Geomechanics and Engineering, *Vol. 37, No. 4 (2024) 395-418*
https://doi.org/10.12989/gae.2024.37.4.395

Assessment of maximum liquefaction distance using soft computing approaches

Kishan Kumar* , Pijush Samui and Shiva S. Choudhary

Department of Civil Engineering, National Institute of Technology Patna, Patna, Bihar 800005, India

(*Received May 25, 2023, Revised March 20, 2024, Accepted May 3, 2024*)

Abstract. The epicentral region of earthquakes is typically where liquefaction-related damage takes place. To determine the maximum distance, such as maximum epicentral distance (R_e), maximum fault distance (R_f), or maximum hypocentral distance (R_h), at which an earthquake can inflict damage, given its magnitude, this study, using a recently updated global liquefaction database, multiple ML models are built to predict the limiting distances (R_e, R_f, or R_h) required for an earthquake of a given magnitude to cause damage. Four machine learning models LSTM (Long Short-Term Memory), BiLSTM (Bidirectional Long Short-Term Memory), CNN (Convolutional Neural Network), and XGB (Extreme Gradient Boosting) are developed using the Python programming language. All four proposed ML models performed better than empirical models for limiting distance assessment. Among these models, the XGB model outperformed all the models. In order to determine how well the suggested models can predict limiting distances, a number of statistical parameters have been studied. To compare the accuracy of the proposed models, rank analysis, error matrix, and Taylor diagram have been developed. The ML models proposed in this paper are more robust than other current models and may be used to assess the minimal energy of a liquefaction disaster caused by an earthquake or to estimate the maximum distance of a liquefied site provided an earthquake in rapid disaster mapping.

Keywords: earthquake; epicentral distance; extreme gradient boosting; liquefaction; machine learning algorithms; regression

1. Introduction

Liquefaction evidence is perceived as a geological marker of paleoseismicity (seismites), because sites affected by previous liquefaction have the potential to liquefy again. However, existing research indicates that waves can also cause liquefaction on the seabed, leading to stratigraphic reorganization (Xu *et al.* 2016). Hence, the differentiation between earthquake-induced liquefaction and wave-induced liquefaction hinges on the depth of liquefaction. Earthquakes typically induce liquefaction at greater depths compared to the shallower depths associated with waves as indicated in the field observations in the silty soil seabed of the Yellow River Delta which revealed liquefaction depths as shallow as 3.65 m (Xu *et al.* 2021). One of the most important challenges in understanding earthquake-induced liquefaction of soil is determining the maximum distance at which such an event can occur. Earthquakes have caused several large-scale liquefaction events worldwide, and many studies have shown a strong correlation between earthquake parameters and the maximum epicentral (or fault) distance from liquefaction sites. These studies are extremely useful for engineers and urban planners conducting rapid disaster assessments due to seismic liquefaction and mitigating seismic risk (Ardeshiri-Lajimi *et al.* 2016, Pirrotta *et al*. 2009, Papadopoulos and Lefkopoulos 1993, Talwani and Cox 1985).

As depicted in Fig. 1, there are typically three types of distance between the seismic source and the furthest liquefaction location. Maximum hypocentral distance (R_h) and fault distance (R_f) are the minimum distances between the furthest point of liquefaction and the fault surface. R_e is the epicentral distance measured in kilometres from the epicentre of the earthquake to the furthest point where there is unambiguous evidence of ground failure caused by liquefaction (Kuribayashi and Tatsuoka 1975).

Table 1 shows many empirical relationships (called M-D relations) between earthquake magnitude and maximum epicentral distance of liquefaction locations based on regional or global liquefaction database (Pirrotta *et al.* 2009, Galli 2000, Gazetas and Botsis 1981, Hu and Liu 2019, Papathanassiou *et al*. 2005). However, just a few of them examined global liquefaction data, notably (Ambraseys, 1988, Hu 2022a, Papadopoulos and Lefkopoulos 1993).

Many studies have been carried out to determine the empirical relationship between earthquake source parameters, such as magnitude, intensity, etc., and the maximum epicentral (or fault distance) distance of liquefaction sites at regional and global scales. These relationships are useful for geotechnical purposes (e.g., microzonation studies) and hazard assessment at regional scale. They are also useful for seismic purposes i.e., estimating the minimum energy an earthquake can produce to induce liquefaction and the minimum magnitude of paleo-earthquake events that have induced liquefaction at the site in question, as well as determining the likely mesoseismic zone. Kuribayashi established a correlation

*Corresponding author, Ph.D. Student
E-mail: kishank.phd22.ce@nitp.ac.in

ISSN: 2005-307X (Print), 2092-6219 (Online)

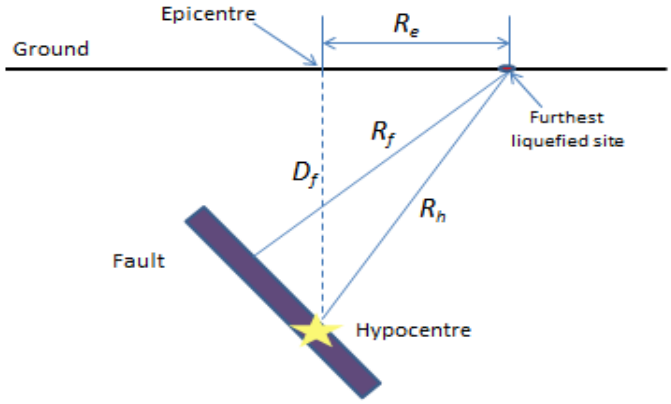


Fig. 1 Illustration of maximum source-to-site, epicentral, and fault distances

between the greatest epicentral distances of liquefaction sites and the associated magnitude of major earthquakes in Japan (Kuribayashi and Tatsuoka 1975). In order to calculate the connections between moment-magnitude and distance for 137 liquefaction episodes dispersed throughout the world, Ambraseys took into account both epicentral and fault distance(Ambraseys 1988). When Papadopoulos and Lefkopoulos (1993) added their Greek data and other liquefaction observations made in various parts of the world, they were able to produce a boundary equation that revisited the global curves Ambraseys had proposed (Papadopoulos and Lefkopoulos 1993). Galli (2000) recently used the magnitude upper bound method to study historical liquefactions brought on by 61 earthquakes that occurred in Italy between 1117 and 1990. The above-mentioned studies were carried out on dataset before the year 2003. However, many cases of seismic liquefaction have took place throughout the world after year 2003, including the Colima earthquake, Bachu earthquake, Canterbury earthquake, Wenchuan, Emilia, Songyuan, Tohoku, Samos island, Muzaffarabad earthquake etc. (Alessio *et al.* 2013, Cetin *et al*. 2022, Hu 2021, Hu and Liu 2019b, Jiang *et al.* 2019, Wang *et al.* 2006). This new dataset (Hu 2022b) substantially expanded the size and scope of the previously existing liquefaction case history database. In addition, all previous models were created using the frequentist ordinary least-squares (OLS) method in regression analysis, which excludes the impact of parameter uncertainty. A new empirical relationship between earthquake magnitude and maximum distance considering parameter uncertainty effect was proposed based on updated database (Hu 2022a, b).

In recent years, artificial intelligence (AI) techniques have advanced rapidly. The development of various engineering research subjects has been greatly boosted by machine learning (ML) technology in particular (Armaghani *et al.* 2022, Bai *et al.* 2021, Javadi and Rezania 2009, Kamran *et al*. 2022, Kumar and Samui 2022, Liu *et al.* 2021). The use of ML techniques along with integration of physical models such as energy-based approach can significantly reduce computational cost and time. The integration of artificial neural networks (ANNs) and adaptive collocation demonstrated the potential in solving partial differential equations (PDEs). This integration presents a promising opportunity to attain precise solutions while minimizing computational costs, particularly in situations when solutions are not smooth (Anitescu *et al.* 2019).Similarly, the use of Deep Neural Networks (DNNs) for approximating solutions to PDEs by leveraging the flexibility and efficiency of DNNs in function approximation by adopting a collocation strategy and, utilizing the energy of mechanical systems as the natural loss function for machine learning. By adopting a collocation strategy and focusing on mechanical problems, the method is validated through various engineering applications, utilizing the energy of mechanical systems as the natural loss function for ML approaches (Samaniego *et al.* 2020). In order to predict soil liquefaction, several researchers have used and developed AI and ML algorithms (Hu and Liu 2019, Kumar *et al.* 2023, Ochoa *et al.* 2018, Samui, 2007). For predicting the potential for soil liquefaction, technologies based on AI and machine learning frequently achieve more accuracy than conventional prediction techniques (Hoang and Bui 2018, Lee and Chern 2013). Several empirical models are offered in the literature review to estimate the maximum distance for liquefaction. Nevertheless, the utilization of ensemble and deep learning-based machine learning models remains somewhat constrained. There is a lack of a comparative analysis of deep learning-based models and the widely used ensemble model (XGB) in predictions of R_e, R_f, and R_h. This study utilizes a dataset that has been recently updated to include 290 earthquake data points up until the year 2020. When selecting a machine learning (ML) model, several factors need to be considered, despite the fact that neural network-based models sometimes incur higher computing expenses compared to standard machine learning methods. Given that this study pertains to sequential or organized data and necessitates the capturing of intricate patterns, the advantages offered by neural networks may

Table 1 List of previous M-D relation

Empirical equations	Date	Region	Sample size	References
$log\ R_e = 0.87M_w - 4.5\ \ for\ M_w > 5.2$ $log\ R_e = 0.77M_w - 3.6\ \ for\ M_w > 6.0$	1872-1968	Japan	44	(Kuribayashi and Tatsuoka 1975)
$M_w = -0.31 + 2.65 \text{ x } 10^{-3}R_e + 4.95\ log\ R_e$ $M_w = 0.18 + 9.2 \text{ x } 10^{-3}R_f + 4.5\ log\ R_f$	1848-1980	worldwide	137	(Ambraseys 1988)
$M_w = -0.44 + 3 \text{ x } 10^{-3}R_e + 4.9\ log\ R_e$ $M_w = -2.5 \text{ x } 10^{-3} + 9.25 \text{ x } 10^{-3}R_f + 4.5\ log\ R_f$	1976-1988	worldwide	30	(Papadopoulos and Lefkopoulos 1993)
$M_s = 1.0 + 3.0\ log\ R_e\ for\ data\ before\ 1990$ $M_s = 1.5 + 3.1\ log\ R_e\ for\ data\ after\ 1990$	1117-1990	Italy	56	(Galli 2000)
$M_w = 2.67 + 0.98\ ln\ R_e$ $M_s = 1.85 + 1.16\ ln\ R_e$	1169-1990	Central-eastern Sicily	14	(Pirrotta *et al.* 2009)
$R_h = 36M_s - 160\ (upper\ bound)$ $R_h = 36M_s - 200\ (mean)$ $R_h = 36M_s - 240\ (lower\ bound)$	Until 1998	Turkey	–	(Aydan *et al.* 2000)
$For\ 5.5 \leq M_s \leq 7.6:$ $M_s = 4.742 + 4.655 \text{ x } 10^{-3}R_e - 0.8907\ log\ R_e$ $For\ 5.5 \leq M_s \leq 7.1:$ $M_s = 5.224 + 7.34 \text{ x } 10^{-3}R_f - 0.488\ log\ R_f$	1509-2003	Broader Aegean	88	(Papathanassiou *et al.* 2005)

M_w, M_s is the moment magnitude scale & surface-wave magnitude scale respectively

surpass the associated processing expenses.

In addition, XGB models demonstrate high performance when applied to tabular data. Consequently, this study proposes three machine learning models based on deep learning and the XGB model.

The present study proposes four machine learning (ML) models namely Long Short-Term Memory (LSTM), Bidirectional Long Short-Term Memory (BiLSTM), Convolutional Neural Network (CNN) and Extreme gradient boosting (XGB) to predict the maximum distance such as maximum epicentral distance (R_e), maximum fault distance (R_f), or maximum hypocentral distance (R_h) given the magnitude of an earthquake using the updated liquefaction database.

Thus, this study aims to (1) predict maximum distances R_e, R_f and R_h using ML algorithms (2) compare the ML models using various performance indicators such as coefficient of determination (R^2), root mean square error (RMSE), mean absolute error (MAE) etc. to verify their effectivity (3) compare the performance of best ML model with empirical method on the updated database.

2. Data preparation

An updated database of the world liquefaction data has been used in this study (Hu 2022b). The updated database contains 290 cases from various historical earthquakes between years 1117-2020. In the present study, 290 data containing magnitude of earthquake and maximum epicentral distance (R_e) has been used. The data is divided into training and testing in the proportion of 70:30 i.e., 203 for training and 87 for testing. For (R_f) prediction, 128 cases have been used and split as 90 data for training and 38 for testing. For (R_h) prediction, 205 cases have been used and the data is split as 144 data for training and 61 for testing.

3. Methodology

This section provides an in-depth summary of the research methodology employed in this study for predicting the maximum distance (R_e, R_f, or R_h) given the magnitude of an earthquake using machine learning algorithms and estimation of the performance of these models in predicting the output. Fig. 2 below shows the methodology flow chart for implementing the ML models

3.1 Data pre-processing

To construct a model using soft computing, the dataset is split into two parts: the training set and the testing set. The model is trained on one set of data (the training set) and evaluated on another set (the testing set) (Dibike *et al.* 2001, Samui and Sitharam 2008). As a first step in investigating data, variables are pre-processed by being transformed into a more acceptable scale. By eliminating the dimensionality of the parameters, normalization ensures that the input variables have the same range of values. (Goh and Goh 2007). In this analysis, input and output variables are normalized to the interval [0, 1] on dividing by their maximum values using Eq. (1).

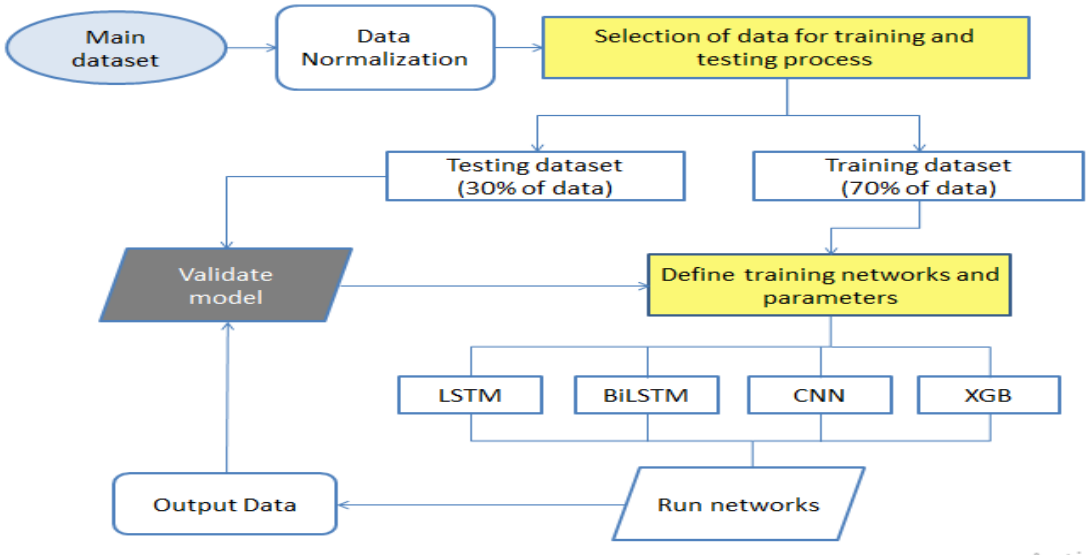


Fig. 2 Methodology flow chart for ML models

$$y = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \quad (1)$$

Where, y represents the scaled input and output variables, x represents the actual values of the variables, x_{max} and x_{min} represents the extremes of the variables.

3.2 Soft computing algorithms

In this study, four ML models namely, LSTM, BiLSTM, CNN, and XGB have been constructed to predict the maximum distance R_e, R_f and R_h. The proposed machine learning models have striking capabilities for learning and estimating the maximum distance associated with earthquake magnitude. The following is a detailed description of these ML models.

3.2.1 Long Short-Term Memory (LSTM) algorithm

LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) architecture that was introduced in 1997 by Hochreiter and Schmidhuber (Hochreiter and Schmidhuber 1997). LSTMs are designed to address the issue of the vanishing gradient that occurs in conventional RNNs, which makes it challenging to learn long-term dependencies in sequential data. The network is made up of many layered STM cells. Three gates—the input, forget, and output gates—are found in every LSTM cell and are used to control the information flow inside the cell. The input gate regulates the quantity of new data that can enter the cell. The forget gate determines which internal cell state data should be discarded. Last but not least, the output gate regulates how much current memory the cell should use to produce the output.

The number of LSTM layers, number of LSTM units, learning rate, dropout rate, and activation functions are the important parameters in LSTM architecture. The model can capture more complicated temporal correlations when there are more LSTM layers in it, but doing so also raises the risk of overfitting and increases computational cost, particularly when dealing with small datasets. The ability of the model to learn and represent the data is determined by the quantity of LSTM units (or neurons) in each layer. Although a larger number of units makes it possible for the model to identify more complex patterns in the data, it also raises the possibility of overfitting, particularly when dealing with small datasets. During the optimization process, the step size is governed by the learning rate. While a lower learning rate may result in slower convergence but more stable training, a greater learning rate may lead to faster convergence but can also induce instability and divergence. Dropout is a regularization strategy that involves randomly removing a portion of the LSTM units during training in order to prevent overfitting. The likelihood of dropping units is determined by the dropout rate; larger dropout rates provide stronger regularization but may also slow down the training process. Each LSTM unit's output is determined by its activation function, which is also essential to the model's capacity to identify non-linear relationships in the data. LSTM networks frequently use relu, tanh, and sigmoid as activation functions.

The model is built by defining the optimizer, the loss function, and any performance indicators to be monitored during training after the model architecture has been established. The LSTM model must next be trained using the prepared data. In order to minimize the loss function, this entails feeding the data into the model and updating the model parameters using backpropagation through time (BPTT). After training the model, evaluation of its performance is carried on a validation or test set. If the performance of the model is not satisfactory, model architecture or hyperparameters are adjusted and the model is retrained until the desired level of performance is achieved. Fig. 3 shows the architecture of LSTM model.

3.2.2 Bidirectional Long Short-Term Memory (LSTM) algorithm

The methodology for implementing a BiLSTM

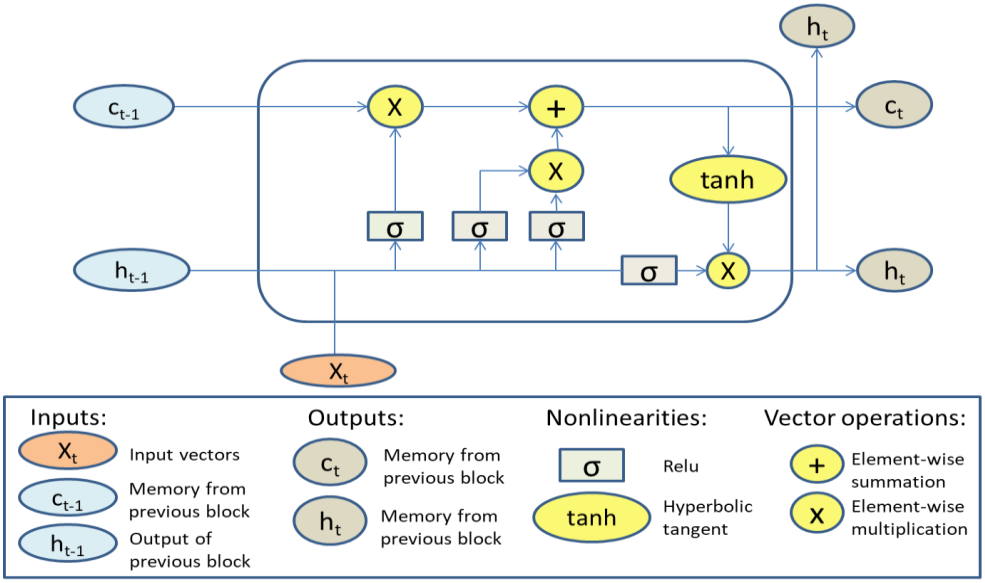


Fig. 3 LSTM architecture

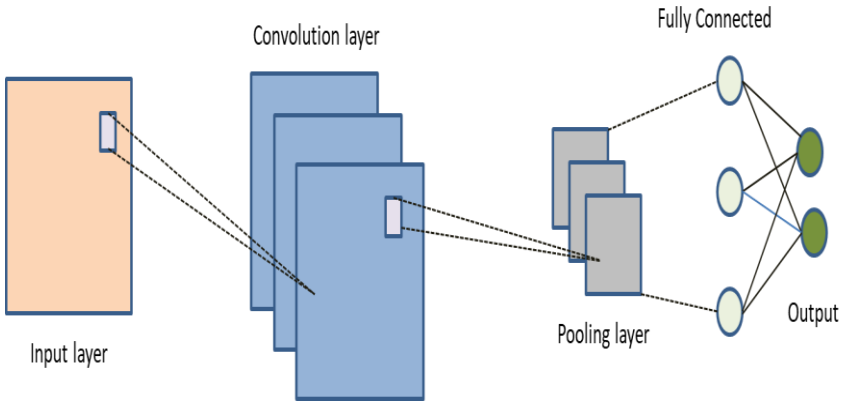


Fig. 4 Illustration of CNN architecture

(Bidirectional Long Short-Term Memory) model is similar to that of a standard LSTM model. Two distinct LSTM layers analyse the input sequence both forwards and backwards in a BiLSTM model, enabling the model to capture dependencies in both directions (Schuster and Paliwal 1997).

In this architecture, the input sequence is initially processed by two distinct LSTM layers, one in the forward direction and one in the reverse direction. Before generating the final output, the outputs of the two LSTM layers are concatenated and passed through a dense layer. Using a forward and backward LSTM layer permits the model to incorporate dependencies in the past and future contexts of the input sequence. The accuracy and generalization capability of BiLSTM model depends on several parameters similar to LSTM model such as number of hidden layers, number of neurons, learning rate, activation functions and choice of optimizer. The dimensionality of the hidden states in both forward and backward LSTM layers determine the memory capacity of the BiLSTM. Increased dimensions provide the ability to capture intricate patterns, but they may necessitate additional computing power. The metric known as batch size is responsible for determining the quantity of samples that are processed during each iteration of training. Increasing the batch sizes can result in quicker convergence, but at the cost of increased memory requirements.

The choice of optimizer (e.g., Adam, RMSprop, SGD) influence the optimization process and convergence speed. BiLSTM model can effectively capture information from both past and future data to make accurate predictions on sequential data.

3.2.3 Convolutional Neural Network (CNN) algorithm

CNNs are a key component of deep learning and are particularly useful for image data analysis. Two-dimensional arrays, such as those found in images, are ideal for use with CNN models. But we can also use CNN for regression analysis. The input data is transformed using a one-dimensional convolutional network in this investigation. The Conv1D class in Keras makes it possible to implement a one-dimensional convolutional layer (LeCun and Bengio 1995). It involves data preparation, defining the model and fitting and predicting and visualizing the output. CNN architecture is built by stacking convolutional, pooling, and fully-connected layers. The key components of 1D CNN include convolutional layers, pooling layers, activation functions and fully connected layers.

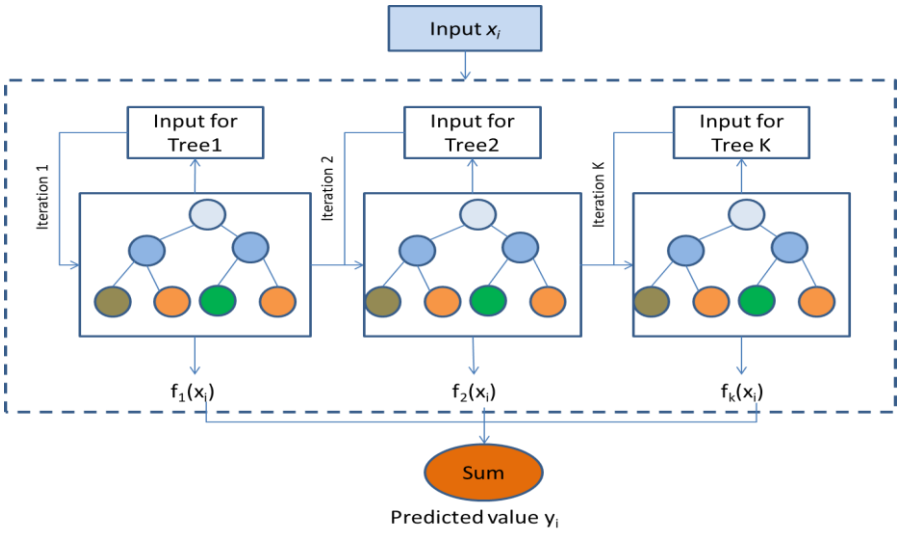


Fig. 5 XGB architecture

Convolutional Layers slide over the input sequence, extracting local features using filters (kernels). Pooling Layers down sample the features to reduce the dimensionality. The Activation Functions introduce non-linearity (e.g., ReLU, sigmoid) and finally the fully Connected Layers combine features for final predictions. Overfitting during training can also be avoided due to the inclusion of dropout layers. Fig. 4 below illustrates the CNN architecture.

3.2.4 Extreme gradient boosting (XGB) algorithm

Extreme Gradient Boosting (XGB) is an ensemble learning technique based on decision trees that integrates a number of ineffective predictive models to create a robust model. It is a member of the family of iterative gradient boosting techniques, which train a series of weak models and gradually combine them into a complete ensemble model (Chen and Guestrin 2016). XGB iteratively adds trees to correct the errors of the previous ones. The objective function defines the loss to be minimized during training and gradient boosting updates weights based on gradients of the loss function. The algorithm fits a decision tree to the training data and uses gradient descent to minimize the loss function, which is often the log loss for classification tasks and the root mean squared error (RMSE) for regression problems.

The highly flexible XGB algorithm's several hyperparameters can be changed to improve the model's effectiveness. Among the crucial hyperparameters are the number of trees (n_estimators), learning rate, minimum child weight (min_child_weight), maximum depth of each tree (max_depth), column subsample (colsample_bytree), and subsample. While more trees might result in better performance, there is a greater chance of overfitting. More intricate patterns in the data can be captured by deeper trees (max_depth), although overfitting is more likely with them. The contribution of each tree to the final prediction is controlled by the learning rate. Reduced learning rates can assist avoid overfitting by resulting in more cautious updates. The portion of training data that will be utilized to grow each tree is called a subsample. By adding randomness to the training process, setting it to a value less than 1 can aid in preventing overfitting. The percentage of features to be randomly sampled for each tree is indicated by column subsampling. In addition to adding more uncertainty to the model, it can aid in avoiding overfitting. The minimal sum of class weights (hessian) required in a child node is determined by the minimum child weight (min_child_weight) option. By the performance of XGB models for specific tasks and datasets, leading to improved results in terms of prediction accuracy and generalization capability. Fig. 5 below shows the XGB architecture.

3.3 Hyperparameter tuning and model building

The best hyperparameter settings for every model must be selected in order to evaluate a ML model's prediction power with less bias. Optimizing the performance of ML algorithms requires fine-tuning their hyperparameters (Bengio 2000). A set of hyperparameters matching the model with the lowest value of a predefined loss function is obtained through hyperparameter tuning. Numerous hyperparameters and configuration options are available for ML algorithms. With huge sets of hyperparameters, hyperparameter tuning thus quickly encounters the curse of dimensionality problem. Generally, it is not feasible to do a comprehensive optimization, therefore we must strike a compromise between the desired performance, computing time, and available resources. To handle this trade-off, a number of hyperparameter tuning techniques have been developed including Grid Search, Random Search, Hyperband, Bayesian Optimization, and Successive Halving.

Grid Search exhaustively explores all possible combinations of pre-defined parameters to find the best combination, ensuring the global optimum, but may become impractical with high-dimensional scenarios due to the curse of dimensionality. Random search is simpler than grid search and explores a fixed number of random parameter combinations, offering ease of parallelization and universality across problem dimensions. While unable to ensure optimal solutions like grid search, its flexibility and randomness, as demonstrated by (Bergstra and Bengio 2012), render it superior in certain

scenarios. Bayesian Optimization optimizes unknown objective functions by updating beliefs with function evaluations, constructing an acquisition function to guide the search for optimal query points.

It balances exploration-exploitation trade-off by updating priors with samples iteratively. In this study, we use random search for hyperparameter optimisation. Table 2 below presents the result of hyperparameters used in this study for each ML models.

The selection of surrogate models is crucial for hyperparameter optimization because these models serve as proxies for the true objective function, which is often expensive or impractical to evaluate directly. Surrogate models learn the underlying relationship between hyperparameters and performance based on observed data, enabling efficient exploration of the hyperparameter space. The importance of surrogate models lies in their ability to balance exploration and exploitation, guiding the optimization process towards promising regions of the hyperparameter space while avoiding wasteful evaluations of unpromising configurations. Different surrogate models have distinct strengths and weaknesses in terms of accuracy, scalability, and robustness, making their selection critical for achieving efficient and effective hyperparameter optimization.

Furthermore, the choice of surrogate model can significantly impact the optimization process's convergence speed and final performance. Models like Gaussian processes offer probabilistic predictions and uncertainty estimates, aiding in robust decision-making, while techniques like random forests provide scalability and versatility. Decision Trees models are often fast at training and predicting data because to their simplicity. The model is flexible, easy to use, and requires minimum data preparation. However, they often fail to generalize data and overfit. Stochastic Gradient Descent (SGD) is a widely used optimization algorithm in machine learning, particularly in training deep neural network. Its efficiency and performance advantage lies in low training and prediction times, making it suitable for regression tasks with large datasets, yet it's limited by linearity and sensitivity to outliers (Bottou 2010). Optimization approaches such as Genetic Algorithms (GA) and Particle Swarm Optimization (PSO), are widely used in several investigations due to their capacity to effectively identify the global optimum within complex regions (Alibrahim and Ludwig 2021, Guo *et al*. 2008). Therefore, selecting an appropriate surrogate model that aligns with the characteristics of the optimization problem and computational constraints is essential for successful hyperparameter optimization.

A resampling-based method for estimating a model's prediction performance is called cross-validation (cv) (James *et al.* 2013). It is a technique aimed at improving model validation by providing a more accurate estimate of model prediction performance. The fundamental concept of cv is to apply a user-defined number of partitions to divide an existing dataset into training and test sets (Fig. 6). The dataset is first split up into k folds, or partitions. The test set comprises the remaining partition, while the training set comprises k−1 partitions. On the training set, the model is trained, and on the test partition, it is assessed. A repeat is made by carrying out k iterations, where a model is trained on the training set and assessed on the test set each time. A test set is used once for each partition. Moreover, in the case of multiple k-fold cv, standard cross validation is repeated several times and the results are aggregated. For instance, the 3x10-fold cv aggregates the overall results after three rounds of cross validation. k-fold cv with repetitions ensures that every data point is used for validation exactly once across all repetitions. Typical values used for k are 3, 5 or 10. The bias of the estimate reduces as the number of folds advances, indicating that the performance estimate accurately reflects the method's actual performance. Therefore, this study uses 5-fold cv with three repetitions is in ML model's performance evaluation.

Overfitting is a common challenge in ML models. It arises when the model picks up noise or unimportant information in the training data rather than the underlying pattern, leading to

Table 2 Hyperparameters used in ML models

Hyperparameters	XGB	LSTM	BiLSTM	CNN
subsample	0.9			
n_estimators	75			
min_child_weight	3	-	-	-
max_depth	2			
colsample_bytree	0.9	-	-	-
learning_rate	0.1	0.01	0.01	0.001
Dropout_rate	-	0.2	0.2	0.2
Hidden layers	-	2	3	2
Metrics	rmse	mse	mse	mse
Optimiser	Gradient descent	Adam	Adam	Adam
Activation functions	-	relu and linear	relu and linear	relu and linear
Batch size	-	16	16	16

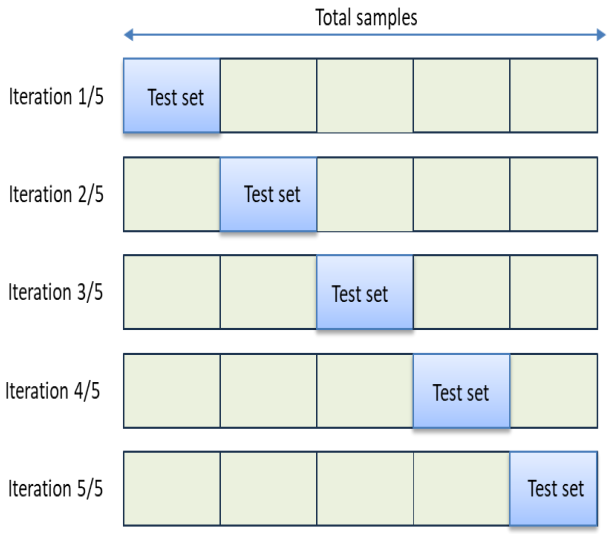


Fig. 6 Illustration of 5-fold cross validation

an unduly complex model that does well on the training set but poorly on the test set. At that point, the model is considered "overfitted" since it fits the training set too closely. A model should ideally have the same test and training accuracy based on its equal performance on both seen and unseen data. The model's nature or an excessively lengthy training period can cause overfitting. Overfitting can be reduced by a variety of strategies, including proper hyperparameter tuning, ensemble approaches, regularization techniques such as dropout layers, and early stopping. Dropout acts as a form of regularization by effectively reducing the capacity of the network during training. By randomly dropping out units, dropout limits the capacity of the network to memorize noise or irrelevant patterns in the training data, thereby preventing overfitting. Early stopping is a technique in machine learning that prevents overfitting by terminating iterative training methods before the model becomes too complicated. Early stopping stops training when the performance on a validation set starts to degrade, preventing the model from continuing to fit the noise in the training data. Often, halting training at the first sign of stagnation is not the best choice because the model may deteriorate before improving considerably. To prevent this, the endpoint is postponed by setting the 'patience' parameter to the number of epochs when no improvement is regarded as acceptable. For example, with 'patience' set to 5, training will halt if the validation loss does not improve after five consecutive epochs. In this study, these techniques have been applied in order to prevent overfitting of the models.

3.4 Statistical performance parameters

In this study, eight significant performance indices have been employed for the evaluation and comparison of observed and projected values: Global performance index (GPI) (Pal and Deswal 2008), coefficient of determination (R^2), root mean square error (RMSE), mean absolute error (MAE) (Kumar *et al.* 2023, Mahmoodzadeh *et al.* 2022), mean biased error (MBE), median absolute deviation (MAD), weighted mean absolute percentage error (WMAPE), and expanded uncertainty (U_{95}) (Adarsh *et al.* 2012). These parameters can be represented by the following mathematical Eqs. (2)-(9)

$$R^2 = \frac{\sum_{k=1}^{n}(d_k - d_{avg})^2 - \sum_{i=1}^{n}(d_k - y_k)^2}{\sum_{k=1}^{n}(d_k - d_{avg})^2} \quad (2)$$

$$RMSE = \sqrt{\frac{\sum_{k=1}^{n}(d_k - y_k)^2}{N}} \quad (3)$$

$$MAE = \frac{\sum_{k=1}^{n}|(y_k - d_k)|}{N} \quad (4)$$

$$MBE = \frac{\sum_{k=1}^{n}(y_k - d_k)}{N} \quad (5)$$

$$MAD = median(|y_1 - d_1|, |y_2 - d_2|, \ldots\ldots\ldots |y_n - d_n|,) \quad (6)$$

$$WMAPE = \frac{\sum_{k=1}^{n}\left|\frac{d_k - y_k}{d_k}\right| \times d_k}{\sum_{k=1}^{n} d_k} \quad (7)$$

$$U_{95} = 1.96\left(RMSE^2 + SD^2\right)^{1/2} \quad (8)$$

$$GPI = RMSE \times MBE \times U_{95} \times t_{stat} \times \left(1 - R^2\right) \quad (9)$$

Where d_k = actual value of kth sample points, y_k = predicted value of kth sample points, d_{avg}= mean of the actual value, N = number of the data sample. SD is the standard deviance of the change between the projected and actual data, 1.96 is the confidence level coverage factor, and U_{95} in Eq. (8) represents uncertainty up to a 95% confidence level. The mathematical relationship between the five statistical elements as presented in Eq. (9) is known as the GPI. The GPI value of a model determines how accurate it is; more accurate models are those with higher GPI values.

3.5 Empirical M-D expressions for best model performance comparison parameters

Three empirical M-D Eqs. (10)-(12) proposed by (Hu, 2022a)are used in this study for performance comparison of best ML model with the result obtained using empirical relation in terms of performance parameters such as R^2, MAE and RMSE.

$$M_w = 5.243 + 2.742 \times 10^{-3} R_e + 0.8669 \times logR_e \quad (10)$$

$$M_w = 5.9391 + 13.818 \times 10^{-3} R_f + 0.2807 \times logR_f \quad (11)$$

$$M_w = 4.7073 + 2.11 \times 10^{-3} R_h + 0.8669 \times logR_h \quad (12)$$

4. Results and discussion

The results of the M-D analysis, which used four ML models and three empirical relations on updated worldwide liquefaction data, are shown below. A model regression plot, rank analysis, error matrix, and Taylor's diagram are used to analyze the performance of the ML models in all M-D analyses. Performance parameters like as R^2, RMSE, and MAE are calculated, and a comparison of the fitting degree and errors of three M-D expressions with ML models is provided.

4.1 Models' regression plots for M_w-R_e, M_w-R_f and M_w-R_h

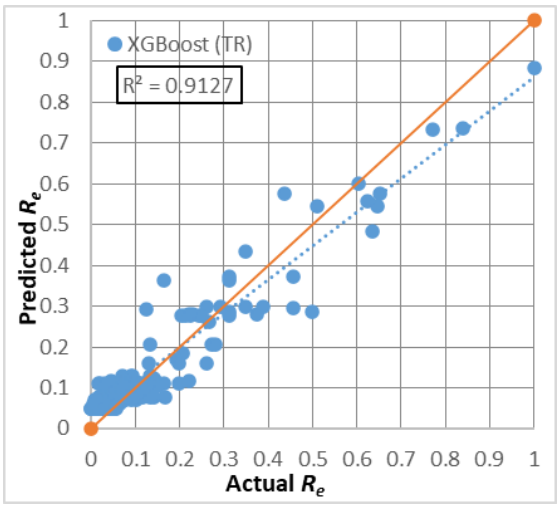
Four machine learning models have been constructed during this study for predicting the maximum distance (R_e, R_f and R_h). Figs. 7(a) and 7(b) presents scatter plots illustrating the relationship between the observed value and the predicted value of the maximal epicentral distance (R_e) derived from the XGB model. The table below and scatter plot indicate that the XGB model achieves the highest R^2 value. Thus, out of all the machine learning algorithms included in the current study,

Table 3 Performance parameters of ML techniques in predicting the R_e for training case

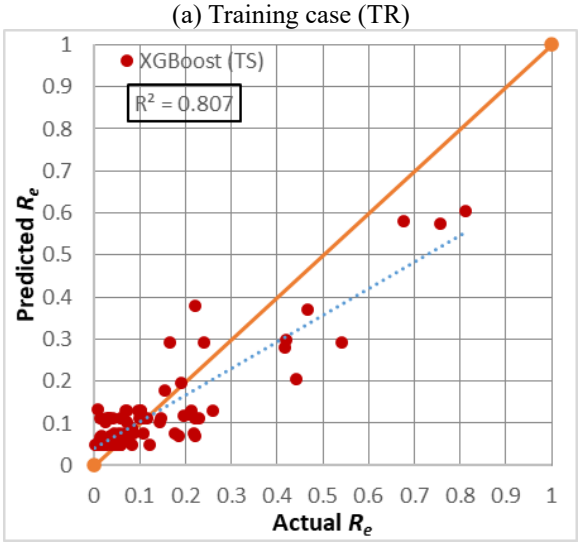
Models	R^2	RMSE	MAE	MBE	MAD	WMAPE	U_{95}	GPI
XGB	0.913	0.052	0.040	0.015	0.035	0.340	0.142	3.94E-05
BiLSTM	0.852	0.063	0.042	0.010	0.023	0.355	0.175	3.85E-05
LSTM	0.858	0.061	0.039	-0.005	0.020	0.332	0.170	-7.79E-06
CNN	0.847	0.070	0.045	-0.016	0.027	0.382	0.191	-1.12E-04

Table 4 Performance parameters of ML techniques in predicting the R_e for testing case

Models	R^2	RMSE	MAE	MBE	MAD	WMAPE	U_{95}	GPI
XGB	0.807	0.078	0.057	-0.005	0.038	0.443	0.217	-1.16E-05
BiLSTM	0.764	0.083	0.055	-0.009	0.027	0.441	0.230	-4.1E-05
LSTM	0.795	0.082	0.055	-0.025	0.028	0.440	0.224	-0.00028
CNN	0.787	0.092	0.060	-0.034	0.030	0.484	0.248	-5.98E-04

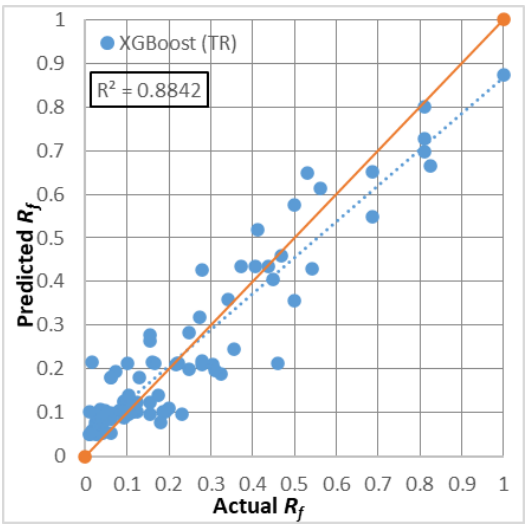


(a) Training case (TR)

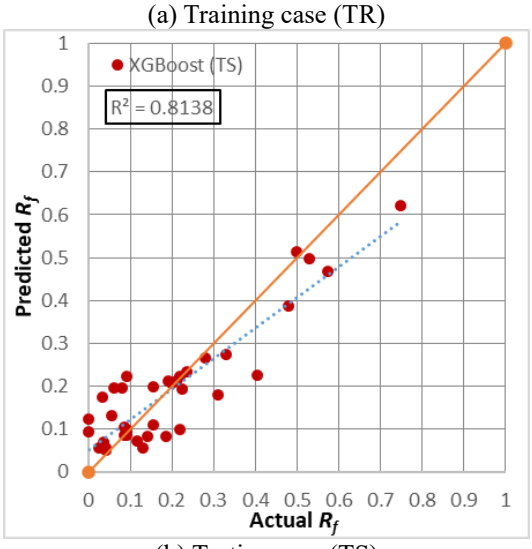


(b) Testing case (TS)

Fig. 7 XGB model actual and predicted R_e trend



(a) Training case (TR)



(b) Testing case (TS)

Fig. 8 XGB model actual and predicted R_f trend

XGB has been demonstrated to be the most dependable and effective. Tables 3 and 4 below compares the various performance parameters of the ML algorithms in predicting the R_e for training and testing phase respectively.

The efficacy of each model is evaluated using various statistical indices, including R^2, RMSE, MAE, MBE, MAD, WMAPE, U_{95}, and GPI. In order for a model to be deemed

Table 5 Performance parameters of ML techniques in predicting the R_f for training case

Models	R^2	RMSE	MAE	MBE	MAD	WMAPE	U_{95}	GPI
XGB	0.884	0.078	0.060	0.003	0.044	0.272	0.217	2.41E-06
BiLSTM	0.792	0.104	0.072	-0.009	0.055	0.323	0.288	-4.4E-05
LSTM	0.794	0.108	0.074	-0.026	0.055	0.334	0.297	-0.00039
CNN	0.812	0.100	0.072	-0.013	0.044	0.325	0.278	-8.4E-05

Table 6 Performance parameters of ML techniques in predicting the R_f for testing case

Models	R^2	RMSE	MAE	MBE	MAD	WMAPE	U_{95}	GPI
XGB	0.814	0.078	0.059	-0.005	0.044	0.273	0.218	-6.8E-06
BiLSTM	0.771	0.089	0.067	-0.012	0.056	0.312	0.248	-5.4E-05
LSTM	0.778	0.096	0.074	-0.030	0.068	0.353	0.260	-0.00034
CNN	0.785	0.090	0.073	-0.023	0.063	0.344	0.246	-0.00017

Table 7 Performance parameters of ML techniques in predicting the R_h for training case

Models	R^2	RMSE	MAE	MBE	MAD	WMAPE	U_{95}	GPI
XGB	0.886	0.056	0.040	0.006	0.027	0.283	0.156	7.96E-06
BiLSTM	0.835	0.067	0.046	-0.001	0.027	0.323	0.188	-1.60E-07
LSTM	0.831	0.068	0.046	0.005	0.031	0.326	0.189	9.25E-06
CNN	0.828	0.071	0.050	0.004	0.038	0.356	0.196	6.86E-06

Table 8 Performance parameters of ML techniques in predicting the R_h for testing case

Models	R^2	RMSE	MAE	MBE	MAD	WMAPE	U_{95}	GPI
XGB	0.862	0.080	0.058	0.007	0.030	0.322	0.222	1.30E-05
BiLSTM	0.864	0.083	0.054	-0.007	0.029	0.299	0.232	-1.13E-05
LSTM	0.857	0.084	0.056	0.000	0.032	0.312	0.233	-3.99E-08
CNN	0.852	0.093	0.062	-0.005	0.041	0.346	0.258	-6.18E-06

correct, it is desired that its RMSE, MAE, MBE, MAD, and WMAPE values are minimized to the greatest extent possible.

Tables 3 and 4 demonstrate that the prediction error for each machine learning model is relatively low. This indicates that the four leading ML models, developed using the Python programming language and compiled using Jupyter Notebook software, have promising performance. The U_{95} statistic quantifies the level of uncertainty with a confidence level of 95%. The model with the lowest U_{95} values is considered superior. A higher GPI value is indicative of superior model performance. The R^2 value obtained by using Eq. (2), is employed to assess the overall performance. Models with R^2 values closer to one exhibit superior performance. The typical least squares regression approach is employed to determine the

line of best fit between the actual and predicted data, as well as its corresponding R^2 value. The XGB model demonstrates superior performance, as seen by its R^2 values of 0.913 for training data sets and 0.807 for testing data sets

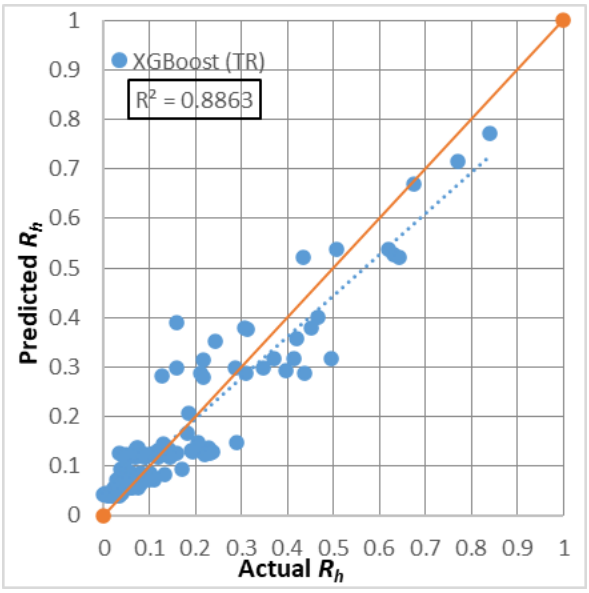
Figs. 8(a) and 8(b) depicts the regression graphs between the observed values and the predicted values of maximal fault distance (R_f) obtained from XGB model. The R^2 value shown in the table and regression plot indicates that the XGB model achieved the highest R^2 value. Thus, out of all the machine learning models included in the current study, XGB has been demonstrated to be the most dependable and effective. Tables 5 and 6 below compares the various performance parameters of the ML algorithms in predicting the R_f for training and testing cases respectively. The line of best fit and the associated R^2 value in the actual vs predicted graph are obtained using the conventional least squares regression method. It is evident that the XGB model, with the highest R^2 values of 0.884 for TR and 0.814 for TS, performs the best.

Figs. 9(a) and 9(b) depicts the regression graph between the observed value and the predicted value of maximal hypocentral distance (R_h) obtained from XGB model. The R^2 value displayed in the scatter plot suggests that XGB model attains highest R^2 value. Thus, out of all the machine learning models included in the current study, XGB has been demonstrated to be the most dependable and effective. Tables 7 and 8 below compares the various performance parameters of the ML algorithms in predicting the R_h for training and testing cases respectively.

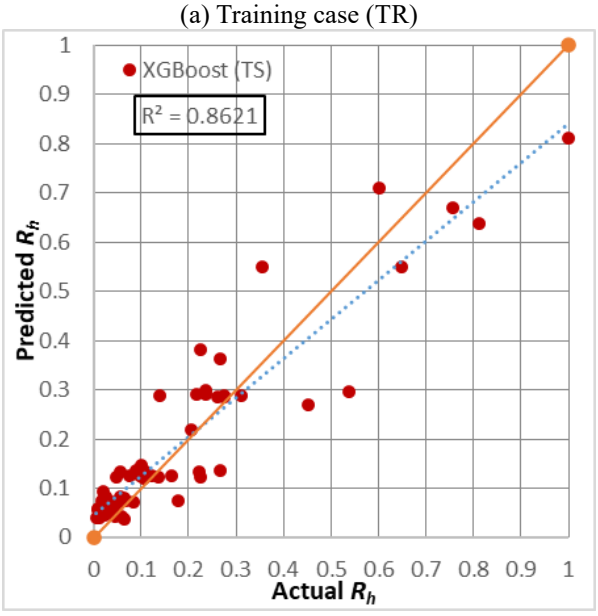
The line of best fit and the associated R^2 value in the actual vs. predicted graph are obtained using the conventional least

Table 9 Average RMSE values obtained after 3 x 5-fold cv for all ML models in predicting R_e, R_f and R_h when data split is (TR:TS=70:30)

Models	XGB		LSTM		BiLSTM		CNN	
M-D	TR	TS	TR	TS	TR	TS	TR	TS
M_w-R_e	0.0699	0.0828	0.0665	0.0705	0.0671	0.0716	0.0723	0.0935
M_w-R_f	0.0937	0.1202	0.1092	0.1148	0.1079	0.1131	0.1106	0.0918
M_w-R_h	0.0828	0.095	0.0739	0.0802	0.0758	0.0814	0.0819	0.1074



(a) Training case (TR)



(b) Testing case (TS)

Fig. 9 XGB model actual and predicted R_h trend

squares regression method. It is clear that the XGB model,which has R^2 values of 0.886 for TR datasets and 0.862 for TS, achieves the best results.

The average RMSE values for all ML models obtained after 5-fold cross validation with 3 repetitions for training and testing data is presented in Table 9.

Table 10 Rank analysis of various performance indicators for M_w-R_e

Performance	XGB		BiLSTM		LSTM		CNN	
parameters	TR	TS	TR	TS	TR	TS	TR	TS
R^2	1	1	4	4	2	2	3	3
RMSE	1	1	3	3	2	2	4	4
MAE	2	2	3	1	1	1	4	3
MBE	2	1	1	2	3	3	4	4
MAD	4	4	2	1	1	2	3	3
WMAPE	2	3	3	2	1	1	4	4
U_{95}	1	1	3	3	2	2	4	4
GPI	2	1	1	2	3	3	4	4
Sub total	15	14	20	18	15	16	30	29
Total score	29		38		31		59	
Rank	1		3		2		4	

Table 11 Rank analysis of various performance indicators for M_w-R_f

Performance	XGB		BiLSTM		LSTM		CNN	
parameters	TR	TS	TR	TS	TR	TS	TR	TS
R^2	1	1	4	4	3	3	2	2
RMSE	1	1	3	2	4	4	2	3
MAE	1	1	2	2	3	4	2	3
MBE	1	1	2	2	4	4	3	3
MAD	1	1	2	2	2	4	1	3
WMAPE	1	1	2	2	4	4	3	3
U_{95}	1	1	3	3	4	4	2	2
GPI	1	1	2	2	4	4	3	3
Sub total	8	8	20	19	28	31	18	22
Total score	16		39		59		40	
Rank	1		2		4		3	

4.2 Rank analysis for different ML models

The performances of the various proposed models are evaluated using a simple rank analysis approach. The rank values for both the training and testing phases of each model are calculated. The effectiveness of the proposed machine learning models is assessed using an overall score. The rank index is determined by summing the individual scores obtained

Table 12 Rank analysis of various performance indicators for M_w-R_h

Performance	XGB		BiLSTM		LSTM		CNN	
parameters	TR	TS	TR	TS	TR	TS	TR	TS
R^2	1	2	2	1	3	3	4	4
RMSE	1	1	2	2	3	3	4	4
MAE	1	3	2	1	2	2	3	4
MBE	3	2	4	4	2	1	1	3
MAD	1	2	1	1	2	3	3	4
WMAPE	1	3	2	1	3	2	4	4
U_{95}	1	1	2	2	3	3	4	4
GPI	2	1	4	4	3	2	1	3
Sub total	11	15	19	16	21	19	24	30
Total score	26		35		40		54	
Rank	1		2		3		4	

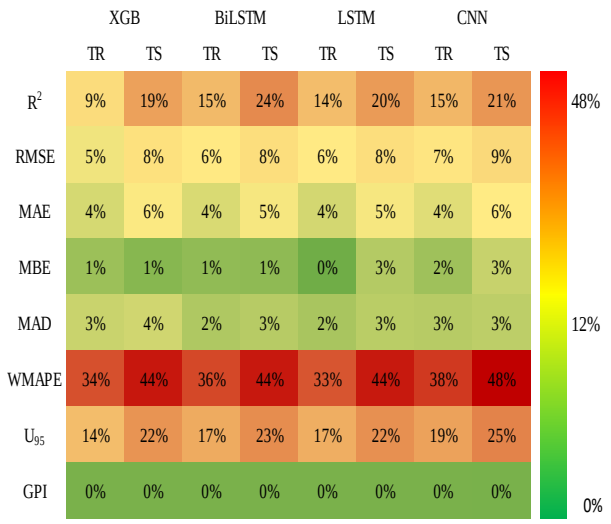
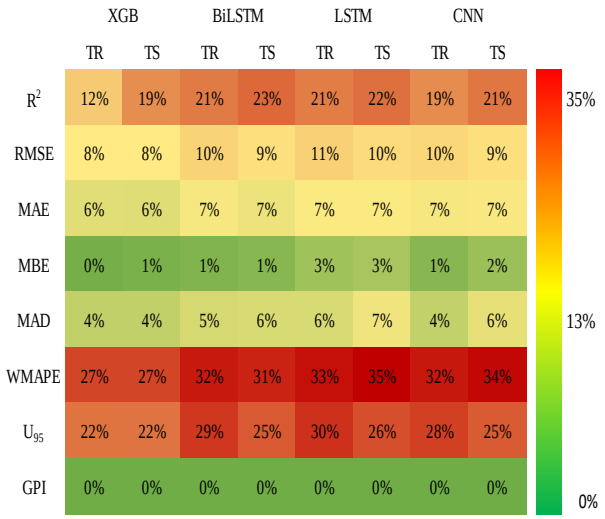
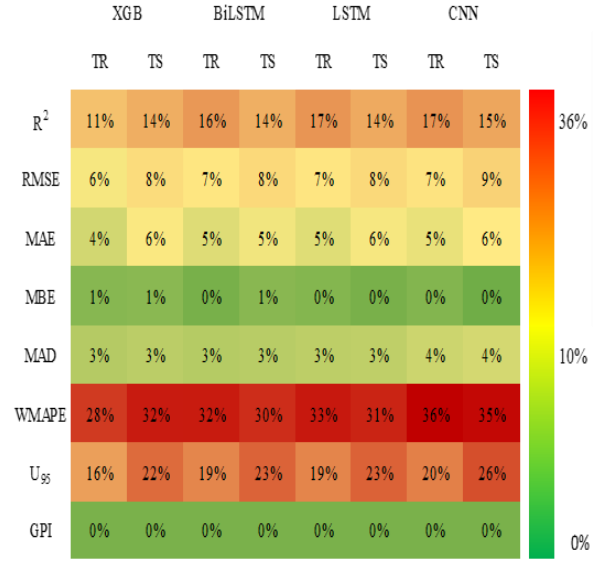


Fig. 10 Error matrix chart of training and testing results for M_w-R_e



(a) Error matrix chart for M_w-R_f



(b) Error matrix chart for M_w-R_h

Fig. 11 Error matrix chart of training and testing results for M_w-R_f and M_w-R_h

from different statistical analysis criteria. The efficiency of the model is enhanced when the magnitude of this parameter is reduced. The rank score of all statistical parameters and overall rank for ML techniques is presented in Tables 10-12 XGB model with lowest overall score performs best for all M-D analysis.

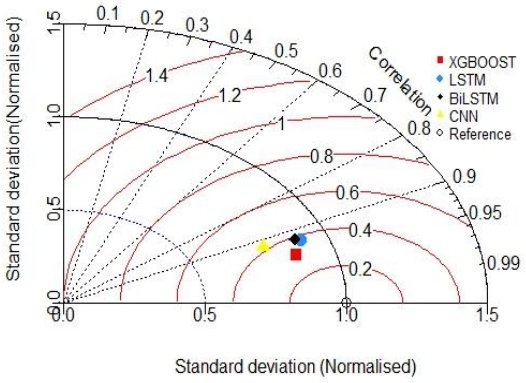
4.3 Error matrix for different ML models

The evaluation of a proposed model's performance can be conducted by doing a comparative analysis between the data obtained from the training set and the test set, utilizing a specialized matrix known as an error matrix. Additionally, a novel graphical concept known as a heat pump matrix is utilized to visually represent the error value (Kumar *et al.* 2022). Ultimately, the model errors are compared throughout a wide range, spanning from 0% to 48%. According to the CNN model, it exhibited the greatest error rates of 38% and 48% intraining and testing phases whereas the XGB model achieved the lowest error (0%) in M_w-R_e analysis as shown in Fig. 10.

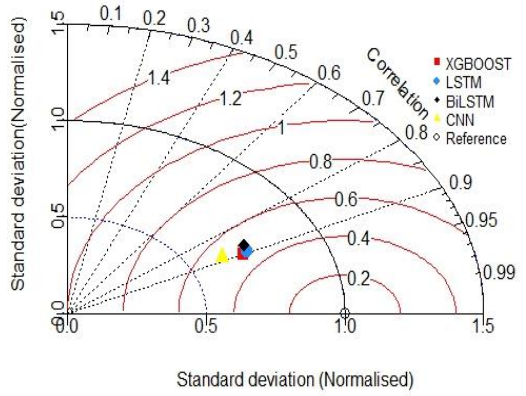
Similarly, for M_w-R_f and M_w-R_h analysis XGB model attained minimum error as compared to other models as shown in Figs. 11(a) and 11(b). The XGB model is shown to have the lowest error based on both training and testing results. This indicates that the XGB model outperforms all other models.

4.4 Taylor's diagram

Taylor diagrams provide a visual representation of the degree to which an observed pattern (or combination of patterns) aligns with a pattern. To assess the similarity between two patterns, statistical measures such as correlation, centered root-mean-square difference, and variation amplitude (measured by standard deviations) are taken into account. Taylor's diagram is shown in Figs. 12-14(a) and 14(b) for various ML models. These diagrams are quite beneficial for examining various features of intricate models or ascertaining the comparative usefulness of multiple models. The model's



(a) Training case



(b) Testing case

Fig. 12 Taylor's plot of ML models for (a) training and (b) testing case (M_w-R_e)

accuracy is determined by the degree of correspondence between the projected value and the reference data. A higher correlation indicates a greater level of agreement between the actual and expected results. The M_w-R_e analysis reveals correlation coefficients of 0.93, 0.92, 0.92, and 0.95 between the actual and projected results for the ML models (LSTM, BiLSTM, CNN, and XGB) on the training data, and 0.89, 0.87, 0.89, and 0.90 on the testing data sets. Similarly, for M_w-R_f and M_w-R_h analysis, correlation between predicted and observed data is higher for the XGB model. The findings of this study also indicates that while predicting the maximum distances (R_e, R_f and R_h), XGB model performs superior than other ML models.

4.5 Effect of split ratio on training, testing and validation data

This section presents the results of the effect of the split ratio on training, testing, and validation data. The data was divided into three split ratios for training (TR), testing (TS), and validation (VS) in the ratios 70:15:15, 60:20:20, and 80:10:10, respectively. Tables 13-15 present the results of

Table 13 Performance parameters of ML techniques in predicting the Re for training, testing and validation stage with split ratio of 70:15:15

Split ratio	Statistical parameter	XGB	LSTM	BiLSTM	CNN
TR (70)	R^2	0.853	0.805	0.803	0.813
	RMSE	0.063	0.069	0.068	0.077
	MAE	0.043	0.049	0.047	0.048
TS (15)	R^2	0.824	0.839	0.845	0.840
	RMSE	0.080	0.081	0.079	0.096
	MAE	0.054	0.055	0.052	0.060
VS (15)	R^2	0.839	0.842	0.836	0.832
	RMSE	0.059	0.071	0.071	0.086
	MAE	0.042	0.048	0.045	0.048

Table 14 Performance parameters of ML techniques in predicting the R_e for training, testing and validation stage with split ratio of 60:20:20

Split ratio	Statistical parameter	XGB	LSTM	BiLSTM	CNN
TR (60)	R^2	0.845	0.809	0.807	0.814
	RMSE	0.061	0.072	0.071	0.080
	MAE	0.040	0.052	0.050	0.049
TS (20)	R^2	0.887	0.849	0.846	0.848
	RMSE	0.075	0.084	0.079	0.098
	MAE	0.048	0.057	0.055	0.060
VS (20)	R^2	0.792	0.798	0.801	0.802
	RMSE	0.069	0.070	0.066	0.081
	MAE	0.043	0.047	0.044	0.049

Table 15 Performance parameters of ML techniques in predicting the R_e for training, testing and validation stage with split ratio of 80:10:10

Split ratio	Statistical parameter	XGB	LSTM	BiLSTM	CNN
TR (80)	R^2	0.876	0.794	0.791	0.802
	RMSE	0.060	0.072	0.075	0.074
	MAE	0.039	0.052	0.055	0.046
TS (10)	R^2	0.840	0.881	0.862	0.909
	RMSE	0.076	0.075	0.074	0.081
	MAE	0.052	0.052	0.054	0.050
VS (10)	R^2	0.823	0.905	0.900	0.913
	RMSE	0.059	0.075	0.076	0.085
	MAE	0.039	0.052	0.058	0.049

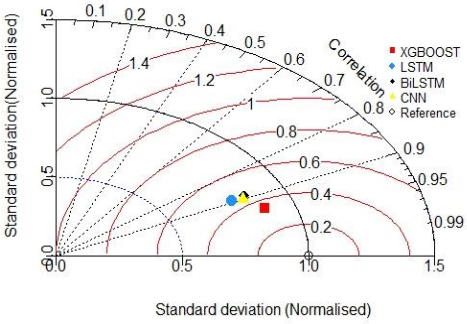
statistical parameters of ML models for different split ratios and Table 16 presents the average RMSE values of ML models for training stage (TR). It can be seen that the deviation in XGB model RMSE value from average RMSE value for split ratio of 70:15:15 is less than the deviation for other split ratios. Also, R^2 value for this split ratio for TR and VS is greater than other split ratios in the prediction of R_e. The supplementary

Table 16 Average RMSE values of ML models for different split ratios for training stage

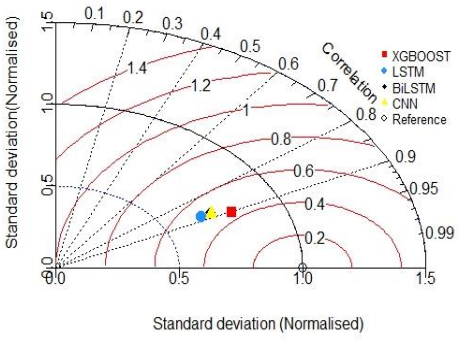
Split ratio	M-D	XGB	LSTM	BiLSTM	CNN
(70:15:15)	M_w-R_e	0.0714	0.0716	0.0703	0.0706
	M_w-R_f	0.0985	0.1049	0.0931	0.1039
	M_w-R_h	0.0803	0.0832	0.0863	0.0841
(60:20:20)	M_w-R_e	0.0702	0.0714	0.0686	0.0755
	M_w-R_f	0.1047	0.0961	0.1019	0.1197
	M_w-R_h	0.0882	0.0815	0.0835	0.0792
(80:10:10)	M_w-R_e	0.0723	0.0733	0.0722	0.0795
	M_w-R_f	0.1011	0.1087	0.1113	0.1003
	M_w-R_h	0.0774	0.078	0.0826	0.0823

Table 17 Performance comparison of XGB model with empirical model

M-D	M-D relation	R^2	RMSE	MAE
M_w-R_e	$Mw = 5.243 + 2.742 \times 10^{-3} R_e + 0.8669 \times logR_e$	0.745	0.380	0.289
	XGB model (TS)	0.807	0.078	0.057
M_w-R_f	$Mw = 5.9391 + 13.818 \times 10^{-3} R_f + 0.2807 \times logR_f$	0.697	0.391	0.301
	XGB model (TS)	0.814	0.078	0.059
M_w-R_h	$Mw = 4.7073 + 2.11 \times 10^{-3} R_h + 0.8669 \times logR_h$	0.813	0.319	0.252
	XGB model (TS)	0.862	0.080	0.058

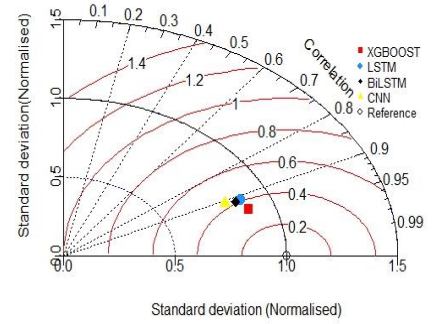


(a) Training case

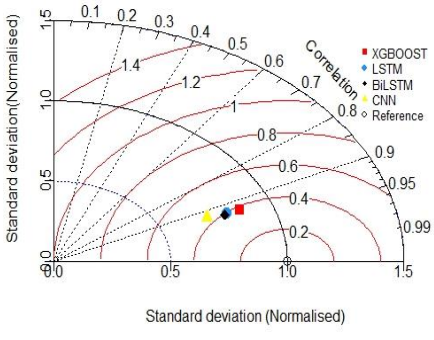


(b) Testing case

Fig. 13 Taylor's plot of ML models for (a) training and (b) testing cases (M_w-R_f)

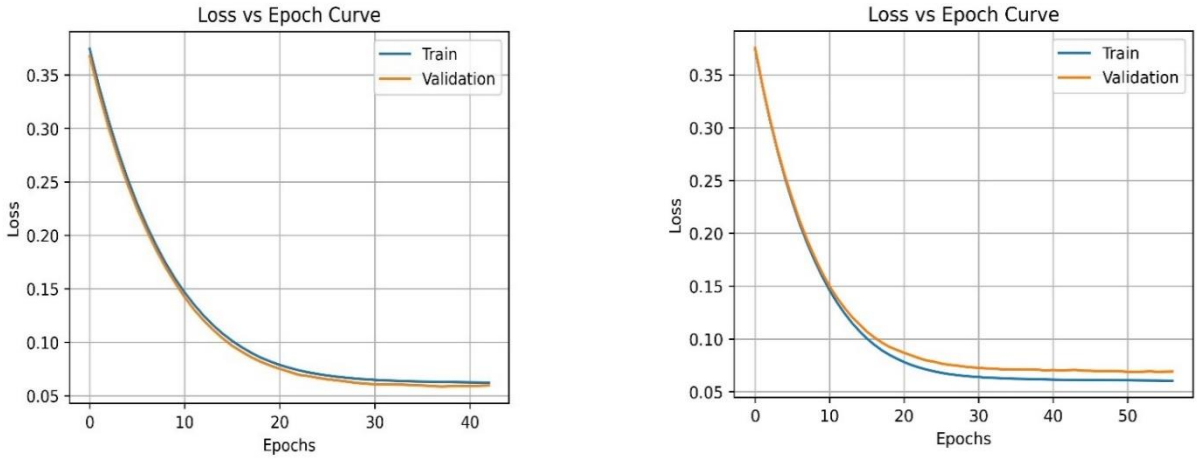


(a) Training case



(b) Testing case

Fig. 14 Taylor's plot of ML models for (a) training and (b) testing cases (M_w-R_h)



(a) Split ratio (TR:TS:VS= 70:15:15) (b) Split ratio (TR:TS:VS =60:20:20)

Fig. 15 Loss vs epoch curves for different split ratios in prediction of R_e for XGB model

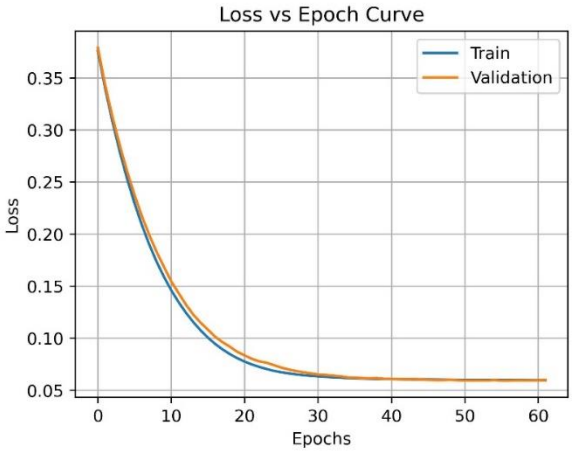


Fig. 16 Loss vs epoch curve for split ratio of TR:TS:VS = (80:10:10) in prediction of Re for XGB model

appendix file includes the additional tables corresponding to the effect of split ratio in prediction of R_f and R_h.

Our training progress for ML models may be clearly seen by examining the loss vs epoch curve. This plot shows loss as a function of epochs and helps detect overfitting and underfitting and provides guidance for early stopping decisions. Each point on the graph reflects loss values recorded in subsequent epochs. A regularization strategy called early stopping stops training when it notices overfitting or inadequate learning over a certain number of successive epochs is used in this study. Figs. 15(a) and 15(b) and Fig. 16 depict the loss vs epoch curves for XGB model corresponding to different split ratios for training and validation stage in the prediction of R_e. The loss vs epoch curves for all ML models for the prediction of R_e, R_f and R_h corresponding to different split ratios is presented in supplementary appendix file.

4.6 Performance comparison of XGB model with empirical method

All four ML models performed better than empirical models for three M-D assessment. Among these models XGB model outperformed all the models with highest R^2 value. Table 17 below presents the R^2, RMSE and MAE values for different M-D relation as well as the values obtained using XGB model in the testing (TS) phase. For M_w-R_e relation R^2, RMSE and MAE values are 0.745, 0.380 and 0.289 using empirical method whereas from XGB model it is 0.807, 0.078 and 0.057 respectively. Similarly, for M_w-R_f relation R^2, RMSE and MAE values are 0.697, 0.391 and 0.301 using empirical method whereas from XGB model it is 0.814, 0.078 and 0.059 respectively. For M_w-R_h relation R^2, RMSE and MAE values are 0.813, 0.319 and 0.252 using empirical method whereas from XGB model it is 0.862, 0.080 and 0.058 respectively. Thus, XGB model performs better as compared to empirical method. It is worth noting that among all the three M-D expressions, M_w-R_h relation and corresponding XGB model both have highest R^2 value and performs best.

5. Conclusions

In this study, a database of 290 records documenting liquefaction events around the world, together with their corresponding magnitudes and distances, from years 1117-2020 is used to predict maximum epicentral distance (R_e), maximum fault distance (R_f), and maximum hypocentral distance (R_h). To predict these distances, four ML models namely LSTM, BiLSTM, CNN, and XGB are developed using Python programming language. To evaluate the performance of the developed models, statistical parameters are calculated, and it can be concluded that the XGB model outperformed all others with the highest predictive accuracy in both training and testing phases for all three magnitude distance studies. Further, three M-D empirical relations are used to compare the results obtained using XGB model. For M_w-R_e relation; R^2, RMSE, and MAE values are 0.745, 0.380, and 0.289 using the empirical method whereas from the XGB model it is 0.807, 0.078, and 0.057 respectively. The rank analysis, error matrix, and Taylor's diagram further demonstrate that the XGB model outperforms all other ML models. This result illustrates that the

proposed XGB ML model is reliable. Among all the three M-D expressions, M_w-R_h equation and the corresponding XGB model both have the highest R^2 value of 0.862 and perform best than other models. The effect of split ratio also suggests that the XGB model performs better as compared to other ML models in terms of R^2 values and deviation in actual and average RMSE values for 70:15:15 data split. As compared to existing models or relations, the proposed machine learning algorithms for the prediction of limiting distances in this study are more robust, which is useful for the assessment of regional liquefaction-induced hazard scenarios. However, the presented models' generalization ability must be validated with additional new data in the future.

References

Adarsh, S., Dhanya, R., Krishna, G., Merlin, R. and Tina, J. (2012), "Prediction of ultimate bearing capacity of cohesionless soils using soft computing techniques", *ISRN Artificial Intelligence*, **2012**, 1-10. https://doi.org/10.5402/2012/628496

Alessio, G., Alfonsi, L., Brunori, C.A., Burrato, P., Casula, G., Cinti, F.R., Civico, R., Colini, L., Cucci, L., De Martini, P.M., Falcucci, E., Galadini, F., Gaudiosi, G., Gori, S., Mariucci, M. T., Montone, P., Moro, M., Nappi, R., Nardi, A. andVillani, F. (2013), "Liquefaction phenomena associated with the Emilia earthquake sequence of May-June 2012 (Northern Italy)", *Nat. Hazard. Earth System Sci.*, **13**(4), 935-947. https://doi.org/10.5194/nhess-13-935-2013.

Alibrahim, H. and Ludwig, S.A. (2021), "Hyperparameter optimization: Comparing genetic algorithm against grid search and bayesian optimization", *Proceedings of the 2021 IEEE Congress on Evolutionary Computation (CEC)*.

Ambraseys, N.N. (1988), "Engineering seismology: Part II", *Earthq. Eng. Struct. D.*, **17**(1), 51-105. https://doi.org/10.1002/eqe.4290170102.

Anitescu, C., Atroshchenko, E., Alajlan, N. and Rabczuk, T. (2019), "Artificial neural network methods for the solution of second order boundary value problems", *Comput, Mater. Continua*, **59**(1). https://doi.org/10.32604/cmc.2019.06641

Ardeshiri-Lajimi, S., Yazdani, M. and Assadi-Langroudi, A. (2016), "A study on the liquefaction risk in seismic design of foundations", *Geomech. Eng.*, **11**(6), 805-820. https://doi.org/10.12989/gae.2016.11.6.805.

Armaghani, D.J., Harandizadeh, H., Momeni, E., Maizir, H. and Zhou, J. (2022), "An optimized system of GMDH-ANFIS predictive model by ICA for estimating pile bearing capacity", *Artif. Intell. Rev.*, **55**(3), 2313-2350. https://doi.org/10.1007/s10462-021-10065-5.

Aydan, Ö., Ulusay, R. and Kumsar, H. (2000), "Liquefaction phenomenon in the earthquakes of Turkey, including recent Erzincan, Dinar and Adana-Ceyhan earthquakes", *Proceedings of the 12th World Conference on Earthquake Engineering*, Auckland, 30 January -4 February.

Bai, X.D., Cheng, W.C., Ong, D.E.L. and Li, G. (2021), "Evaluation of geological conditions and clogging of tunneling using machine learning", *Geomech. Eng.*, **25**(1), 59-73. https://doi.org/10.12989/gae.2021.25.1.059.

Bengio, Y. (2000), "Gradient-based optimization of hyperparameters", *Neural Comput.*, **12**(8), 1889-1900. https://doi.org/10.1162/089976600300015187

Bergstra, J. and Bengio, Y. (2012), "Random search for hyper-parameter optimization", *J. Machine Learning Res.*, **13**(2).

Bottou, L. (2010). "Large-scale machine learning with stochastic gradient descent", *Proceedings of the COMPSTAT'2010: 19th International Conference on Computational Statistics*, Paris France, August 22-27, 2010 Keynote, Invited and Contributed Papers.

Pirrotta, C., Barbano, M.S., Guarnieri, P. and Gerardi, F. (2009), "A new dataset and empirical relationships between magnitude/intensity and epicentral distance for liquefaction in central-eastern Sicily", *Ann. Geophys.*, **50**(6), 763-774. https://doi.org/10.4401/ag-3055.

Cetin, K.O., Mylonakis, G., Sextos, A. and Stewart, J.P. (2022), "Reconnaissance of 2020 M 7.0 Samos Island (Aegean Sea) earthquake", *Bull. Earthq. Eng.*, **20**(14), 7707-7712. https://doi.org/10.1007/s10518-021-01212-y.

Chen, T. and Guestrin, C. (2016), "Xgboost: A scalable tree boosting system", *Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining*, San Francisco, August.

Dibike, Y.B., Velickov, S., Solomatine, D. and Abbott, M.B. (2001), "Model induction with support vector machines: introduction and applications", *J. Comput. Civil Eng.*, **15**(3), 208-216. https://doi.org/10.1061/(ASCE)0887-3801(2001)15:3(208).

Galli, P. (2000), "New empirical relationships between magnitude and distance for liquefaction", *Tectonophysics*, **324**(3), 169-187. https://doi.org/10.1016/S0040-1951(00)00118-9.

Gazetas, G. and Botsis, J. (1981), "Local soil effects and liquefaction in the 1978 Thessaloniki earthquakes", *Proceedings of the International Conference on Recent Advances Geotechnical Earthquake Engineering & Soil Dynamics. University of Missouri-Rolla*, Rolla, Missouri, April.

Goh, A.T.C. and Goh, S.H. (2007), "Support vector machines: their use in geotechnical engineering as illustrated using seismic liquefaction data", *Comput. Geotech.*, **34**(5), 410-421. https://doi.org/10.1016/j.compgeo.2007.06.001.

Guo, X.C., Yang, J.H., Wu, C.G., Wang, C.Y. and Liang, Y.C. (2008), "A novel LS-SVMs hyper-parameter selection based on particle swarm optimization", *Neurocomput.*, **71**(16), 3211-3215. https://doi.org/10.1016/j.neucom.2008.04.027.

Hoang, N.D. and Bui, D.T. (2018), "Predicting earthquake-induced soil liquefaction based on a hybridization of kernel Fisher discriminant analysis and a least squares support vector machine: A multi-dataset study", *Bull. Eng. Geol. Environ.*, **77**(1), 191-204. https://doi.org/10.1007/s10064-016-0924-0.

Hochreiter, S. and Schmidhuber, J. (1997), "Long short-term memory", *Neural Comput.*, **9**(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Hu, J. (2021), "Data cleaning and feature selection for gravelly soil liquefaction", *Soil Dyn. Earthq. Eng.*, **145**, 106711. https://doi.org/10.1016/j.soildyn.2021.106711.

Hu, J. (2022a), "Empirical relationships between earthquake magnitude and maximum distance based on the extended global liquefaction-induced damage cases", *Acta Geotechnica*, **18**, 1-15. https://doi.org/10.1007/s11440-022-01637-y.

Hu, J. (2022b), "The database of earthquake-induced liquefaction", *Mendeley Data*, **VI**. https://doi.org/10.17632/3d2483vxb2.1.

Hu, J. and Liu, H. (2019), "Bayesian network models for probabilistic evaluation of earthquake-induced liquefaction based on CPT and Vs databases", *Eng. Geol.*, **254**(1), 76-88. https://doi.org/10.1016/j.enggeo.2019.04.003.

James, G., Witten, D., Hastie, T. and Tibshirani, R. (2013), *An introduction to statistical learning*, **112**, Springer.

Javadi, A.A. and Rezania, M. (2009), "Applications of artificial intelligence and data mining techniques in soil modeling", *Geomech. Eng.*, **1**(1), 53-74. https://doi.org/10.12989/gae.2009.1.1.053.

Jiang, W., Li, Z.Y. and Lu, K.Y. (2019), "Preliminary analysis of liquefaction characteristics induced by the Songyuan earthquake of May 28 in Jilin", *China*, **39**(3), 52-60.

https://doi.org/10.13197/j.eeev.2019.03.52.Jiangw.006.

Kamran, M., Shahani, N.M. and Armaghani, D.J. (2022), "Decision support system for underground coal pillar stability using unsupervised and supervised machine learning approaches", *Geomech. Eng.*, **30**(2), 107-121. https://doi.org/10.12989/gae.2022.30.2.107.

Kumar, D.R., Samui, P. and Burman, A. (2022), "Prediction of probability of liquefaction using soft computing techniques", *J. Institution of Engineers (India): Series A*, **103**(4), 1195-1208. https://doi.org/10.1007/s40030-022-00683-9

Kumar, K., Samui, P. and Choudhary, S.S. (2023), "State parameter based liquefaction probability evaluation", *Int. J. Geosynthetics Ground Eng.*, **9**(6), 76. https://doi.org/10.1007/s40891-023-00495-2.

Kumar, P., Rao, B., Burman, A., Kumar, S. and Samui, P. (2023), "Spatial variation of permeability and consolidation behaviors of soil using ordinary kriging method", *Groundwater for Sustain. Development*, **20**, 100856. https://doi.org/10.1016/j.gsd.2022.100856.

Kumar, P. and Samui, P. (2022), "Design of an energy pile based on CPT data using soft computing techniques", *Infrastructures*, **7**(12), 169. https://doi.org/10.3390/infrastructures7120169.

Kuribayashi, E. and Tatsuoka, F. (1975), "Brief review of liquefaction during earthquakes in Japan", *Soils Found.*, **15**(4), 81-92. https://doi.org/10.3208/sandf1972.15.4_81.

LeCun, Y. and Bengio, Y. (1995), "Convolutional networks for images, speech, and time series. In *The handbook of brain theory and neural networks*. The MIT Press,Cambridge, Massachusetts, USA.

Lee, C.Y. and Chern, S.G. (2013). Application of a support vector machine for liquefaction assessment", *J. Mar. Sci.Tech. (Taiwan)*, **21**(3), 318-324. https://doi.org/10.6119/JMST-012-0518-3.

Liu, L.L., Yang, C. and Wang, X.M. (2021), "Landslide susceptibility assessment using feature selection based machine learning models", *Geomech. Eng.*, **25**(1), 1-16. https://doi.org/10.12989/gae.2021.25.1.001

Mahmoodzadeh, A., Taghizadeh, M., Mohammed, A.H., Ibrahim, H.H., Samadi, H., Mohammadi, M. and Rashidi, S. (2022), "Tunnel wall convergence prediction using optimized LSTM deep neural network", *Geomech. Eng.*, **31**(6), 545-556. https://doi.org/10.12989/gae.2022.31.6.545.

Ochoa, L.H., Niño, L.F. and Vargas, C.A. (2018), "Fast estimation of earthquake epicenter distance using a single seismological station with machine learning techniques", *Dyna*, **85**(204), 161-168.

Pal, M. and Deswal, S. (2008), "Modeling pile capacity using support vector machines and generalized regression neural network", *J. Geotech. Geoenviron. Eng.*, **134**(7), 1021-1024. https://doi.org/10.1061/(asce)1090-0241(2008)134:7(1021).

Papadopoulos, G.A. and Lefkopoulos, G. (1993), "Magnitude-distance relations for liquefaction in soil from earthquakes", *Bull. Seismol. Soc. Am.*, **83**(3), 925-938. https://doi.org/10.1785/bssa0840062019.

Papathanassiou, G., Pavlides, S., Christaras, B. and Pitilakis, K. (2005), "Liquefaction case histories and empirical relations of earthquake magnitude versus distance from the broader Aegean region", *J. Geodynam.*, **40**(2-3), 257-278. https://doi.org/10.1016/j.jog.2005.07.007.

Samaniego, E., Anitescu, C., Goswami, S., Nguyen-Thanh, V.M., Guo, H., Hamdia, K., Zhuang, X. and Rabczuk, T. (2020), "An energy approach to the solution of partial differential equations in computational mechanics via machine learning: Concepts, implementation and applications", *Comput. Method. Appl. M.*, **362**, 112790. https://doi.org/10.1016/j.cma.2019.112790.

Samui, P. (2007), "Seismic liquefaction potential assessment by using relevance vector machine", *Earthq. Eng. Eng. Vib.*, **6**, 331-336.

Samui, P. and Sitharam, T.G. (2008), "Least-square support vector machine applied to settlement of shallow foundations on cohesionless soils", *Int. J. Numer. Anal. Method. Geomech.*, **32**(17), 2033-2043. https://doi.org/10.1002/nag.731.

Schuster, M. and Paliwal, K.K. (1997), "Bidirectional recurrent neural networks", *IEEE T. Signal Pr.*, **45**(11), 2673-2681. https://doi.org/10.1109/78.650093

Talwani, P. and Cox, J. (1985), "Paleoseismic evidence for recurrence of earthquakes near Charleston, South Carolina", *Science*, **229**(4711), 379-381. https://doi.org/10.1126/science.229.4711.379.

Wang, C.Y., Wong, A., Dreger, D.S. and Manga, M. (2006), "Liquefaction limit during earthquakes and underground explosions: Implications on ground-motion attenuation", *Bull. Seismol. Soc. Am.*, **96**(1), 355-363. https://doi.org/10.1785/0120050019

Xu, G., Liu, Z., Sun, Y., Wang, X., Lin, L. and Ren, Y. (2016), "Experimental characterization of storm liquefaction deposits sequences", *Mar. Geol.*, **382**, 191-199. https://doi.org/10.1016/j.margeo.2016.10.015.

Xu, X., Xu, G., Yang, J., Xu, Z. and Ren, Y. (2021), "Field observation of the wave-induced pore pressure response in a silty soil seabed", *Geo-Marine Lett.*, **41**(1), 13. https://doi.org/10.1007/s00367-020-00680-6

CC

Supplementary Appendix

1. Effect of split ratio on training, testing, and validation data for M_w-R_f and M_w-R_h

Table 18 Effect of split ratio for M_w-R_f when the ratio between TR, TS, and VS is 70:15:15

Split ratio	Statistical parameter	XGB	LSTM	BiLSTM	CNN
TR (70)	R^2	0.831	0.733	0.793	0.728
	RMSE	0.087	0.104	0.095	0.112
	MAE	0.066	0.073	0.072	0.076
TS (15)	R^2	0.761	0.854	0.685	0.884
	RMSE	0.149	0.140	0.153	0.085
	MAE	0.110	0.096	0.114	0.065
VS (15)	R^2	0.772	0.809	0.678	0.835
	RMSE	0.080	0.120	0.090	0.093
	MAE	0.054	0.080	0.056	0.067

Table 19 Effect of split ratio for M_w-R_f when the ratio between TR, TS, and VS is 60:20:20

Split ratio	Statistical parameter	XGB	LSTM	BiLSTM	CNN
TR (60)	R^2	0.750	0.764	0.798	0.715
	RMSE	0.100	0.089	0.096	0.117
	MAE	0.073	0.065	0.070	0.079
TS (20)	R^2	0.804	0.811	0.736	0.886
	RMSE	0.108	0.165	0.123	0.077
	MAE	0.079	0.125	0.088	0.056
VS (20)	R^2	0.866	0.797	0.814	0.807
	RMSE	0.106	0.070	0.106	0.089
	MAE	0.078	0.051	0.067	0.063

Table 20 Effect of split ratio for M_w-R_f when the ratio between TR, TS, and VS is 80:10:10

Split ratio	Statistical parameter	XGB	LSTM	BiLSTM	CNN
TR (80)	R^2	0.814	0.785	0.770	0.776
	RMSE	0.103	0.101	0.098	0.099
	MAE	0.080	0.070	0.069	0.070
TS (10)	R^2	0.873	0.668	0.740	0.749
	RMSE	0.111	0.118	0.154	0.148
	MAE	0.093	0.097	0.103	0.102
VS (10)	R^2	0.717	0.575	0.831	0.838
	RMSE	0.106	0.121	0.094	0.099
	MAE	0.090	0.077	0.075	0.082

Table 21 Effect of split ratio for M_w-R_h when the ratio between TR, TS, and VS is 70:15:15

Split ratio	Statistical parameter	XGB	LSTM	BiLSTM	CNN
TR (70)	R^2	0.872	0.813	0.806	0.821
	RMSE	0.070	0.083	0.085	0.080
	MAE	0.049	0.055	0.061	0.052
TS (15)	R^2	0.840	0.872	0.871	0.872
	RMSE	0.070	0.070	0.074	0.069
	MAE	0.048	0.045	0.053	0.042
VS (15)	R^2	0.763	0.879	0.880	0.879
	RMSE	0.069	0.094	0.087	0.090
	MAE	0.049	0.058	0.059	0.057

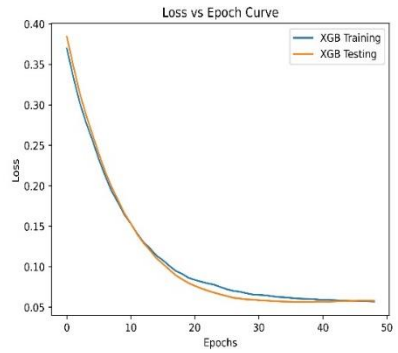
Table 22 Effect of split ratio for M_w-R_h when the ratio between TR, TS, and VS is 60:20:20

Split ratio	Statistical parameter	XGB	LSTM	BiLSTM	CNN
TR (60)	R^2	0.865	0.826	0.825	0.837
	RMSE	0.073	0.078	0.089	0.080
	MAE	0.049	0.059	0.057	0.053
TS (20)	R^2	0.816	0.838	0.851	0.845
	RMSE	0.058	0.075	0.087	0.082
	MAE	0.043	0.053	0.055	0.048
VS (20)	R^2	0.850	0.823	0.819	0.824
	RMSE	0.072	0.079	0.090	0.082
	MAE	0.050	0.057	0.055	0.053

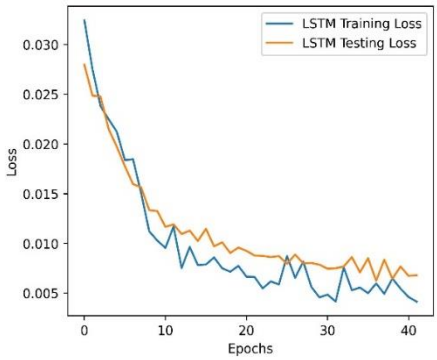
Table 23 Effect of split ratio for M_w-R_h when the ratio between TR, TS, and VS is 80:10:10

Split ratio	Statistical parameter	XGB	LSTM	BiLSTM	CNN
TR (80)	R^2	0.880	0.826	0.827	0.836
	RMSE	0.067	0.083	0.085	0.078
	MAE	0.046	0.054	0.055	0.052
TS (10)	R^2	0.744	0.847	0.857	0.855
	RMSE	0.083	0.095	0.092	0.081
	MAE	0.056	0.063	0.058	0.051
VS (10)	R^2	0.730	0.956	0.945	0.945
	RMSE	0.072	0.093	0.089	0.078
	MAE	0.047	0.053	0.050	0.043

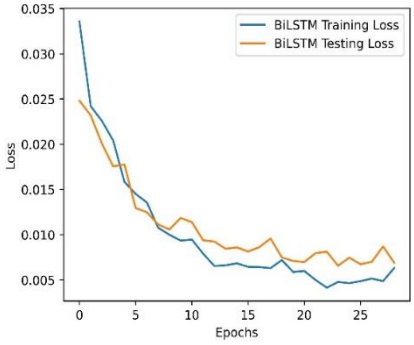
2. Loss vs epoch curves



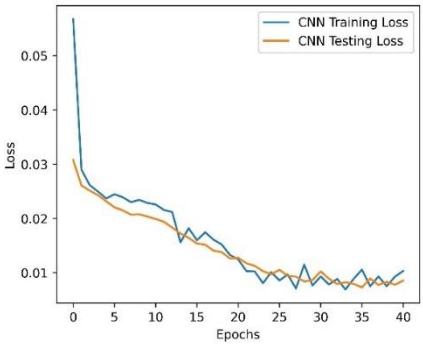
(a) XGB model



(b) LSTM model

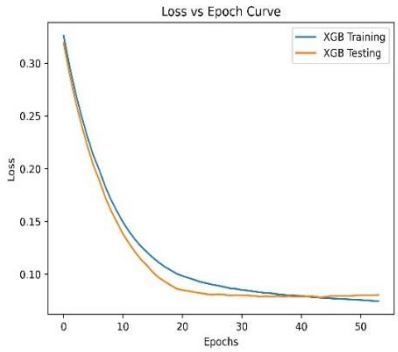


(c) BiLSTM model

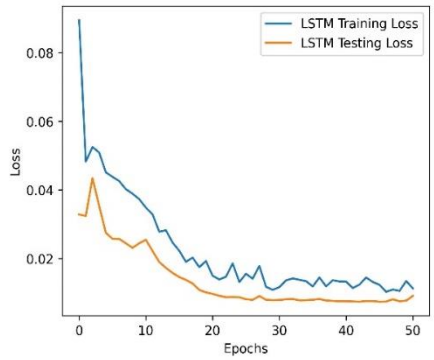


(d) CNN model

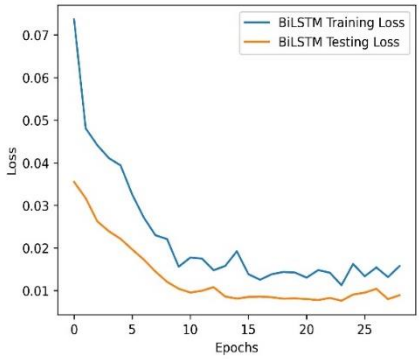
Fig. 17 Loss vs epoch curves of ML models for M_w-R_e for training and testing data divided in the ratio 70:30



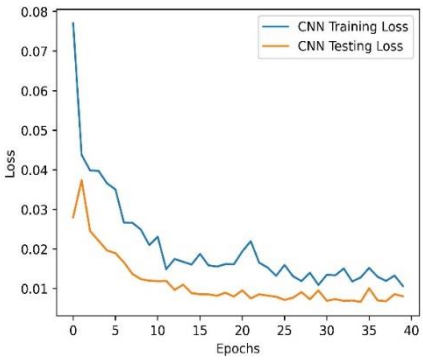
(a) XGB model



(b) LSTM model

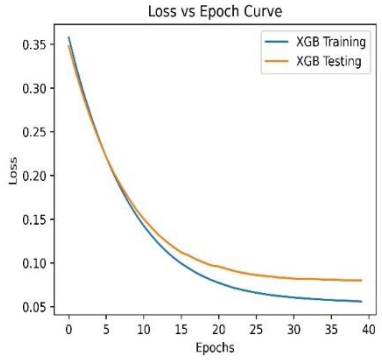


(c) BiLSTM model

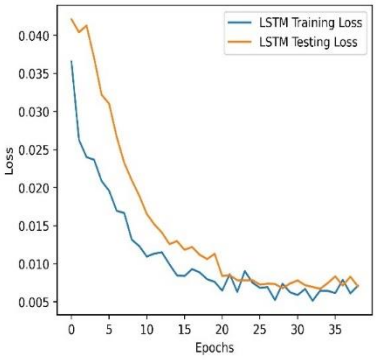


(d) CNN model

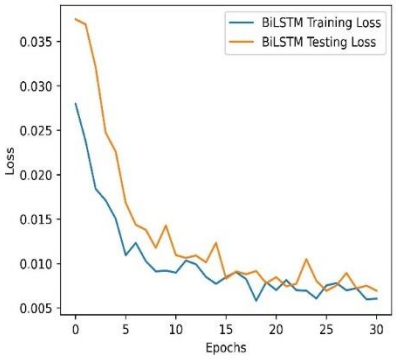
Fig. 18 Loss vs epoch curves of ML models for M_w-R_f for training and testing data divided in the ratio 70:30



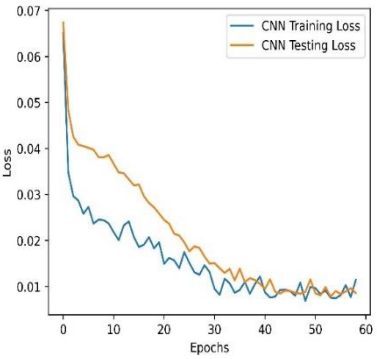
(a) XGB model



(b) LSTM model

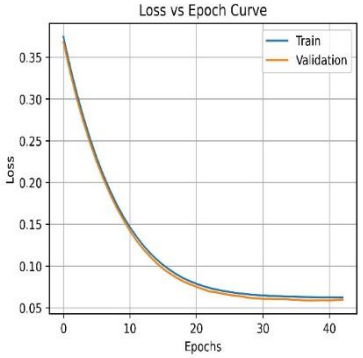


(c) BiLSTM model

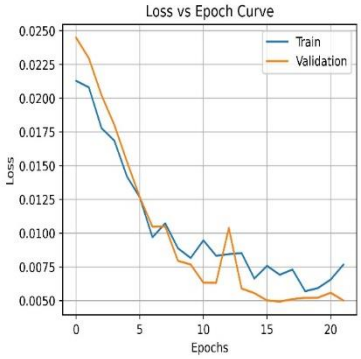


(d) CNN model

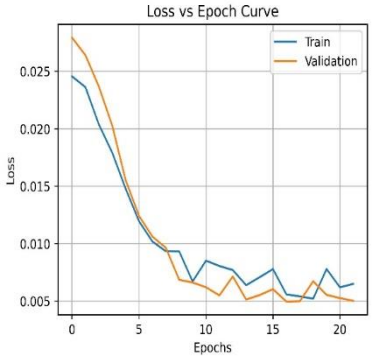
Fig. 19 Loss vs epoch curves of ML models for M_w-R_h for training and testing data divided in the ratio 70:30



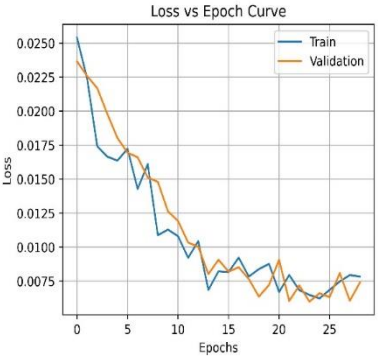
(a) XGB model



(b) LSTM model

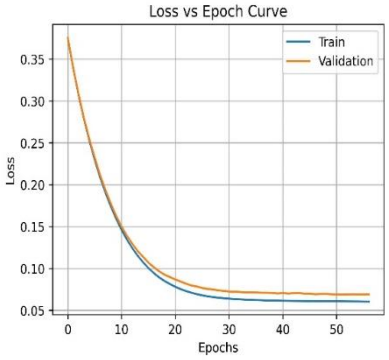


(c) BiLSTM model

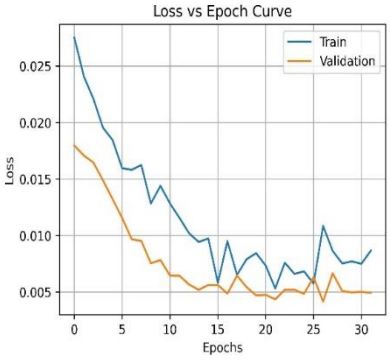


(d) CNN model

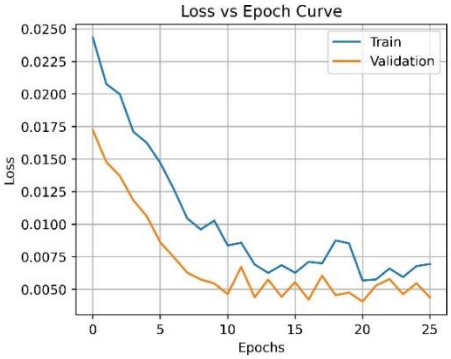
Fig. 20 Loss vs epoch curves for M_w-R_e data split in the ratio (TR:TS: VS= 70:15:15)



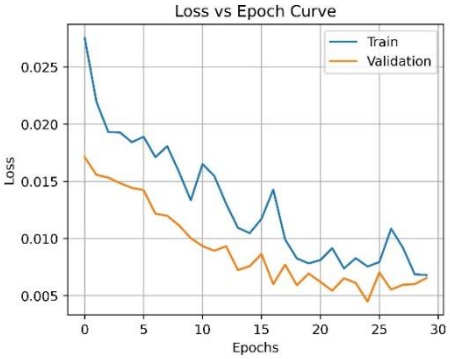
(a) XGB model



(b) LSTM model

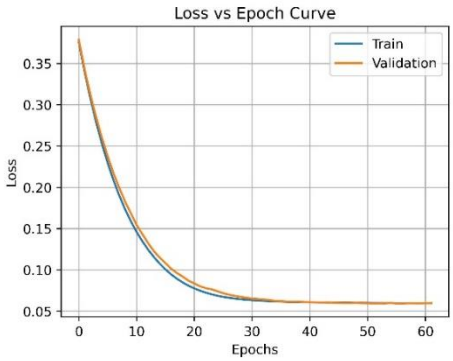


(c) BiLSTM model

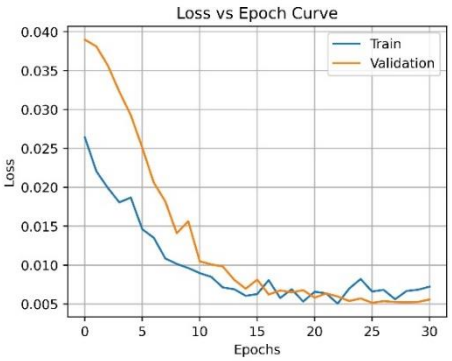


(d) CNN model

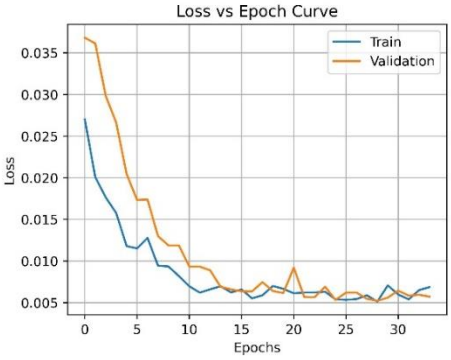
Fig. 21 Loss vs epoch curves for M_w-R_e data split in the ratio (TR:TS: VS= 60:20:20)



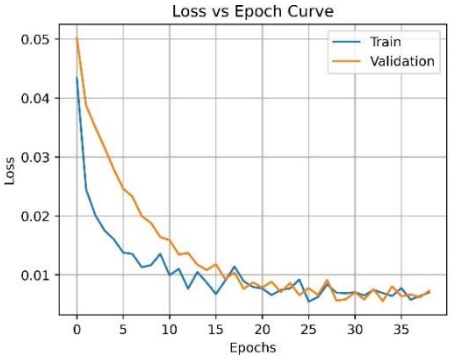
(a) XGB model



(b) LSTM model

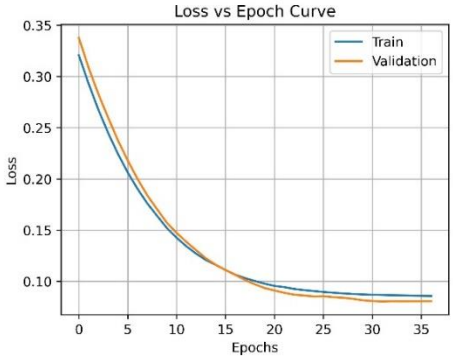


(c) BiLSTM model

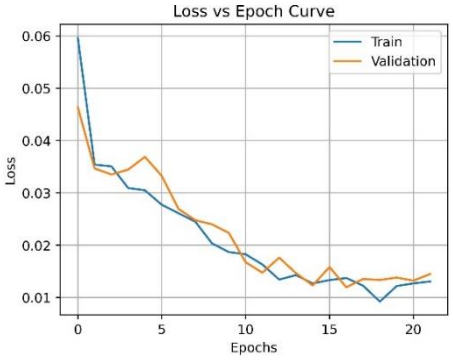


(d) CNN model

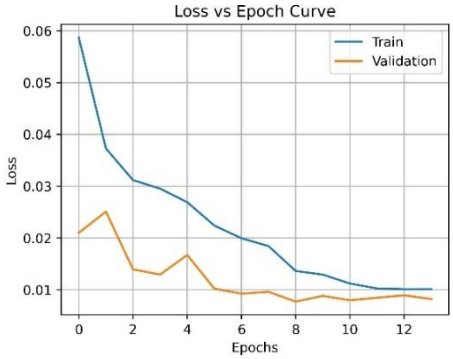
Fig. 22 Loss vs epoch curves for M_w-R_e data split in the ratio (TR:TS: VS= 80:10:10



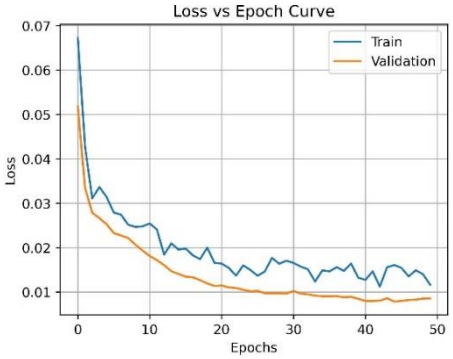
(a) XGB model



(b) LSTM model

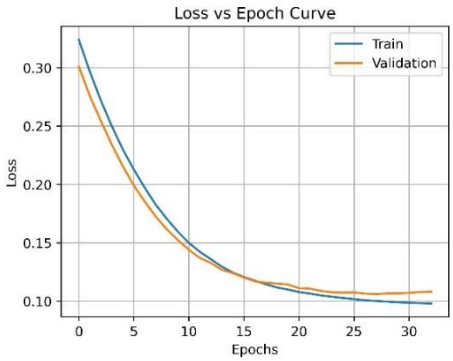


(c) BiLSTM model

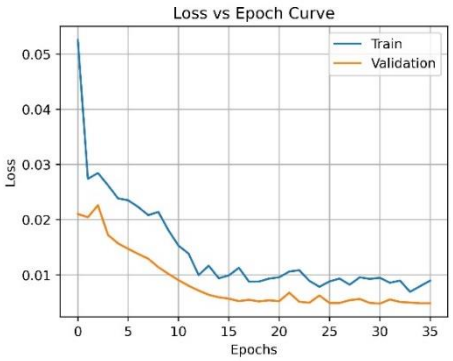


(d) CNN model

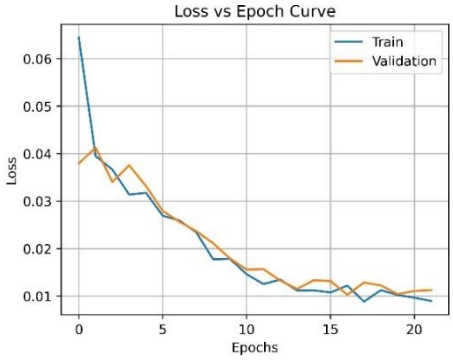
Fig. 23 Loss vs epoch curves for M_w-R_f data split in the ratio (TR:TS: VS= 70:15:15)



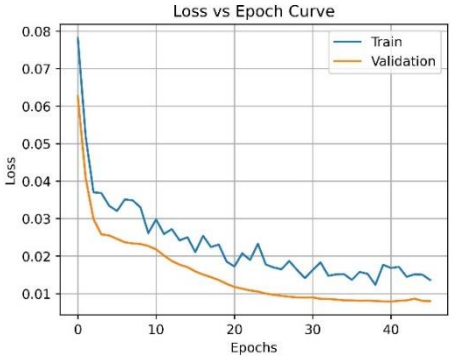
(a) XGB model



(b) LSTM model

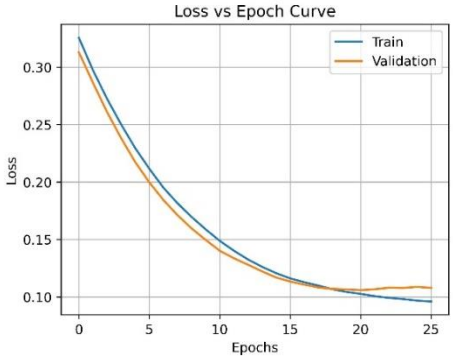


(c) BiLSTM model

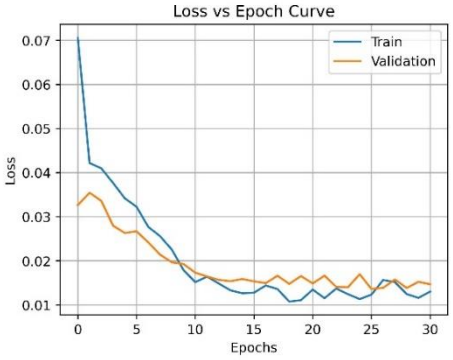


(d) CNN model

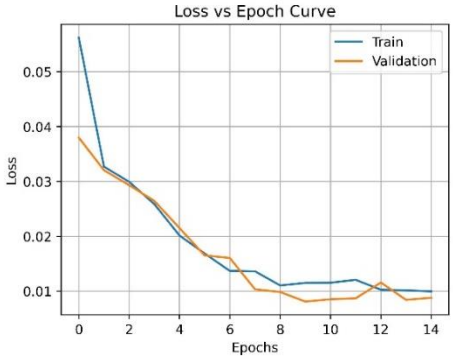
Fig. 24 Loss vs epoch curves for M_w-R_f data split in the ratio (TR:TS: VS= 60:20:20)



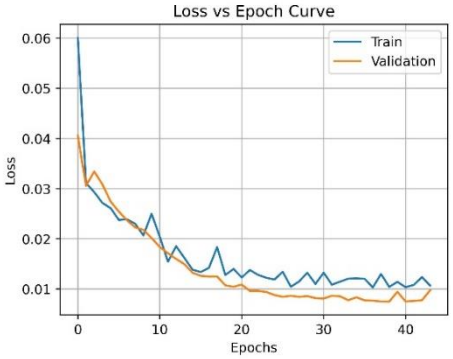
(a) XGB model



(b) LSTM model

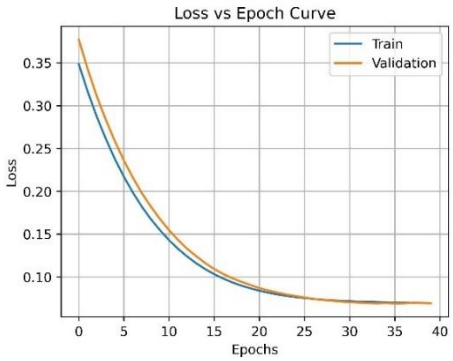


(c) BiLSTM model

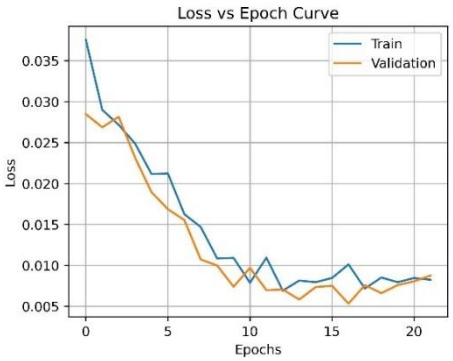


(d) CNN model

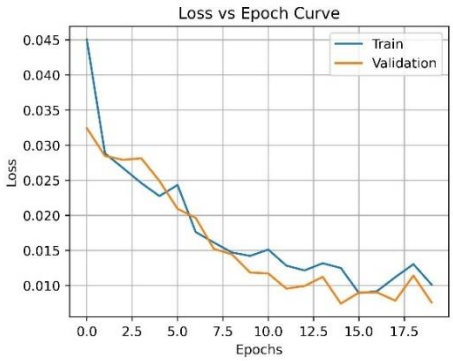
Fig. 25 Loss vs epoch curves for M_w-R_f data split in the ratio (TR:TS: VS= 80:10:10)



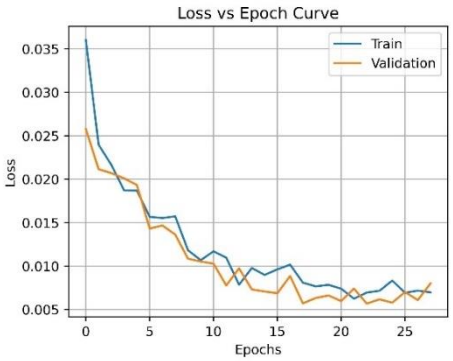
(a) XGB model



(b) LSTM model

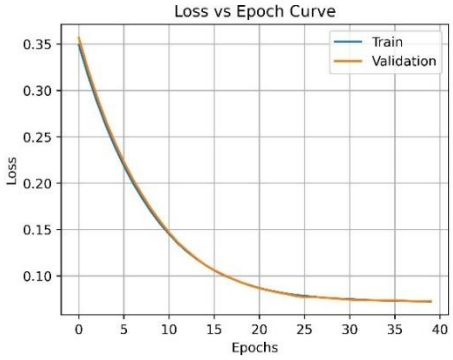


(c) BiLSTM model

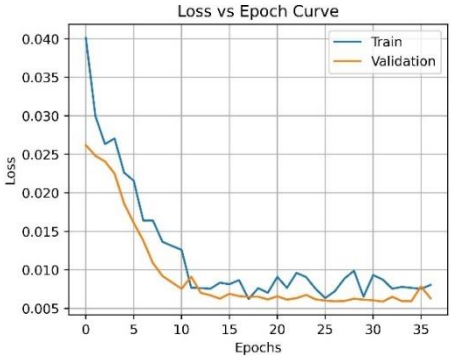


(d) CNN model

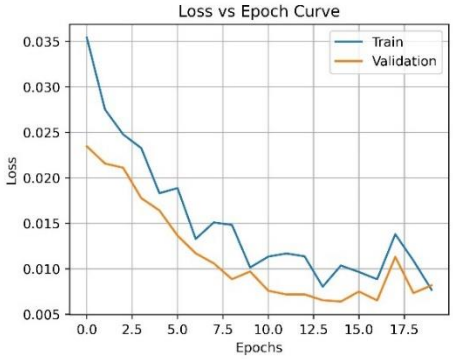
Fig. 26 Loss vs epoch curves for M_w-R_h data split in the ratio (TR:TS: VS= 70:15:15)



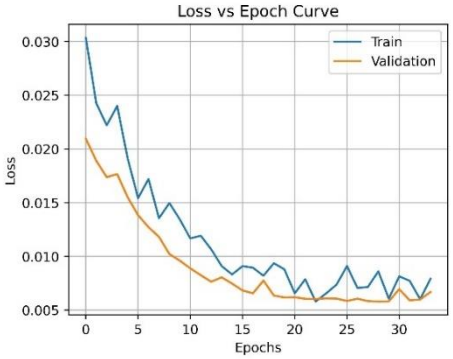
(a) XGB model



(b) LSTM model

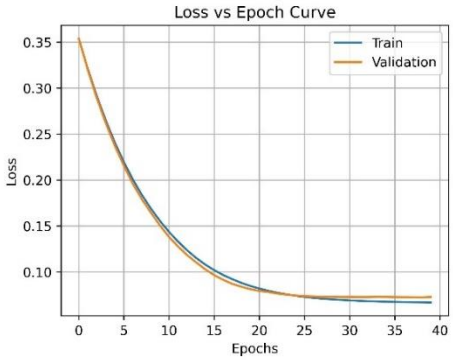


(c) BiLSTM model

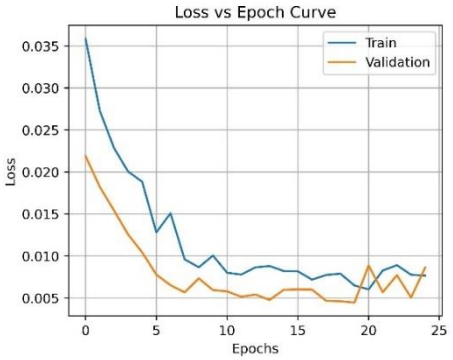


(d) CNN model

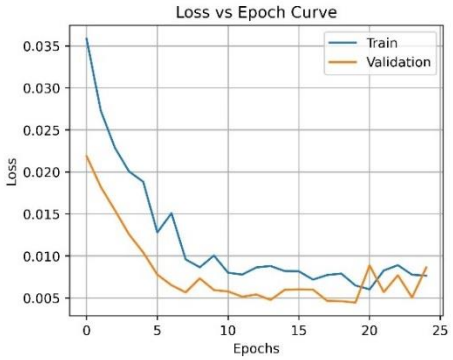
Fig. 27 Loss vs epoch curves for M_w-R_h data split in the ratio (TR:TS: VS= 60:20:20)



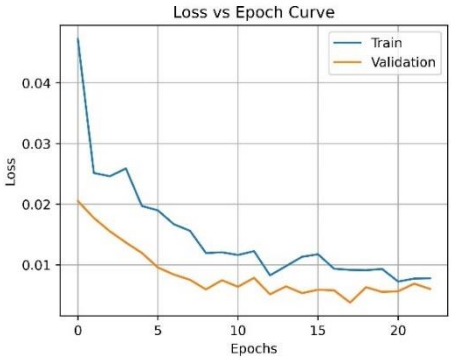
(a) XGB model



(b) LSTM model



(c) BiLSTM model



(d) CNN model

Fig. 28 Loss vs epoch curves for M_w-R_h data split in the ratio (TR:TS: VS= 80:10:10)