Engineering Structures 215 (2020) 110704

Contents lists available at ScienceDirect

Engineering Structures

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

Physics-guided convolutional neural network (PhyCNN) for data-driven seismic response modeling

Ruiyang Zhang[a], Yang Liu[b], Hao Sun[a,c,*]

[a] Department of Civil and Environmental Engineering, Northeastern University, Boston, MA 02115, USA
[b] Department of Mechanical and Industrial Engineering, Northeastern University, Boston, MA 02115, USA
[c] Department of Civil and Environmental Engineering, MIT, Cambridge, MA 02139, USA

ARTICLE INFO

Keywords:
Deep learning
Physics-guided convolutional neural network
PhyCNN
K-means clustering
Seismic response prediction
Serviceability assessment

ABSTRACT

Accurate prediction of building's response subjected to earthquakes makes possible to evaluate building performance. To this end, we leverage the recent advances in deep learning and develop a physics-guided convolutional neural network (PhyCNN) for data-driven structural seismic response modeling. The concept is to train a deep PhyCNN model based on limited seismic input–output datasets (e.g., from simulation or sensing) and physics constraints, and thus establish a surrogate model for structural response prediction. Available physics (e.g., the law of dynamics) can provide constraints to the network outputs, alleviate overfitting issues, reduce the need of big training datasets, and thus improve the robustness of the trained model for more reliable prediction. The surrogate model is then utilized for fragility analysis given certain limit state criteria. In addition, an unsupervised learning algorithm based on K-means clustering is also proposed to partition the datasets to training, validation and prediction categories, so as to maximize the use of limited datasets. The performance of PhyCNN is demonstrated through both numerical and experimental examples. Convincing results illustrate that PhyCNN is capable of accurately predicting building's seismic response in a data-driven fashion without the need of a physics-based analytical/numerical model. The PhyCNN paradigm also outperforms non-physics-guided neural networks.

1. Introduction

Civil infrastructures are vulnerable to natural hazards such as earthquakes, tsunamis and hurricanes, especially under the effect of material aging and structural deterioration. Recent developments in advanced sensor and computation technologies provide a primary tool for monitoring structural behavior and evaluating structural integrity. The majority of existing methodologies focus on extracting structural features (e.g., modal characteristics) and updating models from the measured data, such as the Bayesian probabilistic approach [1–6], Kalman filtering [7–9], seismic interferometry [10–14], etc. Nevertheless, how to effectively utilize sensing data for structural response modeling and prediction under future hazards remains as a challenge.

Conventional approaches for structure response prediction using sensing data include identification-based (or inverse) methods and analytical (or forward) methods. For the identification-based approach, mapping the given excitation to the corresponding response through a black-box model or state-space model [15–19] has been used to simulate and predict the system dynamic response. A comprehensive review of works on system identification and its applications in structural response modeling was given in [20–22]. Alternatively, structural finite element (FE) model updating, which minimizes the errors between the measured response of the real structure and the synthetic response of the parametrized FE model, has been extensively studied and used to predict linear/nonlinear structural response given a new input [23–27]. For instance, Skolnik et al. [28] predicted the seismic responses of a 15-story steel-frame building by an updated FE model. In general, these approaches require excessive computational efforts on updating the FE model when the model is of high fidelity, due to the large number of parameters for updating and limited availability of sensing data. Though low-fidelity models are more computationally cost effective, the accuracy is difficult to be retained under uncertainties especially for nonlinear response modeling. For analytical approaches, autoregressive integrated moving average (ARIMA) is one of the most popular linear models for time series analysis and forecasting [29,30], which is originated from the autoregressive models (AR), the moving average models (MA) and the auto-regressive moving average models (ARMA). These time series modelling methods have served the scientific

* Corresponding author at: Department of Civil and Environmental Engineering, Northeastern University, Boston, MA 02115, USA.
E-mail address: h.sun@northeastern.edu (H. Sun).

https://doi.org/10.1016/j.engstruct.2020.110704
Received 25 October 2019; Received in revised form 19 February 2020; Accepted 21 April 2020
Available online 30 April 2020

community for a long time; however, issues exist in regard to accuracy as well as stationarity and linearity hypothesis.

Recently, considerable attention has been focused on Artificial Intelligence (AI) which has been proven to be a powerful response modeling tool and approximator [31–35]. In particular, support vector machines (SVM) and artificial neural networks (ANN) have been used for the identification and modeling of dynamic responses during the past decade. For example, Zhang et al. [36] employed SVM to identify structural parameters. Dong et al. [37] predicted the dynamic response of an oscillator and a frame structure based on a SVM-based two-stage method. In addition, multi-layer perceptron (MLP) ANN has been applied for predicting structural response under static or dynamic loading conditions. For instance, Lightbody and Irwin [38] applied MLP to predict the viscosity of an industrial polymerization reactor. Wang et al. [39] developed a MLP back-propagation network to predict the seismic response of a bridge structure. Christiansen et al. [40] used previous time step information as input to a one layer MLP network to predict the dynamic response of the simplified model of a wind turbine. Huang et al. [41] identified structural dynamic characteristics and performed damage diagnosis of a building using a back-propagation MLP taking previous time step information as input. Lagaros and Papadrakakis [42] proposed an MLP network to predict the structural nonlinear behavior of 3D buildings when earthquake excitations with increasing intensities are considered.

Nevertheless, a very limited number of studies have been reported in literature for structural response modeling and prediction using more advanced deep learning models such as the recurrent neural network (RNN) and the convolutional neural network (CNN). RNN is designed to learn sequential and time-varying patterns for regression problems [43–46], while CNN is known for its capability in classification of data with grid-like topology (e.g., 1D sequences and 2D images) [47,48]. CNN can be also used for solving regression problems. Recently, Sun et al. [49] proposed a virtual sensor model using CNN to estimate the dynamic responses of two numerical structure given measurements at other locations. However, the network was trained using the partial response measurements as input to predict responses of the rest of DOFs, limiting its applications in structural response prediction under new inputs. Another remarkable work by Wu and Jahanshahi [50] used CNN to estimate structural dynamic response and perform system identification, which showed great capability of CNN for sequence regression. Nevertheless, the hypothesis was that the data is sufficient to train a reliable predictive model. Challenges arise when the available training data is scarce. A potential solution to overcome this limitation is to incorporate scientific laws (e.g., partial differential equations, boundary conditions) into deep neural networks to guide the deep learning from scarce data [51–54]. We herein address limitations in data-driven structural response modeling through developing a novel physics-guided CNN (i.e., PhyCNN), which is capable of accurately predicting nonlinear structural seismic time-history responses in a data-driven manner. The basic concept is to (1) embed available physics knowledge into the deep learning model, (2) train a PhyCNN based on available seismic input–output datasets (e.g., from simulation or sensing), and (3) use the trained PhyCNN as a surrogate model for response prediction. The surrogate model can further be utilized for fragility analysis given certain limit state criteria (e.g., the serviceability state). It is noted that the available physics (e.g., the law off dynamics) can provide constraints to the network outputs, alleviate overfitting issues, reduce the need of big training datasets, and thus improve the robustness of the trained model for more reliable prediction.

This paper is organized as follows. Section 2 presents the proposed PhyCNN architecture for structural response modeling. In Section 3, the performance of PhyCNN is verified through two numerical examples of a nonlinear system. Section 4 presents the experimental validation of PhyCNN based on filed sensing measurements, where data-driven serviceability analysis of a building is also discussed. Section 5 summarizes the conclusions.

2. Physics-guided convolutional neural network (PhyCNN)

Neural networks have been widely recognized as a powerful tool to deal with problems like classification and regression. Among many other neural networks, CNN, which is inspired by the virtual convex of animals [55], can effectively model the grid-structured topology of data (e.g., images), making it especially powerful for image classification. However, CNN is also capable of dealing with regression problems, which is often inconspicuous due to its capability in classification. Traditionally, deep neural networks are trained solely based on data. However, by adding physics (e.g., the governing law of dynamics) into the training phase, the robustness and reliability of learning from the data can be further enhanced. In other words, the embedded physics can inform the learning and constrain the training to a feasible space. In this paper, a 1D regression-oriented PhyCNN architecture is proposed for time series modeling.

To illustrate the concept, let's consider a dynamic system subjected to the ground excitation following the equation of motion that obeys the Newton's second law:

$$\mathbf{M}\ddot{\mathbf{x}}(t) + \mathbf{h}(t) = -\mathbf{M}\boldsymbol{\Gamma}\ddot{\mathbf{x}}_{\mathrm{g}}(t) \tag{1}$$

where $\mathbf{M}$ is the mass matrices; $\mathbf{x}$, $\dot{\mathbf{x}}$, and $\ddot{\mathbf{x}}$ are the relative displacement, velocity, and acceleration vectors to the ground; $\ddot{\mathbf{x}}_{\mathrm{g}}$ represents the ground acceleration; $\boldsymbol{\Gamma}$ is the force distribution vector; and $\mathbf{h}$ is the latent generalized restoring force vector. Normalizing Eq. (1) by $\mathbf{M}$, the governing equation can be expressed as

$$f \coloneqq \ddot{\mathbf{x}}(t) + \mathbf{g}(t) + \boldsymbol{\Gamma}\ddot{\mathbf{x}}_{\mathrm{g}}(t) \to 0 \tag{2}$$

where $\mathbf{g}(t)$ is the mass-normalized restoring force, namely, $\mathbf{g}(t) = \mathbf{M}^{-1}\mathbf{h}(t)$, whose closed form is assumed unknown. Hence, it will be learned by the proposed PhyCNN.

A PhyCNN framework is developed for surrogate modeling of such a nonlinear dynamic system under ground motion excitation. The proposed deep learning framework consists of a 1D CNN and a graph-based tensor differentiator. Fig. 1 shows the basic concept and architecture of PhyCNN in the context of structural response modeling given the ground acceleration as input which contains n sample points from t_1 to t_n. The outputs are state space variables $\mathbf{z}(t)$ including the structural displacement $\mathbf{x}(t)$, the velocity $\dot{\mathbf{x}}(t)$, and the normalized restoring force $\mathbf{g}(t)$, namely, $\mathbf{z}(t) = \{\mathbf{x}(t), \dot{\mathbf{x}}(t), \mathbf{g}(t)\}$, each of which has same number of n sample points ranging from t_1 to t_n. With the convolution operation illustrated in Section 2.1, zero-padding is added to the output sequence of each convolution layer to ensure the identical input/output length. The proposed PhyCNN architecture consists of multiple hidden layers besides the input and output layers, namely, the feature learning layers and the fully connected layers [56,57]. A typical feature leaning layer usually includes a convolution layer, a nonlinear layer (or nonlinear activation function), and a feature pooling layer. The output of each layer is called a feature map since the feature learning layers are used for extracting features from the input or from the output from the previous layer. In the proposed PhyCNN architecture, the dimension of height in a classical CNN is reduced to one, making it possible to take a time-series signal as input, while the dimension of width represents the temporal space. In addition, a graph-based tensor differentiator (e.g., the finite different method) is developed to calculate the derivative of state space outputs $\dot{\mathbf{z}}(t) = \{\mathbf{x}_t(t), \dot{\mathbf{x}}_t(t), \mathbf{g}_t(t)\}$ to construct the physics loss from the governing equation, where the subscript t represents the derivative of the state with respect to time. The basic concept here is to optimize the network hyperparameters $\theta = \{\mathbf{W}_\theta, \mathbf{b}_\theta\}$ such that the PhyCNN can interpret the measurement data (e.g., $\mathbf{x}^m$, $\dot{\mathbf{x}}^m$, $\mathbf{g}^m$) while satisfying the physical equation of motion in Eq. (2), e.g., $f \to 0$. Here, $\mathbf{W}_\theta$ and $\mathbf{b}_\theta$ are the neural network weight and bias parameters. The total loss $J(\theta)$ is then defined as

$$J(\theta) = J_D(\theta) + J_P(\theta) \tag{3}$$

with

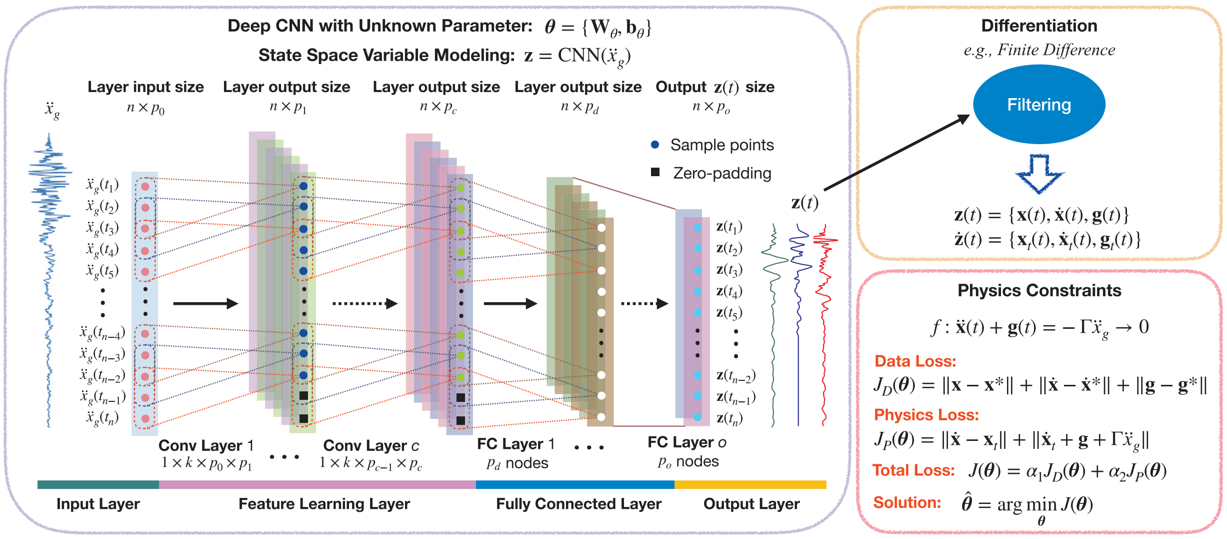

Deep CNN with Unknown Parameter: $\theta = \{\mathbf{W}_\theta, \mathbf{b}_\theta\}$
State Space Variable Modeling: $\mathbf{z} = \text{CNN}(\ddot{x}_g)$
Layer input size $n \times p_0$
Layer output size $n \times p_1$
Layer output size $n \times p_c$
Layer output size $n \times p_d$
Output $\mathbf{z}(t)$ size $n \times p_o$
Sample points
Zero-padding
Conv Layer 1 $1 \times k \times p_0 \times p_1$
Conv Layer c $1 \times k \times p_{c-1} \times p_c$
FC Layer 1 p_d nodes
FC Layer o p_o nodes
Input Layer
Feature Learning Layer
Fully Connected Layer
Output Layer
Differentiation
e.g., Finite Difference
Filtering
$\mathbf{z}(t) = \{\mathbf{x}(t), \dot{\mathbf{x}}(t), \mathbf{g}(t)\}$
$\dot{\mathbf{z}}(t) = \{\mathbf{x}_t(t), \dot{\mathbf{x}}_t(t), \mathbf{g}_t(t)\}$
Physics Constraints
$f: \ddot{\mathbf{x}}(t) + \mathbf{g}(t) = -\Gamma \ddot{x}_g \to 0$
Data Loss:
$J_D(\boldsymbol{\theta}) = \|\mathbf{x} - \mathbf{x}^*\| + \|\dot{\mathbf{x}} - \dot{\mathbf{x}}^*\| + \|\mathbf{g} - \mathbf{g}^*\|$
Physics Loss:
$J_P(\boldsymbol{\theta}) = \|\dot{\mathbf{x}} - \mathbf{x}_t\| + \|\dot{\mathbf{x}}_t + \mathbf{g} + \Gamma \ddot{x}_g\|$
Total Loss: $J(\boldsymbol{\theta}) = \alpha_1 J_D(\boldsymbol{\theta}) + \alpha_2 J_P(\boldsymbol{\theta})$
Solution: $\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

Fig. 1. The proposed physics-guided convolutional neural network (PhyCNN) for time-series modeling. The PhyCNN architecture includes the input layer, the feature learning layer, fully-connected layer, the output layer, and the graph-based tensor differentiator. The inputs are ground accelerations (or ground displacements) and the outputs are state space variables $\mathbf{z}(t)$ including displacement $\mathbf{x}(t)$, velocity $\dot{\mathbf{x}}(t)$, and restoring force $\mathbf{g}(t)$, namely, $\mathbf{z}(t) = \{\mathbf{x}_t(t), \dot{\mathbf{x}}_t(t), \mathbf{g}_t(t)\}$. The derivatives of state space outputs $\dot{\mathbf{z}}(t)$ are calculated through a tensor differentiator using the central finite difference method. The total loss consists of the data loss from the measurements and the physics loss which models the dependency between the output features. Both input and output to the network are time sequences, including p_0 (e.g., ground acceleration and/or ground displacement) and p_o (e.g., state space variables at different floor levels) features, respectively. The size of layer input and output is given. The convolution layer is defined as "height × width × depth × filters (output channels)". An identical kernel size is used for all three convolution layers in this study. Zero-padding is added to the output sequence of each convolution layer due to the convolution operation as illustrated in Section 2.1. Note that the nonlinear activation functions are not shown. in this figure.

$$J_D(\theta) = \frac{1}{N}\|\mathbf{x}^p - \mathbf{x}^m\|_2^2 + \frac{1}{N}\|\dot{\mathbf{x}}^p - \dot{\mathbf{x}}^m\|_2^2 + \frac{1}{N}\|\mathbf{g}^p - \mathbf{g}^m\|_2^2 \tag{4}$$

$$J_P(\theta) = \frac{1}{N}\|\dot{\mathbf{x}}^p - \mathbf{x}_t^p\|_2^2 + \frac{1}{N}\|\dot{\mathbf{x}}_t^p + \mathbf{g}^p + \Gamma\ddot{\mathbf{x}}_g\|_2^2 \tag{5}$$

where $J_D(\theta)$ denotes the data loss based the measurements while $J_P(\theta)$ represents the physics loss which introduces a constraint for the neural network that models the dependency in-between the output features; the superscript p and m denote the prediction and measurement, respectively. Note that the measurements are not necessarily required for the complete state, which could be part of the state variables (e.g., $\mathbf{x}^m$ only) or the accelerations (e.g., $\ddot{\mathbf{x}}^m$). In such a case, the data loss in Eq. (4) should be adjusted accordingly. Note that equal weights are used for $J_D(\theta)$ and $J_P(\theta)$ in this study which lead to satisfactory convergence. Weights may be adjusted to improve the training performance in different problems. During training, θ will be updated and determined by solving the optimization problem $\hat{\theta} := arg\, min_\theta J(\theta)$. Note that the proposed PhyCNN architecture used in this study has five convolution layers ($c = 5$) with identical kernel size and three fully-connected layers. Details of each layer are discussed in the following subsections.

2.1. Convolution layer

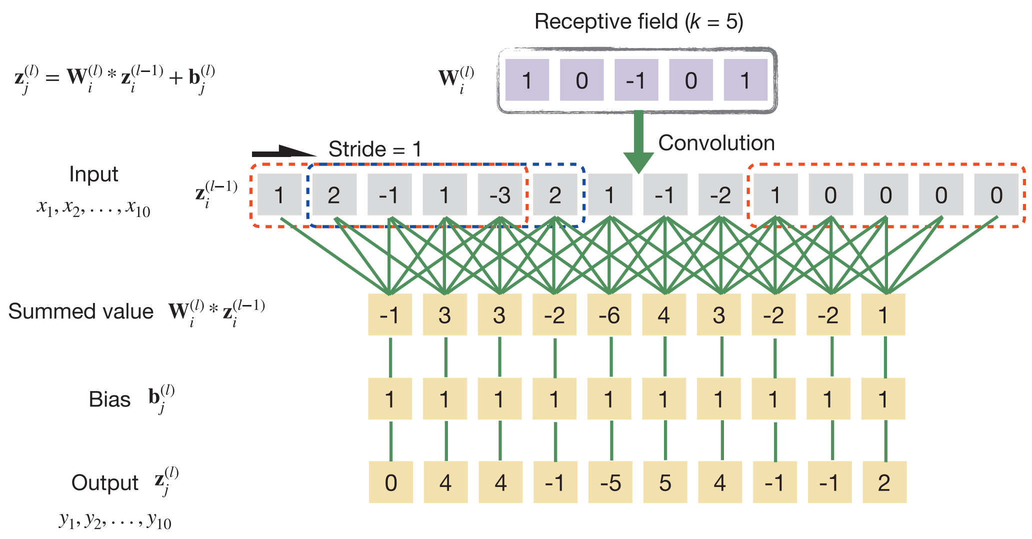
The convolution (Conv) layer is the basis of the CNN architecture, which performs the core operations of feature learning. The size of a convolution layer is defined as "height × width × depth × filters (output channels)", e.g., $1 \times k \times p_0 \times p_1$ for the first Conv layer as shown in Fig. 1. Each Conv layer consists of a set of learnable kernels (also known as filters) with a size of $1 \times k$, which are parameterized by a hyperparameter called the receptive field containing a group of weights shared over the entire input temporal space. The initial weights of a receptive field are typically randomly generated. During the forward pass, the kernels convolve across the temporal space of the input and compute dot products between the entries of a receptive field and a local region of the input which represents a sequence of input time series. The dot products are summed, and the bias is added to the summed value, forming a single entry of the output. The full input space is scanned through sliding the kernels along the temporal space with one single stride. In this way, the time dependency is captured by convolving a sequence of input across the entire temporal space. The stride operation will lead to smaller output length if zero padding is not applied. To ensure the output has the same length n as the input in the temporal space, zero-padding is added at the end of the input time series. The number of required zero-padding, P, is given by $P = k - 1$. The dimension of the convolution layer output $\mathbf{z}^{(l)}$ can be different from the layer input $\mathbf{z}^{(l-1)}$, where l denotes the layer index. The number of filters, p, specifies the dimensionality of the output space. For the jth ($j = 1, 2, \cdots, p$) output feature (i.e., or called the channels in a standard CNN), the corresponding output of a convolution layer can be written as

$$\mathbf{z}_j^{(l)} = \sum_{i=1}^{p_0} \mathbf{W}_i^{(l)} * \mathbf{z}_i^{(l-1)} + \mathbf{b}_j^{(l)} \tag{6}$$

which takes the output of the previous layer $\mathbf{z}^{(l-1)}$ with zero-padding as input. Here, i represents the ith input feature/channel ($i = 1, 2, \cdots, p_0$);$\mathbf{W}_i^{(l)}$ is the receptive field with the kernel size of k for the ith input feature; $\mathbf{b}_j^{(l)}$ is the bias vector added to the summed term; and $*$ represents the 1D convolution operator. Note that for each output feature map, same bias is shared across the temporal space.

A simple example is presented in Fig. 2 to illustrate the convolution operation in the temporal space. Herein, a single-channel input sequence ($p_0 = 1$) with a length of $n = 10$ is considered, with a receptive filed size of $k = 5$ for the jth filter. The kernel slides across the temporal space with a stride $s = 1$, resulting in the output with a length of 10. The weights of the receptive field are [1, 0, − 1, 0, 1], shared across the entire temporal space.

Conventionally, a convolution layer is always followed by a nonlinear activation function, which introduces nonlinearity and makes learning easier by adapting with variety of data and differentiating between the output. The activation layer increases the nonlinearities of the model and the overall network without affecting the receptive fields of the convolution layer. Common nonlinear activation functions include rectified linear unit (ReLU) [58], hyperbolic tangent (Tanh), and



Receptive field (k = 5)
$\mathbf{z}_j^{(l)} = \mathbf{W}_i^{(l)} * \mathbf{z}_i^{(l-1)} + \mathbf{b}_j^{(l)}$
$\mathbf{W}_i^{(l)}$
1 0 -1 0 1
Convolution
Stride = 1
Input
$x_1, x_2, \ldots, x_{10}$
$\mathbf{z}_i^{(l-1)}$
1 2 -1 1 -3 2 1 -1 -2 1 0 0 0 0
Summed value $\mathbf{W}_i^{(l)} * \mathbf{z}_i^{(l-1)}$
-1 3 3 -2 -6 4 3 -2 -2 1
Bias $\mathbf{b}_j^{(l)}$
1 1 1 1 1 1 1 1 1 1
Output $\mathbf{z}_j^{(l)}$
$y_1, y_2, \ldots, y_{10}$
0 4 4 -1 -5 5 4 -1 -1 2

Fig. 2. Illustration of the convolution process in a single convolution layer with $i = p_0 = 1$, $n = 10$, $k = 5$ and $s = 1$.

sigmoid function. In this paper, the ReLU function shown in Eq. (7) is employed, which gives slightly better performance compared to Tanh and sigmoid.

$$f(x) = \begin{cases} 0, & \text{for} \quad x < 0 \\ x, & \text{for} \quad x \geq 0 \end{cases} \tag{7}$$

2.2. Pooling layer

The pooling layer is often used to reduce the spatial size of the feature maps when dealing with classification problems with large input data. Popular pooling layers include max pooling and average pooling, which take either maximum or mean values from a pooling window. For example, Fig. 3 shows the max pooling operation in a standard CNN. In PhyCNN, pooling layer will perform a down-sampling operation in the temporal space, resulting in smaller output length, which is undesired for regression problems like time-series prediction. Although zero-padding can be applied to keep same output length as input, the time dependencies are alternated. Therefore, the pooling layer is restricted in the proposed PhyCNN architecture for structural response modeling.

2.3. Fully-connected layer

Exactly as its name implies, the fully-connected (FC) layer has full connections to all activations in the previous layer, as observed in regular neural networks. A FC layer multiplies the input by a weight matrix and then adds a bias vector. FC layers are typically used in the last stage of the CNN to connect to the target output layer and construct the desired number of output classes. For regression problems like time-series prediction, nonlinear activation functions such as ReLU, Tanh, and sigmoid are inappropriate for the last FC layer since they map the output in the range of [0, infinite), $(-1, 1)$, and (0, 1), respectively. Other potential alternatives include parametric rectified linear unit (PReLU) [59] and exponential linear unit (ELU) [60]. In this paper, the Tanh function is used as the activation within the FC layers and the linear activation function is applied for the output layer.

2.4. Dropout layer

Dropout layers can be added after each convolution layer and fully-connected layer to reduce overfitting by preventing complex co-adaptations on training data [61], which has remained as a common issue in machine learning. The key idea is to randomly disconnect the connections and drop units from the connected layer with a certain dropout rate during training. Dropout layers can also improve the training speed. Typically, they are applied before the FC layer which has more learnable parameters and is more likely to cause overfitting. Although convolution layers are less likely to overfit due to their particular structure where weights are shared over the spatial space, dropout layers can still be applied to the convolution layers which have huge parameters. In this study, dropout layers with a dropout rate of 0.2 are applied before the FC layers.

2.5. PhyCNN for response modeling

The proposed PhyCNN architecture takes the ground motion (e.g., ground accelerations) as input and the structural responses (e.g., story displacements) as output to learn the feature mapping between the input and output. First, the model is trained with either synthetic database or field sensing measurements. Then the trained model can be used to predict the structural responses under new seismic excitations. To train the proposed PhyCNN architecture, both the input and output dataset must be formatted as a three-dimensional array, where the entries are samples in the first dimension, time history steps in the

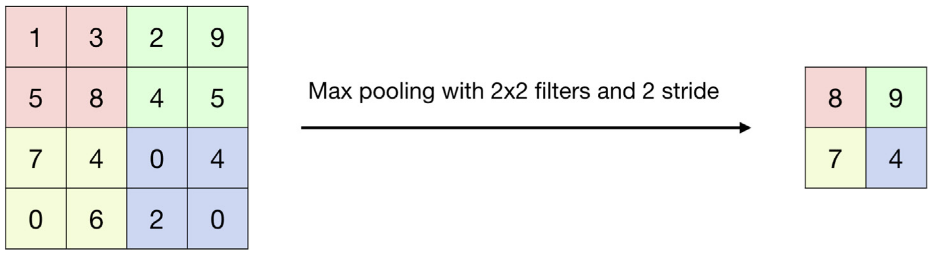


Fig. 3. An example of max pooling operation.

second dimension, and input or output features in the last dimension. The detailed neural network architecture is illustrated in Fig. 1, including five convolution layers and three fully-connected layers in addition to the input and output layer. Each convolution layer has 64 filters with a kernel size of 50 used in this study. The number of filters (nodes) and kernel length can be adjusted to get better performance for different problems. Note that the number of nodes for the last FC layer must be equal to the number of output features.

The entire training process is performed in a Python environment using Keras [62]. Keras is a high-level open source deep learning library built on top of TensorFlow which offers easy and fast prototyping neural networks. TensorFlow, a symbolic math library for machine learning applications developed by Google Brain Team [63], is served as the backend engine in Keras. It offers flexible data flow architecture enabling high-performance training of various types of neural networks across a variety of platforms (CPUs, GPUs, TPUs). The simulations are performed on a standard PC with 28 Intel Core i9-7940X CPUs and 2 NVIDIA GTX 1080 Ti video cards. It is noted that, for all case studies presented in this paper, it takes about 20 min for model training and less than 0.01 s for inference. The data and codes used in this paper will be publicly available on GitHub at https://github.com/zhry10/PhyCNN after the paper is published.

3. Numerical validation

We first demonstrate the performance of the PhyCNN approach for predicting structural displacements through two numerical examples. In the first example, we assume that the field measurements of all states, including $\mathbf{x}$, $\dot{\mathbf{x}}$, and $\mathbf{g}$, are available for training. Note that $\mathbf{g}$ can be inferred from the measurement of $\ddot{\mathbf{x}}$, e.g., $\mathbf{g} = -\ddot{\mathbf{x}} - \mathbf{\Gamma}\ddot{\mathbf{x}}_g$. Noteworthy, these responses can be recorded from numerical simulations when using PhyCNN for reduced order surrogate modeling. In the second example, only the measurements of $\ddot{\mathbf{x}}$ are available for training. The aim of having the aforementioned two scenarios is to show the versatility of the proposed PhyCNN for dealing with various availability of measurement data, which, for example, takes account for the access of field sensing measurements in real applications. In both examples, a single degree-of-freedom (DOF) nonlinear system subjected to ground motion excitation is investigated, whose equation of motion of the nonlinear system is expressed as:

$$m\ddot{x} + \underbrace{c\dot{x} + k_1 x + k_2 x^3}_{h} = -m\mathbf{\Gamma}\ddot{x}_g \tag{8}$$

where $m = 1$ kg is the mass, $c = 1$ Ns/m is the damping coefficient, $k_1 = 20$ N/m is the linear stiffness coefficient, and $k_2 = 200$ N/m is the nonlinear stiffness coefficient. The mass-normalized restoring force reads $g = h/m$.

3.1. Case 1: Available Measurements of $\mathbf{x}$, $\dot{\mathbf{x}}$, $\mathbf{g}$

A synthetic database, consisting of 100 samples (i.e., independent seismic sequences), was generated by numerical simulation of the 1DOF nonlinear system excited by a suite of earthquake records selected from the PEER strong motion database [64] with a 10% probability of exceedance in 50 years. Each simulation was executed up to 50 s with a sampling frequency of 20 Hz resulting in 1,001 data points for each record. However, only 10 datasets are randomly selected and considered as known datasets for training, while the rest are considered as unknown datasets to show the prediction performance. The PhyCNN architecture shown in Fig. 1 is implemented to develop the surrogate model. The input and output are formatted into the shape of [10, 1001, 1] and [10, 1001, 3] for the training datasets. To show the performance of the proposed approach with physics constraint, our method is also compared with the regular CNN without the physics loss (denoted as CNN).

Fig. 4 summarizes the prediction performance of both PhyCNN and CNN. Fig. 4(a) shows the regression analysis across all 90 prediction datasets for both PhyCNN (top-left) and CNN (bottom-left). It can be clearly seen that the prediction accuracy is greatly increased by embedding the physics constraints into deep learning. The time histories of predicted displacements are presented in Fig. 4(b) corresponding four different levels of correlation coefficients (noted by r), namely, 0.95, 0.92, 0.87, 0.61 using PhyCNN and 0.60, 0.72, 0.66, 0.37 using CNN. The PhyCNN prediction matches the reference well in both magnitudes and phases. Even for the worst case of $r = 0.61$, the proposed PhyCNN approach is able to reasonably predict the structural dynamics. On the contrary, CNN produces less satisfactory prediction especially in predicting the displacement magnitudes. Another salient feature of PhyCNN is that it also accurately predicts the states of velocity $\dot{x}$ and nonlinear restoring force g, as illustrated by the regression analysis in Fig. 5(a) and (c). Fig. 5(b) shows examples of predicted time histories of $\dot{x}$ and g using PhyCNN indicating a good agreement with the ground truth. The predicted nonlinearity is given in Fig. 6 which shows an example of the predicted hysteresis of the normalized nonlinear restoring force versus displacement and velocity.

3.2. Case 2: available measurements of $\ddot{\mathbf{x}}$ only

In most engineering practices, only accelerometers are installed on the building to record acceleration time histories. In such cases, it is impossible to train a standard deep learning model to predict structural displacements using acceleration measurements only. One way is to calculate the displacements from the acceleration measurements first and then train a deep learning model based on the interpreted displacements. However, it is well known that inevitable large numerical errors exist due to the integration of accelerations to displacements. This somehow limits the application of standard neural networks such as CNN in real-world applications.

However, with physics embedded into the training, the proposed PhyCNN is capable of accurately predicting the displacements based on only the acceleration measurements for training. This is realized by the graph-based tensor differentiator to construct the physics graph. Fig. 7 shows the network architecture specified in this study which is similar to the general state space format used in the previous study as shown in Fig. 1. The input to the CNN is the ground motion and the output displacement is passed into the differentiator to calculate the acceleration. The model will be trained and optimized to minimize the objective function defined as follows based on acceleration measurements.

$$J(\theta) = \frac{1}{N}\|\ddot{\mathbf{x}}^p - \ddot{\mathbf{x}}^m\|_2^2 \tag{9}$$

In this case, only acceleration measurements are considered as known and used to calculate the loss. The model is trained using 50 randomly selected datasets and tested with the rest 50 datasets considered as unknown. Fig. 8 shows the predicted acceleration compared to the ground truth. It is seen that the majority of correlation coefficients are greater than 0.9. Two example acceleration time histories are presented in Fig. 8(b) with different levels of accuracy. Even for the worst case with $r = 0.75$, a reasonably good matching in both magnitudes and phases is observed between the PhyCNN prediction and the ground truth. Fig. 9 shows the predicted displacements of the nonlinear system. From the regression analysis in Fig. 9(a), the correlation coefficients are mainly greater than 0.8. Fig. 9(b) shows the comparison of displacement time histories. It can be clearly seen that the PhyCNN can produce accurate displacement prediction even under the circumstances that only limited acceleration measurements are available for training. This study clearly demonstrates another benefit of the proposed PhyCNN in structural response modeling in the context of latent response prediction.

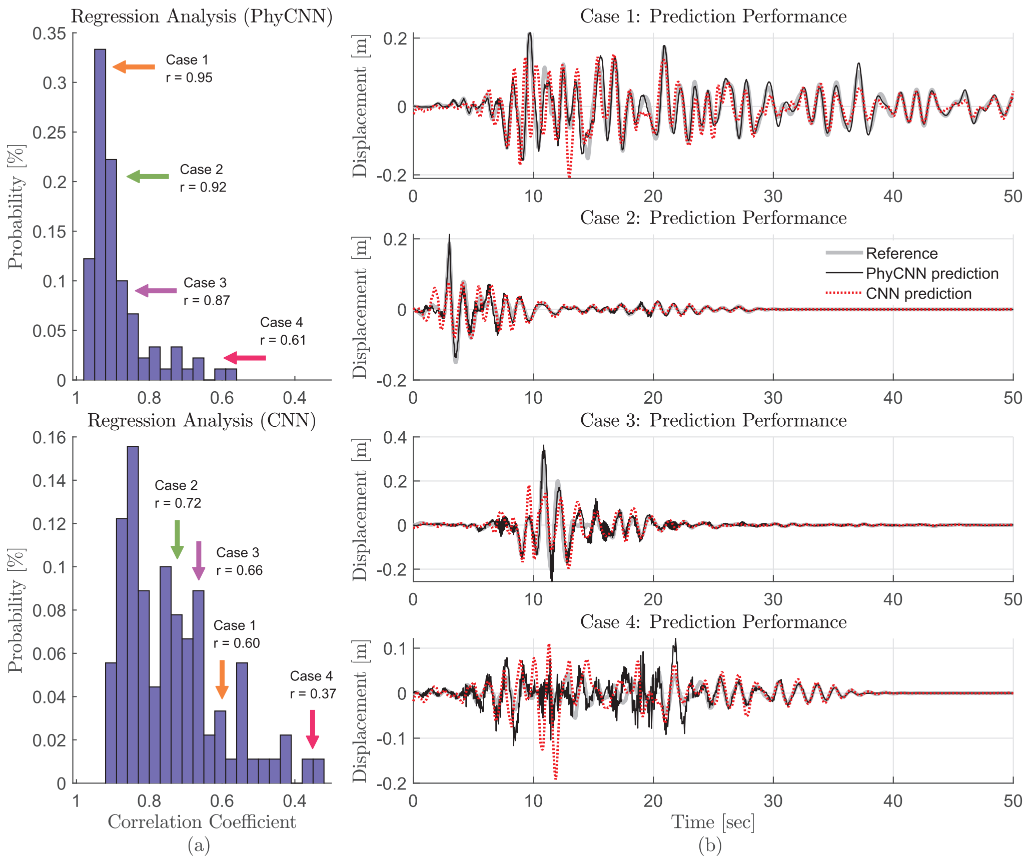


Fig. 4. Regression analysis in (a) and four examples of predicted displacements in (b) for unknown earthquakes using PhyCNN and CNN.

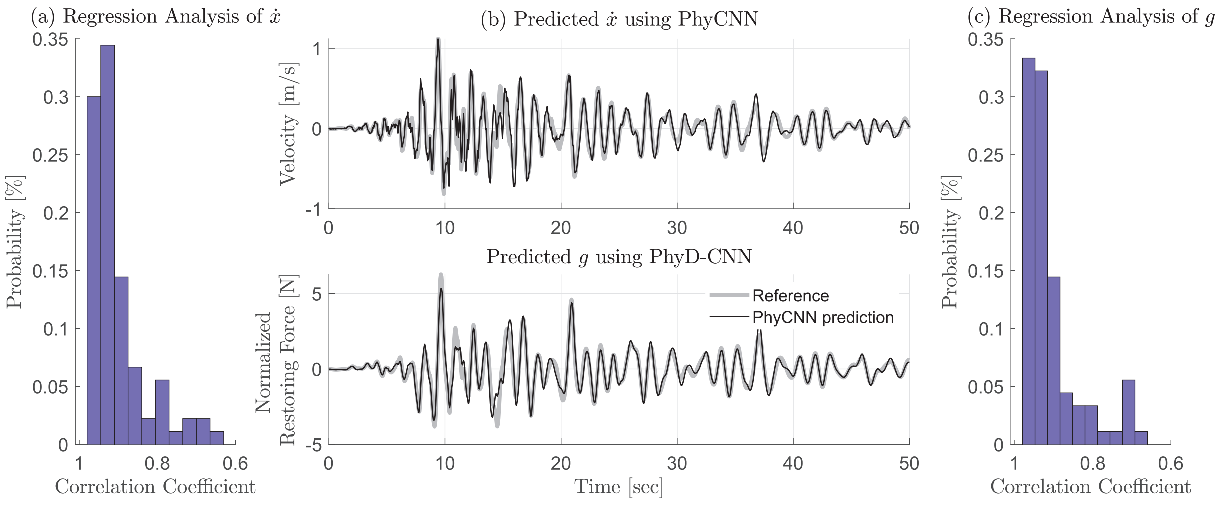


Fig. 5. Prediction performance of $\dot{x}$ and g using PhyCNN: (a, c) regression analysis of $\dot{x}$ and g respectively; (b) an example of predicted time history of $\dot{x}$ and g.

4. Experimental validation of PhyCNN performance

The PhyCNN architecture is further demonstrated using filed sensing data. A 6-story hotel building in San Bernardino, CA, from the Center for Engineering Strong Motion Data (CESMD), is selected and investigated [65]. The deep PhyCNN model illustrated in Fig. 1 was trained for the instrumented building with ground accelerations as input and the structural displacements as output. However, the measurement data used to train the PhyCNN are the acceleration time histories only (referred to the modified PhyCNN in Fig. 7). A data clustering technique is proposed to partition the data sets for training, validation and prediction. Based on the trained PhyCNN model, the serviceability of the 6-story hotel building can be further analyzed given new seismic inputs.

4.1. 6-Story Hotel Building in San Bernardino, CA

The 6-story hotel building in San Bernardino, California (CA) is a mid-rise concrete building designed in 1970 with a total of nine accelerometers installed on the 1st floor, 3rd and roof floors in both directions. The sensors, with their locations shown in Fig. 10, have recorded multiple seismic events in history from 1987 to 2018. Table 1 summarizes a total of 23 available datasets on CESMD used in this example. The historically recorded data is then used to train the

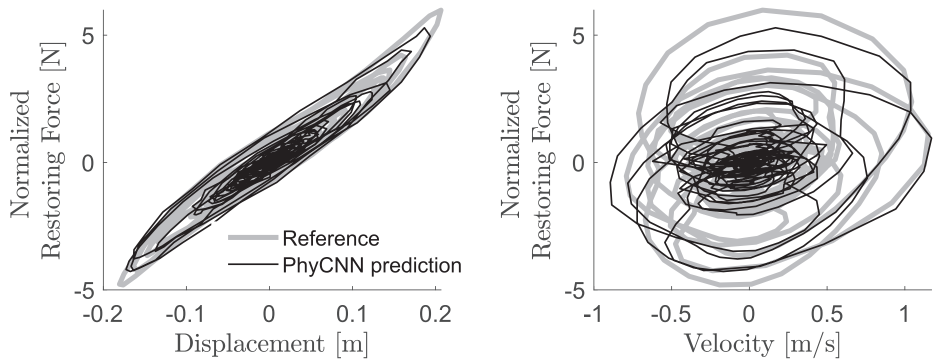


Fig. 6. Predicted hysteresis of nonlinear restoring force versus displacement (left) and nonlinear restoring force versus velocity (right).

PhyCNN. The trained surrogate model is then used to predict structural displacement time histories given new ground motions and develop a fragility function for serviceability assessment of the building.

Selecting training/validation datasets plays a critical role in deep learning. Commonly, the database is divided into training, validation and prediction datasets randomly (e.g., with ratios of 70%, 15%, 15%, respectively). In the case when the database is very limited, dataset partition could have a significant influence on the generalizability of the trained model. To extensively utilize the limited sensing data in this study, an unsupervised learning technique based on the K-means algorithm is proposed for data clustering. To better illustrate the concept, the 6-story hotel building in San Bernardino, CA is taken as an example.

4.2. K-means clustering

Before clustering, the raw sensing measurements with different sampling rates and high-frequency noise were first preprocessed. The measured accelerations were passed through a 2-pole Butterworth high-pass filter with a cutoff frequency of 0.1 Hz to remove the low-frequency behavior. The displacement time-series are further obtained from the high-pass filtered accelerations and used to model the input–output (ground acceleration-structural displacements) relationship. The entire historical datasets summarized in Table 1 are divided into training, validation and prediction datasets using the K-means clustering and the convex envelope technique discussed in the following. Both training and validation datasets are considered as known where both input and output are fully given during the training process, while prediction dataset is considered as unknown where only ground acceleration is given. The historical data is clustered based on both input excitations and output structural responses. Fig. 11(a) shows the relationship of peak ground acceleration (PGA) versus the peak structural displacement for all datasets in logarithmic scale. It can be seen that the peak structural displacements of most samples are less than 1 cm as shown in the green region, which is considered as the boundary of interest for training. The other two samples in the yellow region which yield large displacement under Northridge and Landers earthquakes, are used to test the performance of the trained PhyCNN model under a larger level of response whose information might not be fully covered during training process.

The 21 samples within the boundary of interest (green region) are partitioned into several clusters using the K-means algorithm [66] which is a popular data mining approach in the unsupervised learning setting that groups datasets into a certain number of clusters. It starts with the random selection of a set of k cluster centroids, e.g., $C = c_1, c_2, \ldots, c_k$. Next, each observation is assigned to the cluster, whose mean has the least squared Euclidean distance, given by

$$\arg\min_{c_i \in C} dist(c_i, x)^2 \tag{10}$$

where *dist* calculates the Euclidean distance. The new centroid is then determined by taking the mean of all the observations assigned to that

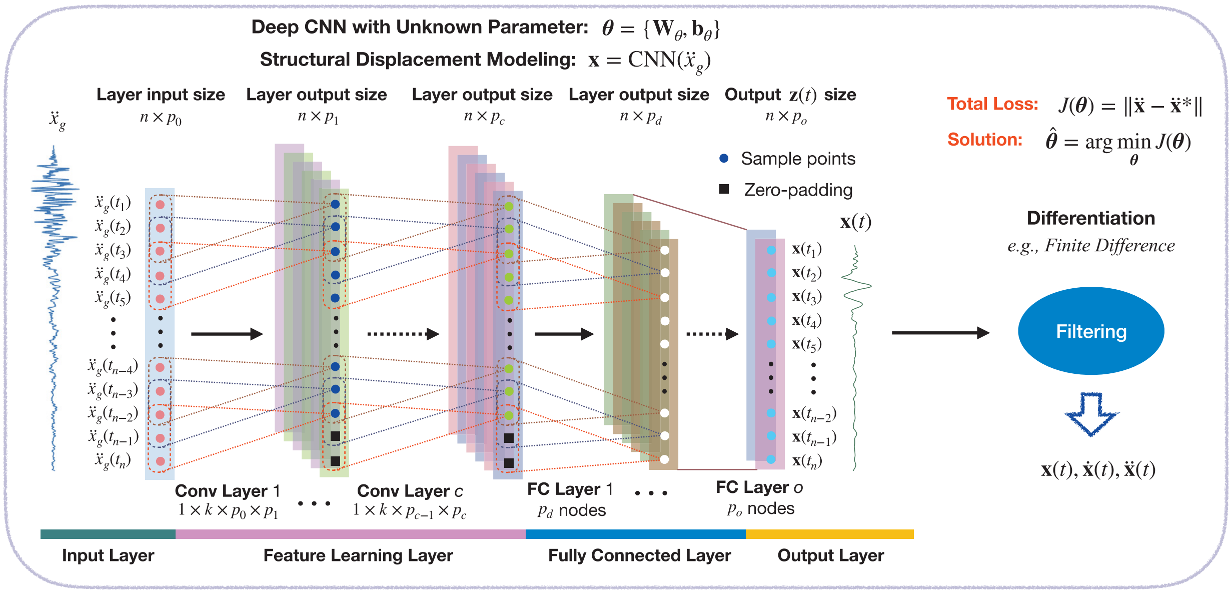

Deep CNN with Unknown Parameter: $\theta = \{\mathbf{W}_\theta, \mathbf{b}_\theta\}$
Structural Displacement Modeling: $\mathbf{x} = \mathrm{CNN}(\ddot{x}_g)$
Layer input size $n \times p_0$
Layer output size $n \times p_1$
Layer output size $n \times p_c$
Layer output size $n \times p_d$
Output $\mathbf{z}(t)$ size $n \times p_o$
Total Loss: $J(\theta) = \|\ddot{\mathbf{x}} - \ddot{\mathbf{x}}^*\|$
Solution: $\hat{\theta} = \arg\min_\theta J(\theta)$
$\ddot{x}_g$
● Sample points
■ Zero-padding
$\mathbf{x}(t)$
Differentiation
e.g., Finite Difference
Filtering
$\ddot{x}_g(t_1)$ … $\ddot{x}_g(t_n)$
$\mathbf{x}(t_1)$ … $\mathbf{x}(t_n)$
$\mathbf{x}(t), \dot{\mathbf{x}}(t), \ddot{\mathbf{x}}(t)$
Conv Layer 1 $1 \times k \times p_0 \times p_1$
Conv Layer c $1 \times k \times p_{c-1} \times p_c$
FC Layer 1 p_d nodes
FC Layer o p_o nodes
Input Layer
Feature Learning Layer
Fully Connected Layer
Output Layer

Fig. 7. The modified PhyCNN for structural displacement prediction without displacement measurements for training. The only available measurements are the structural accelerations which are used to train the PhyCNN model.

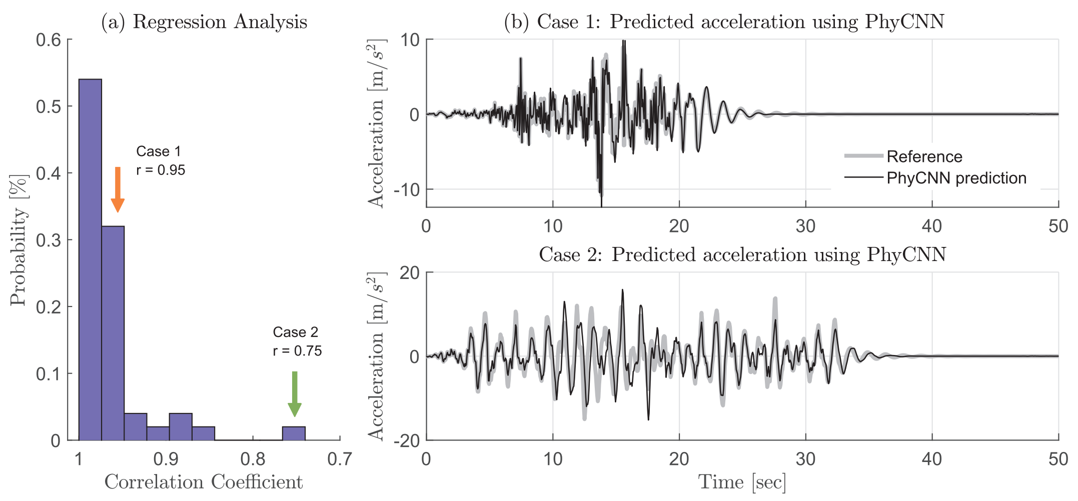


Fig. 8. Prediction performance of structural acceleration $\ddot{x}$ using PhyCNN: (a) regression analysis; (b) examples of predicted time history of $\ddot{x}$.

cluster as shown in Eq. (11):

$$c_i = \frac{1}{|S_i|}\sum_{x_i \in S_i} x_i \tag{11}$$

where S_i is the set of all observations assigned to the ith cluster. The algorithm converges when the cluster assignments no longer change. To determine the optimal number of clusters k for the given observations, the elbow method is used which calculates the distortions for different numbers of clusters [67]. As shown in Fig. 12, the optimal value for k is determined as $k = 4$ where adding another cluster doesn't give much better modeling of data. The limitation of this method is that it cannot always be unambiguously identified. In such cases, other approaches such as Silhouette [68,69] and Cross-validation [70] can be used to find the optimal numbers of clusters, which will be investigated in the future work.

The 21 samples in the green region are divided into four clusters shown in Fig. 11(b). A total number of 11 datasets are selected for training by picking up the datasets on the convex envelope which defines the boundary of interest, as well as the datasets closest to the cluster centroids. The validation datasets can be determined by randomly and evenly picking from each cluster. In this study, since the datasets in Cluster 2 and Cluster 4 are insufficient, the validation datasets are selected from Cluster 1 and Cluster 3 (two for each). The rest 6 datasets plus the 2 datasets out of the boundary (in yellow region) are considered as the prediction datasets to demonstrate the performance of the proposed PhyCNN architecture both within and out of the boundary of interest. A summary of the datasets for training, validation and prediction purposes is illustrated in Fig. 11(c). The training and prediction performance for this 6-story hotel building is presented in the following subsection.

4.3. Predicted displacements using PhyCNN

The training and validation datasets discussed above are used to train the PhyCNN for the 6-story hotel building in San Bernardino, CA, consisting of 11 and 4 samples respectively, each of which contains sequences (with 7,200 data points) of ground motion accelerations as input and the story accelerations as the measurement data. During training, the training datasets are fed into the PhyCNN architecture used in the previous numerical example in Section 3.2 with accelerations as the only field measurements. The trained PhyCNN model is then used to predict structural displacements under new earthquakes. By simply feeding a new ground acceleration into the trained PhyCNN model, it accurately predicts the structural displacements under that excitation. Fig. 13 shows the predicted story displacements of the 3rd floor and roof for Big Bear Lake 2014 and Loma Linda 2016 earthquakes. It can be clearly seen that the PhyCNN prediction matches the historical sensing data very well for earthquakes with different magnitudes and frequency contents. To better illustrate the prediction error, the probability density function (PDF) of the normalized error

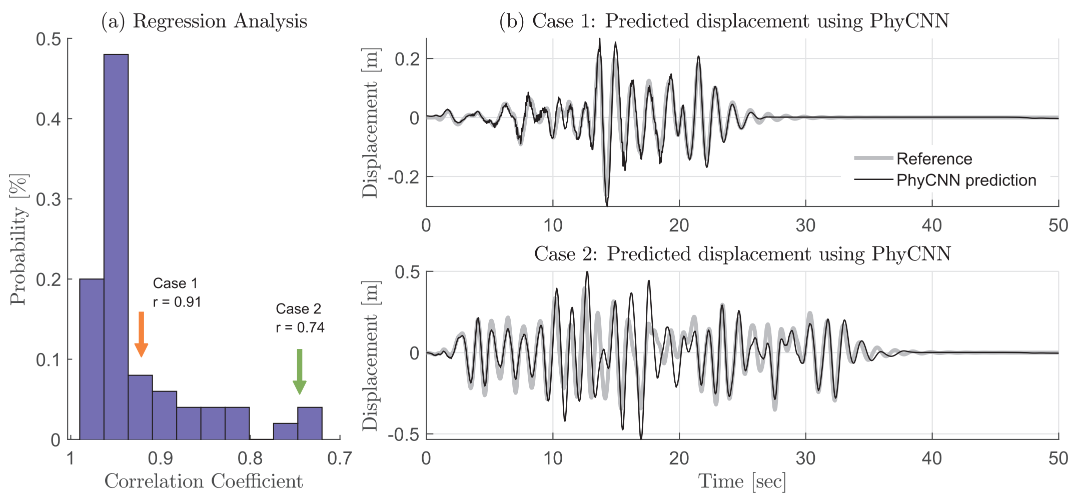


Fig. 9. Prediction performance of structural displacement x using PhyCNN: (a) regression analysis; (b) examples of predicted time history of x.

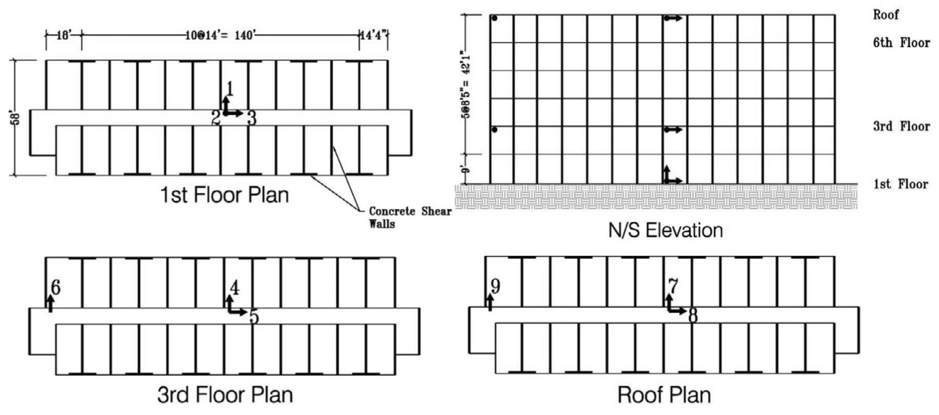


Fig. 10. Sensor layout of the 6-story hotel in San Bernardino, California (Station Number: 23287) (http://www.strongmotioncenter.org/).

distribution defined in Eq. (12) is presented in Fig. 14. It can be seen that the prediction error is mainly located within 5% for the 3rd floor and roof with a confidence interval (CI) of 97% and 93%, respectively. This demonstrates the high prediction accuracy of the proposed PhyCNN approach.

$$\mathscr{P} = \text{PDF}\left\{\frac{y_{true} - y_{predict}}{\max(|y_{true}|)}\right\} \tag{12}$$

The extrapolation ability of the proposed PhyCNN is further verified using the two samples out of the boundary interest (in the yellow region as shown in Fig. 11(a)). Fig. 15 shows the predicted structural displacements under Northridge 1994 and Landers 1992 earthquakes. It is observed that the proposed PhyCNN model is able to well predict structural responses for larger earthquakes, which offers confidence in applying the proposed method for building serviceability or fragility assessment.

4.4. Seismic serviceability analysis of the building

The trained PhyCNN model is used as a surrogate model for structural seismic response prediction which can be further employed to develop fragility functions based on certain limit states for seismic serviceability analysis. The use of limit states in seismic risk assessment reflects the vulnerability of the building structures against earthquakes. The serviceability limit state, as one of the limit states, indicates the structural performance under operational service conditions and aims to minimize any future structural damage due to relatively low earthquakes [71]. For serviceability assessment, the fragility function can be used to describe the probability of exceedance of the serviceability limit state for a specific earthquake intensity measure (IM). The probability of exceeding a given damage level (DL) is defined as a cumulative lognormal distribution function as follows

$$P(\text{DL}|\text{IM} = x) = \Phi\left[\frac{ln(x/\mu)}{\beta}\right] \tag{13}$$

where $P(\text{DL}|\text{IM} = x)$ denotes the probability that a ground motion with IM $= x$ exceeds a given performance level (e.g., serviceability limit state); $\Phi(\cdot)$ is the standard normal cumulative distribution function (CDF); μ is the median of the fragility function (the IM level with 50% probability of exceeding the given DL); and β is the standard deviation of the natural logarithm of the IM which describes the variability for structural damage states.

To calibrate the fragility function, we need to estimate the parameters μ and β. Shinozuka et al. [72,73] estimated μ and β using the maximum likelihood estimation (MLE), denoted by $\mathscr{L}(\cdot)$. In the MLE approach, the damage state is related to a Bernoulli random variable. If the limit state is reached, y_i is set as 1; otherwise, $y_i = 0$. The likelihood function is given by

$$\mathscr{L}(\mu, \beta) = \prod_{i=1}^{N} \Phi\left[\frac{\ln(x_i/\mu)}{\beta}\right]^{y_i}\left[1 - \Phi\left[\frac{\ln(x_i/\mu)}{\beta}\right]\right]^{1-y_i} \tag{14}$$

where Π denotes a product over N earthquake ground motions. Using an optimization algorithm, the two parameters μ and β can be obtained when the likelihood function in the logarithmic space is maximized.

Table 1
Historical data information.

Ind.	Earthquake	Epicenter Distance (km)	PGA(g)	Peak Floor Disp. (cm)	Ind.	Earthquake	Epicenter Distance (km)	PGA (g)	Peak Floor Disp. (cm)
				Training Dataset					
1	Borrego Springs 2010	102.5	0.024	0.406	7	Beaumont 2011	22.6	0.028	0.058
2	Devore 2015	18.6	0.054	0.319	8	Lahabra 2014	60.7	0.024	0.181
3	Fontana 2014	17.3	0.034	0.102	9	Loma Linda 2017	4.8	0.025	0.047
4	Inglewood 2009	99.4	0.008	0.052	10	Ontario 2011	27.8	0.004	0.015
5	Ocotillo 2010	197.3	0.007	0.135	11	Yorba Linda 2012	50.3	0.003	0.021
6	San Bernardino 2009	5.1	0.094	0.852					
				Validation Dataset					
12	Banning 2016	38.2	0.019	0.102	14	Redlands 2010	11.5	0.019	0.145
13	Banning 2010	38.9	0.005	0.017	15	Trabuco Canyon 2018	40.9	0.01	0.04
				Prediction Dataset					
16	Beaumont 2010	28.1	0.005	0.026	20	Devore 2012	23.5	0.01	0.052
17	Big Bear Lake 2014	33.4	0.011	0.107	21	Loma Linda 2013	4.6	0.012	0.058
18	Fontana 2015	15.5	0.01	0.034	22	Northridge 1994	117.4	0.07	2.67
19	Loma Linda 2016	6.9	0.008	0.034	23	Landers 1992	79.9	0.08	9.38

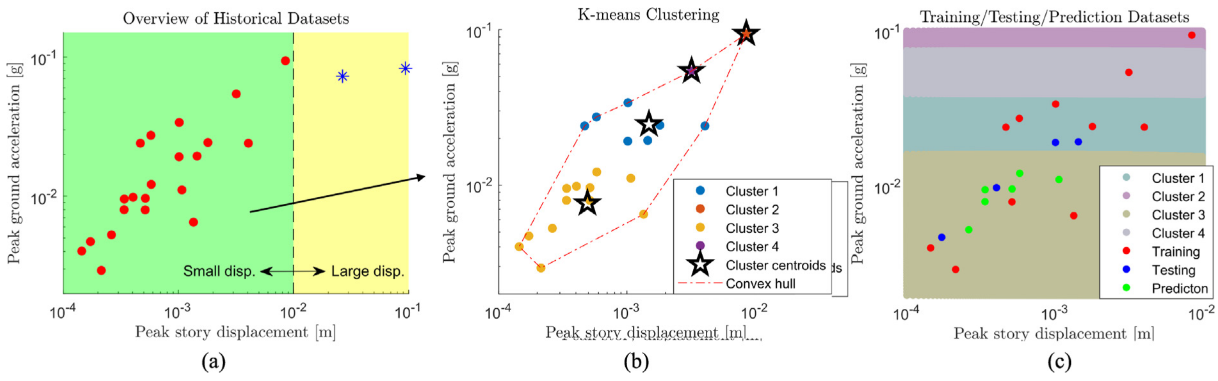


Fig. 11. Data clustering using K-means clustering: (a) overview of historical datasets; (b) illustration of clusters ($k = 4$) and convex envelope; (c) training/validation/prediction datasets determined based on K-means algorithm and convex envelope.

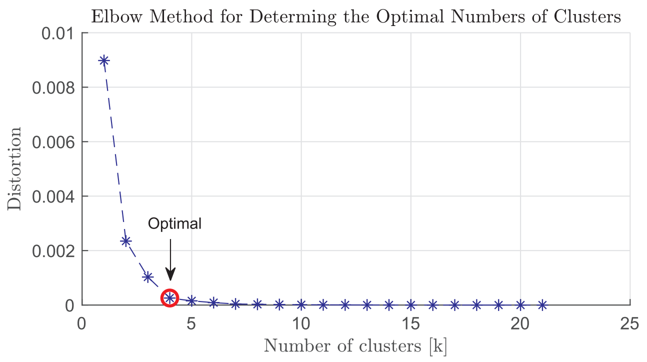


Fig. 12. Identification of optimal number of clusters at the Elbow point.

The serviceability assessment is conducted based on the performance-based engineering method. According to ASCE/SEI 41-06 standard [74], the building performance levels include operational, immediate occupancy, life safety, and collapse prevention. Both the operational and immediate occupancy performance level can be considered as the serviceability limit state. ASCE/SEI 41–06 also provides the recommended values for the maximum inter-story drift for each performance level and type of the structure. An alternative standard for fragility analysis is HAZUS [75], in which the damage states of both structural and nonstructural components are defined. However, the inter-story drifts are typically unavailable due to the limitation of the sensor locations. Instead, the drift angle, defined as the ratio of the story deflection to story height, can be calculated and used as the threshold for serviceability assessment. Typical values of the drift angle for serviceability check lie in the range of [1/600, 1/100] for different building types and materials [76]. In this paper, the threshold of the serviceability limit state is defined as 0.5% for the maximum drift angle under the earthquake with a 10% probability of exceedance in a 50-year period. For serviceability assessment of a building, the structural responses can be predicted under a group of new ground motions using the trained PhyCNN model. Thus, the fragility function is obtained using Eq. (13) based on the serviceability limit state.

The serviceability of the 6-story hotel building in San Bernardino, CA is assessed based on the trained PhyCNN model described in Section 4.3. A suite of 100 ground motion records is input to the trained PhyCNN model to predict the structural displacements in an incremental dynamic analysis (IDA) setting. The 100 earthquake ground motions are selected from the PEER strong motion database [64] in the area of San Bernardino with a 10% probability of exceedance in 50 years. The mean response spectrum of the selected ground motion records matches the design spectrum of the 6-story hotel building. Fig. 16 shows the predicted displacements under two example new earthquakes. All the predicted displacements are then used to determine the fragility function with respect to the serviceability limit state. The fragility curve of the serviceability limit state is obtained based on Eq. (13) and Eq. (14) and shown in Fig. 17. It is seen that the probabilities of exceeding the serviceability limit state are around 47%, 78%, and 90% for future earthquakes with PGA of 0.1 g, 0.2 g, and 0.3 g, respectively. It is noted that the data-driven fragility curve can provide valuable information to guide the design of maintenance and rehabilitation strategies for the building.

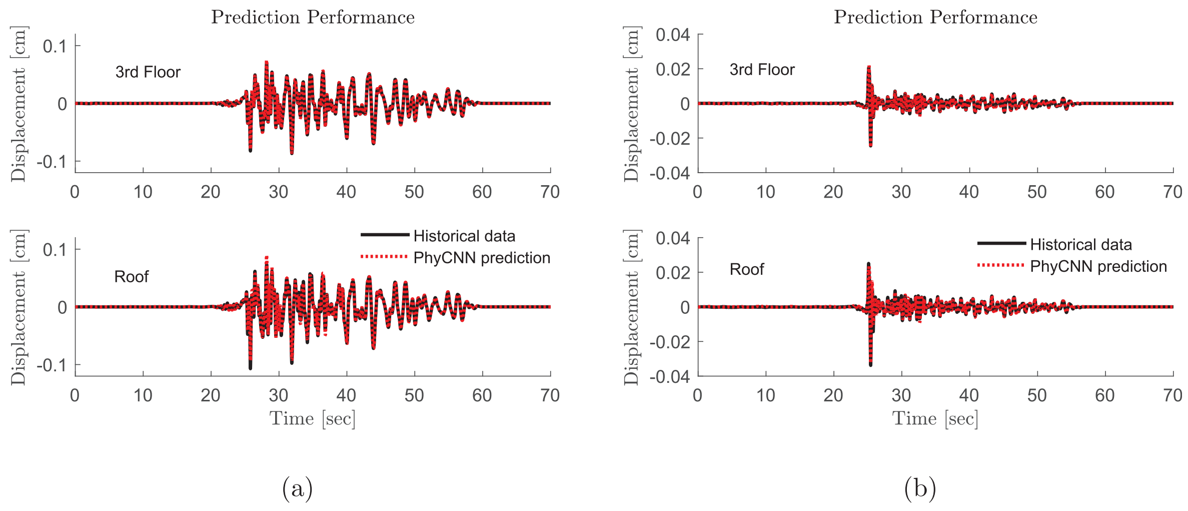


Fig. 13. Prediction performance of the proposed PhyCNN model.

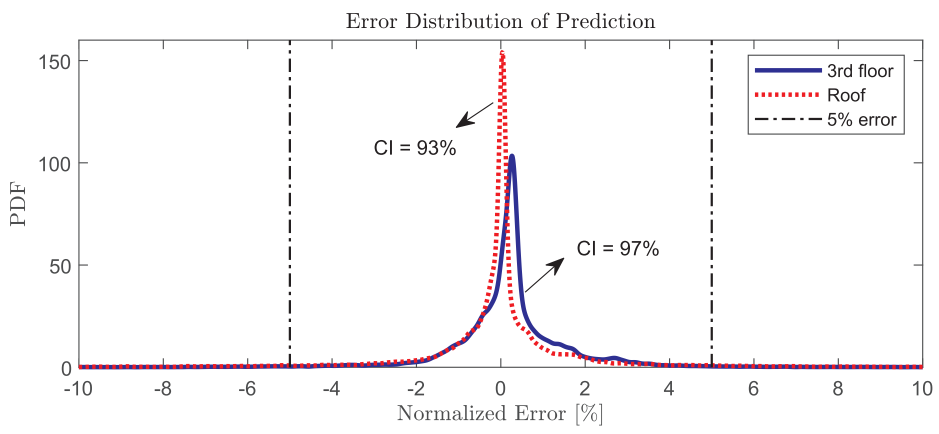


Fig. 14. Error distribution of the prediction datasets using the proposed PhyCNN model.

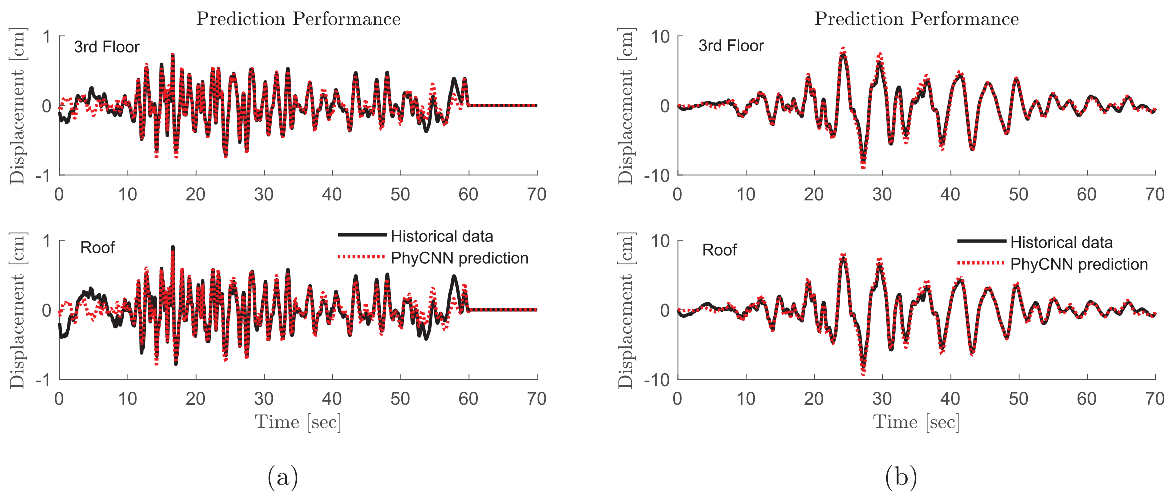


Fig. 15. Prediction performance of the proposed PhyCNN model under larger seismic intensities.

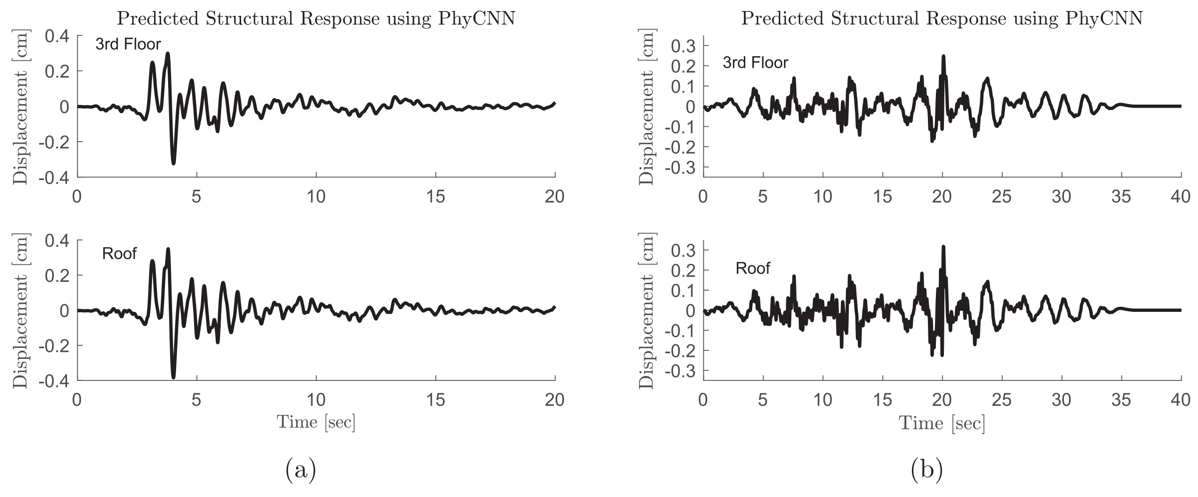


Fig. 16. Predicted structural responses under two example earthquakes using PhyCNN.

5. Conclusions

This paper presents a novel physics-guided convolutional neural network (PhyCNN) architecture to develop data-driven surrogate models for modeling/prediction of seismic response of building structures. The deep PhyCNN model includes several convolution layers and fully-connected layers to interpret the data, a graph-based tensor differentiator, and physics constraints. The key concept to leverage available physics (e.g., the law off dynamics) that can provide constraints to the network outputs, alleviate overfitting issues, reduce the need of big training datasets, and thus improve the robustness of the trained model for more reliable prediction. The performance of the proposed approach was illustrated by both numerical and experimental examples with limited datasets either from simulations or field sensing. The results show that the proposed deep PhyCNN model is an effective, reliable and computationally efficient approach for seismic structural

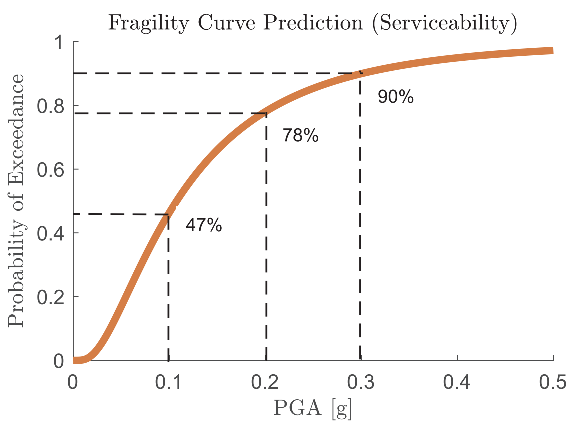


Fig. 17. Predicted fragility curve of the serviceability limit state for the 6-story hotel building in San Bernardino using PhyCNN.

response modeling. The trained model can further serve as a basis for developing fragility function for building serviceability assessment. Overall, the proposed algorithm is fundamental in nature which is scalable to other structures (e.g., bridges) under other types of hazard events.

Declaration of Competing Interest

None.

Acknowledgement

This work was supported by the Engineering for Civil Infrastructure program at National Science Foundation under grant CMMI-2013067, and the TIER 1 Seed Grant program at Northeastern University, which are greatly acknowledged. The data and codes used in this paper will be publicly available on GitHub at https://github.com/zhry10/PhyCNN after the paper is published.

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.engstruct.2020.110704.

References

[1] Yuen K-V. Bayesian methods for structural dynamics and civil engineering. John Wiley & Sons; 2010.
[2] Yuen K-V. Updating large models for mechanical systems using incomplete modal measurement. Mech Syst Signal Process 2012;28:297–308.
[3] Yuen K-V, Kuok S-C. Efficient bayesian sensor placement algorithm for structural identification: a general approach for multi-type sensory systems. Earthquake Eng Struct Dynam 2015;44(5):757–74.
[4] Sun H, Büyüköztürk O. Probabilistic updating of building models using incomplete modal data. Mech Syst Signal Process 2016;75:27–40.
[5] Yan G, Sun H, Büyüköztürk O. Impact load identification for composite structures using Bayesian regularization and unscented Kalman filter. Struct Control Health Monit 2017;24(5):e1910.
[6] Chen Z, Zhang R, Zheng J, Sun H. Sparse Bayesian learning for structural damage identification. Mech Syst Signal Process 2020;140:106689.
[7] Yang JN, Lin S, Huang H, Zhou L. An adaptive extended Kalman filter for structural damage identification. Struct Control Health Monit 2006;13(4):849–67.
[8] Wu M, Smyth AW. Application of the unscented Kalman filter for real-time nonlinear structural system identification. Struct Control Health Monit 2007;14(7):971–90.
[9] Xie Z, Feng J. Real-time nonlinear structural system identification via iterated unscented Kalman filter. Mech Syst Signal Process 2012;28:309–22.
[10] Nakata N, Snieder R, Kuroda S, Ito S, Aizawa T, Kunimi T. Monitoring a building using deconvolution interferometry. i: Earthquake-data analysis. Bull Seismol Soc Am 2013;103(3):1662–78.
[11] Nakata N, Snieder R. Monitoring a building using deconvolution interferometry. ii: Ambient-vibration analysis. Bull Seismol Soc Am 2013;104(1):204–13.
[12] Nakata N, Tanaka W, Oda Y. Damage detection of a building caused by the 2011 Tohoku-oki earthquake with seismic interferometry. Bull Seismol Soc Am 2015;105(5):2411–9.
[13] Mordret A, Sun H, Prieto GA, Toksöz MN, Büyüköztürk O. Continuous monitoring of high-rise buildings using seismic interferometry. Bull Seismol Soc Am 2017;107(6):2759–73.
[14] Sun H, Mordret A, Prieto GA, Toksöz MN, Büyüköztürk O. Bayesian characterization of buildings using seismic interferometry on ambient vibrations. Mech Syst Signal Process 2017;85:468–86.
[15] Sjöberg J, Zhang Q, Ljung L, Benveniste A, Delyon B, Glorennec P-Y, et al. Nonlinear black-box modeling in system identification: a unified overview. Automatica 1995;31(12):1691–724.
[16] Braun JE, Chaturvedi N. An inverse gray-box model for transient building load prediction. HVAC&R Res 2002;8(1):73–99.
[17] Moaveni B, Conte JP, Hemez FM. Uncertainty and sensitivity analysis of damage identification results obtained using finite element model updating. Comput-Aided Civil Infrastruct Eng 2009;24(5):320–34.
[18] Belleri A, Moaveni B, Restrepo JI. Damage assessment through structural identification of a three-story large-scale precast concrete structure. Earthquake Eng Struct Dynam 2014;43(1):61–76.
[19] Yousefianmoghadam S, Behmanesh I, Stavridis A, Moaveni B, Nozari A, Sacco A. System identification and modeling of a dynamically tested and gradually damaged 10-story reinforced concrete building. Earthquake Eng Struct Dynam 2018;47(1):25–47.
[20] Sohn H, Farrar CR, Hemez FM, Shunk DD, Stinemates DW, Nadler BR, et al. A review of structural health monitoring literature. USA: Los Alamos National Laboratory; 1996-2001.
[21] Kerschen G, Worden K, Vakakis AF, Golinval J-C. Past, present and future of nonlinear system identification in structural dynamics. Mech Syst Signal Process 2006;20(3):505–92.
[22] Wu R-T, Jahanshahi MR. Data fusion approaches for structural health monitoring and system identification: past, present, and future. Struct Health Monit 2018. 1475921718798769.
[23] Brownjohn JM, Xia P-Q. Dynamic assessment of curved cable-stayed bridge by model updating. J Struct Eng 2000;126(2):252–60.
[24] Yuen K-V, Katafygiotis LS. Model updating using noisy response measurements without knowledge of the input spectrum. Earthquake Eng Struct Dynam 2005;34(2):167–87.
[25] Weber B, Paultre P. Damage identification in a truss tower by regularized model updating. J Struct Eng 2009;136(3):307–16.
[26] Song W, Dyke S. Real-time dynamic model updating of a hysteretic structural system. J Struct Eng 2013;140(3):04013082.
[27] Sun H, Betti R. A hybrid optimization algorithm with bayesian inference for probabilistic model updating. Comput-Aided Civil Infrastruct Eng 2015;30(8):602–19.
[28] Skolnik D, Lei Y, Yu E, Wallace JW. Identification, model updating, and response prediction of an instrumented 15-story steel-frame building. Earthquake Spectra 2006;22(3):781–802.
[29] Fishwick PA. Neural network models in simulation: a comparison with traditional modeling approaches. Proceedings of the 21st conference on Winter simulation. ACM; 1989. p. 702–9.
[30] Ediger VŞ, Akar S. Arima forecasting of primary energy demand by fuel in turkey. Energy Policy 2007;35(3):1701–8.
[31] Irie B, Miyake S. Capabilities of three-layered perceptrons. in: IEEE international conference on neural networks 1988; Vol. 1, p. 218.
[32] Hornik K. Approximation capabilities of multilayer feedforward networks. Neural Networks 1991;4(2):251–7.
[33] Chen S, Billings S. Neural networks for nonlinear dynamic system modelling and identification. Int J Control 1992;56(2):319–46.
[34] Tianping C, Hong C. Approximations of continuous functions by neural networks with application to dynamic system. IEEE Trans Neural Networks 1993;4(6):910–8.
[35] Chen T, Chen H. Approximation capability to functions of several variables, nonlinear functionals, and operators by radial basis function neural networks. IEEE Trans Neural Networks 1995;6(4):904–10.
[36] Zhang J, Sato T, Iai S. Novel support vector regression for structural system identification. Struct Control Health Monit 2007;14(4):609–26.
[37] Yinfeng D, Yingmin L, Ming L, Mingkui X. Nonlinear structural response prediction based on support vector machines. J Sound Vib 2008;311(3–5):886–97.
[38] Lightbody G, Irwin GW. Multi-layer perceptron based modelling of nonlinear systems. Fuzzy Sets Syst 1996;79(1):93–112.
[39] Ying W, Chong W, Hui L, Renda Z. Artificial neural network prediction for seismic response of bridge structure. In: 2009 International conference on artificial intelligence and computational intelligence, IEEE, 2009, 2009;2. p. 503–6.
[40] Christiansen NH, Høgsberg JB, Winther O. Artificial neural networks for nonlinear dynamic response simulation in mechanical systems. In: NSCM-24; 2011.
[41] Huang C, Hung S, Wen C, Tu T. A neural network approach for structural identification and diagnosis of a building from seismic response data. Earthquake Eng Struct Dynam 2003;32(2):187–206.
[42] Lagaros ND, Papadrakakis M. Neural network based prediction schemes of the nonlinear seismic response of 3d buildings. Adv Eng Softw 2012;44(1):92–115.
[43] Mandic DP, Chambers J. Recurrent neural networks for prediction: learning algorithms, architectures and stability. John Wiley & Sons Inc; 2001.
[44] Medsker LR, Jain L. Recurrent neural networks. Des Appl, 5.
[45] Yu Y, Yao H, Liu Y. Aircraft dynamics simulation using a novel physics-based learning method. Aerosp Sci Technol 2019;87:254–64.
[46] Zhang R, Chen Z, Chen S, Zheng J, Büyüköztürk O, Sun H. Deep long short-term memory networks for nonlinear structural seismic response prediction. Comput Struct 2019;220:55–68.

[47] Cha Y-J, Choi W, Büyüköztürk O. Deep learning-based crack damage detection using convolutional neural networks. Comput-Aided Civil Infrastruct Eng 2017;32(5):361–78.
[48] Atha DJ, Jahanshahi MR. Evaluation of deep learning approaches based on convolutional neural networks for corrosion detection. Struct Health Monitor 2018;17(5):1110–28.
[49] Sun S-B, He Y-Y, Zhou S-D, Yue Z-J. A data-driven response virtual sensor technique with partial vibration measurements using convolutional neural network. Sensors 2017;17(12):2888.
[50] Wu R-T, Jahanshahi MR. Deep convolutional neural network for structural dynamic response estimation and system identification. J Eng Mech 2018;145(1):04018125.
[51] Raissi M. Deep hidden physics models: deep learning of nonlinear partial differential equations. J Mach Learn Res 2018;19(1):932–55.
[52] Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys 2019;378:686–707.
[53] Sun L, Gao H, Pan S, Wang JX. Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data, arXiv preprint arXiv:1906.02382.
[54] Zhu Y, Zabaras N, Koutsourelakis P-S, Perdikaris P. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. J Comput Phys 2019;394:56–81.
[55] Ciresan DC, Meier U, Masci J, Gambardella LM, Schmidhuber J. Flexible, high performance convolutional neural networks for image classification. Twenty-second international joint conference on artificial intelligence. 2011.
[56] Lawrence S, Giles CL, Tsoi AC, Back AD. Face recognition: a convolutional neural-network approach. IEEE Trans Neural Networks 1997;8(1):98–113.
[57] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature 2015;521(7553):436.
[58] Nair V, Hinton GE. Rectified linear units improve restricted boltzmann machines. Proceedings of the 27th international conference on machine learning (ICML-10). 2010. p. 807–14.
[59] Xu B, Wang N, Chen T, Li M. Empirical evaluation of rectified activations in convolutional network, arXiv preprint arXiv:1505.00853.
[60] Trottier L, Gigu P, Chaib-draa B, et al. Parametric exponential linear unit for deep convolutional neural networks. 2017 16th IEEE International conference on machine learning and applications (ICMLA). IEEE; 2017. p. 207–14.
[61] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 2014;15(1):1929–58.
[62] Chollet F, et al., Keras; 2015.
[63] Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, et al. Tensorflow: large-scale machine learning on heterogeneous distributed systems, arXiv preprint arXiv:1603.04467.
[64] Chiou B, Darragh R, Gregor N, Silva W. Nga project strong-motion database. Earthquake Spectra 2008;24(1):23–44.
[65] Haddadi H, Shakal A, Stephens C, Savage W, Huang M, Leith W, et al. Center for engineering strong-motion data (cesmd). Proceedings of the 14th world conference on earthquake engineering. 2008.
[66] Ng A. Clustering with the k-means algorithm. Mach Learn.
[67] Kodinariya TM, Makwana PR. Review on determining number of cluster in k-means clustering. Int J 2013;1(6):90–5.
[68] Pollard KS, Van Der Laan MJ. A method to identify significant clusters in gene expression data.
[69] Kaufman L, Rousseeuw PJ. Finding groups in data: an introduction to cluster analysis Vol. 344. John Wiley & Sons; 2009.
[70] Smyth P. Clustering using monte carlo cross-validation. In: Kdd, Vol. 1; 1996. p. 26–133.
[71] Dymiotis-Wellington C, Vlachaki C. Serviceability limit state criteria for the seismic assessment of rc buildings. In: Proceedings of the 13th world conference on earthquake engineering. Vancouver, Canada; 2004. p. 1–10.
[72] Shinozuka M, Feng MQ, Kim H-K, Kim S-H. Nonlinear static procedure for fragility curve development. J Eng Mech 2000;126(12):1287–95.
[73] Shinozuka M, Feng MQ, Lee J, Naganuma T. Statistical analysis of fragility curves. J Eng Mech 2000;126(12):1224–31.
[74] Committee ASRS, et al., Seismic rehabilitation of existing buildings (asce/sei 41–06), American Society of Civil Engineers, Reston, VA.
[75] FEMA H. Multi-hazard loss estimation methodology, earthquake model, Washington, DC, USA: Federal Emergency Management Agency.
[76] Griffis LG. Serviceability limit states under wind load. Eng J 1993;30(1):1–16.