The 2025 World Congress on
Advances in Structural Engineering and Mechanics (ASEM25)
BEXCO, Busan, Korea, August 11-14, 2025

Data-driven approaches for structural response estimation under earthquakes

*Sung Hei Luk[1)]

[1)] Department of Construction Environment and Engineering, THEi, Hong Kong, China
[1)] henrylsh@thei.edu.hk

ABSTRACT

This paper investigates using machine learning (ML) to predict the structural responses of buildings during earthquakes. It highlights the potential of data-driven methods to provide quick estimations. In the study, numerous building models are analyzed with non-linear time-history analysis using past earthquake histories with a variety of magnitudes. Input parameters and output responses are consolidated as training and testing datasets for development of ML prediction models. In addition to the classical ML algorithms, the feasibility of using ensemble artificial neural networks (ANN), transformer-based architectures, and convolutional neural network (CNN) algorithms is explored in this study.

Evaluations show that data-driven methods can effectively estimate seismic responses of buildings. Boosting methods, such as XGBoost and AdaBoost, perform well with reasonable computational efforts. The results show that vanilla ANN models can be improved by using ensemble techniques, transformers, and CNN algorithms. A transformer-based model is proposed based on feature embedding, input masking, and transformer modules. The predictive performance of the proposed model outperforms most of the ensemble algorithms in seismic response prediction.

1. INTRODUCTION

Seismic vulnerability of buildings and structures is an important task in structural engineering to protect human life and identify the possibility of building collapse during earthquakes. It involves understanding how different structures response under future earthquakes and developing strategies to mitigate potential hazards. By evaluating seismic responses of buildings, they can design buildings that are more resilient and capable of withstanding earthquakes, thereby reducing the risk of catastrophic failures. Engineers use various analytical methods, such as non-linear static (pushover) analysis or non-linear time-history analysis (NLTHA) to predict the structural responses of buildings. Pushover analysis is a simplified method to rapidly estimate the capacities and failure mechanisms of buildings under earthquakes. According to this method,

[1)] Lecturer

earthquake actions are approximated by lateral forces with specified load patterns applied monotonically to buildings (Saleemuddin and Sangle, 2017). The performance of buildings can be measured based on different approaches, such as coefficient methods or spectrum methods. Some modified pushover methods, such as modal pushover analysis, are developed to improve the accuracy of the analysis. On the other hand, non-linear dynamic analysis is known as the most accurate method to assess seismic performance of buildings. This method involves solving the equations of motion for multiple degrees of freedom (MDOF) systems subjected to ground motions (Nazari and Saatcioglu, 2017). However, the computation efforts are much higher than pushover methods and therefore, it is difficult to apply in large-scale projects owing to the hardware limitations and time constraints.

Recently, data-driven approaches via machine learning methods have been widely adopted to handle complicated tasks particularly in the areas of computer vision and natural language processing (Liu, 2020). These successful applications mainly result from the development of efficient machine learning algorithms, enhancement of hardware and rich relevant training data. In civil engineering, data-driven methods are successfully applied in different disciplines, such as construction management, structural health monitoring, transportation, image recognition, architectural design, etc. (Pan and Zhang, 2021; Thai, 2022). Data-driven methods can be also adopted in seismic vulnerability assessment to provide rapid estimation on seismic risk of buildings (Wen et al., 2022; Mahmoudi et al., 2023; Bhatta et al., 2024; Demir et al., 2024).

In literature, prediction models were developed for classification (e.g., predicting damage level) and regression tasks (e.g., predicting inter-story drift) (Géron, 2019). Different machine learning algorithms were adopted, such as artificial neural networks, support vector machines, decision trees, and ensemble methods. Wen et al. (2022) developed regression models to predict inter-story drift ratio and peak floor acceleration using StruNet, which is based on neural network algorithm. Earthquake records and selected structure parameters, such as number of stories, story height, column spacing and transverse span, were used as input features. Mahmoudia et al. (2023) investigated several algorithms for damage classification and the results showed that K-Nearest Neighbor algorithm was the most accurate algorithm. The hyperparameters of models were adjusted using Bayesian optimization algorithm. Some damage features, such as Arias intensity, cumulative absolute velocity, spectral acceleration, energy ratio and drift, were considered. Bhatta et al. (2024) selected several algorithms to develop models for damage assessment of RC buildings. It was found that random forests were the most effective algorithm. The prediction model could be applied on a regional scale to predict the damaged classes of RC buildings, based on simple parameters such as number of stories, story height, height of building, fundamental period, building age and plan configuration, peak ground acceleration and velocity. Demir et al. (2024) selected a total of 21 ground motion intensity measures to develop prediction models to predict maximum drift ratio of buildings. Regular and irregular RC frames were considered. The study found that both random forest and XGBoost algorithms could achieve comparable efficiency and accuracy. In short, selection of appropriate algorithms and suitable input features are key factors for developing prediction models.

Among all machine learning algorithms, artificial neural networks (ANNs) or deep learning approaches are known as very effective methods to handle complex tasks in the real world, such as image and text generation, since ANNs and deep learning are good in processing data with spatial or semantic relationships. On the other hand, the ability of ANNs to handle tabular data is not widely recognized. Tabular data is a common data structure where information is organized in a table format. Rows represent different samples, while columns represent various features with discrete (i.e., categorical) or continuous (i.e., numerical) values. Previous studies revealed that ensemble methods, such as tree-based boosting methods, performed better in handling tabular data compared with vanilla ANNs models (Shwartz-Ziv and Armon, 2021; Luk, 2023; Borisov et al., 2024). Recently, some studies were conducted using transformer-based models to improve the performance of ANN models on tabular data. Arik and Pfister (2019) introduced TabNet, which utilizes sequential attention mechanisms for instance-wise feature selection to effectively process tabular data. Huang et al. (2020) developed TabTransformer based on self-attention algorithms to process tabular data. In the model, categorical features were embedded and passed through transformer modules. Gorishniy et al. (2021) proposed the use of ResNet-like architecture and feature tokenizer transformer (FT-Transformer) architecture on tabular data. The latter method transformed all features before sending them to the transformer modules. Gorishniy et al. (2022) explored two techniques, including piecewise linear encoding (binning) and periodic activation functions, for embedding numerical features to enhance the capabilities of deep learning models. Such techniques could reduce the gaps between deep learning methods and boosting methods. Somepalli et al. (2022) developed SAINT model, where row and column attentions and embedding of all input features with MLP were involved. The study showed that ANN models could perform competitively with tree-based boosting methods. Chen et al. (2024) proposed ExcelFormer by introducing semi-permeable attention module and gated linear units in transformer architecture and data augmentation methods to enrich datasets. The results showed that the proposed techniques could greatly improve the predictive performance of ANN models. Instead of transformer-based approaches, other techniques, such as the use of tabular-to-image transformation techniques (Briner et al., 2023) or combining graph-structure data format and graph neural network (Alkhatib et al., 2024), are also possible methods to improve ANNs on handling tabular data. More studies are therefore needed to investigate suitable data structure, data pre-processing techniques and machine learning algorithms in these applications.

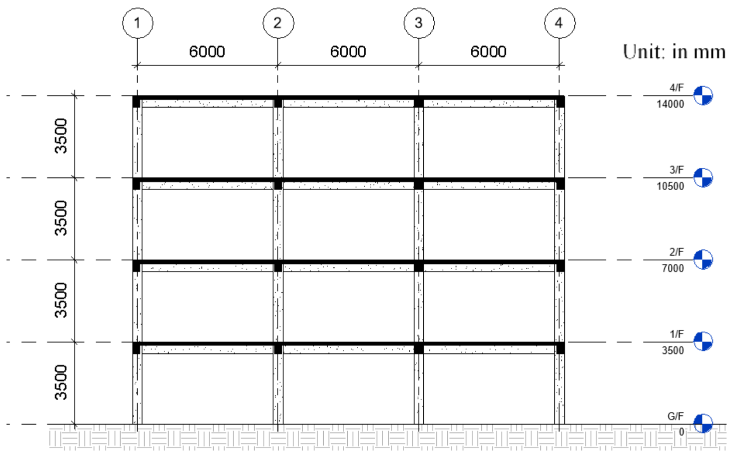
This study aims to investigate suitable machine learning algorithms to develop prediction models for buildings under earthquakes. A variety of machine learning algorithms are adopted to study their efficiency, suitability, and accuracy in prediction or regression problems. Techniques for enhancing the performance of ANNs, such as ensemble methods, embedding techniques, network architectures, and the influences of the corresponding hyperparameters, are explored in detail.

2. METHODOLOGY / DATA-DRIVEN METHODS

The methodology of development of prediction models involves several key steps, including dataset generation via non-linear time-history analysis, selection of effective machine learning algorithms, and training and evaluation of the selected machine learning models. Details of each step are presented in the following sections.

2.1 Non-linear time-history analysis

In this study, a total of 40 building models were modelled and designed based on code of practice for structural use of concrete in Hong Kong (HKCP2013). Different design variables, including height and width of buildings, dimension of structural members, material property and reinforcement ratio, were considered. Models were developed with heights ranging from 14 m to 70 m and widths ranging from 18 m to 30 m. Concrete with elastic modules of 23.7 kN/m^2, 25.1 kN/m^2, 27.7 kN/m^2 and 30 kN/m^2, respectively, were used. The elastic modules and yield strength of reinforcement were 200 kN/m^2 and 500 kN/m^2, respectively. The steel ratio of beams ranges from 0.71% to 2.38%, while the steel ratio of columns ranges from 1.20% to 4.36%. A sample 4-storey building is shown in Fig. 1. Variety of beam and column sizes were considered.



Model parameters
Height = 14 m
Width = 18 m
Storey height = 3.5 m
Bay width = 6 m
Concrete =
Beam (hxb) = 500x300 mm
Column = 400 mm square

Fig. 1 Sample of 4-story building model

Incremental dynamic analysis (IDA) was adopted to assess the seismic performance of buildings (Vamvatsikos and Cornell, 2002). All the analyses were carried out using ETABS, which is a widely used finite element software for tall buildings analysis and design in practice. IDA involved conducting extensive non-linear time-history analyses (NLTHA) with scaled intensity measures, where peak ground acceleration (PGA) was selected as the intensity measure. In this study, 16 selected past earthquakes were selected mainly from ATC-63 ground motion set (FEMA P-695). The values of PGA were scaled from 0.1g to 1.0g to simulate earthquakes with different magnitudes. In the finite element models, material non-linearity was modelled using plastic hinges at member's ends. Geometric non-linearity was considered via activating the second-order order P-delta effect. The model was first loaded under gravity load with combination of 1.0G$_k$ and 0.3Q$_k$, where G$_k$ and Q$_k$ are the dead load and imposed load, respectively. After that, ground accelerations were applied to the model to simulate the earthquake attack. Rayleigh damping with damping ratio of 5% was adopted in analyses. The structural responses, such as top drifts and inter-story drift ratios, were determined by finite element analysis. The input parameters and outputs were consolidated in tabular format for training ML models in the next step.

2.2 Classical and ensemble machine learning algorithms

Selection of effective and efficient machine learning (ML) algorithms is a crucial task for the development of prediction models. Prediction models are normally developed using supervised learning, in which the ML models are trained using labeled datasets. This study aims to develop regression models to predict seismic responses of buildings. Basic ML algorithms, ensemble methods and artificial neural network-based models were considered. Input features $\mathbf{X} = (x_1, x_2, \dots)$ and target y obtained from previous IDA were consolidated for training ML models based on a variety of algorithms. All the ML models were developed in Python version 3.11. Several python libraries, such as scikit-learn (Pedregosa et al., 2011), xgboost, lightgbm and tensorflow, were utilized to develop prediction models. The hyperparameters of ML models were evaluated using grid search method. The suitability and performance of different ML algorithms were investigated and compared.

Several basic ML algorithms, including linear regression, stochastic gradient descent (SGD), decision tree (DT), k-nearest neighbor (KNN) and support vector regression (SVR) were adopted as baseline models for comparison. Details about the description of these baseline models can be found in the previous study (Luk, 2025). The hyperparameters of basic ML models are presented in Table 1.

Table 1. Hyperparameters for the selected basic ML models

Model	**Hyperparameters**
Linear regression	-
SGD	loss=squared_error, penalty='l2', alpha=0.0001, tol=0.0001
DT	criterion=squared_error, max_depth=14
KNN	n_neighbors=2, weights=distance
SVR	C=20, gamma=scale, kernel=rbf

Ensemble methods aim to improve the performance and generality of prediction models by combining multiple ML algorithms together sequentially or in parallel (Mienye and Sun, 2022). Depending on the combining approaches, ensemble methods can be classified into voting methods, stacking methods, bagging methods and boosting methods (Luk, 2023). In this study, several ensemble methods, including random forest (Ho, 1995), gradient boosted decision tree (Friedman, 2001), adaptive boosting (Freund and Schapire, 1995), XGBoost (Chen and Guestrin, 2016), and LightGBM (Ke et al., 2017) were adopted. Table 2 presents the hyperparameters of selected ensemble models. Below are some descriptions of boosting algorithms.

<u>Gradient boosted decision tree</u> (GBDT) is known as an efficient and accurate algorithm based on gradient boosting methods (Friedman, 2001). In this method, many weak learners are combined sequentially to minimize errors from previous weak learners. In GBDT, DT is selected as the weak learners. The concept of gradient descent is used to minimize the selected loss function (residual) for the next weak learner. By combining

all the predictions together, the accuracy of the final prediction can be greatly enhanced. Hyperparameters, such as learning rate, number of estimators, and DT-related parameters, need to be adjusted.

Table 2. Hyperparameters for the selected ensemble models

Model	**Hyperparameters**
Random forest (RF)	n_estimators=350, criterion=squared_error
GBDT	loss=squared_error, learning_rate=0.2, n_estimators=500, criterion=friedman_mse
AdaBoost	estimator=decision tree, n_estimators=100, learning_rate=1.0, loss=linear
XGBoost	booster=gbtree, eta=0.2, lambda=0.01, alpha=0.01, n_estimators=200, max_depth=30, subsample=0.8, colsample_bytree=0.8
LightGBM	boosting_type=gbdt, num_iterations=100, lambda_l1=0.005, lambda_l2=0.0001, num_leaves=50, feature_fraction=0.80, bagging_fraction=0.80
Stacking	basic_learner=[GBDT, AdaBoost, XGBoost] meta_learner=linear

eXtremely gradient boosting (XGBoost) is an accurate and efficient algorithm developed based on the concept of GBDT (Chen and Guestrin, 2016). The basic framework of XGBoost is similar to GBDT, as discussed in the previous paragraph. In addition to the classical gradient boosting algorithms, XGBoost introduces second-order derivatives, bagging algorithm for sub-sampling and regularization to improve performance and to reduce overfitting, which enhance the generality of the prediction models in practice.

Light gradient boosting machine (LightGBM) is a fast and high-performance gradient boosting model based on GBDT (Ke et al., 2017). Unlike XGBoost that uses level-wise tree growth, LightGBM uses leaf-wise tree growth and histogram-based algorithms to reduce memory usage and computation effort, which makes LightGBM generally faster than XGBoost.

Adaptive boost (Adaboost) is based on boost algorithms but slightly different from gradient boosting algorithms. Instead of using gradient descent, Adaboost assigns weights to all samples, each with the same weight initially. After training, higher weights are assigned to the misclassified samples so that the errors from the previous weak learner can be minimized in the next weak learner (Freund and Schapire, 1995). The predictions are then combined to obtain the final outputs.

2.3 Artificial neural network (ANN)

ANNs and deep learning are widely used nowadays in different complex tasks, particularly in generative AI, image recognition and large language models. Typical examples of deep learning algorithms include convolutional neural networks (CNNs) for image processing and transformer models for natural language processing (NLP). The potential of ANNs for tabular data is often overlooked, though deep learning may help to enhance their performance on such problems. In this study, four types of ANN models, including vanilla ANNs, ensemble ANNs, transformer-based models, and CNN-based models were explored and compared with basic and ensemble methods. Descriptions of these models are presented in the next section.

Multi-layer perceptron (MLP), or feedforward neural network, is a simple form of ANNs in which the model is formed by fully connecting nodes together. At a particular node within a hidden layer, the information is manipulated via trainable weights (w_i) and bias (b_i) and then passing to non-linear activation functions, such as rectified linear unit (relu) and Gaussian error linear unit (gelu). The size of the input layer depends on the number of input features, while the size of the output layer depends on the target (or label). For regression problems, the size of output layer is one with no activation function. Fig. 2 shows a sample network architecture with three input features and two hidden layers used for regression purposes. The weights and bias between nodes are updated through backpropagation, which is based on the differences between predictions and given data. The key hyperparameters in MLP include the number of hidden layers and the number of units for each hidden layer. One to three hidden layers with units of 16 to 128 per layer were considered in this study.

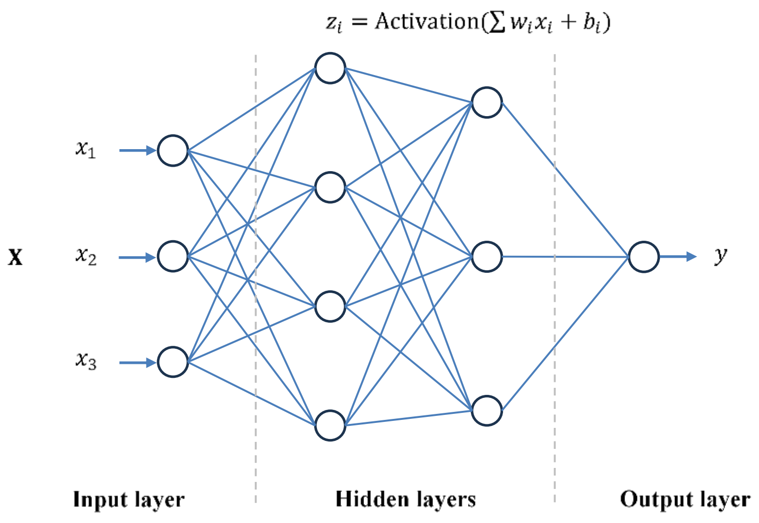


Fig. 2 Multi-layer perceptron with three numerical input features as example

Ensemble artificial neural networks are an advanced technique by combining multiple neural network models together based on ensemble methods, such as stacking and boosting methods (Luk, 2025). The advantages are to enhance the overall predictive performance and to reduce bias.

In this study, stacking and gradient boosting algorithms were adopted to enhance the ANNs performance to process tabular data. Ensemble ANN models based on stacking method (StackANNs) were developed by using three vanilla MLP models as base learners. Each MLP model was created using one to two hidden layers, each with 16, 32 or 64 units. The predictions from each base learner were concatenated and sent to a meta-learner to generate the final predictions. Linear model was adopted as the meta-learner.

For the ensemble ANNs models based on gradient boosting algorithm (GBANNs), MLP models were arranged sequentially. Each MLP model was trained using the inputs $\mathbf{X}$ and residuals $r = y - \hat{y}$, where y is the target and $\hat{y}$ is the prediction from previous weak learners. The final predictions were computed by summing up all the predictions together with a selected learning rate. A total of 20 weak learners were adopted for GBANN models in this study.

Transformer-based models are based on the attention model (Vaswani et al., 2017), which is a significant breakthrough in NLP and LLM. In literature, attention mechanisms utilize scaled dot-products between query, key and value matrices to learn the relationships between words or tokens in a paragraph to get better results in generative models. Mathematically, it can be computed as

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}\right)\boldsymbol{V} \tag{1}$$

where $\boldsymbol{Q}$, $\boldsymbol{K}$, $\boldsymbol{V}$ are the query, key and value matrices, respectively, which are formed using trainable weights matrices and input vectors, and d_k is related to the dimension of key vectors. In addition, attention mechanisms support parallelization to enhance training efficiency.

To improve the ANN ability to process tabular data, transformer architecture was adopted in this study. TabTransformer (Haung et al., 2020), a transformer-based model for tabular data, was based on the original attention model (Vaswani et al., 2017) and only categorical features were embedded. FT-Transformer (Gorishniy et al., 2023) embeded all numerical and categorical features using linear operations. In this study, both categorical and numerical features were embedded separately with different embedding methods. The modified model is called Tabular Embedding Transformer (TET). Fig. 3 illustrates the architecture of the proposed TET model.

Similar to TabTransformer, categorical features were embedded, with pre-defined embedding dimensions. Moreover, numerical features were embedded into a vector space using selected embedding methods. The proposed model supported several embedding methods, including periodic activation functions, linear operation and binning embedding. For example, the embedded numerical feature $E(x)$ based on periodic activation functions (Tancik, et al., 2020; Gorishniy et al., 2022) were defined as the following operation,

$$E(x) = \text{concat}[\sin(v), \cos(v)] \tag{2}$$

where $v = [2\pi c_1 x, 2\pi c_2 x, \dots]$ and c_i are trainable parameters. This embedding method was to transform numerical features using harmonic functions with suitable frequencies that were determined via training. The size of the embedded features depended on the pre-defined embedding dimension.

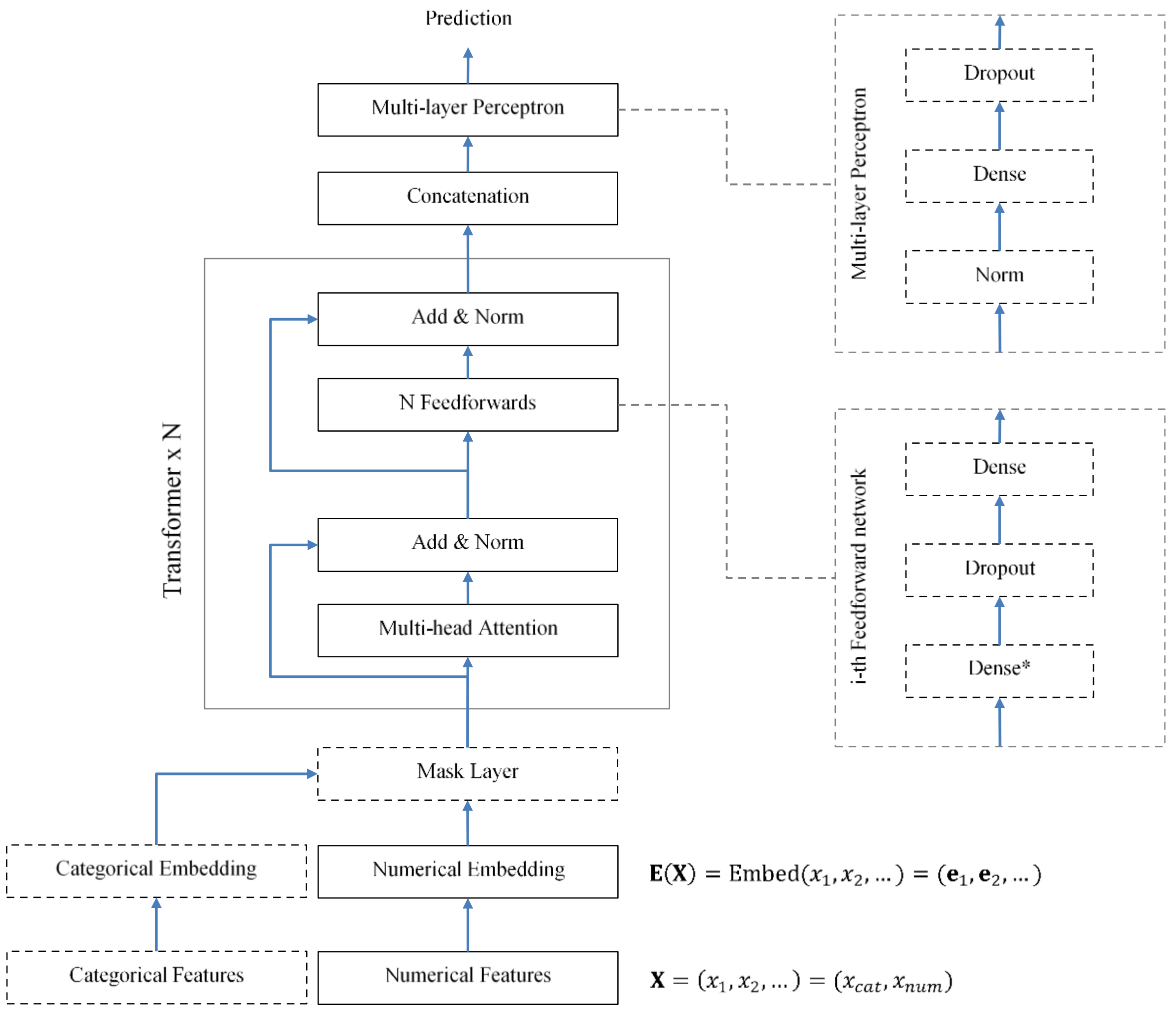


Fig. 3 Illustration of Tabular Embedding Transformer (TET) architecture

In the revised model, a mask layer, which is a layer with multiplication operation between each embedded feature and learnable parameter, was introduced to take account of the importance of different input features. The embedded features were then passed to the transformer modules, which were composed of multi-head attention layers, normalization layers, and feedforward layers. Different from classical transformer modules, three feedforward layers with different networks' architectures (e.g. number of units and hidden layers) were connected in parallel in this study, following the logic of ensemble method to reduce bias. After that, the results from transformer modules were concatenated and passed to the multi-layer perceptron layer to generate the final outputs. The hyperparameters for the TET models are presented in Table 3. Studying the effects of embedding and transformer to ANN models was an important scope in this study. Fig. 4 shows the network for model TET-P16T1M32-1.

Table 3. Network architectures of the TET models

Model	**Networks' Architecture**		
	Embedding dimension	Numbers of transformer	MLP arch. (units per layer)
TET-P16T1M32-1*	16	1	32
TET-P16T1M32-2	16	1	32,32
TET-P16T2M32-1	16	2	32
TET-P16T3M32-1	16	3	32
TET-P32T1M32-1	32	1	32
TET-P32T2M32-1	32	2	32
TET-P32T3M32-1	32	3	32

*TET-P**x**T**y**M**p**-**q**, where
P**x**: P = Periodic embedding; x = embedding dimension
T**y**: T = Transformer; y = number of transformer
M**p**-**q**: M = MLP; p = units per layer, q = number of hidden layers

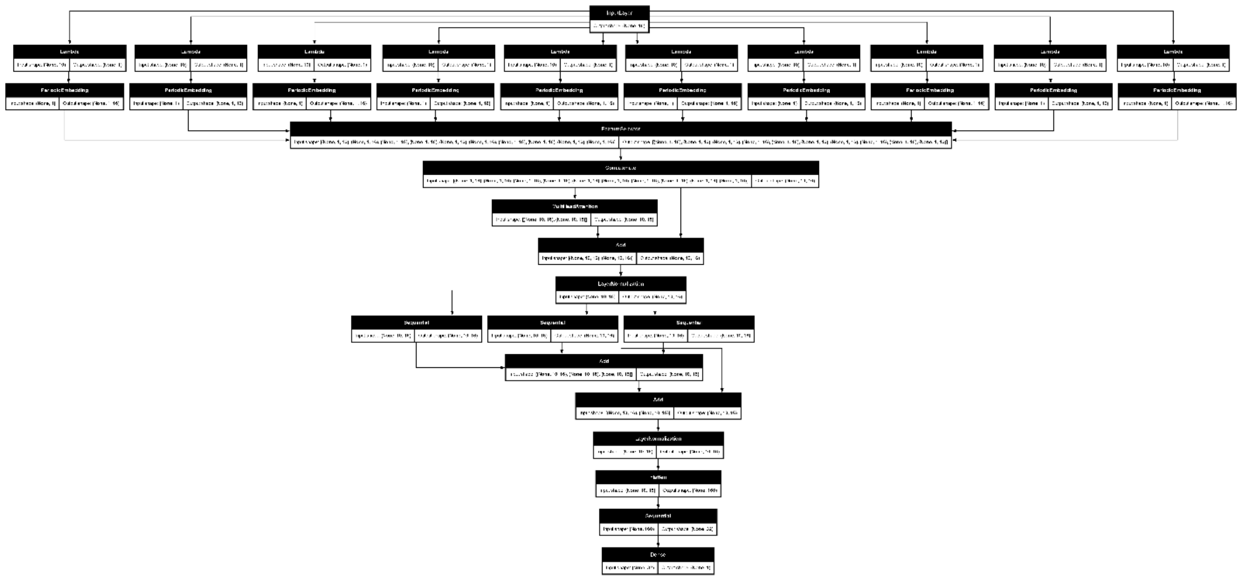


Fig 4. Sample TET model: TET-P16T1M32-1

CNN-based models are the other approaches to enhance the performance of vanilla ANN models to handle tabular data (Sharma et al., 2019; Zhu et al., 2021; Sharma and Kumar, 2022). This method first transforms tabular data into 2-dimensional images based on different tabular-to-image algorithms, such as data wrangling methods, dimensionality reduction-based methods, feature permutation-based methods and embedding methods (Alenizy and Berri, 2025). The images are then sent to CNN models for classification or regression purposes.

In this study, the CNN-based model first transformed tabular data into image datasets using novel algorithm for convolving tabular data (NCTD) method (Alenizy and Berri, 2025). This method transformed one sample (i.e., one row in tabular data) into a 2D image by data duplication and rotation, as shown in Fig. 5. Hence, tabular data with N samples gave N $(d\text{x}d)$ images, and each image had unique spatial structure. The images were then expanded to $2d\text{x}2d$ by stacking itself vertically and horizontally to enrich the spatial structure. Next, the images were sent to 2D-CNN kernels and maximum pooling layers to identify important features. In this study, two 3x3 kernels and 2x2 pooling layers were adopted in the CNN architecture. The fully connected layer involved one hidden layer only. Table 4 summarizes the model hyperparameters.

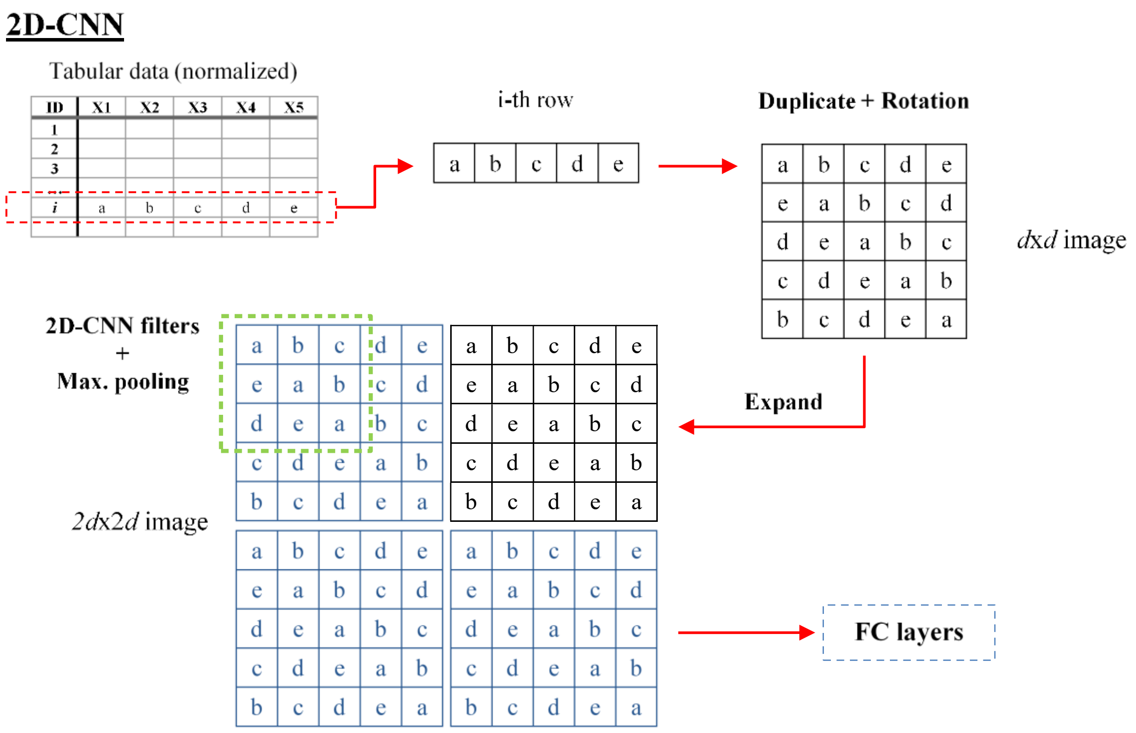


Fig 5. Tabular to image transformation (NCTD method)

Table 4. Networks' architecture of the CNN-based models

Model	**CNN Layers**	**FC Layers**

2D-CNN1-M16	Conv2D[filters=32, kernel_size=(3,3), activation='relu']
	MaxPooling2D[pool_size=2]
	Conv2D[filters=16, kernel_size=(3,3), activation='relu']
	MaxPooling2D[pool_size=2]
		Dense[units=16, activation='relu']
		Dense[units=1]

2D-CNN2-M16	Conv2D[filters=64, kernel_size=(3,3), activation='relu']
	MaxPooling2D[pool_size=2]
	Conv2D[filters=32, kernel_size=(3,3), activation='relu']
	MaxPooling2D[pool_size=2]
		Dense[units=16, activation='relu']
		Dense[units=1]

2D-CNN1-M32	Conv2D[filters=32, kernel_size=(3,3), activation='relu']
	MaxPooling2D[pool_size=2]
	Conv2D[filters=16, kernel_size=(3,3), activation='relu']
	MaxPooling2D[pool_size=2]
		Dense[units=32, activation='relu']
		Dense[units=1]

2D-CNN2-M32	Conv2D[filters=64, kernel_size=(3,3), activation='relu']
	MaxPooling2D[pool_size=2]
	Conv2D[filters=32, kernel_size=(3,3), activation='relu']
	MaxPooling2D[pool_size=2]
		Dense[units=32, activation='relu']
		Dense[units=1]

2.4 Validation of the proposed models

Large-scale datasets available in sklearn, Kaggle or OpenML were used to verify the performance of the modified models before developing the prediction models for estimating the seismic responses of buildings. Descriptions of the datasets are summarized in Table 5. Three datasets were for classification, while the remaining three were for regression. Before training the model, the numerical data was first normalized using max-min normalization. The categorical data was transformed into numerical numbers using ordinary encoding (i.e. discrete values 0, 1, 2, …, k). The performance was evaluated using k-fold validation procedures with k = 5.

Table 5. Datasets for validation

Index	**Name of Dataset**	**No. of Sample**	**No. of num**	**No. of cat**	**Task**	**Batch size**
1	Titanic[1]	890	5	2	Classification	16
2	Adult-census[2]	30162	6	8	Classification	128
3	Otto Group Products[3]	61878	93	0	Classification	256
4	House price[4]	1460	36	43	Regression	32
5	California housing[5]	20640	8	0	Regression	128
6	Ames Housing[6]	2929	37	43	Regression	32

[1] https://www.kaggle.com/datasets/pranjalyadav92905/titanic-eda-data
[2] https://www.openml.org/search?type=data&status=active&id=1590
[3] https://www.kaggle.com/competitions/otto-group-product-classification-challenge/data
[4] https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data?select=train.csv
[5] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_california_housing.html#sklearn.datasets.fetch_california_housing
[6] https://www.kaggle.com/datasets/prevek18/ames-housing-dataset

The performance of the proposed models (TET and 2D-CNN) and reference models (XGBoost, vanilla MLP and TabTransformer) is summarized in Table 6. Accuracy was selected as an evaluation metric for classification problems, while root mean square error (RMSE) was taken as metrics for regression problems. The results showed that the performance of the proposed TET models for both classification and regression problems was comparable or sometime better than XGBoost, showing good predictive ability. Moreover, the CNN-based models normally performed well for most of the datasets. Hence, they were used to develop prediction models to estimate seismic responses of buildings.

Table 6. Validation results for classification and regression (best results are underlined)

1.Classification (Metric: Accuracy)			
Model	**Name of Dataset**		
	Titanic	Adult-census	Otto Group Products
XGBoost	0.819	0.866	0.813
Vanilla MLP	0.788	0.834	0.792
TabTransformer	0.787	0.831	0.751
TET	<u>0.845</u>	<u>0.870</u>	<u>0.831</u>
2D-CNN	0.820	0.851	0.708

2.Regression (Metric: RMSE)			
Model	**Name of Dataset**		
	House price	California housing	Ames Housing
XGBoost	31418	<u>0.469</u>	24741
Vanilla MLP	39109	0.733	50243
TabTransformer	39634	0.742	34310
TET	<u>29487</u>	0.632	<u>22361</u>
2D-CNN	37375	0.625	36997

2.5 Input features and output label for prediction models

To develop ML models for predicting seismic responses of buildings and structures, representative datasets ($\boldsymbol{X}$,y) should be available. Numerous input features were considered in the former studies (Luk, 2023; Luk, 2025). Among all input features, five earthquake parameters and five structural parameters were finally selected in this study, including peak ground acceleration (PGA), peak ground velocity, (PGV), peak ground displacement (PGD), ratio between PGV to PGA (V/A), spectral acceleration at fundamental period $S_a(T_1)$, height of building (H), building width (W), maximum axial load ratio (v_d), moment of inertia of column (I_c) and fundamental period (T_1) for training and testing of ML models. For the output prediction, maximum inter-story drift ratio (MIDR), which is a widely used indicator in literature and design guidelines, was selected. The input features were normalized using max-min normalization.

In this study, k-fold validation procedure with k = 5 was adopted to assess the predictive performance of each ML model. This approach first divided the entire dataset into k (= 5) data subsets. 4 subsets were used for training and 1 subset was used for testing. At the end of the procedure, each subset of data was used for both training and testing to obtain unbiased model comparisons. After that, the average values of each metric were computed and reported.

3. RESULTS AND DISCUSSION

Four evaluation metrics, including mean square errors (MSE), mean absolute errors (MAE), root mean square errors (RMSE) and R2 score, were adopted to evaluate the performance of selected ML models. The following sections present the performance of basic ML models, ensemble models, vanilla ANN models, ensemble ANN models, transformer-based models, and CNN-based models, separately.

3.1 Performance of basic and ensemble algorithms

The performance of basic and ensemble models under consideration is summarized in Table 7. The results reveal that ensemble methods normally outperform the basic methods. Owing to the complex relationships between input and output variables, the performance of linear regression and stochastic gradient descent methods is unsatisfactory. The values of all evaluation metrics are generally worse than the other methods. The other basic methods (SVR, DT and KNN) can achieve reasonable performance, particularly SVR with MSE of 0.353, in which the performance is comparable to some ensemble models. The previous study (Luk, 2025) also found that support vector machines could achieve high accuracy for classification problems.

Ensemble methods generally have good performance in all metrics, as shown in Table 7. Boosting methods are known as very efficient and effective methods in literature. Among all boosting methods, AdaBoost, where DT is selected as the basic learner, performs the best with MSE of 0.259, followed by XGBoost with MSE of 0.293. Both models can achieve small errors and high R2 scores. The performance of GBDT and LightGBM is lower than XGBoost. On the other hand, the performance of RF, which is based on bagging method, is comparable to LightGBM.

Boosting methods can be improved by using the stacking method, which introduces a meta-learner to improve the predictions from base learners. In this study, XGBoost, AdaBoost and GBDT are selected as the base learners, while linear model is adopted as the meta-learner. The results show that higher R2 values and lower errors are observed compared with the other ensemble models. The results also indicate the effectiveness of boosting methods in handling tabular data.

Table 7. Results of basic and ensemble models (best results are underlined)

Model	**MSE**	**MAE**	**RMSE**	**R2**
Basic models				
Linear regression	1.072	0.579	1.033	0.686
Stochastic gradient descent	1.087	0.583	1.041	0.682
SVR	0.353	0.252	0.588	0.897
DT	0.637	0.371	0.797	0.813
KNN	0.429	0.340	0.654	0.874
Ensemble models				
Random forest (RF)	0.336	0.267	0.578	0.902
XGBoost	0.293	0.250	0.530	0.914
LightGBM	0.336	0.305	0.577	0.902
AdaBoost (DT)	0.259	0.243	0.508	0.924
GBDT	0.301	0.273	0.548	0.912
Stacking	0.258	0.239	0.507	0.924

3.2 Performance of vanilla MLP models

Table 8 shows that the overall performance of vanilla MLP models is lower than some basic models (e.g., SVR and DT) and ensemble models (XGBoost and RF), but better than linear regression in this problem. In terms of computational efforts for training and testing, MLP models normally take longer time compared with basic and ensemble models. The performance of vanilla MLP models depends on the model's architecture. The best vanilla MLP model is MLP-64,64 with two hidden layers and 64 units per layer, and the corresponding MSE is 0.562. The results also indicate that increasing numbers of hidden layers and units per layer are normally helpful to improve model performance. For MLP models with a single hidden layer, the MSE decreases from 0.770 (MLP-16) to 0.655 (MLP-128) when the units per layer increase from 16 to 128. This is because more units are helpful in capturing the non-linear relationship between input features and targets. Similar trends are observed when the numbers of hidden layers increase, except model MLP-16,16. However, the overall performance of all vanilla MLP models is still lower than boosting methods.

Table 8. Results of vanilla MLP models (best results are underlined)

Model	**MSE**	**MAE**	**RMSE**	**R2**
MLP-16	0.770	0.540	0.874	0.775
MLP-16,16	0.856	0.576	0.925	0.749
MLP-16,16,16*	0.720	0.551	0.846	0.789
MLP-32	0.762	0.533	0.870	0.777
MLP-32,32	0.756	0.530	0.867	0.779
MLP-64	0.689	0.526	0.828	0.798
MLP-64,64	0.562	0.474	0.748	0.835
MLP-128	0.684	0.562	0.825	0.800
MLP-128,128	0.655	0.510	0.806	0.807

*MLP-16,16,16 represents vanilla MLP model with three hidden layers and 16 units per layer

3.3 Performance of ensemble ANN models

Table 9 presents the performance of ensemble ANN models. For StackingANN models, their performance is better than vanilla MLP models presented in the previous section. The R2 score of MLP-16 is 0.775, while the value increases to 0.830 for StackANN-3(16)-L, which is developed by stacking three vanilla MLP models with 16 units together. Similar trends are found for the other models. In addition, the best model is StackANN-3(64,64)-L, in which the performance is comparable with some ensemble models (RF and LightGBM), with a R2 score of 0.906.

The performance of GBANN models has also improved compared with vanilla MLP models. The R2 score of MLP-32 is 0.777, while the value increases to 0.823 for GBANN-N20-(32), which is formed by connecting 20 vanilla MLP models sequentially. However, GBANN models take much longer training since they involve training large numbers of MLP models. In addition, GBANN models are under-performed compared with ensemble models. In short, combining ensemble techniques with ANNs can enhance performance of neural network models, but the performance is normally poor than classical ensemble methods (e.g., XGBoost and AdaBoost) for handling tabular data.

Table 9. Results of ensemble ANN models (best results are underlined)

Model	**MSE**	**MAE**	**RMSE**	**R2**
Stacking methods				
StackANN-3(16)-L	0.580	0.377	0.758	0.830
StackANN-3(32)-L*	0.557	0.370	0.743	0.837
StackANN-3(64)-L	0.477	0.352	0.688	0.860
StackANN-3(16,16)-L	0.532	0.371	0.726	0.845
StackANN-3(32,32)-L	0.410	0.355	0.639	0.880
StackANN-3(64,64)-L	0.321	0.303	0.563	0.906
Gradient boosting				
GBANN-N20-(16)	0.683	0.430	0.824	0.800

GBANN-N20-(32)[#]	0.605	0.410	0.777	0.823
GBANN-N20-(64)	0.547	0.391	0.738	0.840
GBANN-N20-(16,16)	0.660	0.472	0.811	0.807
GBANN-N20-(32,32)	0.564	0.443	0.749	0.835
GBANN-N20-(64,64)	<u>0.441</u>	<u>0.390</u>	<u>0.662</u>	<u>0.871</u>

[*] StackANN-3(32)-L represents stacking ANN model with 3 MLP-32 as the base learners and L represents linear model as meta learner
[#] GBANN-N20-(32) represents GBANN model with 20 numbers of estimators and MLP-32 as weak learner

3.4 Performance of transformer-based models

The performance of the proposed transformer-based models with selected hyper-parameters, such as embedding dimensions, number of transformer blocks and settings of MLP layer, is summarized in Table 10. The results show that all the TET models outperform the ensemble models, such as XGBoost and RF. The best model is TET-P16T2M32-1 with embedding dimensions of 16 and 2 transformer modules. The R2 value is 0.974, which is higher than all the ensemble models. The results also reveal that an increase in embedding dimension and number of transformer modules normally give better performance. Therefore, the proposed transformer-based models are capable of handling tabular data. Another benefit of using neural network-based models is that they can be modified to receive other complex data, such as images, to further enhance the model's capacity.

Table 10. Results of TET models (best results are underlined)

Model	**MSE**	**MAE**	**RMSE**	**R2**
TET-P16T1M32-1	0.148	0.215	0.380	0.956
TET-P16T1M32-2	0.147	0.220	0.379	0.957
TET-P16T2M32-1	0.124	0.209	0.345	0.963
TET-P16T3M32-1	0.116	0.203	0.337	0.966
TET-P32T1M32-1	0.097	0.186	0.305	0.971
TET-P32T2M32-1	<u>0.088</u>	<u>0.174</u>	<u>0.286</u>	<u>0.974</u>
TET-P32T3M32-1	0.103	0.189	0.313	0.969

3.5 Performance of CNN-based models

The evaluation metrics of CNN-based models are reported in Table 11. The performance of most CNN-based models is comparable with the ensemble methods, such as RF and LightGBM, and ensemble ANN models, but slightly lower than transformer-based models. The best CNN-based model is 2D-CNN2-M16 which is composed of 64 3x3 kernels followed by 32 3x3 kernels and a fully connected layer with 16 units in hidden layer. The MSE and R2 score are 0.288 and 0.915, respectively. The results also reveal that increasing the number of kernels can help to enhance predictive performance. In short, the use of CNN-based algorithms can help to improve the performance of vanilla MLP models. More studies can be conducted to explore tabular-to-image transformation methods and CNN architecture.

Table 11. Results of CNN-based models (best results are underlined)

Model	**MSE**	**MAE**	**RMSE**	**R2**
2D-CNN1-M16	0.327	0.322	0.569	0.904
2D-CNN2-M16	0.288	0.289	0.535	0.915
2D-CNN1-M32	0.351	0.316	0.589	0.897
2D-CNN2-M32	0.301	0.297	0.543	0.912

4. CONCLUSIONS

This paper investigated the capabilities of using data-driven methods based on ML algorithms, including basic algorithms, ensemble methods, and neural network-based models to predict seismic responses of RC buildings. Modifications on neural network architectures using ensemble techniques, embedding and transformers, and CNN algorithms were also explored. The key findings are summarized as follows:

- SVR model could achieve satisfactory performance for regression problems based on tabular data compared with the other basic algorithms, such as DT and KNN.
- Among the boosting methods considered in this study, AdaBoost achieved the highest performance levels. XGBoost performed slightly lower than AdaBoost, followed by GBDT and LightGBM. The performance of boosting models could be slightly enhanced via the stacking method.
- The performance of vanilla MLP models was generally lower than ensemble methods in processing tabular data. An increase in the number of units per layer and the number of hidden layers could improve their performance.
- Ensemble ANN, which combined ensemble methods and ANNs together, could improve the predictive performance of vanilla MLP models, but the performance was generally lower than classical ensemble models (e.g. XGBoost, GBDT).
- Two ensemble methods, namely stacking and boosting methods, were adopted. This study showed that StackANN was an effective way to enhance the performance of ANN models. StackANN models could potentially achieve similar performance to ensemble models. On the other hand, GBANN models showed satisfactory results, but the metrics were slightly lower than StackANN. Besides, GBANN models took longer time for training and testing.
- This study proposed a Tabular Embedding Transformer (TET) model based on feature embedding techniques, feature masking and transformer architecture for handling tabular data. The results revealed that such techniques could significantly enhance the model's accuracy so that the TET models could outperform ensemble methods in some problems.
- Tabular-to-image transformation with CNN architecture were effective algorithms to enhance the predictive performance of ANN models. The performance of the CNN-based models under consideration was comparable with most of the ensemble models.
- The proposed models could be used to rapidly estimate the MIDR of buildings under different levels of earthquake. The results were valuable to understand the seismic performance and possible damage levels of buildings under earthquakes.

Further studies will involve investigating cutting-edge techniques, such as tabular-to-image transformation algorithms, tabular-to-graph transformation and graph neural network, auto-encoder and applications of transfer learning to enhance the predictive performance of the ANN models to process tabular data.

ACKNOWLEDGEMENT
The work described in this paper was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. UGC/FDS25/E05/21)

REFERENCES

Alenizy, H.A., and Berri, J. (2025), "Transforming tabular data into images via enhanced spatial relationships for CNN processing," *Scientific Reports*, **15**, 17004.

Alkhatib, A., Ennadir, S., Bostrom, H., and Vazirgiannis, M. (2024), "Interpretable graph neural networks for tabular data," *arXiv:2308.08945v2*.

Arik, S.O., and Pfister, T. (2019), "TabNet: Attentive interpretable tabular learning," *arXiv:1908.07442*.

Asgarkhani, N., Kazemi, F., Jakubczyk-Gałczynska, A., Mohebi, B., and Jankowski, R. (2024), "Seismic response and performance prediction of steel buckling-restrained braced frames using machine-learning methods," *Engineering Applications of Artificial Intelligence*, **128**, 107388.

Bhatta, S. and Dang J. (2023), "Seismic damage prediction of RC buildings using machine learning," *Earthquake Engineering and Structural Dynamics*, **65**, 1-24.

Bhatta, S., Kang, X., and Dang, J. (2024), "Machine learning prediction models for ground motion parameters and seismic damage assessment of buildings at a regional scale," *Resilient Cities and Structures*, **3**, 84-102.

Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., and Kasneci, G. (2024), "Deep neural networks and tabular data: A survey," *IEEE Transactions on Neural Networks and Learning System*, **35**(6).

Briner, N., Cullen, D., Halladay, J., Miller, D., Primeau, R., Avila, A., Basnet, R., and Doleck, T. (2023), "Tabular-to-image transformations for the classification of anonymous network traffic using deep residual networks," *IEEE Access*, **11**, 113110-113113.

Buildings Department. *The Hong Kong Code of Practice for Structural Use of Concrete 2013* (2020 Edition). HKSAR.

Chen, J., Yan, J., Chen, Q., Chen, D.Z., Wu, J., and Sun, J. (2024), "Can a deep learning model be a sure bet for tabular prediction?," *in KDD*, Aug 25-29,2024, Barcelona, Spain, 288-296.

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*, Association for Computing Machinery, New York, August.

Demertzis, K., Kostinakis, K., Morfidis, K., and Iliadis, L. (2023), "An interpretable machine learning method for the prediction of R/C buildings' seismic response," *Journal of Building Engineering*, **63**, 1-26.

Demir, A., Sahin, E.K., and Demir, S. (2024), "Advanced tree-based machine learning methods for predicting the seismic response of regular and irregular RC frames," *Structures*, **64**, 106524.

FEMA P695 (2008). *Quantification of Building Seismic Performance Factors*. Federal Emergency Management Agency, Washington, DC, USA.

Freund, Y., and Schapire, R. (1995), "A decision-theoretic generalization of on-line learning and an application to boosting."

Friedman, J. (2001), "Greedy Function Approximation: A Gradient Boosting Machine," *The Annals of Statistics*, **29**(5).

Géron A. (2019). *Hands-on Machine Learning with Scikit-Learn, Keras, and Tensorflow*, 2nd edition. Published by O'Reilly Media, Inc.

Gorishniy, Y., Rubachev, I., Khrulkov, V., and Babenko, A. (2021), "Revisiting deep learning models for tabular data," *arXiv:2106.11959.*

Gorishniy, Y., Rubachev, I., and Babenko, A. (2022), "On embeddings for numerical features in tabular deep learning," *arXiv:2203.05556*.

Ho, T.K. (1995), "Random decision forests," *Proceedings of 3rd International Conference on Document Analysis and Recognition*, Montreal, Canada, Aug.

Huang, X., Khetan, A., Cvitkovic, M. and Karnin Z. (2020), "TabTransformer: Tabular data modeling using contextual embeddings," https://arxiv.org/abs/2012.06678.

Ke G.; Meng Q.; Finley T.; Wang T.; Chen W.; Ma W.; Ye Q.; Liu T.Y. (2017), "LightGBM: A highly efficient gradient boosting decision tree," *Advances in Neural Information Processing Systems, 31st Conference on Neural Information Processing Systems* (NIPS 2017), Long Beach, CA, USA.

Mahmoudi, H., Bitaraf, M., Salkhordeh, M. and Soroushian, S. (2023), "A rapid machine learning-based damage detection algorithm for identifying the extent of damage in concrete shear-wall buildings," *Structures*, **47**, 482-499.

Mienye, I.D. and Sun, Y. (2022), "A Survey of Ensemble Learning: Concepts, Algorithms, Applications, and Prospects," *IEEE Access*, **10**, 99129-99149.

Nazari, Y.R. and Saatcioglu, M. (2017), "Seismic vulnerability assessment of concrete wall buildings through fragility analysis." *Journal of Building Engineering*, **12**, 202-209.

Liu, Y. (2020). *Python Machine Learning by Example*, 3rd Edition. Packt Publiching Ltd., Birmingham, UK.

Luk, S.H. (2023), "Damage Prediction of Buildings using Machine Learning Algorithms", *The 2023 World Congress on Advances in Structural Engineering and Mechanics* (ASEM23), Seoul, Korea on August 16-18.

Luk, S.H. (2025), "Machine learning-based methods for the seismic damage classification of RC buildings," *Buildings*, **15**, 2395.

Pan Y. and Zhang L. (2021), "Roles of artificial intelligence in construction engineering and management: A critical review and future trends," *Automation in Construction*, **122**.

Pedregosa F., Varoquaux G., Gramfort, A., Michel, V., Thirion B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Du-bourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011), "Scikit-learn: machine learning in Python," *The Journal of Machine Learning Research*, **12**, 2825-2830.

Saleemuddin M.Z.M. and Sangle K.K. (2017), "Seismic damage assessment of reinforced concrete structure using non-linear static analysis," *KSCE Journal of Civil Engineering*, **21**(4), 1319-1330.

Sharma, A., Vans, E., Shigemizu, D., Boroevich, K.A., and Tsunoda, T. (2019), "DeepInsight: A methodology to transform a non-image data to an image for convolution neural network architecture," *Scientific Reports*, **9**, 11399.

Sharma, A., and Kumar, D. (2022), "Classification with 2-D convolutional neural networks for breast cancer diagnosis," *Scientific Reports*, **12**, 21857.

Shwartz-Ziv, R., and Armon, A. (2021), "Tabular data: deep learning is not all you need," *arxiv:2106.03253*.

Somepalli, G., Goldblum, M., Schwarzschild, A., Bruss, C.B., and Goldstein, T. (2021), "SAINT: Improved neural networks for tabular data via row attention and contrastive pre-training," *arXiv:2106.01342*.

Tancik, M., Srinivasan, P.P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J.T., and Ng, R. (2020), "Fourier features let networks learn high frequency functions in low dimensional domains," *NeurIPS*.

Thai, H.T. (2022), "Machine learning for structural engineering: a state-of-the-art review," *Structures*, **38**, 448-491.

Vamvatsikos D. and Cornell C.A. (2002), "Incremental dynamic analysis," *Earthquake Engineering and Structural Dynamics*, **31**(3), 491-514.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., and Polosukhin, I. (2017), "Attention is all you need," *31st Conference on Neural Information Processing Systems* (NIPS 2017), Long Beach, CA, USA.

Wen, W., Zhang, C., and Zhai, C. (2022), "Rapid seismic response prediction of RC frames based on deep learning and limited building information," *Engineering Structures*, **267**, 114638.

Zhu, T., Brettin, T., Xia, F., Partin, A., Shukla, M., Yoo, H., Evrard Y.A., Doroshow, J.H., and Stevens, R.L. (2021), "Converting tabular data into images for deep learning with convolutional neural networks," *Scientific Reports*, **11**, 11325.