used, and the RMSE of predictions was reported as 6.178 MPa—a value close to the one obtained by the M5P model used in this study (Table 5). Lastly, in a study conducted by Chou and Pham (2015), the SVM algorithm was combined with FFA, and applied to predict compressive strength of concretes from Dataset 1. Based on the reported results, the hybrid [SVM + FFA] model outperformed other standalone (e.g., SVM) and ensemble models (e.g., ANN + SVM), and yielded predictions with RMSE of 5.631 MPa.

Going back to Table 5, it is clear from all five statistical parameters that the RF and the hybrid RF-FFA models have superior prediction performance compared to MLP, SVM, and M5P models. Based on the values of CPI—the unified measure of prediction performance—the ML models can be ranked as RF-FFA > RF > SVM > M5P > MLP-ANN. This order is similar to the one that emerged in the section on Highly Nonlinear and Periodic Trigonometric Functions, wherein periodic trigonometric functions were used to generate datasets. Here again, the superiority of the RF model—compared to MLP-ANN, SVM, and M5P models—is attributed to the large number of unpruned trees that are grown [i.e., 450 trees, with 5 leaves per tree, as described in the Random Forests (RF) section]. Such depth in the model's structure allows

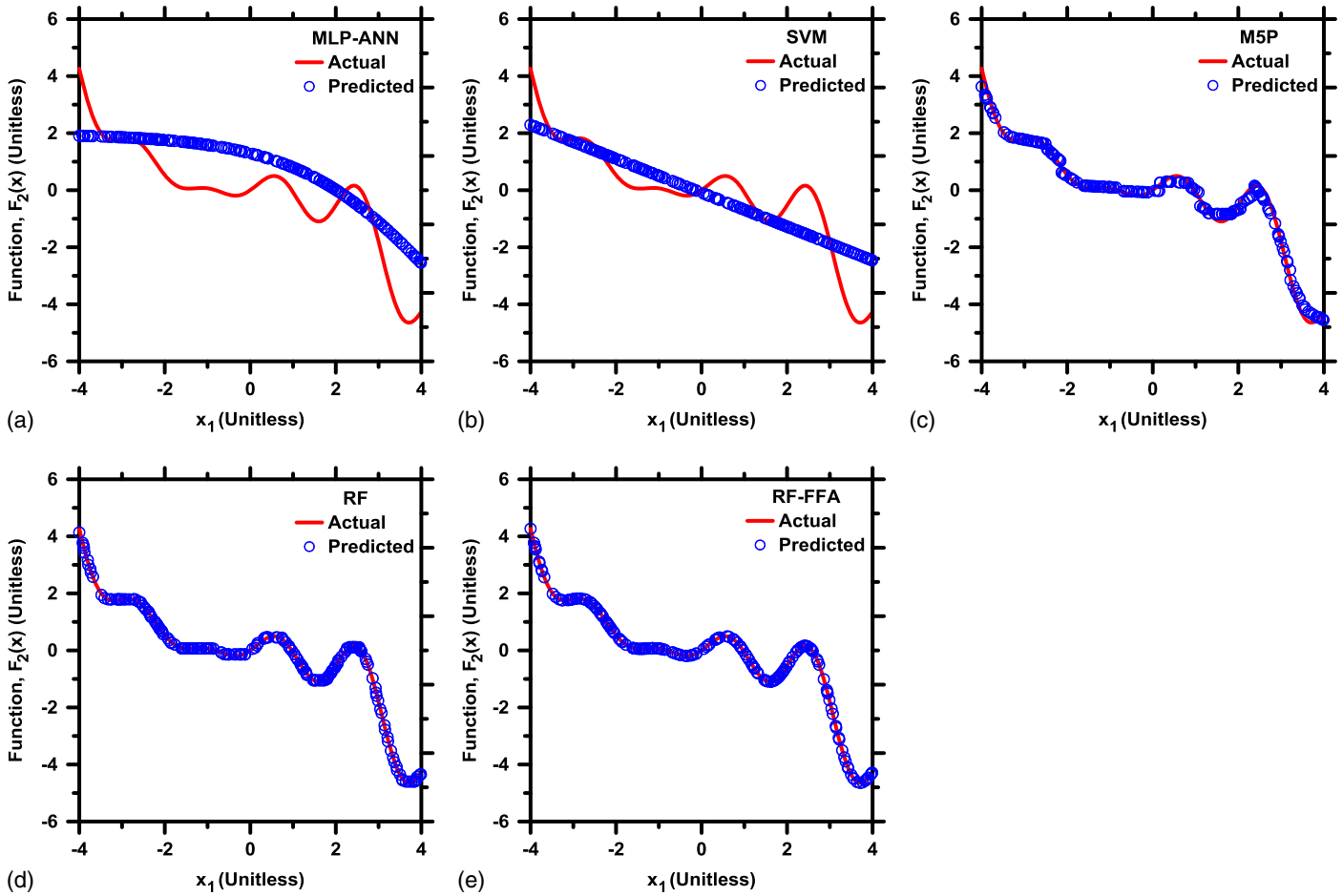

Fig. 2. Predictions made by ML models: (a) MLP-ANN; (b) SVM; (c) M5P; (d) RF; and (e) RF-FFA compared against actual values of the trigonometric function, $F_2(x)$ (Eq. 18). Both the actual values and predictions are plotted against x_1, an input variable ranging from −4 to 4. Here, $x_1 = x_2 = x_3 = -5x_4$.

Table 3. Prediction performance of ML models, measured on the basis of the test set developed using Eq. (17). Five statistical parameters (i.e., R, R^2, MAE, MAPE, and RMSE) and the composite performance index (CPI) are shown

ML model	R	R^2	MAE	MAPE	RMSE	CPI
MLP-ANN	0.8425	0.7098	2.4633	102.87	2.8552	1.0000
SVM	0.8707	0.7581	1.2353	51.588	1.5245	0.6334
M5P	0.9718	0.9443	0.6113	25.530	0.7891	0.2212
RF	0.9995	0.9990	0.0753	3.1436	0.0977	0.0106
RF-FFA	0.9999	0.9998	0.0354	1.4790	0.0563	0.0000

Table 4. Prediction performance of ML models, measured on the basis of the test set developed using Eq. (18). Five statistical parameters (i.e., R, R^2, MAE, MAPE, and RMSE) and the composite performance index (CPI) are shown

ML model	R	R^2	MAE	MAPE	RMSE	CPI
MLP-ANN	0.8598	0.7392	1.0794	91.133	1.2602	0.9633
SVM	0.8450	0.7140	0.7201	60.797	0.9570	0.8175
M5P	0.9963	0.9926	0.1272	10.737	0.1648	0.0797
RF	0.9998	0.9996	0.0239	2.0158	0.0312	0.0104
RF-FFA	1.0000	1.0000	0.0063	0.5359	0.0107	0.0000

more logical splits in the data, which, in turn, results in development of logical input–output correlations and mitigates overfitting and generalization errors. Even further enhancement in prediction performance was achieved when the RF model was combined with FFA. This enhancement is attributed to the FFA's ability to optimize the number of trees and leaves per tree of the RF model—based on intrinsic characteristics of the dataset, and all without any user intervention.

On a closing note for this subsection, it is pointed out that the RMSE of predictions produced by the RF-FFA model are lower (i.e., RMSE = 4.0098 MPa) than the values reported in all other studies found in the authors' literature review (Akande et al. 2014; Behnood et al. 2017; Chou et al. 2010, 2014; Chou and Pham 2015; Duan et al. 2013; Gupta et al. 2006; Kasperkiewicz et al. 1995; Nagwani and Deo 2014; Omran et al. 2016; Veloso de Melo and Banzhaf 2017; Yeh 1998a; Yeh and Lien 2009; Young et al. 2019; Zarandi et al. 2008). Admittedly, the RMSE value alone cannot be used to assert that the RF-FFA model is superior compared to others. This is mainly because such comparison of prediction performance of ML models, developed and implemented by different users (in spite of utilization of the same database), is complex on account of differences in (1) description of cumulative statistical error (e.g., in some papers, R^2—rather than RMSE—was used to assess accuracy); (2) splitting of parent dataset into training and test sets (e.g., in some papers, the parent dataset was split as per 80% and 20% or 66.66% and 33.33% between the training and test

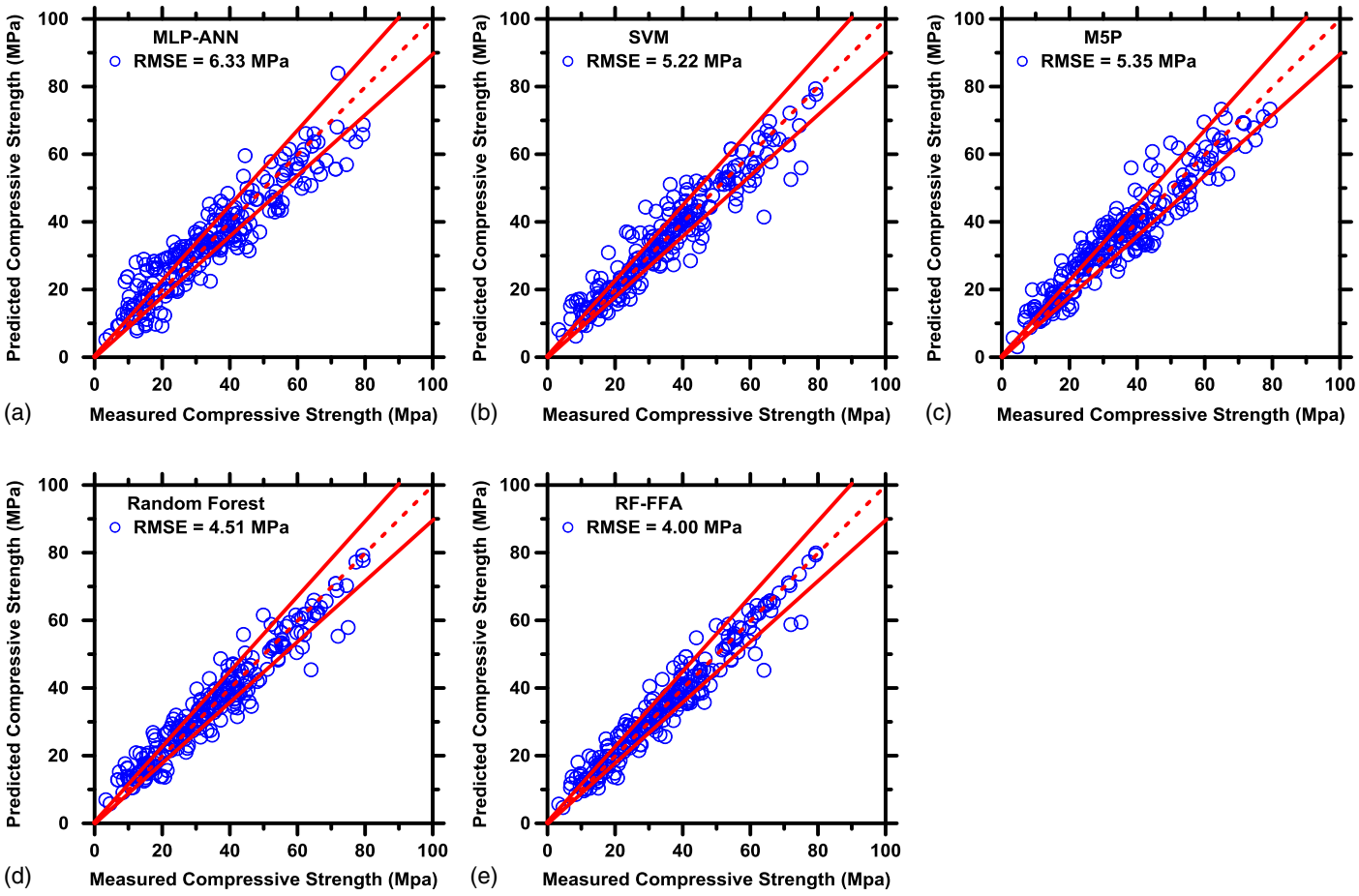

Fig. 3. Predictions made by ML models: (a) MLP-ANN; (b) SVM; (c) M5P; (d) RF; and (e) RF-FFA compared against actual compressive strength of concretes (drawn from Dataset 1). The dashed line represents the line of ideality and the solid lines represent a ±10% bound.

Table 5. Prediction performance of ML models, measured on the basis of the test set of Dataset 1. Five statistical parameters (i.e., R, R^2, MAE, MAPE, and RMSE) and the composite performance index (CPI) are shown

ML model	R	R^2	MAE (MPa)	MAPE (%)	RMSE (MPa)	CPI
MLP-ANN	0.9308	0.8664	5.0421	36.143	6.3300	1.0000
SVM	0.9525	0.9073	3.5756	25.624	5.2234	0.4385
M5P	0.9502	0.9029	4.2369	30.367	5.3518	0.5884
RF	0.9654	0.9320	3.2674	23.443	4.5103	0.1999
RF-FFA	0.9720	0.9448	2.7301	19.571	4.0098	0.0000

sets—as opposed to 75% and 25%, as used in this study); (3) total number of data-records used for training and testing of the ML models (e.g., in some papers, all 1,030 data-records of Dataset 1 were used, whereas in some only a fraction of them were used); and (4) methodology used for optimization of model parameters (e.g., some papers used the cross-validation method to optimize model parameters using the training dataset, whereas, in this study, the FFA was used to optimize hyper-parameters of the RF model). Notwithstanding, the low RMSE (i.e., 4.0098 MPa)—combined with low values of MAE and MAPE and high values of R and R^2 (Table 5)—produced by the hybrid RF-FFA model certainly suggest that the model is a promising tool for prompt, reliable, and accurate predictions of age-dependent compressive strength of concretes using their mixture design variables as inputs. It is worth mentioning that incorporation of FFA within the RF model does not increase computational complexity and, therefore, does not increase computational time compared to the standalone RF model in a significant manner. In fact, compared to parameter optimization conducted using the traditional multifold CV method, the FFA is more efficient because it is more convenient (e.g., it eliminates the need for trail-and-error or CV method based optimization of parameters), requires less time for computations, and produces more accurate predictions. Lastly, it should be pointed out that Dataset 1 provides a singular, albeit important, corroboration of superiority of RF-FFA over the standalone RF model as well as other ML models. It is conceivable that utilization of a higher-quality training database—for example, one with a large number of data-records, or one wherein significant physical (e.g., particle size distribution) and chemical (e.g., composition) characteristics of concrete components (e.g., cement and fly ash) and experimental process parameters (e.g., temperature and relative humidity of curing) are also described—will lead to even better prediction performance of the RF-FFA model.

Compressive Strength of Concrete: Dataset 2

In the previous subsection, it was shown that the proposed hybrid RF-FFA model produced predictions of concrete compressive strength with RMSE of 4.0098 MPa—suggesting a reasonably high degree of accuracy, especially in comparison to predictions produced by ML models reported in the literature as well as other ML models presented in this study (i.e., MLP-ANN, SVM, M5P, and RF). The dataset used in the section Compressive Strength

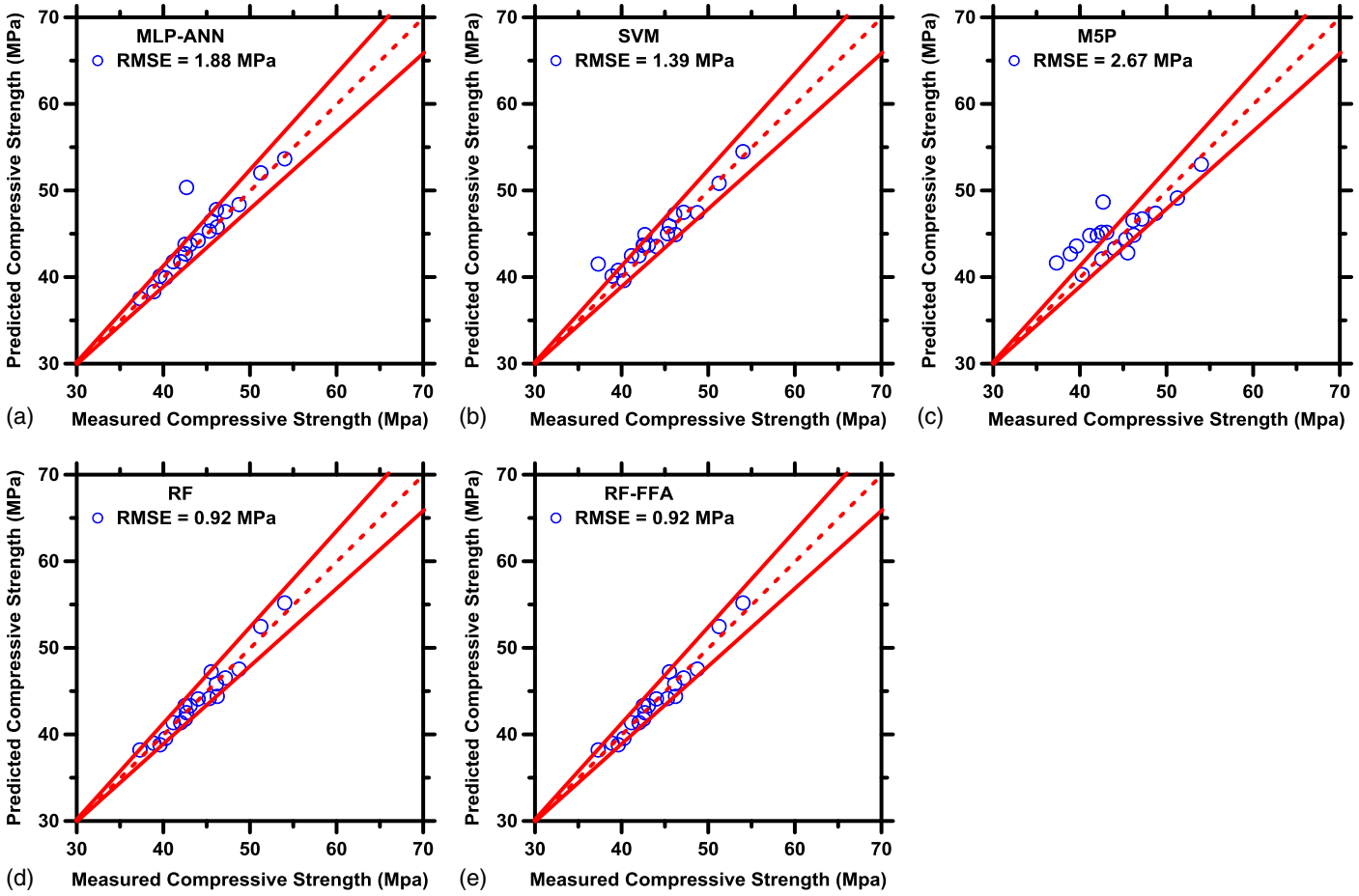

Fig. 4. Predictions made by ML models: (a) MLP-ANN; (b) SVM; (c) M5P; (d) RF; and (e) RF-FFA compared against actual compressive strength of concretes (drawn from Dataset 2). The dashed line represents the line of ideality and the solid lines represent a ±10% bound.

of Concrete: Dataset 1 is composed of 1,030 data-records, providing the RF-FFA model an adequate number of data-records (i.e., $0.75 \times 1{,}030 = 772$) for developing logical input–output correlations and, thus, making accurate predictions. It is, however, important to examine if the RF-FFA model is able to retain its superior prediction performance when a much smaller dataset is used for training (and testing). Such examination is deemed necessary because generating large datasets of concrete performance is very time-consuming; thus, it is important to evaluate whether or not the proposed RF-FFA model is applicable to smaller concrete datasets that are more abundant and easily found in literature. Toward this, the prediction performance of the RF-FFA model was evaluated using Dataset 2 (described in the "Data Collection" section) and benchmarked against the performance of other ML models. Readers are reminded that Dataset 2 consists of 76 data-records, featuring different concrete mixture designs and their compressive strengths at 28 days. The mixture design variables were used as inputs; the 28-day compressive strength was used as an output. Predictions of compressive strength of concretes from the test set of Dataset 2, as produced by the ML models, are shown in Fig. 4; statistical errors pertaining to the predictions are summarized in Table 6.

Akin to the results shown in the section "Compressive Strength of Concrete: Dataset 1," all five ML models were able to predict the age-dependent compressive strength of concretes from Dataset 2 with reasonable accuracy. The RMSE of predictions made by the ML models range from 0.9213 to 2.6754 MPa, attesting to the high degree of accuracy of predictions. These RMSE values are lower than those reported in some prior studies (Chopra et al. 2014, 2016), albeit similar to those reported in a recent study (Chopra et al. 2018)—wherein ANN, RF, and decision tree models were used for making predictions. Upon comparing the overall prediction performances, based on the values of CPI (Table 6), the following order emerges: RF-FFA = FA > SVM > MLP-ANN > M5P. This order, once again, suggests that prediction performance of the RF-FFA model is superior compared to other ML models presented in this study. Although the aforementioned order is broadly similar to the one obtained from predictions of strength of concretes from Dataset 1, there are a few small differences. Firstly, in Dataset 2, the prediction performance of the M5P model is the worst; this was not the case when Dataset 1 was used. It is expected that the deterioration in performance of the M5P model is

Table 6. Prediction performance of ML models, measured on the basis of the test set of Dataset 2. Five statistical parameters (i.e., R, R^2, MAE, MAPE, and RMSE) and the composite performance index (CPI) are shown

ML model	R	R^2	MAE (MPa)	MAPE (%)	RMSE (MPa)	CPI
MLP-ANN	0.9201	0.8464	0.9163	27.352	1.8783	0.2857
SVM	0.9565	0.9149	1.0635	31.744	1.3841	0.1876
M5P	0.8003	0.6400	2.1480	64.127	2.6754	1.0000
RF	0.9778	0.9561	0.7718	23.041	0.92313	0.0000
RF-FFA	0.9778	0.9561	0.7718	23.041	0.92313	0.0000

due to the much smaller volume of Dataset 2 (i.e., 76 data-records as opposed the to 1,030 of Dataset 1)—thus resulting in inferior quality of splits in the training dataset, and, consequently, poor input–output linear correlations within each split. The poor prediction performance of the M5P model indicates—as was also suggested in a prior study (Chopra et al. 2018)—that, when the dataset volume is small, decision tree models with limited number of trees (1) cannot ensure homogeneity in data clustered in each node, (2) cannot maintain diversity among the different nodes, and, therefore, (3) are unable to make predictions in an accurate manner. Secondly, it is also interesting to note in Table 6 that both the RF and RF-FFA models have similar prediction performances. The implication of this equivalency is that when the dataset is small, the application of FFA—for optimization of the two hyper-parameters (i.e., number of trees and number of leaves per tree in the forest) of the RF model—is redundant and does not necessarily elicit any substantial improvement in prediction performance. However, when the dataset is large—for example, Dataset 1—the application of FFA is beneficial in that it produces substantial improvement in prediction performance of the RF model by optimizing its two hyper-parameters in relation to the nature and volume of the dataset (Table 5).

Conclusions

This study developed and presented a novel hybrid machine learning (ML) model (RF-FFA) for prediction of compressive strength of concrete, in relation to its mixture design and age, by combining the random forests (RF) model with the firefly algorithm (FFA). The firefly algorithm—a metaheuristic optimization technique—was used to optimize the two hyper-parameters of the RF model (i.e., the number of trees and the number of leaves per tree in the forest) in relation to the volume and nature of the dataset, and without any user intervention.

The RF-FFA model was trained to develop correlations between input variables and output of two different categories of datasets; such correlations were subsequently leveraged by the model to make predictions. The first category included two separate datasets featuring highly nonlinear and periodic relationships between input variables and output as given by trigonometric functions. The second category included two real-world datasets, composed of mixture design variables and age of concretes as inputs and their compressive strengths as outputs. The performance of the hybrid RF-FFA model was benchmarked against commonly used standalone ML models—support vector machine (SVM), multilayer perceptron artificial neural network (MLP-ANN), M5Prime model tree algorithm (M5P), and RF. The metrics used for evaluation of prediction accuracy of the ML models included five different statistical measures (i.e., R, R^2, MAE, RMSE, and MAPE) as well as a composite performance index (CPI).

The prediction performances of MLP-ANN and SVM models were reasonable for concrete datasets; however, their inability to identify and converge to global minima rendered their prediction performances poor when datasets generated from trigonometric functions were used. The prediction performance of the M5P model, in general, was commensurable to, or slightly superior compared to, those of MLP-ANN and SVM models. However, on account of limited size (or depth) of the decision tree and utilization of multivariate linear regression models, prediction performance of the M5P model was consistently inferior compared to those of RF and RF-FFA models. The superiority of the RF model was attributed to the large number of unpruned trees, which, in turn, results in development of logical input–output correlations and mitigates overfitting and generalization errors. Even further enhancement in prediction performance was achieved when the RF model was combined with FFA (i.e., the RF-FFA model). This enhancement in prediction performance was attributed to the FFA's ability to optimize the number of trees and leaves per tree of the RF model based on the volume and nature of the dataset.

The high degree of prediction accuracy (i.e., RMSE of $\approx$4.0 and $\approx$0.92 MPa for the large and small datasets, respectively) produced by the hybrid RF-FFA model suggests that the model is a promising tool for prompt and reliable prediction of composition-dependent properties of concrete, provided that the training is accomplished using adequate number of data-records. It is expected that utilization of a higher-quality database—wherein the dataset volume is large, and/or wherein influential physical (e.g., particle size distribution) and chemical (e.g., composition) attributes of concrete components (e.g., cement and fly ash) and curing conditions (e.g., temperature and relative humidity of curing) are also described—will further bolster the prediction performance of the RF-FFA model.

Data Availability Statement

The datasets (i.e., Dataset 1 and Dataset 2, referenced in the preceding sections), machine learning models, and code generated or used during the study are available from the corresponding author (A. Kumar; kumarad@mst.edu) by request.

Acknowledgments

Funding for this research was provided by the National Science Foundation [NSF, CMMI: 1661609]. Computational tasks were conducted in the Materials Research Center and Department of Materials Science and Engineering at Missouri S&T. The authors gratefully acknowledge the financial support that has made these laboratories and their operations possible.

References

Akande, K. O., T. O. Owolabi, S. Twaha, and S. O. Olatunji. 2014. "Performance comparison of SVM and ANN in predicting compressive strength of concrete." *IOSR J. Comput. Eng.* 16 (5): 88–94. https://doi.org/10.9790/0661-16518894.

Behnood, A., V. Behnood, M. M. Gharehveran, and K. E. Alyamac. 2017. "Prediction of the compressive strength of normal and high-performance concretes using M5P model tree algorithm." *Constr. Build. Mater.* 142 (Jul): 199–207. https://doi.org/10.1016/j.conbuildmat.2017.03.061.

Biau, G., L. Devroye, and G. Lugosi. 2008. "Consistency of random forests and other averaging classifiers." *J. Mach. Learn. Res.* 9 (Sep): 2015–2033.

Breiman, L. 1996. "Bagging predictors." *Mach. Learn.* 24 (2): 123–140.

Breiman, L. 2001. "Random forests." *Mach. Learn.* 45 (1): 5–32. https://doi.org/10.1023/A:1010933404324.

Carrasquilla, J., and R. G. Melko. 2017. "Machine learning phases of matter." *Nat. Phys.* 13 (5): 431–434. https://doi.org/10.1038/nphys4035.

Chandwani, V., V. Agrawal, and R. Nagar. 2015. "Modeling slump of ready mix concrete using genetic algorithms assisted training of artificial neural networks." *Expert Syst. Appl.* 42 (2): 885–893. https://doi.org/10.1016/j.eswa.2014.08.048.

Chen, X., and H. Ishwaran. 2012. "Random forests for genomic data analysis." *Genomics* 99 (6): 323–329. https://doi.org/10.1016/j.ygeno.2012.04.003.

Chopra, P., R. K. Sharma, and M. Kumar. 2014. "Predicting compressive strength of concrete for varying workability using regression models." *Int. J. Eng. Appl. Sci.* 6 (4): 10–22.

Chopra, P., R. K. Sharma, and M. Kumar. 2015. "Artificial neural networks for the prediction of compressive strength of concrete." *Int. J. Appl. Sci. Eng.* 13 (3): 187–204.

Chopra, P., R. K. Sharma, and M. Kumar. 2016. "Prediction of compressive strength of concrete using artificial neural network and genetic programming." *Adv. Mater. Sci. Eng.* 2016 (2): 1–10. https://doi.org/10.1155/2016/7648467.

Chopra, P., R. K. Sharma, M. Kumar, and T. Chopra. 2018. "Comparison of machine learning techniques for the prediction of compressive strength of concrete." *Adv. Civ. Eng.* 2018 (3): 1–9. https://doi.org/10.1155/2018/5481705.

Chou, J.-S., C.-K. Chiu, M. Farfoura, and I. Al-Taharwa. 2010. "Optimizing the prediction accuracy of concrete compressive strength based on a comparison of data-mining techniques." *J. Comput. Civ. Eng.* 25 (3): 242–253. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000088.

Chou, J.-S., and A.-D. Pham. 2015. "Smart artificial firefly colony algorithm-based support vector regression for enhanced forecasting in civil engineering." *Comput.-Aided Civ. Infrastruct. Eng.* 30 (9): 715–732. https://doi.org/10.1111/mice.12121.

Chou, J.-S., C.-F. Tsai, A.-D. Pham, and Y.-H. Lu. 2014. "Machine learning in concrete strength simulations: Multi-nation data analytics." *Constr. Build. Mater.* 73 (Dec): 771–780. https://doi.org/10.1016/j.conbuildmat.2014.09.054.

Clarke, S. M., J. H. Griebsch, and T. W. Simpson. 2005. "Analysis of support vector regression for approximation of complex engineering analyses." *J. Mech. Des.* 127 (6): 1077–1087. https://doi.org/10.1115/1.1897403.

Cunningham, P., J. Carney, and S. Jacob. 2000. "Stability problems with artificial neural networks and the ensemble solution." *Artif. Intell. Med.* 20 (3): 217–225. https://doi.org/10.1016/S0933-3657(00)00065-8.

Deepa, C., K. SathiyaKumari, and V. Pream Sudha. 2010. "Prediction of the compressive strength of high performance concrete mix using tree based modeling." *Int. J. Comput. Appl.* 6 (5): 18–24.

Dietterich, T. G. 2000. "Ensemble methods in machine learning." In *Proc., Int. Workshop on Multiple Classifier Systems*, 1–15. New York: Springer.

Duan, Z.-H., S.-C. Kou, and C.-S. Poon. 2013. "Prediction of compressive strength of recycled aggregate concrete using artificial neural networks." *Constr. Build. Mater.* 40 (Mar): 1200–1206. https://doi.org/10.1016/j.conbuildmat.2012.04.063.

Fang, S. F., M. P. Wang, W. H. Qi, and F. Zheng. 2008. "Hybrid genetic algorithms and support vector regression in forecasting atmospheric corrosion of metallic materials." *Comput. Mater. Sci.* 44 (2): 647–655. https://doi.org/10.1016/j.commatsci.2008.05.010.

Frank, E., M. Hall, L. Trigg, G. Holmes, and I. H. Witten. 2004. "Data mining in bioinformatics using Weka." *Bioinformatics* 20 (15): 2479–2481. https://doi.org/10.1093/bioinformatics/bth261.

Gardner, M. W., and S. R. Dorling. 1998. "Artificial neural networks (the multilayer perceptron): A review of applications in the atmospheric sciences." *Atmos. Environ.* 32 (14): 2627–2636. https://doi.org/10.1016/S1352-2310(97)00447-0.

Garg, P., and J. Verma. 2006. "In silico prediction of blood brain barrier permeability: An artificial neural network model." *J. Chem. Inf. Model.* 46 (1): 289–297. https://doi.org/10.1021/ci050303i.

Goh, A. T. C. 1995. "Back-propagation neural networks for modeling complex systems." *Artif. Intell. Eng.* 9 (3): 143–151. https://doi.org/10.1016/0954-1810(94)00011-S.

Gupta, R., M. A. Kewalramani, and A. Goel. 2006. "Prediction of concrete strength using neural-expert system." *J. Mater. Civ. Eng.* 18 (3): 462–466. https://doi.org/10.1061/(ASCE)0899-1561(2006)18:3(462).

Hartigan, J. A., and M. A. Wong. 1979. "Algorithm AS 136: A K-means clustering algorithm." *J. R. Stat. Soc. Ser. C (Appl. Stat.)* 28 (1): 100–108.

Hearst, M. A., S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. 1998. "Support vector machines." *IEEE Intell. Syst. Appl.* 13 (4): 18–28. https://doi.org/10.1109/5254.708428.

Hegazy, T., P. Fazio, and O. Moselhi. 1994. "Developing practical neural network applications using back-propagation." *Comput.-Aided Civ. Infrastruct. Eng.* 9 (2): 145–159. https://doi.org/10.1111/j.1467-8667.1994.tb00369.x.

Holmes, G., A. Donkin, and I. H. Witten. 1994. "Weka: A machine learning workbench." In *Proc., 2nd Australian and New Zealand Conf. on Intelligent Information Systems, 1994*, 357–361. New York: IEEE.

Ibrahim, I. A., and T. Khatib. 2017. "A novel hybrid model for hourly global solar radiation prediction using random forests technique and firefly algorithm." *Energy Convers. Manage.* 138 (Apr): 413–425. https://doi.org/10.1016/j.enconman.2017.02.006.

Jain, A., S. P. Ong, G. Hautier, W. Chen, W. D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, and G. Ceder. 2013. "Commentary: The materials project: A materials genome approach to accelerating materials innovation." *APL Mater.* 1 (1): 011002. https://doi.org/10.1063/1.4812323.

James, G., D. Witten, T. Hastie, and R. Tibshirani, eds. 2013. *An introduction to statistical learning: With applications in R: Springer texts in statistics*. New York: Springer.

Jennings, H. M., A. Kumar, and G. Sant. 2015. "Quantitative discrimination of the nano-pore-structure of cement paste during drying: New insights from water sorption isotherms." *Cem. Concr. Res.* 76 (Oct): 27–36. https://doi.org/10.1016/j.cemconres.2015.05.006.

Kasperkiewicz, J., J. Racz, and A. Dubrawski. 1995. "HPC strength prediction using artificial neural network." *J. Comput. Civ. Eng.* 9 (4): 279–284. https://doi.org/10.1061/(ASCE)0887-3801(1995)9:4(279).

Li, G., and X. Zhao. 2003. "Properties of concrete incorporating fly ash and ground granulated blast-furnace slag." *Cem. Concr. Compos.* 25 (3): 293–299. https://doi.org/10.1016/S0958-9465(02)00058-6.

Liu, Y., T. Zhao, W. Ju, and S. Shi. 2017. "Materials discovery and design using machine learning." *J. Materiomics* 3 (3): 159–177. https://doi.org/10.1016/j.jmat.2017.08.002.

Lukasik, S., and S. Żak. 2009. "Firefly algorithm for continuous constrained optimization tasks." In *Proc., Int. Conf. on Computational Collective Intelligence*, 97–106. New York: Springer.

Manning, D. G., and B. B. Hope. 1971. "The effect of porosity on the compressive strength and elastic modulus of polymer impregnated concrete." *Cem. Concr. Res.* 1 (6): 631–644. https://doi.org/10.1016/0008-8846(71)90018-4.

Martius, G., and C. H. Lampert. 2016. "Extrapolation and learning equations." Preprint, submitted October 10, 2016. https://arxiv.org/abs/1610.02995.

Moré, J. J. 1978. "The Levenberg–Marquardt algorithm: Implementation and theory." In *Numerical analysis*, 105–116. New York: Springer.

Mueller, T., A. G. Kusne, and R. Ramprasad. 2016. "Machine learning in materials science: Recent progress and emerging applications." *Rev. Comput. Chem.* 29 (Apr): 186–273.

Nagwani, N. K., and S. V. Deo. 2014. "Estimating the concrete compressive strength using hard clustering and fuzzy clustering based regression techniques." *Sci. World J.* 2014: 1–16. https://doi.org/10.1155/2014/381549.

Oluokun, F. A., E. G. Burdette, and J. H. Deatherage. 1991. "Elastic modulus, Poisson's ratio, and compressive strength relationships at early ages." *ACI Mater. J.* 88 (1): 3–10.

Omran, B. A., Q. Chen, and R. Jin. 2016. "Comparison of data mining techniques for predicting compressive strength of environmentally friendly concrete." *J. Comput. Civ. Eng.* 30 (6): 04016029. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000596.

Pilania, G., C. Wang, X. Jiang, S. Rajasekaran, and R. Ramprasad. 2013. "Accelerating materials property predictions using machine learning." *Sci. Rep.* 3 (1): 2810. https://doi.org/10.1038/srep02810.

Polikar, R. 2006. "Ensemble based systems in decision making." *IEEE Circuits Syst. Mag.* 6 (3): 21–45. https://doi.org/10.1109/MCAS.2006.1688199.

Poon, C. S., L. Lam, and Y. L. Wong. 2000. "A study on high strength concrete prepared with large volumes of low calcium fly ash." *Cem. Concr. Res.* 30 (3): 447–455. https://doi.org/10.1016/S0008-8846(99)00271-9.

Powers, T. C., and T. L. Brownyard. 1946. "Studies of the physical properties of hardened portland cement paste." *ACI J. Proc.* 43 (9): 249–336.

Quinlan, J. R. 1992. "Learning with continuous classes." In *Proc., Australian Joint Conf. on Artificial Intelligence*, 343–348. Singapore: World Scientific.

Sarstedt, M., and E. Mooi. 2014 "Cluster analysis." In *A concise guide to market research*, 273–324. New York: Springer.

Schaffer, C. 1993. "Selecting a classification method by cross-validation." *Mach. Learn.* 13 (1): 135–143.

Schalkoff, R. J. 1997. *Artificial neural networks*. New York: McGraw-Hill.

Smola, A. J., and B. Schölkopf. 2004. "A tutorial on support vector regression." *Stat. Comput.* 14 (3): 199–222. https://doi.org/10.1023/B:STCO.0000035301.49549.88.

Svetnik, V., A. Liaw, C. Tong, J. C. Culberson, R. P. Sheridan, and B. P. Feuston. 2003. "Random forest: A classification and regression tool for compound classification and QSAR modeling." *J. Chem. Inf. Comput. Sci.* 43 (6): 1947–1958. https://doi.org/10.1021/ci034160g.

Vapnik, V. 2000. *The nature of statistical learning theory.* New York: Springer.

Veloso de Melo, V., and W. Banzhaf. 2017. "Improving the prediction of material properties of concrete using kaizen programming with simulated annealing." *Neurocomputing* 246: 25–44.

Wang, Y., and I. H. Witten. 1997. "Induction of model trees for predicting continuous classes." In *Proc., European Conf. on Machine Learning*. Prague, Czechia: Univ. of Economics.

Ward, L., A. Agrawal, A. Choudhary, and C. Wolverton. 2016. "A general-purpose machine learning framework for predicting properties of inorganic materials." *npj Comput. Mater.* 2 (1): 16028. https://doi.org/10.1038/npjcompumats.2016.28.

Yang, X.-S. 2009. "Firefly algorithms for multimodal optimization." In *Stochastic algorithms: Foundations and applications: Lecture notes in computer science*, edited by O. Watanabe and T. Zeugmann, 169–178. Berlin: Springer.

Yang, X.-S., and X. He. 2013. "Firefly algorithm: Recent advances and applications." *Int. J. Swarm Intell.* 1 (1): 36–50. https://doi.org/10.1504/IJSI.2013.055801.

Yao, X. 1999. "Evolving artificial neural networks." *Proc. IEEE* 87 (9): 1423–1447. https://doi.org/10.1109/5.784219.

Yeh, I.-C. 1998a. "Modeling of strength of high-performance concrete using artificial neural networks." *Cem. Concr. Res.* 28 (12): 1797–1808. https://doi.org/10.1016/S0008-8846(98)00165-3.

Yeh, I.-C. 1998b. "Modeling concrete strength with augment-neuron networks." *J. Mater. Civ. Eng.* 10 (4): 263–268. https://doi.org/10.1061/(ASCE)0899-1561(1998)10:4(263).

Yeh, I.-C., and L.-C. Lien. 2009. "Knowledge discovery of concrete material using genetic operation trees." *Expert Syst. Appl.* 36 (3): 5807–5812. https://doi.org/10.1016/j.eswa.2008.07.004.

Young, B. A., A. Hall, L. Pilon, P. Gupta, and G. Sant. 2019. "Can the compressive strength of concrete be estimated from knowledge of the mixture proportions?: New insights from statistical analysis and machine learning methods." *Cem. Concr. Res.* 115 (Jan): 379–388. https://doi.org/10.1016/j.cemconres.2018.09.006.

Zarandi, M. F., I. B. Türksen, J. Sobhani, and A. A. Ramezanianpour. 2008. "Fuzzy polynomial neural networks for approximation of the compressive strength of concrete." *Appl. Soft Comput.* 8 (1): 488–498. https://doi.org/10.1016/j.asoc.2007.02.010.

Zdeborová, L. 2017. "Machine learning: New tool in the box." *Nat. Phys.* 13 (5): 420–421. https://doi.org/10.1038/nphys4053.

Zhang, G., B. E. Patuwo, and M. Y. Hu. 1998. "Forecasting with artificial neural networks: The state of the art." *Int. J. Forecasting* 14 (1): 35–62. https://doi.org/10.1016/S0169-2070(97)00044-7.