Front. Struct. Civ. Eng. 2024, 18(7): 1084–1102
https://doi.org/10.1007/s11709-024-1077-z

RESEARCH ARTICLE

Bayesian Optimized LightGBM model for predicting the fundamental vibrational period of masonry infilled RC frames

Taimur RAHMAN[a], Pengfei ZHENG[b*], Shamima SULTANA[c]

[a] *Department of Civil Engineering, World University of Bangladesh, Dhaka 1230, Bangladesh*
[b] *School of Civil Engineering, Zhengzhou University, Zhengzhou 450001, China*
[c] *Department of Computer Science & Engineering, University of Asia Pacific, Dhaka 1205, Bangladesh*
[*] *Corresponding author. E-mail: pfzhce15@zzu.edu.cn*

ABSTRACT The precise prediction of the fundamental vibrational period for reinforced concrete (RC) buildings with infilled walls is essential for structural design, especially earthquake-resistant design. Machine learning models from previous studies, while boasting commendable accuracy in predicting the fundamental period, exhibit vulnerabilities due to lengthy training times and inherent dependence on pre-trained models, especially when engaging with continually evolving data sets. This predicament emphasizes the necessity for a model that adeptly balances predictive accuracy with robust adaptability and swift data training. The latter should include consistent re-training ability as demanded by real-time, continuously updated data sets. This research implements an optimized Light Gradient Boosting Machine (LightGBM) model, highlighting its augmented predictive capabilities, realized through the astute use of Bayesian Optimization for hyperparameter tuning on the FP4026 research data set, and illuminating its adaptability and efficiency in predictive modeling. The results show that the R^2 score of LightGBM model is 0.9995 and *RMSE* is 0.0178, while training speed is 23.2 times faster than that offered by XGBoost and 45.5 times faster than for Gradient Boosting. Furthermore, this study introduces a practical application through a streamlit-powered, web-based dashboard, enabling engineers to effortlessly utilize and augment the model, contributing data and ensuring precise fundamental period predictions, effectively bridging scholarly research and practical applications.

KEYWORDS masonry-infilled RC frame, fundamental period, LightGBM, FP4026 research dataset, machine learning, data-driven approach, Bayesian Optimization

1 Introduction

The fundamental period of vibration is a crucial structural design parameter for reinforced concrete (RC) buildings, particularly in regions susceptible to earthquakes. Predicting fundamental period with high accuracy is essential for ensuring safety and resilience of earthquake-resistant structures. The addition of infilled walls considerably modifies the seismic response of RC buildings, including the fundamental period. Therefore, it is essential for structural engineers to have an effective method for predicting the fundamental period of structures with such walls.

Article history: Received Aug 10, 2023; Accepted Oct 27, 2023

Past studies have emphasized the influence of infill on structural dynamics [1–7]. In particular, the diagonal strut method is the most prevalent technique for modeling infill panels [8]. Building upon these findings, researchers have turned their attention to understanding the fundamental period of masonry-infilled RC structures, in order to predict seismic response and develop appropriate design strategies [9–17]. Those investigations have significantly improved the understanding of the critical factors influencing the dynamic behavior of masonry-infilled RC frames and have helped to identify potential areas for improvement in both modeling and design.

Machine learning techniques are now applied to complex engineering challenges, in a diverse array of

applications [18–30]. In parallel with this, optimization techniques have matured, emerging as crucial instruments in addressing multifarious structural engineering conundrums. This is evidenced by recent explorations into innovative algorithms and strategies, applied to various facets of engineering inquiry [31–35]. By integrating these powerful tools with the foundational knowledge of infill panel behavior, researchers have been able to explore innovative approaches for predicting fundamental period of structures with or without masonry-infilled RC walls. Using both traditional machine learning and deep learning techniques, these studies have improved the accuracy and efficiency of predictions, enhancing the understanding of the dynamic behavior of infilled RC frames under seismic loading.

Artificial Neural Networks (ANNs) have accurately predicted the fundamental period of infilled RC frames using the FP4026 Research Database [8]. The Artificial Bee Colony (ABC) algorithm optimized neural network models, accurately estimating structural vibration period [36]. Different AI models involving neural networks, statistical regression, and a neuro-fuzzy approaches have also produced high accuracy in fundamental period calculation [37]. Techniques involving neural networks and extreme gradient boosting have been used in another study to predict the period of masonry-infilled RC frames [38] with a genetic algorithm optimizing model predictions on an existing database, and generating a new database. Advanced machine learning techniques based on bagging and boosting have accurately predicted fundamental period [39], with Shapley Additive Explanations revealing each independent variable's contribution. Similarly, machine learning coupled with simple nonlinear formulations has estimated the period of masonry-infilled RC frames [40]. Lastly, a recent deep-neural-network-based model has estimated the fundamental period of infilled RC frames [41]. An Android application, 'Building Period Estimator', was developed from this model, although potential limitations when updated with new user-contributed data. In the field of structural engineering, LightGBM has demonstrated not only its effectiveness in solving complex problems but also its time efficiency, owing to its capability to handle large data sets and parallel learning techniques [42–47]. Its gradient-boosting architecture and efficient tree-growing algorithms have proven to be advantageous for managing large data sets and producing accurate results. Similarly, Bayesian Optimization has shown time efficiency in addressing optimization tasks [48–52]. Its ability to efficiently explore the design space and converge to an optimal solution with minimal function evaluations reduces computational overhead, making it a highly efficient method for addressing problems requiring extensive search and optimization efforts. Notably, Bayesian Optimization outperformed conventional methods in predicting wind pressure for low-rise buildings, reducing the computational time by 88% to 94% [48]. Considering the effectiveness of LightGBM and the high efficiency of Bayesian Optimization in large data set learning, a combined method has much to offer.

The FP4026 Research Database is a rigorously curated collection of analytical data that elucidates the fundamental period of vibration in RC structures with masonry infill. Spanning cases up to 24 floors, the database integrates a multitude of geometrical and mechanical parameters—ranging from the number of storeys and spans to the span length and the percentage of openings in the infill walls. It also incorporates the stiffness of the infill wall panels, a crucial parameter often overlooked in conventional studies. Developed to serve both academic and industry professionals, the database aims to elevate the precision and reliability of seismic design calculations, the FP4026 Research data set has demonstrated significant achievements in terms of accuracy of prediction of vibration of RC-infilled walls. However, there is still a need for more comprehensive and practical data; incorporating additional data from real-world structures can enrich the data set.

Embarking on a thorough exploration into predicting the fundamental period of RC buildings with infilled walls, this research addresses gaps and demands that have been identified in existing scholarly publications and practical applications. Existing models have the shortcomings of substantial training times and inherent dependence on pre-trained models, especially as data sets grow, by means of freshly contributed updates. This scenario mandates a unique model that synergistically melds predictive precision with adaptive capabilities, not only ensuring its evolution with the addition of new user-contributed data, but also offering applicability and relevance.

The motivation for this study emanates from a multifaceted necessity to augment predictive accuracy to offer not only time-efficiency but also robust adaptability to the evolution of practical data. Integrating additional data—particularly those encapsulating a broad spectrum of variations and anomalies encountered in tangible building design and construction—is paramount. This requires the engagement of structural engineers, allowing them to fortify the database with pragmatic data derived from prior work. Thereby, the model's predictive competency can be enhanced and continuous refinement can be achieved, incorporating theoretical and practical robustness.

Adhering steadfast commitment to interweaving technological advancements with practical insights, this research fine-tunes a LightGBM model via Bayesian Optimization, and thereby aspires to create a tool that cannot merely evolve perpetually but can also offer robustness for the structural engineering community. Through comparative analysis, the research assesses the performance

of the LightGBM model relative to numerous machine learning models. Furthermore, the formulation of a user-centric, web-based dashboard, which enables engineers not only to augment the model with their data but also to utilize the pre-trained model for nuanced fundamental period predictions, reflects the desire to bridge theoretical and practical domains in structural engineering.

2 Methodology

The methodology, illustrated in Fig. 1, begins with the selection of the FP4026 research database, which serves as the primary data set for investigating the fundamental periods of infilled RC frame structures. An exploratory data analysis (EDA) is then conducted to gain a deep understanding of the data set parameters and identify any hidden patterns. This involves assessing correlations using violin plots to guide subsequent research steps. Following the EDA, the data set undergoes cleaning and preprocessing to prepare it for machine learning models. This step addresses any missing or inconsistent data and normalizes and transforms variables, as needed, to ensure the data are in an appropriate format for model training and evaluation.

The study then moves on to model selection and comparison. The LightGBM machine learning model is chosen due to its proven computational efficiency and fast training speed. To optimize the model's performance, Bayesian Optimization is employed for hyperparameter tuning. This optimization process is embedded within a *K*-fold cross-validation framework which mitigates the risk of overfitting, ensures a more unbiased model, and provides a robust estimate of the model's performance on unseen data. The objective of this step is to fine-tune the model using advanced optimization methods and maximize the prediction accuracy of the fundamental period of infilled RC frame structures. The performance of the LightGBM model is then compared with that of other ML models, including Decision Tree, Random Forest,

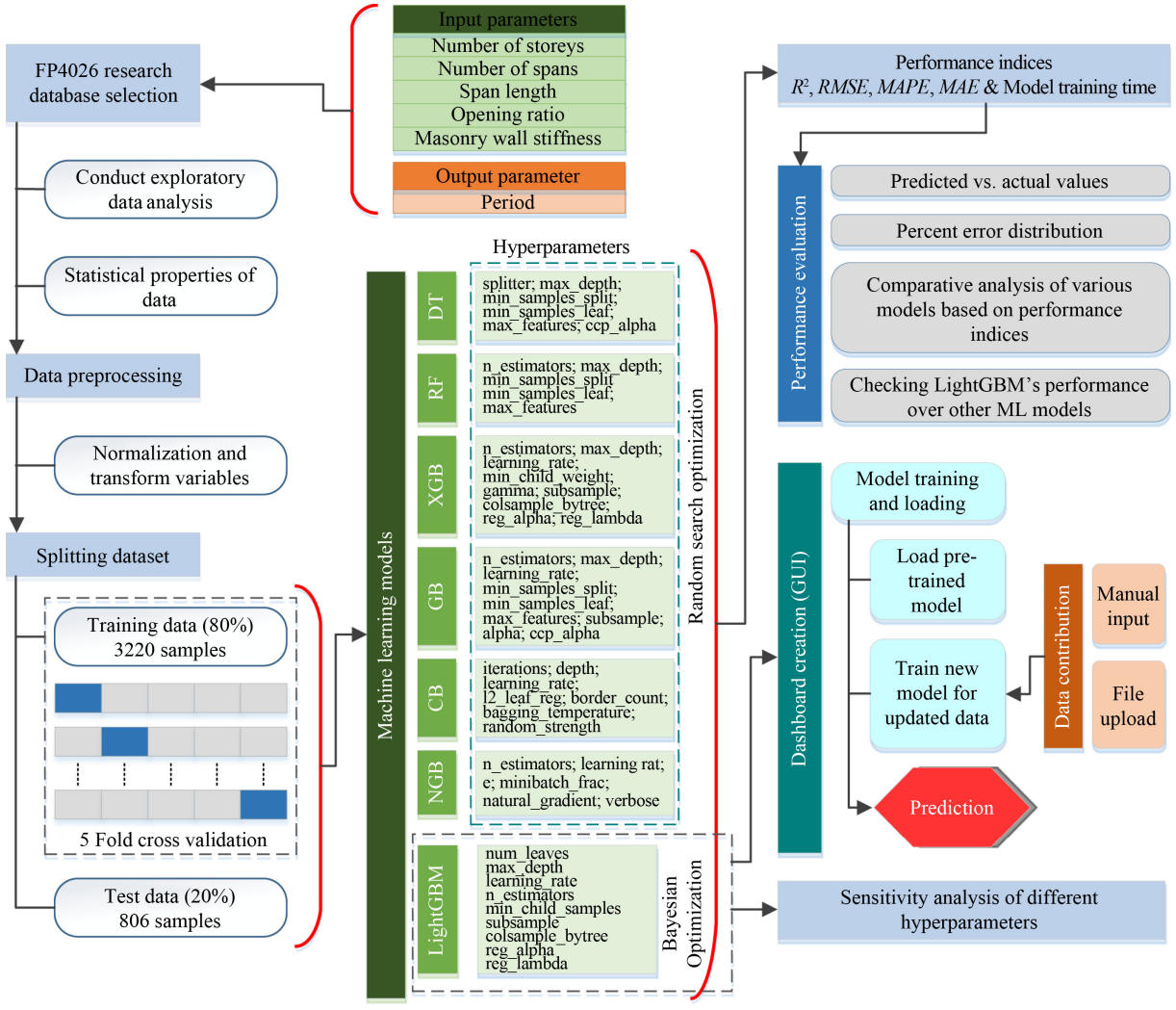


Fig. 1 Methodology overview.

XGBoost, and Gradient Boosting, by calculating their metric scores, percent error plot, and predicted values vs. actual values plot. Once optimized, the model is trained and evaluated using the preprocessed data set. Performance metrics, including the Coefficient of Determination (R^2), Root Mean Squared Error (*RMSE*), Mean Absolute Percentage Error (*MAPE*), Mean Absolute Error (*MAE*) and model training time, are used to assess the model's effectiveness. The results are then compared.

Lastly, a user-friendly web-based dashboard is developed using the Streamlit framework. This dashboard enables structural engineers to input data and use a pretrained model to predict the fundamental period of vibration, and offers an accessible platform for collaboration and prediction. The dashboard serves as a crucial component of this study's methodology by providing a valuable tool for improving the data set and the model's prediction capabilities.

3 Data set

The FP4026 Research Database is an extensive data set comprising fundamental period data for 4026 RC structures featuring masonry wall infill. The database incorporates various parameters, including storey numbers, spans, span lengths, infill wall opening percentages, and masonry wall rigidity. The frames in the database adhere to Eurocode 8 specifications, and their fundamental period values are analytically determined using SeismoStruct software. This valuable resource can aid in developing new code proposals for estimating the fundamental period of masonry wall-infilled RC structures. This study's data set is accessible from an online data repository [53]. Recently, seminal contributions by Somala et al. [39] and Latif et al. [38] have offered a nuanced grasp of various data mining attributes associated with the FP4026 data set, including correlation matrices, relationships among features, and the significance of variables. Charalampakis et al. [40] have already furnished a comprehensive overview of the data set, and a concise summary of their work is also presented in the current study.

The output variable for the equivalent strut nonlinear cyclic model is the fundamental time period of the masonry-infilled RC frame. Specifically, there are seven distinct values for wall stiffness (2.25×10^5, 4.50×10^5, 7.50×10^5, 11.25×10^5, 15.00×10^5, 20.00×10^5, and 25.00×10^5 kN/m), five distinct values for opening percentage (0%, 25%, 50%, 75%, and 100%), three distinct values for the number of spans (2, 4, and 6), and four distinct values for span length (3.0, 4.5, 6.0, and 7.5 m). The number of storeys ranges from 1 to 22, with a unit step for each parameter.

This study presents a series of violin plots to illustrate the relationships between various building characteristics and their influence on the structural response, specifically the period (in s), as shown in Fig. 2. Violin plots for all five input variables are generated. Combining the characteristics of box plots and kernel density plots, each violin plot generated by the Python Seaborn library provides an insightful visual representation of the data distribution. Box plots illustrate the central tendency, spread, and potential outliers of the data, whereas kernel density plots offer a smooth estimate of the probability density function. The combination of these two methods within the violin plot permits a more thorough comprehension of the underlying data distribution, both in terms of summary statistics and density information.

Swarm plots are overlaid on violin plots to further enhance the clarity of data representation. Individual data points are represented as dots arranged along the categorical axis with a slight separation to prevent overlap. This visual approach helps reveal the actual data points and provides a clearer understanding of the data's density, distribution, and potential patterns or clusters. Due to their ability to convey a vast amount of information in a single, compact visualization, violin plots have been chosen to represent the data set in this study. By displaying both summary statistics (as in box plots) and data distribution (as in kernel density plots), violin plots provide a more comprehensive view of the relationships between building characteristics and structural response. The statistical properties of the training and test data also have been given in Table 1.

4 Model development and performance evaluation

4.1 Light Gradient Boosting Machine (LightGBM)

LightGBM, or Light Gradient Boosting Machine, created by Microsoft, is a gradient boosting framework renowned for efficiently handling large data sets and delivering high performance without high memory usage [54]. It is based on the gradient boosting framework, an ensemble learning method aiming to optimize a specific objective function by combining multiple weak learners, typically decision trees. The core concept of gradient boosting involves iteratively fitting weak learners to the negative gradient of the objective function concerning the current model prediction [55]. Mathematically, this can be expressed as:

$$F_m(x) = F_{m-1}(x) + \eta \sum_{k=1}^{K} \gamma_{mk} h_{mk}(x), \tag{1}$$

where $F_m(x)$ is the updated model at iteration$m$, $F_{m-1}(x)$ is the previous model, η is the learning rate, K is the number of weak learners, $h_{mk}(x)$ represents the kth weak

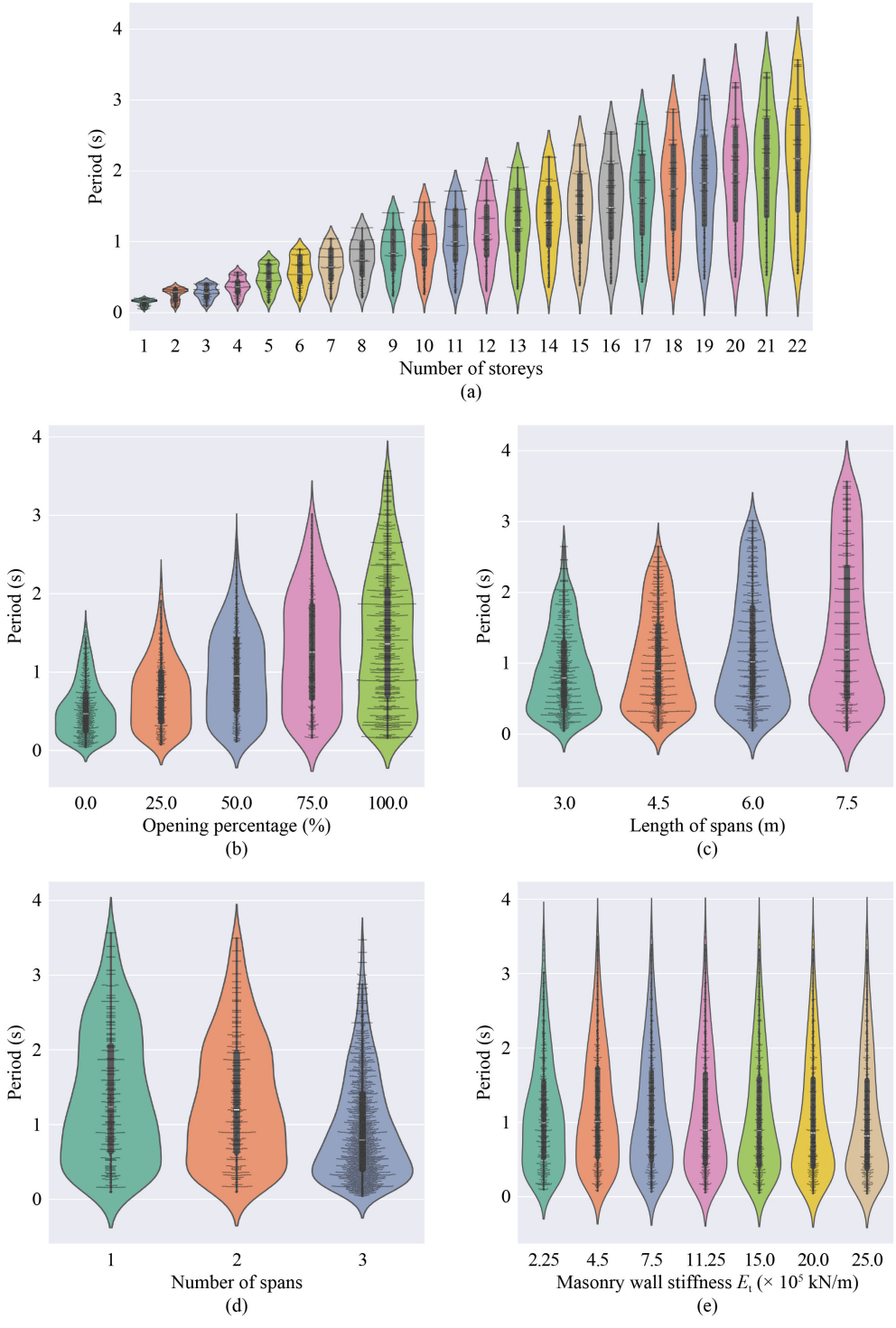


Fig. 2 Violin plots showing distribution of period (s) for various building characteristics: (a) fundamental period vs. number of storey; (b) fundamental period vs. opening percentage (%); (c) fundamental period vs. length of spans (m); (d) fundamental period vs. number of spans; (e) fundamental period vs. masonry wall stiffness, E_t (× 10^5 kN/m).

learner, and γ_{mk} is the corresponding weight.

A unique aspect of LightGBM is its implementation of a leaf-wise tree growth strategy, contrasting with the conventional level-wise strategy used by the majority of gradient boosting algorithms. The leaf-wise approach enables LightGBM to grow trees by selecting the leaf with the highest delta loss to split rather than expanding all leaves at the same level. This strategy can be expressed mathematically as:

$$\Delta Loss = \frac{(\sum_{i\in I_L} g_i)^2 + (\sum_{i\in I_R} g_i)^2}{\sum_{i\in I_L} h_i + \lambda + \sum_{i\in I_R} h_i + \lambda} - \frac{(\sum_{i\in I} g_i)^2}{\sum_{i\in I} h_i + \lambda}, \quad (2)$$

where I_L and I_R represent the left and right child nodes

Table 1 Statistical properties of the training and test data

Dataset	Property Unit	Nos. of storey	Nos. of spans	Length of spans (m)	Opening percentage (%)	Masonry wall stiffness ($\times 10^5$ kN/m)	Period (s)
Training (3220 samples)	Min	1.00	2.00	3.00	0.00	2.25	0.04
	Max	22.00	6.00	7.50	100.00	25.00	3.57
	Mean	11.61	4.95	4.99	63.07	11.77	1.11
	SD	6.34	1.55	1.57	40.11	7.79	0.78
	Skewness	−0.01	−1.05	0.17	−0.51	0.38	0.80
	kurtosis	−1.21	−0.52	−1.19	−1.37	−1.17	−0.11
Test (806 samples)	Min	1.00	2.00	3.00	0.00	2.25	0.04
	Max	22.00	6.00	7.50	100.00	25.00	3.50
	Mean	11.07	4.94	5.01	63.12	11.74	1.09
	SD	6.37	1.55	1.59	40.27	7.79	0.81
	Skewness	0.06	−1.03	0.14	−0.51	0.37	0.92
	kurtosis	−1.17	−0.56	−1.22	−1.38	−1.16	0.09

after splitting, I is the set of instances in the current leaf, g_i and h_i are the first and second-order gradients of the loss function with respect to prediction, respectively, λ is the regularization term.

The leaf-wise strategy, combined with the best-first growth policy, allows LightGBM to attain greater accuracy and improved convergence compared to level-wise tree growth. However, it may also cause overfitting when dealing with smaller data sets, which can be mitigated by adjusting hyperparameters such as tree maximum depth, minimum samples per leaf, and learning rate. Another benefit of LightGBM is its compatibility with parallel and GPU learning, which can substantially decrease training time and enhance the framework's scalability. Moreover, LightGBM provides various regularization techniques, including L1 and L2 regularization, to prevent overfitting and enhance the model's generalization [54].

In summary, LightGBM is a powerful gradient-boosting framework that offers several advantages over traditional gradient-boosting methods, including its leaf-wise tree growth strategy, efficient memory usage, and parallel learning capabilities. These features make it an attractive option for tackling complex problems in various domains, including structural engineering.

4.2 Bayesian optimization

Bayesian Optimization is a powerful global optimization method for costly black-box functions, and has found widespread application in various domains such as hyperparameter tuning, optimization of resource-intensive experiments, and structural engineering [56–58]. The technique is built upon use of a probabilistic model, typically a Gaussian process ($\mathcal{GP}$), to represent the objective function and quantify the uncertainty in function evaluations [59]. A Gaussian process can be mathematically defined as follows:

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')), \tag{3}$$

where $m(x)$ is the mean function and $k(x, x')$ is the covariance function or kernel. Widely used kernels include the squared exponential kernel and the Matérn kernel.

The primary concept behind Bayesian Optimization is utilization of an acquisition function to steer the search toward the optimum by balancing exploration (probing areas with high uncertainty) and exploitation (probing areas with high expected improvement) [60]. A common acquisition function is the Expected Improvement (EI), which is defined as:

$$EI(x) = \mathbb{E}[\max(f(x) - f(x^+), 0)], \tag{4}$$

where x^+ signifies the current best solution. The optimization procedure iteratively selects the subsequent point to evaluate based on the acquisition function:

$$x_{t+1} = \underset{x}{\operatorname{argmax}} \mathrm{EI}(x). \tag{5}$$

Following each evaluation, the $\mathcal{GP}$ model is updated, and the acquisition function is re-optimized to identify the next point for evaluation.

4.3 Hyperparameter tuning

In this research, the Light Gradient Boosting Machine (LightGBM) model is evaluated with various hyperparameter combinations, including boosting type, number of leaves, maximum depth, learning rate, number of estimators, minimum child samples, subsample, column sample by tree, L1 regularization term, and L2 regularization term. The boosting type is chosen from two options: 'gbdt' and 'dart'. The number of leaves parameter is adjusted to regulate the complexity of individual trees, where a higher number might improve

performance but also increase the risk of overfitting. The maximum depth parameter is determined to control the overall complexity of the model. The learning rate is set to dictate the step size for weight updates during training, with a smaller rate, possibly leading to more accurate models, but requiring increased computational resources. The parameter, number of estimators, is used to establish the total number of trees to be constructed. The parameter, minimum child samples, is adjusted to manage overfitting by requiring a minimum number of data points in each leaf node. The subsample parameter is employed to introduce randomness and enhance generalization, while the parameter, column sample by tree, is set to govern feature selection for each tree. Lastly, L1 and L2 regularization terms are incorporated to manage overfitting and promote sparsity in weight values. The values of the aforementioned hyperparameters are presented in Table 2. Through tuning these hyperparameters, this study aimed to identify the optimal combination to yield the best predictive performance while minimizing the risk of overfitting with reduced computational resources needed for model training.

The performance of the model is assessed using a 5-fold cross-validation with a random state set to 42, ensuring reproducibility. The R^2 score is employed as the scoring metric for evaluating the model's performance during the hyperparameter tuning process.

4.4 Performance metrics

Upon identifying the best hyperparameters, the LightGBM model is trained using the entire training data set, predictions are made on the test data set, and R^2, *RMSE*, and *MAE* are computed. The mathematical equations for the goodness-of-fit indicators are as follows, Eqs. (6)−(9):

$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}, \tag{6}$$

where SS_{res} = sum of squares of residuals (or errors), and SS_{tot} = total sum of squares. The closer R^2 is to 1, the better the fit of the model. An R^2 of 0 indicates that the model does not explain any of the variance in the dependent variable.

RMSE is a measure of the difference between the predicted values and the actual values. It is calculated as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}, \tag{7}$$

where y_i = actual value of the ith observation, $\hat{y}_i$ = predicted value of the ith observation, and n = the number of observations.

MAPE and *MAE* are measures of the accuracy of a prediction. The lower the value of these, the better the model's accuracy. These are calculated as follows:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100\%, \tag{8}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|. \tag{9}$$

The model training time, in addition to R^2, *RMSE*, *MAPE*, and *MAE*, is taken into consideration as a performance index. The model training time refers to the time taken to train the model on a specific data set. This metric is particularly relevant in practical applications where computational resources and time are limited. A model with a faster training time can provide timely predictions, enhancing its utility in real-world scenarios. It is also worth noting that a model's training time can vary based on the complexity of the model, the size of the data set, and the computational resources available.

4.5 Model training

Bayesian Optimization initiates the training process by generating suggestions for hyperparameters. These suggested hyperparameters are then incorporated into the LightGBM model and the model, with these hyperparameters, is trained using *K*-fold cross-validation, yielding a

Table 2 Hyperparameters for the LightGBM model

Hyperparameter	Value range
Maximum number of decision leaves (num_leaves)	(20, 150)
Maximum tree depth (max_depth)	(5, 50)
Learning rate (learning_rate)	(0.05, 0.5)
Number of estimators (n_estimators)	(100, 1000)
Minimum sum of instance weight in a child node (min_child_samples)	(5, 20)
Fraction of observations to be randomly sampled for each tree (subsample)	(0.5, 1)
Column sample by tree (colsample_bytree)	(0.8, 1)
L1 regularization term (reg_alpha)	(0, 0.1)
L2 regularization term (reg_lambda)	(0, 0.5)

trained model. The performance of this trained model is then evaluated, and the performance metrics are generated. These performance metrics serve two primary functions: they are fed back into the Bayesian optimization process to refine the subsequent suggestions for hyperparameters, and they are also used to compare and identify the best-performing model. Once the best model is identified, it is retrained on the entire training set to produce the final trained model. This final model is used to make predictions on the test set. These predictions are then evaluated, and performance metrics on the test set are generated. This systematic and cyclic process of model tuning and evaluation is depicted in Fig. 3. Each step in the process is designed to enhance the model's predictive accuracy and computational efficiency, ultimately contributing to the superior performance of the LightGBM model in predicting the fundamental period of infilled RC frame structures.

A box and whisker plot, shown in Fig. 4, is created to visualize the distribution of R^2 scores for each hyperparameter. In the sensitivity analysis, each hyperparameter is iterated over its specified range while keeping the other hyperparameters constant. This analysis uses the following ranges for each hyperparameter: 'num_leaves' (20 to 150 in steps of 10), 'max_depth' (5 to 50 in steps of 1), 'learning_rate' (0.05 to 0.5 in steps of 0.05), 'n_estimators' (100 to 1000 in steps of 100), 'min_child_samples' (5 to 20 in steps of 2), 'subsample' (0.5 to 1 in steps of 0.05), 'colsample_bytree' (0.8 to 1 in steps of 0.05), 'reg_alpha' (0.0 to 0.1 in steps of 0.05), and 'reg_lambda' (0.0 to 0.5 in steps of 0.05). For each hyperparameter value, the model is trained and evaluated using the updated values. For example, for the hyperparameter 'num_leaves', the R^2 scores range from 0.9994 to 0.9997. The 'max_depth' hyperparameter exhibits a more consistent performance with all the R^2 scores concentrated around 0.9995. In this study, for the 'learning_rate', the R^2 scores ranged from approximately 0.9995 to 0.9997, showing a relatively high model performance.

The other hyperparameters also had similar levels of variations in the R^2 scores. This plot, Fig. 4, visually demonstrates the variation in model performance concerning different hyperparameter values. By analyzing the plot, we gain a better understanding of each hyperparameter's influence on the model's accuracy. This analysis is crucial for making informed decisions when fine-tuning the model, which in turn enhances its performance and robustness. As a result, the optimized model provides more accurate predictions of the fundamental period of RC-infilled walls.

Figure 5 presents a focused sensitivity analysis of individual hyperparameters for the LightGBM model.

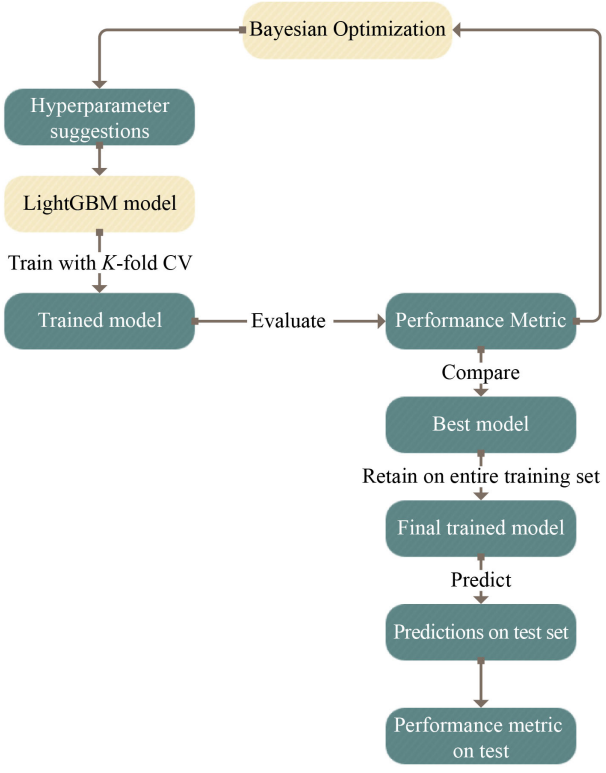


Fig. 3 The Bayesian Optimization process for LightGBM hyperparameter tuning.

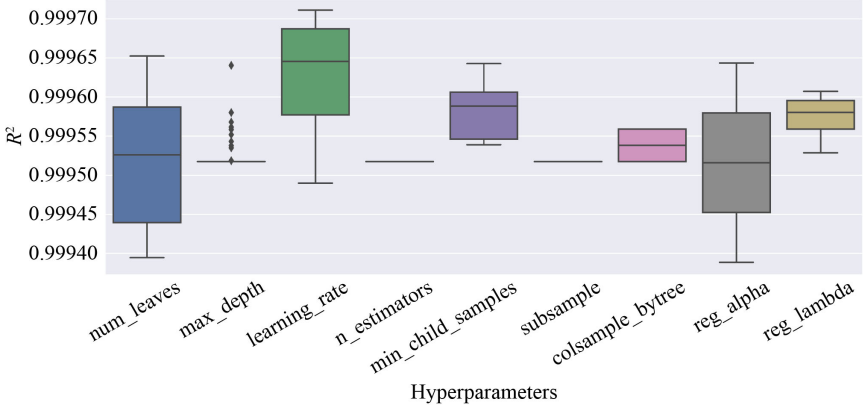
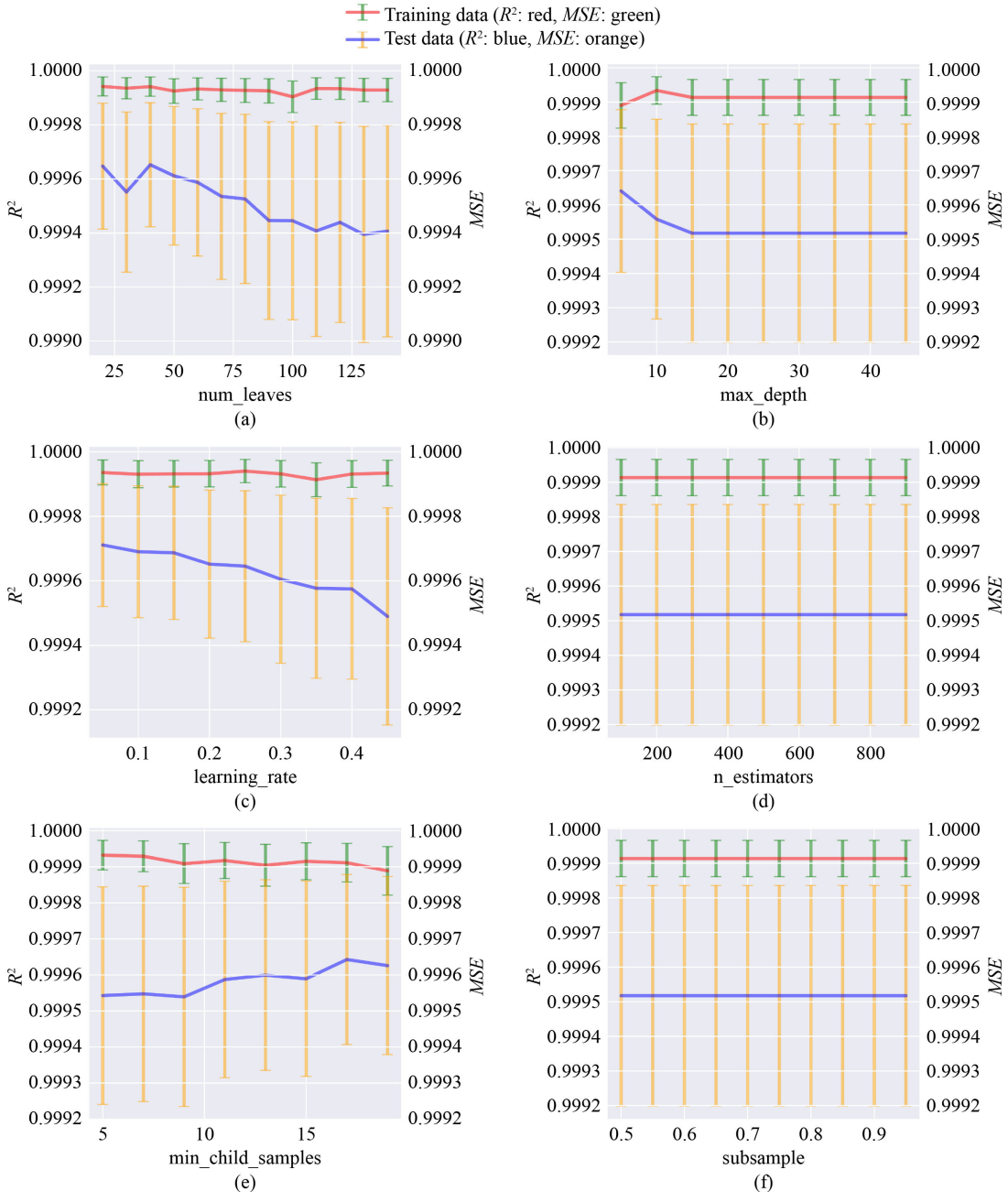


Fig. 4 R^2 scores of different hyperparameters.

The objective of this analysis is to evaluate the performance impact of each individual hyperparameter on the model, independent of the others. This univariate approach allows us to scrutinize the role of each hyperparameter more clearly, setting the stage for the more complex, multivariate optimization carried out by Bayesian Optimization. Each plot in the figure illustrates the effect of varying a single hyperparameter while maintaining constant values for the other hyperparameters. Both the R^2 scores and mean squared error (*MSE*) are presented for the training and test data sets. In each plot, the *x*-axis represents the different values of a particular hyperparameter, while the left *y*-axis displays the corresponding R^2 scores. The right *y*-axis represents the associated *MSE*. For the training data, R^2 scores are represented by red lines, while *MSE* range is depicted using green error bars. Similarly, for the test data, blue lines indicate R^2 scores and orange error bars represent ranges of *MSE*. For instance, evaluation of the 'num_leaves' hyperparameter revealed that a value of 20 resulted in an R^2 score of 0.99994 and a range of *MSE* values around 0.000035 for the training data, and an R^2 score of 0.99965 and a range of *MSE* values around 0.000233 for the test data. On the other hand, a 'num_leaves' value of 140 gave an R^2 score of 0.99993 and a range of *MSE* values around 0.000043 for the training data and an R^2 score of 0.99941 and a range of *MSE* values around 0.000391 for the test data. When the



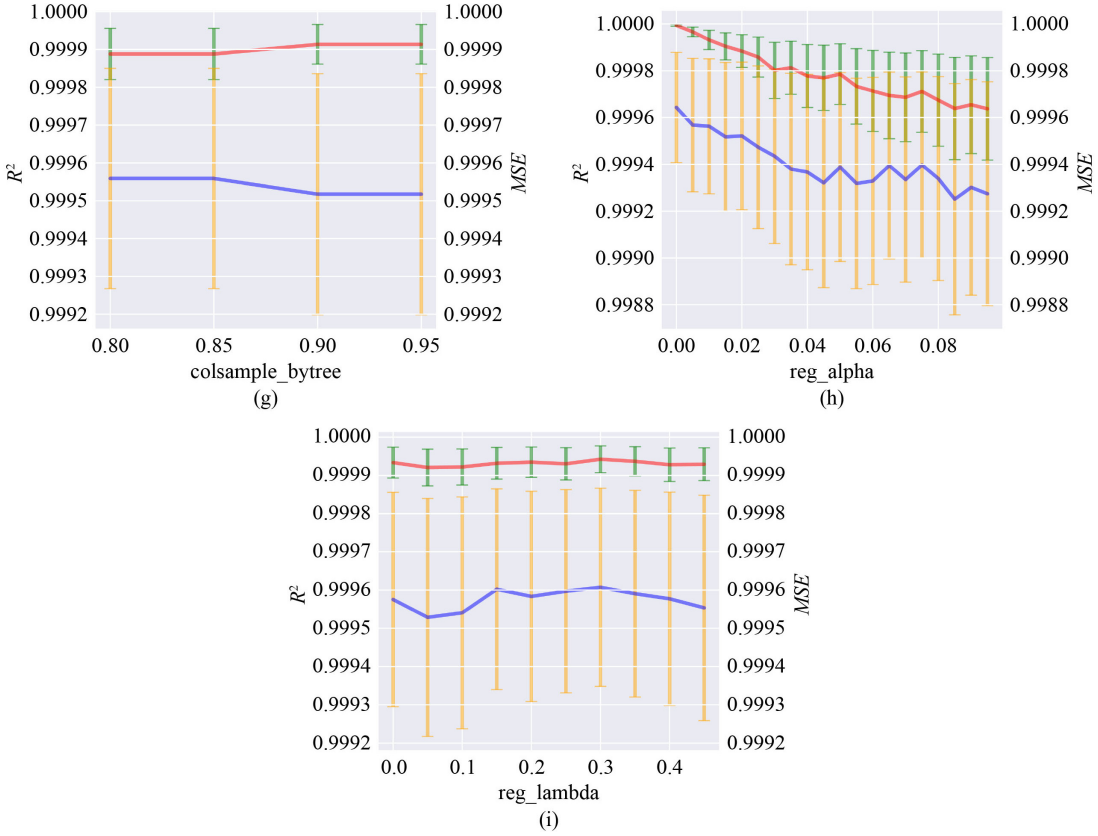


Fig. 5 Sensitivity Analyses of LightGBM Hyperparameters: (a) number of leaves sensitivity analysis; (b) maximum depth sensitivity analysis; (c) learning rate sensitivity analysis; (d) number of estimators sensitivity analysis; (e) minimum child samples sensitivity analysis; (f) subsample sensitivity analysis; (g) column sample by tree sensitivity analysis; (h) regularization alpha sensitivity analysis; (i) regularization lambda sensitivity analysis.

'max_depth' hyperparameter was varied from 5 to 45, the R^2 score for the training data remained constant at 0.99991, as did the change of *MSE* at 0.000052. The same trend was observed in the test data, where the R^2 score was 0.99952, and the range of *MSE* was 0.000319, irrespective of the 'max_depth' value. For the 'learning_rate' hyperparameter, a rate of 0.05 resulted in an R^2 score of 0.99994 and a range of *MSE* values around 0.000039 for the training data, and an R^2 score of 0.99971 and a range of *MSE* values around 0.000191 for the test data. When the learning rate was increased to 0.45, the R^2 score decreased slightly to 0.99993 for the training data and to 0.99949 for the test data, while the range of *MSE* increased to 0.000040 and 0.000337 for the training and test data, respectively. A similarly detailed analysis can be performed for the remaining hyperparameters.

This detailed sensitivity analysis can guide users in fine-tuning the model to achieve higher accuracy and improved performance in predicting the fundamental period of RC-infilled walls. Additionally, it allows for the determination of optimal values or ranges for each hyperparameter, ultimately leading to the best model performance. Figure 5 helps to visualize the trade-offs between overfitting and underfitting, as well as the robustness and generalizability of the model across different settings.

The study continues with an evaluation of experimental and predicted period values illustrated in Fig. 6, as well as the error distribution plot shown in Fig. 7. Figure 6 presents an evaluation of experimental and predicted period values using the optimal LightGBM model. The model displays a high degree of precision in predicting the period values, as evidenced by the metric scores for both training and test sets. For the training set, the model demonstrates an impressive R^2 score of 0.9999, signifying an excellent fit of the data. Further evaluation metrics, such as *RMSE*, *MAPE*, and *MAE*, are also notably low at 0.0001, 0.6983%, and 0.0049, respectively. This indicates a minimal discrepancy between actual and predicted values. Moreover, when applied to the test set, the model sustains its excellent performance with an R^2 score of 0.9995. The *RMSE*, *MAPE*, and *MAE* scores for the test set are marginally higher than for the training set, at 0.0003, 1.4213%, and 0.0098, respectively, but still, demonstrate the model's reliable predictive capability.

The study also includes the error distribution plot shown in Fig. 7. Error distribution plots are created to gain better insights into the dispersion of errors in the model's predictions. The percentage error for each data point in the training and test sets is computed and

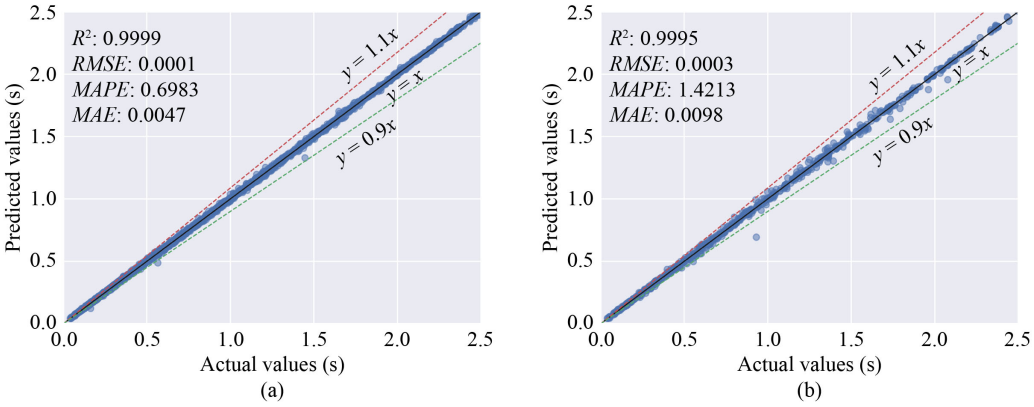


Fig. 6 Predicted vs. actual values (LightGBM model): (a) training set; (b) test set.

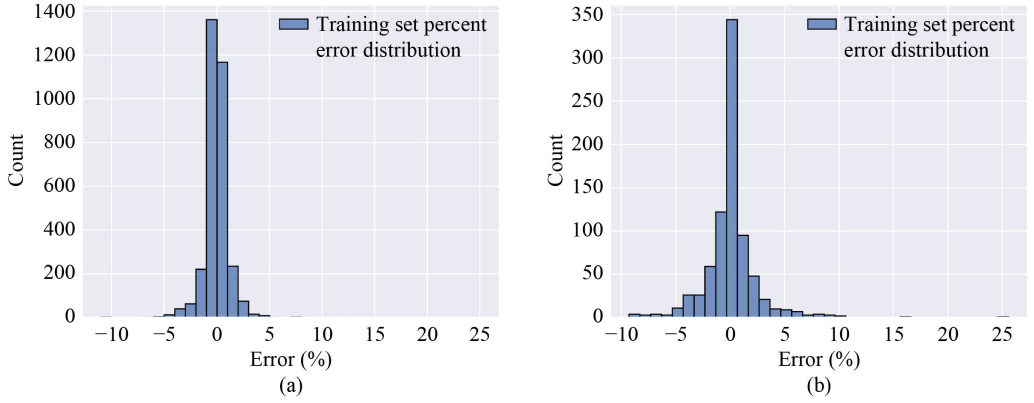


Fig. 7 Percent Error Distribution (LightGBM model): (a) training set; (b) test set.

depicted as histogram. These visualizations represent the model's prediction errors, enabling the identification of any possible biases or trends in the errors. The histograms also aid in determining whether the error distributions are symmetric and centered around zero, which would indicate a high-performing model.

4.6 Model Comparison

The performance of the LightGBM model was compared with other machine learning models, including Decision Tree, Random Forest, XGBoost, Gradient Boosting, CatBoost and NGBoost, to determine the most effective model for the given problem. The selection of these models was based on a comprehensive review of previous studies and conducted within the scope of these studies. Table 3 presents a detailed overview of the hyperparameters considered for each predictive model. These hyperparameters were chosen based on their relevance and impact on the model's performance, as demonstrated in the literature.

To ensure a fair and unbiased comparison, each model underwent a rigorous hyperparameter tuning process using the same cross-validation approach. The *K*-Fold cross-validation technique was employed, which involved partitioning the data into *K* equally sized folds. The model was then trained on $(K-1)$ folds and validated on the remaining fold. This process was repeated *K* times, and the average performance across all iterations was used as the model's performance metric. For instance, the XGBoost model's hyperparameter tuning was performed using a grid search with *K*-Fold cross-validation. The random grid search explored the hyperparameter space systematically, evaluating the model's performance for each combination of hyperparameters. This exhaustive search enabled the identification of the best hyperparameters for the model, optimizing its overall performance. Once the best hyperparameters were found, the XGBoost model was trained accordingly.

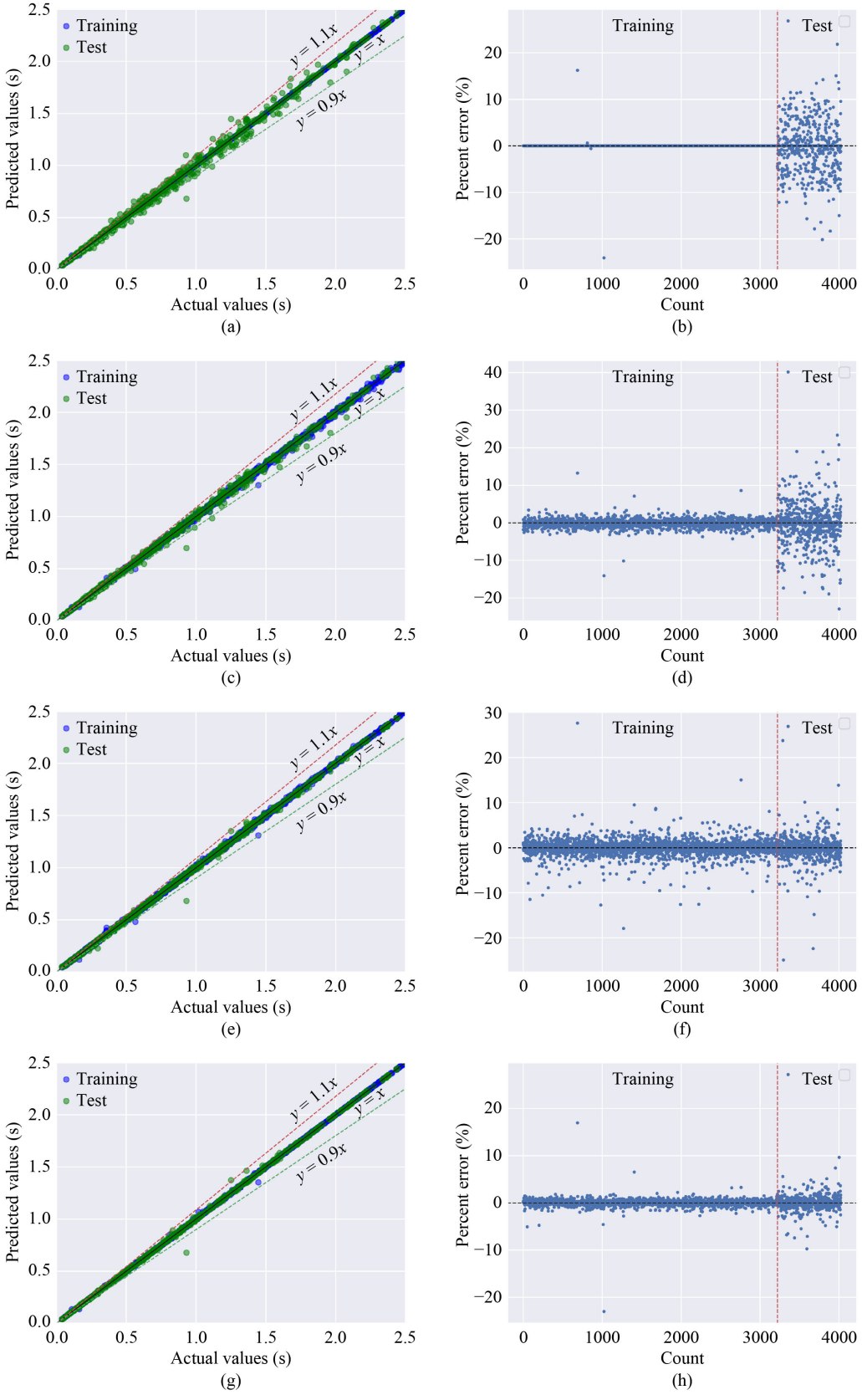
This same rigorous hyperparameter tuning process, involving the random search optimization with *K*-Fold cross-validation, was applied to each of the other machine learning models. Figure 8 illustrates a comprehensive comparison of these various machine learning models, showing their predicted vs. actual values and Percent Error Distribution, providing valuable insights into each model's performance and accuracy. The LightGBM model exhibited a test set mean percent error of 0.09% and *MAPE* of 1.42%. The Decision Tree model showed a mean error of −0.13% and a *MAPE* of 2.52%. The Random Forest reported a mean error of −0.85% and a *MAPE* of 3.42%. The XGBoost model had a mean error

Table 3 Hyperparameters for the predictive models

Model	Hyperparameter	Value
Decision Tree	splitter of nodes (splitter)	'best', 'random'
	maximum tree depth (max_depth)	'None', 1, 3, 5, 7, 9, 11
	minimum samples for split (min_samples_split)	2, 5, 10, 20, 50
	minimum samples of leaf node (min_samples_leaf)	1, 2, 5, 10, 20, 50
	number of features (max_features)	'None', 'auto', 'sqrt', '$\log_2$'
	minimal cost-complexity pruning (ccp_alpha)	0.0, 0.01, 0.1, 1, 10
Random Forest	number of estimators (n_estimators)	100, 500, 1000
	maximum tree depth (max_depth)	'None', 5, 10, 15, 20
	minimum samples for split (min_samples_split)	2, 5, 10
	minimum samples of leaf node (min_samples_leaf)	1, 2, 4
	number of features (max_features)	'auto', 'sqrt', '$\log_2$'
XGBoost	number of estimators (n_estimators)	100, 500, 1000
	maximum tree depth (max_depth)	'None', 1, 3, 5, 7, 9, 11
	learning rate (learning_rate)	0.01, 0.1, 0.2
	minimum sum of instance weight in a child node (min_child_weight)	1, 3, 5
	minimum loss reduction (gamma)	0, 0.1, 0.2
	fraction of observations to be randomly sampled for each tree (subsample)	0.8, 1
	column sample by tree (colsample_bytree)	0.8, 1
	L1 regularization term (reg_alpha)	0, 0.1, 1
	L2 regularization term (reg_lambda)	0.1, 1, 10
Gradient Boosting	number of estimators (n_estimators)	100, 500, 1000
	maximum tree depth (max_depth)	'None', 1, 3, 5, 7, 9, 11
	learning rate (learning_rate)	0.01, 0.1, 0.2
	minimum samples for split (min_samples_split)	2, 5, 10
	minimum samples of leaf node (min_samples_leaf)	1, 3, 5
	number of features (max_features)	'None', 'sqrt', '$\log_2$'
	fraction of observations to be randomly sampled for each tree (subsample)	0.8, 1
	regularization parameter (alpha)	0.1, 0.5, 0.9
	minimal cost-complexity pruning (ccp_alpha)	0, 0.1, 1
CatBoost	number of iterations (iterations)	100, 500, 1000
	maximum tree depth (depth)	3, 5, 7, 9, 11
	learning rate (learning_rate)	0.01, 0.1, 0.2
	L2 regularization (l2_leaf_reg)	1, 3, 5, 7, 9
	border count (border_count)	32, 64, 128
	bagging temperature (bagging_temperature)	0, 0.5, 1
	random strength (random_strength)	0.2, 0.5, 0.8
NGBoost	number of estimators (n_estimators)	100, 500, 1000
	learning rate (learning_rate)	0.01, 0.1, 0.2
	mini-batch fraction (minibatch_frac)	0.5, 0.7, 1.0
	natural gradient (natural_gradient)	'True', 'False'
	verbosity (verbose)	'True', 'False'

of −0.17% and a *MAPE* of 1.28%. The Gradient Boosting model showed a mean error of 0.06% and a *MAPE* of 0.75%. In addition to these models, CatBoost model reported a *MAPE* of 0.59% and the NGBoost model exhibited a *MAPE* of 2.53%.

The performance of the LightGBM model is compared



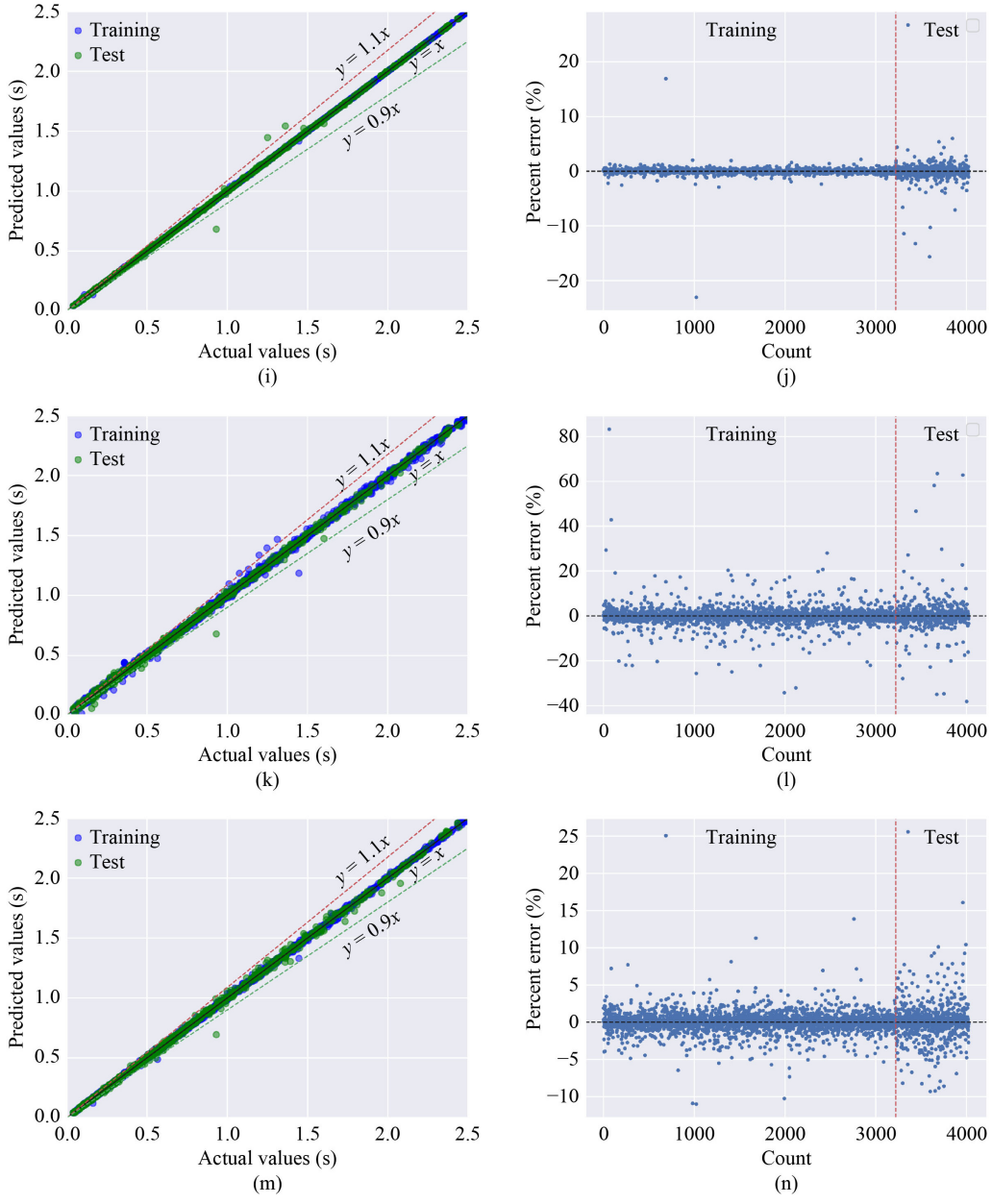


Fig. 8 Predicted vs. actual values and Percent Error Distribution for the compared models: (a) predicted vs. actual values (Decision Tree model); (b) Percent Error Distribution (Decision Tree model); (c) predicted vs. actual values (Random Forest model); (d) Percent Error Distribution (Random Forest model); (e) predicted vs. actual values (XGBoost model); (f) Percent Error Distribution (XGBoost model); (g) predicted vs. actual values (Gradient Boosting model); (h) Percent Error Distribution (Gradient Boosting model); (i) predicted vs. actual values (CatBoost model); (j) Percent Error Distribution (CatBoost model); (k) predicted vs. actual values (NGBoost model); (l) Percent Error Distribution (NGBoost model); (m) predicted vs. actual values (LightGBM model); (n) Percent Error Distribution (LightGBM model).

with that of other models using calculated metrics, as demonstrated in Table 4. Although the LightGBM model does not achieve the highest R^2 and lowest *RMSE* scores compared to XGBoost, Gradient Boosting, and CatBoost, its most notable advantage lies in its significantly faster training time. Specifically, the LightGBM model achieves an R^2 score of 0.9995 on the test set, which is slightly lower than the scores of XGBoost (0.9997) and Gradient Boosting (0.9998), but higher than ANN (0.993), Decision Tree (0.9982), Random Forest (0.9992), and NGBoost (0.992). The test *RMSE* for the LightGBM model is 0.0178, which is higher than those of XGBoost (0.0149) and Gradient Boosting (0.0126), but lower than ANN (0.068), Decision Tree (0.0344), and Random Forest (0.0232). Most importantly, the LightGBM model exhibits significantly faster training times, requiring only 8.44 s. This is 1.3 × slower than Decision Tree (6.42 s), 6.9 × faster than Random Forest (58.24 s), 23.2 × faster

Table 4 Performance evaluation of different models

Machine learning model	R^2		*RMSE*		*MAPE*		*MAE*		Model training time (s)
	Train	Test	Train	Test	Train	Test	Train	Test	
LightGBM	0.9999	0.9995	0.0072	0.0178	0.6983	1.4213	0.0047	0.0098	8.44
Decision Tree	1	0.9982	0.0007	0.0344	0.0129	2.5217	0	0.0178	6.42
Random Forest	0.9999	0.9992	0.0094	0.0232	0.6442	1.7469	0.0049	0.0124	58.24
XGBoost	0.9999	0.9997	0.0081	0.0149	0.8285	1.2754	0.0053	0.0081	196.23
Gradient Boosting	1	0.9998	0.0035	0.0126	0.275	0.7461	0.0018	0.0049	383.92
CatBoost	1	0.9997	0.0017	0.0137	0.1671	0.5858	0.0011	0.0036	410.69
NGBoost	0.9995	0.9992	0.0172	0.0224	1.59	2.5290	0.0101	0.0135	973.04
ANN [40]	0.991	0.993	0.0727	0.068	–	–	–	–	–

than XGBoost (196.23 s), and 45.5 × faster than Gradient Boosting (383.92 s). The CatBoost model took 410.69 s for training, making it approximately 48.7 × slower than LightGBM. The NGBoost model exhibited the longest training time among the models studied, requiring 973.04 s, making it approximately 115.3 × slower than the LightGBM model.

The performance of several machine learning models, including LightGBM, Decision Tree, Random Forest, XGBoost, Gradient Boosting, CatBoost, and NGBoost, has been comprehensively visualized using a radar chart (Fig. 9). The test data from Table 4 has been normalized to a range of 0 to 100 to enable a fair comparison of various performance criteria. The axes of the radar chart represent the performance indicators R^2, *RMSE*, *MAPE*, *MAE*, and model training time. Examining the radar chart allows for the identification of models that excel in specific areas, aiding in the selection of the most suitable model for a given situation. LightGBM is the optimal choice when considering model training speed and performance metrics. It achieves a high training speed and relatively higher metric scores. While the Decision Tree model trains slightly faster (6.42 s) than LightGBM (8.44 s.), it falls short in other key metrics like R^2 and *RMSE*. Specifically, LightGBM outperforms the Decision Tree in test set R^2 (0.9995 vs. 0.9982) and *RMSE* (0.0178 vs. 0.0344). Additionally, the Decision Tree has a higher Percent Error, making it less reliable for predicting critical structural design parameters like the fundamental period. In summary, LightGBM provides a well-balanced solution, offering competitive performance metrics while maintaining a fast training speed.

Considering the scope of this research, LightGBM demonstrates superiority over other models for several reasons. The primary research objectives of this study include developing a machine learning model with a faster training runtime, optimized hyperparameters through Bayesian Optimization, and a comparatively better prediction score for the fundamental period of infilled RC frame structures. LightGBM satisfies these objectives by achieving a high training speed and significantly higher metric scores compared to other models. LightGBM is based on the gradient boosting framework, which employs gradient-based optimization techniques to minimize the loss function. It addresses the challenges of slow training times and large memory consumption that can be associated with other gradient boosting algorithms. It employs a unique tree-growing strategy called 'Leaf-wise' that optimizes the tree structure by selecting the leaf with the highest delta loss to grow. This strategy results in a more balanced and efficient tree structure, contributing to enhanced performance compared to other algorithms.

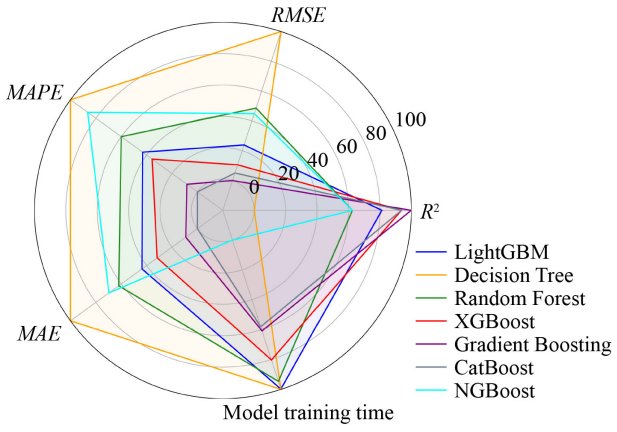


Fig. 9 Model performance comparison on evaluation metrics.

Furthermore, LightGBM's compatibility with Bayesian Optimization for hyperparameter tuning further enhances its efficiency. Bayesian Optimization reduces the number of iterations required to find optimal hyperparameters by leveraging a probabilistic model that incorporates prior knowledge about the function's behavior. This approach leads to more effective hyperparameter optimization and faster convergence, which is a significant advantage in the context of this study. In addition to its performance advantages, LightGBM's scalability and flexibility make it an excellent choice for the study's objective of creating a user-friendly, web-based dashboard for structural engineers. The algorithm's fast training speed, coupled with its accuracy and efficiency, ensures that the model remains accessible and practical for industry professionals. This, in turn, fosters collaboration and enhances the

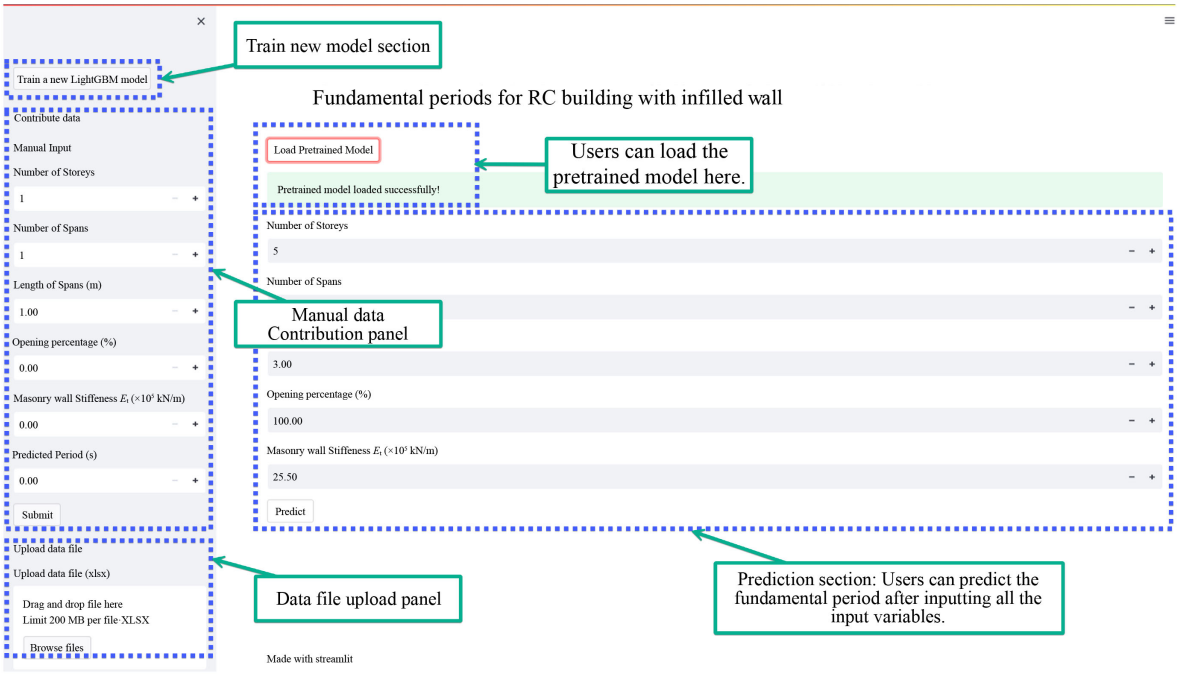
overall prediction capabilities of the machine learning model.

4.7 Dashboard creation

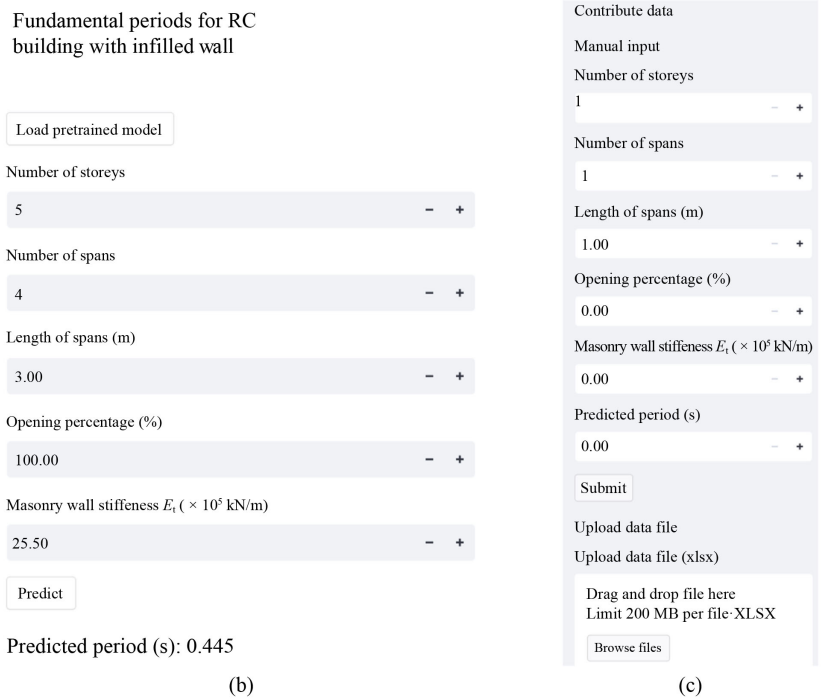
Subsequently, a user-friendly web-based dashboard is developed for use by working engineers, powered by the Streamlit framework. In Fig. 10, users can easily navigate the main home page (Fig. 10(a)), calculate the predicted period (Fig. 10(b)) for RC buildings with infilled walls, and contribute data to enhance the model's performance (Fig. 10(c)).

The dashboard comprises several sections.

i. Model training and loading: Users have the option to load a pre-trained model, which can be useful if they want to utilize the pre-existing LightGBM model without retraining. Alternatively, they can train a new LightGBM model using the provided data file. The training process involves splitting the data set into training and testing



(a)



(b) (c)

Fig. 10 Fundamental periods dashboard overview: (a) dashboard homepage; (b) predicted period (s) calculation; (c) data contribution.

sets, using Bayesian Optimization to search for the best hyperparameters, and then training the model on the entire training set.

ii. Data contribution: Engineers can contribute their data to the existing data set through two methods: manual input or file upload. For manual input, users can enter the relevant variables (Number of storeys, Number of spans, Length of spans, Opening percentage, and Masonry wall stiffness) and the predicted period in the sidebar. Once the data are submitted, it will be appended to the existing data set and saved in the data file. Users can also upload an Excel file (.xlsx format) containing their data.

iii. Prediction: Once a pre-trained model is loaded or a new model is trained, users can input the required variables (Number of storeys, Number of spans, Length of spans, Opening percentage, and Masonry wall stiffness) and click the 'Predict' button to obtain the predicted fundamental period.

The 'Workflow of the User Interaction Process on the Dashboard', depicted in Fig. 11, provides a schematic representation of the workflow that users will follow when interacting with the developed dashboard. It summarizes the options available to users, including loading a pre-trained model, training a new model, contributing data manually or through a file upload, and making predictions.

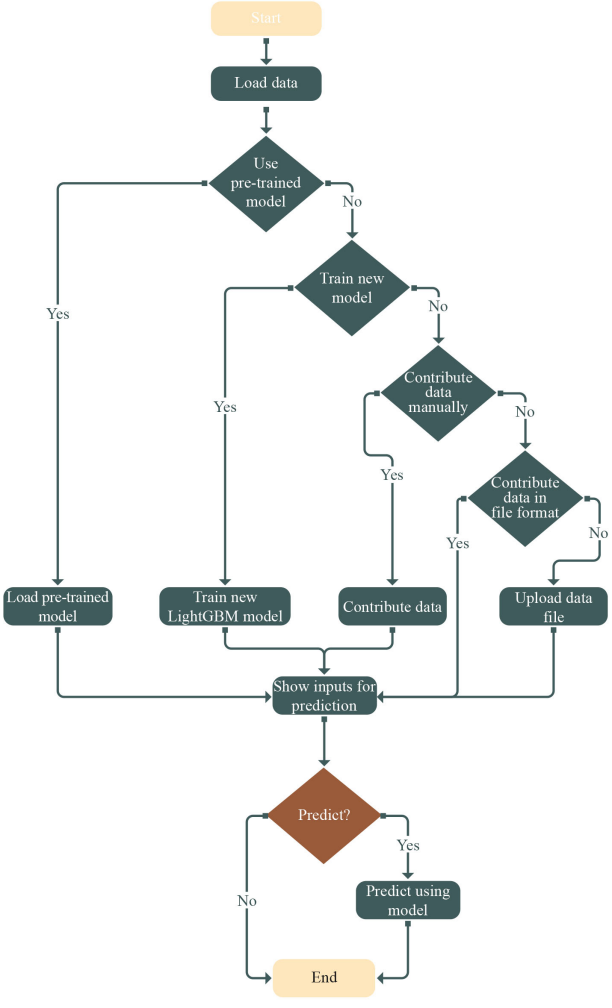


Fig. 11 Workflow of the user interaction process on the dashboard.

The dashboard's design follows the best practices of user experience, providing clear instructions and intuitive controls for engineers to input their data and obtain predictions with ease. This user-friendly interface fosters collaboration among engineers and researchers, ultimately enhancing the machine learning model's overall prediction capabilities for practical applications in the field.

5 Conclusions

Through analysis and comparative assessment, this study underscores the proficiency of the LightGBM model, showing its benefits relative to its machine learning contemporaries such as Decision Tree, Random Forest, XGBoost, Gradient Boosting, CatBoost and NGBoost. The salient strength of the LightGBM model, enhanced by Bayesian Optimization, resides not only in its relatively superior training quality (R^2: 0.9995) and minimal *RMSE* (0.0178), but also in its training speed - being approximately 23.2 and 45.5 times faster than XGBoost and Gradient Boosting, respectively. The findings reveal that, despite not achieving the pinnacle in every metric, the LightGBM model provides a balanced blend of time efficiency and predictive accuracy. The additional creation of a web-based dashboard, engineered via Streamlit, leverages the enhanced LightGBM model, furnishing users with a versatile platform that encompasses features such as utilization of pre-existing models or training of new ones, and notably provides swift fundamental period predictions through user-input variables. In essence, the amalgamation of Bayesian Optimization-augmented LightGBM with a highly intuitive web-based dashboard provides a significant pathway toward bridging machine learning and structural design, serving to fortify collaborative and accurate predictive capabilities in the sector. This research marks a notable advancement in integrating machine learning with structural design, offering new avenues for future research and the practical application in the industry.

Competing interests The authors declare that they have no competing interests.

References

1. Chiou Y J, Tzeng J C, Liou Y W. Experimental and analytical study of masonry infilled frames. Journal of Structural

Engineering, 1999, 125(10): 1109–1117
2. Colangelo F. Pseudo-dynamic seismic response of reinforced concrete frames infilled with non-structural brick masonry. Earthquake Engineering & Structural Dynamics, 2005, 34(10): 1219–1241
3. De Angelis A, Pecce M R. The structural identification of the infill walls contribution in the dynamic response of framed buildings. Structural Control and Health Monitoring, 2019, 26(9): e2405
4. Fardis M N, Panagiotakos T B. Seismic design and response of bare and masonry-infilled reinforced concrete buildings. Part II: Infilled structures. Journal of Earthquake Engineering, 1997, 1(3): 475–503
5. Gago A S, Alfaiate J, Lamas A. The effect of the infill in arched structures: Analytical and numerical modelling. Engineering Structures, 2011, 33(5): 1450–1458
6. Singh H, Paul D K, Sastry V V. Inelastic dynamic response of reinforced concrete infilled frames. Computers & Structures, 1998, 69(6): 685–693
7. Wang F, Zhao K, Zhang J, Yan K. Influence of different types of infill walls on the hysteretic performance of reinforced concrete frames. Buildings, 2021, 11(7): 310–328
8. Asteris P G, Tsaris A K, Cavaleri L, Repapis C C, Papalou A, Di Trapani F, Karypidis D F. Prediction of the fundamental period of infilled RC frame structures using artificial neural networks. Computational Intelligence and Neuroscience, 2016, 2016: 1–12
9. Asteris P G, Repapis C C, Repapi E V, Cavaleri L. Fundamental period of infilled reinforced concrete frame structures. Structure and Infrastructure Engineering, 2017, 13(7): 929–941
10. Asteris P G, Repapis C C, Cavaleri L, Sarhosis V, Athanasopoulou A. On the fundamental period of infilled RC frame buildings. Structural Engineering and Mechanics, 2015, 54(6): 1175–1200
11. Asteris P G, Repapis C C, Tsaris A K, Di Trapani F, Cavaleri L. Parameters affecting the fundamental period of infilled RC frame structures. Earthquakes and Structures, 2015, 9(5): 999–1028
12. Chethan K, Babu R, Venkataramana K, Sharma A. Influence of masonry infill on fundamental natural frequency of 2D RC frames. Journal of Structural Engineering, 2010, 37(2): 135–141
13. Jiang R, Jiang L, Hu Y, Ye J, Zhou L. A simplified method for estimating the fundamental period of masonry infilled reinforced concrete frames. Structural Engineering and Mechanics, 2020, 74(6): 821–832
14. Koçak A, Kalyoncuoğlu A, Zengin B. Effect of infill wall and wall openings on the fundamental period of RC buildings. Earthquake Resistant Engineering Structures IX, 2013, 132: 121–131
15. Kose M M. Parameters affecting the fundamental period of RC buildings with infill walls. Engineering Structures, 2009, 31(1): 93–102
16. Masi A, Vona M. Experimental and numerical evaluation of the fundamental period of undamaged and damaged RC framed buildings. Bulletin of Earthquake Engineering, 2010, 8(3): 643–656
17. Ricci P, Verderame G M, Manfredi G. Analytical investigation of elastic period of infilled RC MRF buildings. Engineering Structures, 2011, 33(2): 308–319
18. Dimiduk D M, Holm E A, Niezgoda S R. Perspectives on the impact of machine learning, deep learning, and artificial intelligence on materials, processes, and structures engineering. Integrating Materials and Manufacturing Innovation, 2018, 7(3): 157–172
19. Jasmine P H, Arun S. Machine learning applications in structural engineering—A review. IOP Conference Series: Materials Science and Engineering, 2021, 1114(1): 012012
20. Lee S, Ha J, Zokhirova M, Moon H, Lee J. Background information of deep learning for structural engineering. Archives of Computational Methods in Engineering, 2018, 25(1): 121–129
21. Salehi H, Burgueño R. Emerging artificial intelligence methods in structural engineering. Engineering Structures, 2018, 171: 170–189
22. Sun H, Burton H V, Huang H. Machine learning applications for building structural design and performance assessment: State-of-the-art review. Journal of Building Engineering, 2021, 33: 101816
23. Hamdia K M, Zhuang X, Rabczuk T. An efficient optimization approach for designing machine learning models based on genetic algorithm. Neural Computing & Applications, 2021, 33(6): 1923–1933
24. Nariman N A, Hamdia K, Ramadan A M, Sadaghian H. Optimum design of flexural strength and stiffness for reinforced concrete beams using machine learning. Applied Sciences, 2021, 11(18): 8762–8777
25. Guo H, Zhuang X, Alajlan N, Rabczuk T. Physics-informed deep learning for melting heat transfer analysis with model-based transfer learning. Computers & Mathematics with Applications, 2023, 143: 303–317
26. Guo H, Zhuang X, Chen P, Alajlan N, Rabczuk T. Stochastic deep collocation method based on neural architecture search and transfer learning for heterogeneous porous media. Engineering with Computers, 2022, 38(6): 5173–5198
27. Guo H, Zhuang X, Fu X, Zhu Y, Rabczuk T. Physics-informed deep learning for three-dimensional transient heat transfer analysis of functionally graded materials. Computational Mechanics, 2023, 72(3): 513–524
28. Guo H, Zhuang X, Rabczuk T. A deep collocation method for the bending analysis of Kirchhoff plate. Computers, Materials & Continua, 2019, 59(2): 433–456
29. Samaniego E, Anitescu C, Goswami S, Nguyen-Thanh V M, Guo H, Hamdia K, Zhuang X, Rabczuk T. An energy approach to the solution of partial differential equations in computational mechanics via machine learning: Concepts, implementation and applications. Computer Methods in Applied Mechanics and Engineering, 2020, 362: 112790
30. Zhuang X, Guo H, Alajlan N, Zhu H, Rabczuk T. Deep autoencoder based energy method for the bending, vibration, and buckling analysis of Kirchhoff plates with transfer learning. European Journal of Mechanics. A, Solids, 2021, 87: 104225
31. Sang-To T, Le-Minh H, Abdel Wahab M, Thanh C L. A new metaheuristic algorithm: Shrimp and Goby association search algorithm and its application for damage identification in large-scale and complex structures. Advances in Engineering Software, 2023, 176: 103363
32. Minh H L, Khatir S, Rao R V, Abdel Wahab M, Cuong-Le T. A variable velocity strategy particle swarm optimization algorithm (VVS-PSO) for damage assessment in structures. Engineering with Computers, 2023, 39(2): 1055–1084

33. Ho L V, Trinh T T, De Roeck G, Bui-Tien T, Nguyen-Ngoc L, Abdel Wahab M. An efficient stochastic-based coupled model for damage identification in plate structures. Engineering Failure Analysis, 2022, 131: 105866
34. Nghia-Nguyen T, Kikumoto M, Nguyen-Xuan H, Khatir S, Abdel Wahab M, Cuong-Le T. Optimization of artificial neutral networks architecture for predicting compression parameters using piezocone penetration test. Expert Systems with Applications, 2023, 223: 119832
35. Tran V T, Nguyen T K, Nguyen-Xuan H, Abdel Wahab M. Vibration and buckling optimization of functionally graded porous microplates using BCMO-ANN algorithm. Thin-walled Structures, 2023, 182: 110267
36. Asteris P G, Nikoo M. Artificial bee colony-based neural network for the prediction of the fundamental period of infilled frame structures. Neural Computing & Applications, 2019, 31(9): 4837–4847
37. Mirrashid M, Naderpour H. Computational intelligence-based models for estimating the fundamental period of infilled reinforced concrete frames. Journal of Building Engineering, 2022, 46: 103456
38. Latif I, Banerjee A, Surana M. Explainable machine learning aided optimization of masonry infilled reinforced concrete frames. Structures, 2022, 44: 1751–1766
39. Somala S N, Karthikeyan K, Mangalathu S. Time period estimation of masonry infilled RC frames using machine learning techniques. Structures, 2021, 34: 1560–1566
40. Charalampakis A E, Tsiatas G C, Kotsiantis S B. Machine learning and nonlinear models for the estimation of fundamental period of vibration of masonry infilled RC frame structures. Engineering Structures, 2020, 216: 110765
41. Bioud N, Laid I, Benbouras M A. Estimating the fundamental period of infilled RC frame structures via deep learning. Urbanism. Architecture. Constructions, 2023,14:1–22
42. Cakiroglu C, Bekdaş G, Kim S, Geem Z W. Explainable ensemble learning models for the rheological properties of self-compacting concrete. Sustainability, 2022, 14(21): 14640
43. Chakraborty D, Elhegazy H, Elzarka H, Gutierrez L. A novel construction cost prediction model using hybrid natural and light gradient boosting. Advanced Engineering Informatics, 2020, 46: 101201
44. Chun P, Izumi S, Yamane T. Automatic detection method of cracks from concrete surface imagery using two-step light gradient boosting machine. Computer-Aided Civil and Infrastructure Engineering, 2021, 36(1): 61–72
45. Kookalani S, Cheng B, Torres J L C. Structural performance assessment of GFRP elastic gridshells by machine learning interpretability methods. Frontiers of Structural and Civil Engineering, 2022, 16(10): 1249–1266
46. Mangalathu S, Jang H, Hwang S H, Jeon J S. Data-driven machine-learning-based seismic failure mode identification of reinforced concrete shear walls. Engineering Structures, 2020, 208: 110331
47. Naser M Z, Kodur V, Thai H T, Hawileh R, Abdalla J, Degtyarev V V. StructuresNet and FireNet: Benchmarking databases and machine learning algorithms in structural and fire engineering domains. Journal of Building Engineering, 2021, 44: 102977
48. Ding Z, Zhang W, Zhu D. Neural-network based wind pressure prediction for low-rise buildings with genetic algorithm and Bayesian optimization. Engineering Structures, 2022, 260: 114203
49. Lookman T, Alexander F, Rajan K. Information Science for Materials Discovery and Design. Switzerland: Springer, 2016
50. Mathern A, Steinholtz O S, Sjöberg A, Önnheim M, Ek K, Rempling R, Gustavsson E, Jirstrand M. Multi-objective constrained Bayesian optimization for structural design. Structural and Multidisciplinary Optimization, 2021, 63(2): 689–701
51. Sajedi S, Liang X. Deep generative Bayesian optimization for sensor placement in structural health monitoring. Computer-Aided Civil and Infrastructure Engineering, 2022, 37(9): 1109–1127
52. Zhang W, Wu C, Zhong H, Li Y, Wang L. Prediction of undrained shear strength using extreme gradient boosting and random forest based on Bayesian optimization. Geoscience Frontiers, 2021, 12(1): 469–477
53. Asteris P G. The FP4026 Research Database on the fundamental period of RC infilled frame structures. Data in Brief, 2016, 9: 704–709
54. Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q, Liu T. LightGBM: A highly efficient gradient boosting decision tree. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. New York: Curran Associates Inc., 2017, 3149–3157
55. Friedman J H. Greedy function approximation: A gradient boosting machine. Annals of Statistics, 2001, 29(5): 1189–1232
56. Brochu E, Cora V M, de Freitas N. A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning. 2010, arXiv: 1012.2599
57. Frazier P I. A Tutorial on Bayesian Optimization. 2018. arXiv: 1807.02811
58. Shahriari B, Swersky K, Wang Z, Adams R P, de Freitas N. Taking the human out of the loop: A review of Bayesian Optimization. proceedings of the IEEE, 2016, 104(1): 148–175
59. Rasmussen C E. Gaussian Processes in Machine Learning. In: Bousquet O, von Luxburg U, Rätsch G, eds. Advanced Lectures on Machine Learning. Berlin: Springer, 2004, 63–71
60. Snoek J, Larochelle H, Adams R P. Practical Bayesian optimization of machine learning algorithms. In: Proceedings of the 25th International Conference on Neural Information Processing Systems—Volume 2. New York: Curran Associates Inc., 2012, 2951–2959