Engineering Structures 229 (2021) 111582

Contents lists available at ScienceDirect

Engineering Structures

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

# DynNet: Physics-based neural architecture design for nonlinear structural response modeling and prediction

Soheil Sadeghi Eshkevari [a,*], Martin Takáč [b], Shamim N. Pakzad [a], Majid Jahani [b]

[a] Department of Civil and Environmental Engineering, Lehigh University, Bethlehem, PA 18015, USA
[b] Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA 18015, USA

ARTICLE INFO

*Keywords:*
Deep learning
Physics-based neural network
Ordinary differential equation
Structural dynamics
Earthquake engineering
Dynamic response prediction

ABSTRACT

Data-driven models for predicting dynamic responses of linear and nonlinear systems are of great importance due to their wide application from probabilistic analysis to inverse problems such as system identification and damage diagnosis. In this study, a physics-based recurrent neural network model is designed that is able to estimate the dynamics of linear and nonlinear multiple degrees of freedom systems given the ground motions. The model is able to estimate a complete set of responses, including displacement, velocity, acceleration, and internal forces. Compared to the most advanced counterparts, this model requires smaller number of trainable variables while the accuracy of predictions is higher for long trajectories. In addition, the architecture of the recurrent block is inspired by differential equation solver algorithms and it is expected that this approach yields more generalized solutions. In the training phase, we propose multiple novel techniques to substantially accelerate the learning process using smaller datasets, such as hardsampling, utilization of a trajectory loss function, and implementation of a trust-region optimization approach. Numerical case studies are conducted to examine the strength of the network to learn different nonlinear behaviors. It is shown that the network is able to capture different nonlinear behaviors of dynamic systems with high accuracy and with no need for prior information or very large datasets.

## 1. Introduction

Dynamic response prediction of structural systems has been a great tool for design and assessment of individual buildings as well as reliability analysis of infrastructure and large urban areas. Traditionally, this process is executed by building numerical models of dynamic systems and predicting responses using numerical differential equation solvers such as Newmark-$\beta$ method. However, this approach is suitable for structures with known physical properties (i.e., mass, stiffness, and damping matrices) with very accurate analytical models for the nonlinear components of structures. Structural health monitoring (SHM) methods have been effective in identifying mechanical properties of the existing structures. Yet, the dynamic response simulation of an existing system requires a comprehensive SHM phase for model updating [1–4]. In addition, for an accurate simulation of a structure with nonlinear components, emerging technologies such as real-time hybrid simulation are proposed [5–7]. This approach is also limited to individual nonlinear structural components and requires advanced experimental and numerical resources.

Artificial intelligence has been one of the most useful and promising tools in science and technology over the past few decades. In particular, machine learning has demonstrated a great potential for learning and predicting nonlinear behaviors and trends in large and noisy datasets [8]. Neural Networks (NN) have shown an exceptional potential as universal function approximators with minimal need for prior information about the underlying knowledge of a problem [9,10]. However, in engineering applications, black-box function approximators are less favored due to the fact that for many of those, solid underlying equations/models exist. Knowledge-based machine learning approach intends to bridge this gap by contributing governing equations into machine learning models [11].

### 1.1. Artificial intelligence in structural engineering

In general, the major applications of machine learning in structural engineering can be divided into the following categories: (a) system identification (SID); (b) damage detection; and (c) dynamic response prediction of structural systems. A detailed overview of machine

* Corresponding author. Current address: MIT Senseable City Lab.
*E-mail address:* ssadeghi@mit.edu (S. Sadeghi Eshkevari).

https://doi.org/10.1016/j.engstruct.2020.111582
Received 10 April 2020; Received in revised form 23 September 2020; Accepted 11 November 2020
Available online 21 December 2020

learning algorithms for damage detection is given in [12,13]. In summary, the methods use machine learning algorithms (e.g., support vector machines (SVM) and multi-layer perceptrons (MLP)) for classification between damaged and undamaged states of structural components based on low-level inputs (e.g., motion sensor data). A multi-stage damage detection method is proposed in [14] in which signal features are extracted using wavelet transforms and an MLP network diagnoses whether a damage has occurred. Gui et al. [15] proposed a method for feature extraction from sensor signals and damage classification based on these extracted features using SVM. More recently, end-to-end damage detection algorithms are emerging in which feature extraction and damage detection stages are combined in a single estimator. Abdeljaber et al. [16] proposed a vibration-based convolutional neural network (CNN) for direct damage detection and localization based on sensor time signals. Gulgec et al. [17] proposed a one-step vision-based damage detection and localization method via CNN which uses 2D strain fields as input.

Fewer studies have investigated data-driven methods for system identification due to the inherent model-dependency of this problem. Some efforts have been made to reconstruct underlying equations using data-driven algorithms. Brunton et al. [18] proposed a look-up approach to reconstruct the governing equation of dynamic systems using sparse identification. More recent studies investigate machine learning solutions with model-guided constraints. Raissi and Karniadakis [19] introduced hidden physics models that are able to identify underlying physics of dynamic systems using small datasets. In structural engineering, Sadeghi Eshkevari et al. [20] proposed a data-driven approach for bridge modal identification using mobile sensing data. The model is highly constrained by the modal superposition law of structural dynamics and could successfully identify complete modal properties.

In addition to diagnosis and monitoring tasks that are the objectives of the previous studies, data-driven approaches for dynamic response prediction of structural systems have been of great importance and interest. Finite element analysis (FEA) along with nonlinear time history analysis (NTHA) has enabled very accurate dynamic response estimations; however, both techniques are computationally expensive and require detailed information of the system. By the emergence of probabilistic reliability analyses of individual and clusters of structures subject to hazards (e.g., earthquake or hurricanes), it is realistically impractical to carry out extensive FEA and NTHA analyses of increasingly larger systems [21,22]. Therefore, faster, reliable, and more flexible approaches are highly required.

### 1.2. Data-driven dynamic response prediction

Dynamic response prediction of structures using statistical methods have been widely investigated over the last few years. The approaches span from model-based predictions to data-driven models such as autoregressive moving average (ARMA) models or neural networks. A model-based full state predictor was proposed in [23] that incorporates a prior nonlinear model of the building for experimental response prediction. Mattson and Pandit [24] proposed an autoregressive model to predict major trends of the dynamic response; however, the effect of the exogenous input was remained unmet and considered as a residual. In fact, despite their simplicity, ARMA-based models are limited to stationary and linear systems. To address that, Bornn et al. [25] proposed an autoregressive SVM model that incorporates nonlinear functionalities within the prediction equation. Neural networks (NN) have been the most recent approach for dynamic response prediction due to their flexibility and great performance in regression problems. The pioneer studies were focused on simple MLP models for partial one-step ahead response predictions (i.e., predictions include some but not all of the followings: displacement, velocity, acceleration, and internal force of all degrees of freedom). Lightbody and Irwin [26] proposed a single layer neural network in which the output is a weighted sum of multiple trainable AR models with *Tanh* activation. The study was a breakthrough that enhanced estimator complexities from an individual linear model to a nonlinear ensemble of linear models. By recent computational developments, deeper MLP networks were utilized for more comprehensive dynamic response predictions of nonlinear cases. Lagaros and Papadrakakis [27] proposed a MLP for one-step ahead response prediction of nonlinear buildings. The method showed great performance both numerically and experimentally, however, the prediction was limited to displacement time histories. Note that in general there is no guarantee for reasonable predictions of other response components (e.g., velocity and accelerations) by using direct integration or differentiation on a single component when using data-driven regression methods. Therefore, yet more comprehensive predictive models are required.

Theoretically, MLPs perform ideally when the input features are fully independent. In the dynamic response prediction problem, however, a high inter-dependency between responses at consequent time steps exist. Therefore, other neural network architectures have been also utilized for this specific problem. CNNs are known for their strength in extracting local (e.g., spatial or temporal) features and incorporating inter-dependency of the input nodes [28]. In addition, the variable space of the models is dramatically reduced since fixed sized kernels are being trained rather than large matrices from fully-connected layers. CNNs are mostly used for computer vision applications in which 2D kernels are applied on pixel pallets. In signal processing, 1D kernels are more proper choices. A dynamic response predictor for linear systems using CNNs is introduced in [29]. More recently, Wu and Jahanshahi [30] proposed a CNN-based algorithm for different partial dynamic response predictions. The most advanced case included the prediction of acceleration responses at the roof level of a multi degrees of freedom (MDOF) system given the ground motion.

Comprehensive dynamic response prediction of nonlinear systems has been investigated in a few recent studies. Zhang et al. [31] confirms that recurrent neural networks (RNN) are great candidates in terms of the architecture for structural dynamic response modeling, however, technically they suffer from gradient-vanishing issues during the training process. In fact, RNN models have been a frequently used architecture in the previously mentioned models (i.e., all one-step ahead response prediction models are basically RNN models). Based on this argument, Zhang et al. [32] proposes a long short-term memory (LSTM) architecture for the response modeling in order to address the gradient-vanishing issue. The primary difference of LSTM models compared to the vanilla RNN models is the special architecture that allows for learning long-term temporal dependencies. This difference also resolves the gradient-vanishing issue of the RNN models. The study successfully predicted displacements, velocities, accelerations, and internal forces using the ground motion in different nonlinear cases. However, the

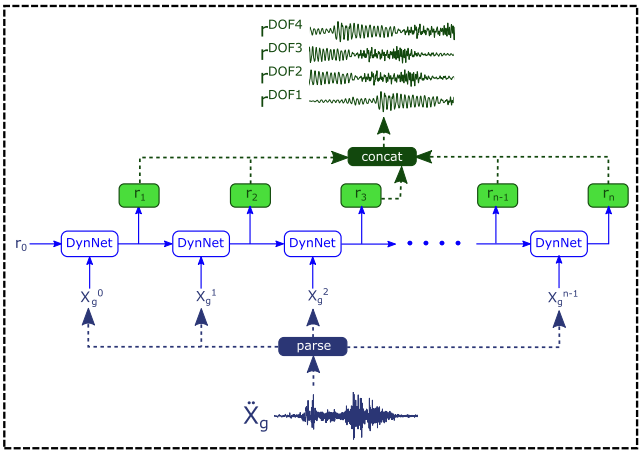


**Fig. 1.** Schematic diagram of DynNet and the conversion process from the ground motion to the structural response.

model consists of a large trainable variable space which requires very long training process. For instance, the DeepLSTM-s model [32] requires 50,000 epochs for training and the network includes ~130,000 trainable variables.

The same research team has also recently introduced physics-guided models using CNN and LSTM architectures for dynamic response prediction problem [31,33]. The studies propose additional terms in the loss function of the problem which penalizes deviations in the equation of motion and equilibrium when predicted outputs are plugged in. The studies showed that imposing this new physical constraint improved the prediction accuracy and reduced overfitting issues. Despite their high accuracy and completeness, the NN architectures are vanilla versions of the common NN types with no guidance from the physics or the problem. This results in over-complicated networks that require high number of training epochs (e.g., 10,000 for the first phase of the training only). In addition, LSTM model requires a fixed signal length which is limiting.

In our study, we focus on designing the architecture of a recurrent neural cell that updates the state from the current time step to the next (i.e., one-step ahead predictor) with the neural connections that are inspired by exact numerical differential equation solvers. We believe that an ideal network is able to predict a response merely based on the current time step of a full state space, as it is hardcoded in the simulation algorithms such as Newmark-$\beta$.

### 1.3. Motivation

As the ubiquity of data-driven methods grows, the generalization and reliability of these models become more important. The vast majority of the available research train neural networks with no consideration for the solid knowledge that governs the actual problem in hand. In addition, for engineering applications as opposed to the pure data science problems, the available data is not extremely large and does not cover the entire domain of application possibilities (e.g., data is available for a limited domain of linear responses in operational conditions). These two concerns demand for incorporating physical constraints into the architecture design of the NNs. On the other hand, as the problem holds more constraints, the training process eventually becomes harder. This study proposes a new approach to impose a special architecture that is inspired by implicit numerical solvers of the differential equations into a recurrent cell for full response prediction of nonlinear MDOF systems. The proposed network is termed DynNet in this article, standing for Dynamic Network. Moreover, this study recommends multiple techniques so that the training process becomes smoother and more reliable.

DynNet is a recurrent cell that performs one-step ahead prediction of the full state space of a MDOF nonlinear dynamic system given a desired ground motion. The schematic structure of the network is presented in Fig. 1. This architecture has no limitation for the length of the signal ($n$ is the number of the ground motion discrete samples). Our contribution is to design the architecture based on implicit dynamic simulation algorithms for nonlinear time history analysis (e.g., nonlinear Newmark-$\beta$ method). The key idea is that if the numerical algorithm is suitable and sufficiently accurate for nonlinear response analysis, a similar architecture has to be successful in learning the same nonlinear model from raw data. In addition, the architecture design is inspired by Residual Networks [34] (i.e., ResNets) that have shown outstanding performances in learning partial differential equations from raw data. DynNet has significantly smaller variable space compared to the most accurate counterparts.

In terms of network optimization, this study utilizes second order trust region method which significantly reduces required training iterations. Training dynamic blocks for one-step ahead prediction is highly sensitive to instability. To overcome this challenge, we introduce a projection loss function. In addition, to accelerate learning ability of the network for nonlinear transitions, an importance sampling technique is proposed and implemented. Although DynNet is strongly constrained which results in a more involved training, its smaller variable space and high constraints enable network training with very limited amount of data. The physical interpretability of DynNet also alleviates modeling of severe nonlinear behaviors as well as very long signals, as we will show in the next sections.

In the following section, the detailed architecture of the network is elaborated. In Section 3 the technical approaches for faster and more robust training process of DynNet are presented (e.g., the optimization algorithm, the customized loss function, and the importance sampling technique). In Section 4 two numerical case studies are presented in which different types of nonlinearity are imposed. The summary of the method along with the highlights are presented in Section 5.

## 2. Physics-based neural network architecture design

### 2.1. Numerical solution for direct problems

For simulation of dynamic systems, implicit numerical solvers process responses at time step $i$ to derive responses at time step $i+1$. In fact, regardless of the complexity and level of nonlinearity of the problem, simulators require no further information for one-step ahead predictions. Relying on this fact, an ultimate simulator that learns from data should be a dynamic cell that is able to perform the one-step ahead prediction with high accuracy and low cumulative error. In addition, considering the causality of dynamic systems as well as their short memory (i.e., a few recent samples are sufficient for the next step prediction), LSTM models seem unnecessarily over-complicated. DynNet is a robust one-step ahead dynamic cell that is very sharp in learning nonlinearities as well as robust to noise. In this study, we do not use a simplified version of existing networks such as CNN or LSTM, but instead we design the internal cell connections in a way that conform with common dynamic numerical solvers. The nonlinear version of Newmark's algorithm is shown in Algorithm 1 [35].

**Algorithm 1.** Newmark's Method for Nonlinear Systems.

1: **Input:** $u_i, \dot{u}_i, \ddot{u}_i, S_i, \ddot{x}_i^g$, TangentStiffness(.), NonlinearForce(.)
2: $a_1, a_2, a_3, C_1, C_2, C_3, C_4, C_5, C_6, M, \Gamma :=$ Constant
3: $\widehat{P}_{i+1} = M\Gamma\ddot{x}_i^g + a_1 u_i + a_2 \dot{u}_i + a_3 \ddot{u}_i$
4: $R^{(0)} = \widehat{P}_{i+1}$
5: $j = 0$
6: $K_i^t =$ TangentStiffness$(u_i, \dot{u}_i, \ddot{u}_i, S_i)$
7: **while** $abs(R^{(j)}) <$ *threshold* **do**
8: $\quad R^{(j)} = \widehat{P}_{i+1} - S_{i+1}^{(j)} - a_1 u_{i+1}^{(j)}$
9: $\quad (K_{i+1}^t)^{(j)} = (K_{i+1}^t)^{(j)} + a_1$
10: $\quad \Delta u^{(j)} = ((K_{i+1}^t)^{(j)})^{-1} R^{(j)}$
11: $\quad u_{i+1}^{(j+1)} = u_{i+1}^{(j)} + \Delta u^{(j)}$
12: $\quad S_{i+1}^{(j+1)} =$ NonlinearForce$(u_{i+1}^{(j+1)}, S_i^{(j)})$
13: $\quad j = j + 1$
14: **end while**
15: $u_{i+1} = u_{i+1}^{(j)}$
16: $S_{i+1} = S_{i+1}^{(j)}$
17: $\dot{u}_{i+1} = C_1(u_{i+1} - u_i) + C_2 \dot{u}_i + C_3 \ddot{u}_i$
18: $\ddot{u}_{i+1} = C_4(u_{i+1} - u_i) + C_5 \dot{u}_i + C_6 \ddot{u}_i$
19: **Return** $u_{i+1}, \dot{u}_{i+1}, \ddot{u}_{i+1}, S_{i+1}$

In this algorithm, $u_i, \dot{u}_i, \ddot{u}_i$ are displacement, velocity, and acceleration vectors of current time step $i$, respectively. $S_i$ and $\ddot{x}_i^g$ are respectively the internal force vector and ground motion acceleration at time $i$. In this algorithm, the detailed expressions for the constant coefficients are discounted and can be found extensively in [36]. The algorithm consists of a majority of linear expressions and some nonlinear functions - TangentStiffness(.) and NonlinearForce(.) - that depend on the defined nonlinearity of the system (the first function returns the tangent stiffness and the second function derives nonlinear story forces based on the nonlinear model). In particular, the algorithm can be divided into three blocks: (a) initialization (lines 1 to 5 in Algorithm 1); (b) equilibrium solver (lines 6 to 14 in Algorithm 1); and (c) post-processing (lines 15 to 19 in Algorithm 1). In this organization, blocks (a) and (c) merely

include linear operations. For instance, in lines 17 and 18, the equaions for deriving velocity and acceleration of the future time step are linear.

In addition, block (b) contains a `while` loop which certifies the equilibrium (i.e., Newton–Raphson root finding solution). Intuitively, this `while` loop incrementally adds up values to its estimation of $u_{i+1}$ every time the loop runs. This mechanics resemble the architecture of Residual Networks (ResNet) [34] in which the output of the network is added to the input and fed back to the network repeatedly. Studies have shown that ResNets outperform other architectures in learning differential equations from data [37,38] due to their inherent resemblance to the Euler's method.

### 2.2. DynNet components

DynNet is designed to benefit from two intuitive ideas: (1) following the structure of numerical implicit simulators; and (2) inspired by ResNet structures for nonlinearity learning. The architecture of the network is given in Fig. 2. The input of the network is identical to the Newmark's algorithm. All connections in the network are linear except for the internal connections of the ResNet block. The network initially adjusts the dimension of the input vector via a linear embedding layer. Then, the velocity and acceleration of the structure in addition to the ground motion acceleration of the current time step are fed into a linear layer to produce $R_u^n$ (equivalent to $R$ in Algorithm 1). Then, the internal force, displacement, and $R_u$ are concatenated and passed into the ResNet block. The ResNet block is expanded in Fig. 2 as well. This block is the sole component of the network that is able to learn the nonlinear behavior of the dynamic system. The block is conveniently arranged with stacked fully-connected layers that are connected with leaky rectified linear units (i.e., LeakyReLU activation functions). The output of the fifth fully-connected layer is added to the input of the ResNet block to produce the terminal state of the ResNet block. This terminal state is fed back to the ResNet block for $N$ times ($N$ is a user defined parameter). After $N$ repetitions, the output is linearly mapped to $S_{i+1}$ and $X_{i+1}$. Given $X_{i+1}$, the velocity and acceleration of the next time step are derived by another linear map. Once the prediction of the time step $i+1$ is found, it will be fed back to DynNet for the response prediction of the consecutive time step (e.g., $i+2$).

Note that for inference using DynNet, the only required input is the ground motion accelerations as well as the initial condition of the states (the latter can be assumed a vector of zeros if structure is at rest). The network then predicts the full state at the next time step using the initial condition and repeatedly feeds it back to predict the successive state (therefore, the input and output dimensions of an n-DOF systems are $2+4\times n$ and $4\times n$, respectively. Note that two consecutive values of ground motion are needed for the one-step ahead inference). However, in the training phase the full state responses are required to be available which can be obtained from finite element models or implicitly derived based on a subset of the full state and the governing equations. The concentrated learning ability that is placed in the ResNet block enables easy replacement of the simple MLP network with other nonlinear structures (e.g., CNN or deeper networks). This feature decouples the nonlinearity learning and state transitioning tasks in the network. In other words, for very involved types of nonlinearities, one simply requires to modify the structure of the ResNet block (e.g., add extra layers or increase the number of hidden nodes in each layer). However, in this study we found a five layer MLP sufficiently strong for the test cases. The variable space of the network is highly dependent on the user-defined embedding dimension. In this study, embedding size is set to eight for all cases, yielding $5,320$ trainable variables. The dimension is significantly lower compared to other recently developed networks for the same purpose (e.g., $130,000$ variables in Zhang et al. [32]).

## 3. Accelerating techniques for the training phase

### 3.1. Selecting optimizer

Stochastic first-order methods, including SGD [39] and ADAM [40], are currently standard optimization methods for training neural network problems. These methods have a low per-iteration cost, enjoy optimal complexity, and are easy to implement and applicable to many machine learning tasks. However, these methods have several issues: (i) they are highly sensitive to the choice of hyper-parameters (such as batch size and learning rate); and more importantly (ii) they are not effective for ill-conditioned problems, meaning that for a small change in the inputs, the outputs can change dramatically. The second issue is quite likely when dealing with nonlinear structural systems. For instance, in an elasto-plastic model, there is a bounded relationship between force and displacement within the elastic range. However, the variations of displacements become extremely large when the system experiences larger forces (i.e., forces beyond the elastic limit).

On the other hand, second-order methods by utilizing second-order (i.e., curvature) information can address the aforementioned issues.

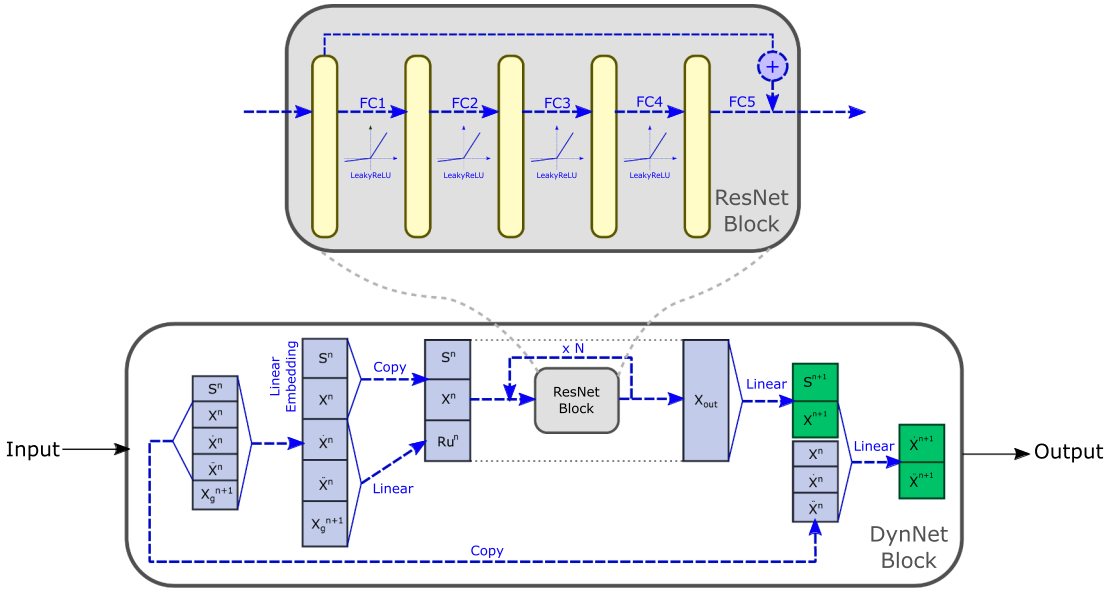


**Fig. 2.** The components of the DynNet recurrent cell.

One class of second-order methods are Hessian-free methods, in which no Hessian is needed to be constructed explicitly, and only Hessian-vector multiplications are needed in order to update the neural network parameters. In our study, we utilize a method in the Hessian-free class which is called *Newton trust-region approach* (TRCG). This is motivated by the results presented in Fig. 3 that illustrate the performance of TRCG and some of the well-known stochastic first-order methods with different choices of hyperparameters. As is clear from the results, the performance of TRCG by utilizing the curvature information is noticeably better than the stochastic first-order methods in terms of loss function value with respect to both iteration and epoch number. Similar behaviour is also observed in [41,42]. In every iteration of TRCG, the following non-convex quadratic sub-problem needs to be solved:

$$p_k \in \arg\min_{p\in\mathbb{R}^d} Q_k(p) = \quad p^T g_k + \frac{1}{2} p^T H_k p \qquad \text{s.t.} \quad \|p\| \leqslant \Delta_k, \tag{1}$$

where $g_k$ is the (stochastic) gradient, $H_k$ is the (stochastic) Hessian, and $\Delta_k$ is the trust-region radius at iteration $k$. The above sub-problem can be approximately and efficiently solved using CG-Steihaug [43] which is summarized in Algorithm 2. The output of Algorithm 2, $p_k$, is the search direction in order to update the neural network parameters. In other words, assume we are at $k^{\text{th}}$ iteration, and the neural network parameters are updated as $\omega_{k+1} := \omega_k + p_k$. More details regarding the trust-region algorithm, the strategy for updating $\Delta_k$, and accepting or rejecting the steps can be found in [43].

**Algorithm 2.** CG-Steihaug [43].

**Input:** $\epsilon$ (termination tolerance), $g_k$ (current gradient).
Set $z_0 = 0$, $r_0 = g_k$, $d_0 = -r_0$
**if** $\|r_0\| < \epsilon$ **then**
  **return** $p_k = z_0 = 0$
**end if**
**for** $j = 0, 1, 2, \ldots$ **do**
  **if** $d_j^T H_k d_j \leqslant 0$ **then**
    Find $\tau \geqslant 0$ such that $p_k = z_j + \tau d_j$ minimizes $m_k(p_k)$ and satisfies $\|p_k\| = \Delta_k$
    **return** $p_k$
  **end if**
  Set $\alpha_j = \dfrac{r_j^T r_j}{d_j^T H_k d_j}$ and $z_{j+1} = z_j + \alpha_j d_j$
  **if** $\|z_{j+1}\| \geqslant \Delta_k$ **then**
    Find $\tau \geqslant 0$ such that $p_k = z_j + \tau d_j$ and satisfies $\|p_k\| = \Delta_k$
    **return** $p_k$
  **end if**
  Set $r_{j+1} = r_j + \alpha_j H_k d_j$
  **if** $\|r_{j+1}\| < \epsilon_k$ **then**
    **return** $p_k = z_{j+1}$
  **end if**
  Set $\beta_{j+1} = \dfrac{r_{j+1}^T r_{j+1}}{r_j^T r_j}$ and $d_{j+1} = -r_{j+1} + \beta_{j+1} d_j$
**end for**

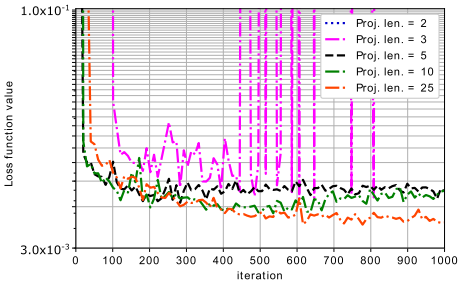


**Fig. 4.** Optimization trends using different projection lengths.

### 3.2. Projection loss

In order to train a recurrent block for the one-step ahead prediction, the simplest approach is to minimize the residual between the predictions and the actual values over a mini-batch in each iteration. However, this approach for training produces unstable networks, which are prone to divergence when predicting a long trajectory of responses given the initial conditions. To address this issue, we introduce and utilize the projection loss that is the basis for the training process in this study.

Projection loss is calculated as the mean squared error of a sequence of responses predicted by DynNet when compared with the corresponding actual responses. To produce the sequence of predicted responses, the only given value is the initial conditions at some randomly selected time steps. This initial condition is then fed into the DynNet and the responses are fed back for $p$ times to predict a trajectory starting from the random initial condition ($p$ is a user-defined projection length). Compared to the conventional loss function, the projection loss can effectively control the instability issue of the trained neural network. Fig. 4 demonstrates the effect of loss functions with different projection lengths on the testing error. In this figure a preliminary training analysis based on a four-DOF model with elastic perfectly plastic material is presented (nonlinear type 1 in Section 4).

As shown in Fig. 4, the length of the projection directly affects the robustness of the optimization. In fact, when the projection length is two, the network's inference diverges (i.e., after multiple steps of recurrence, DynNet outputs explode and it is outside the shown range in the figure). The best results on the testing data are observed when the projection length equals to 25. Note that as the projection length in the loss function increases, the model becomes more optimal for longer trajectory predictions, however, the training time linearly increases as well. In fact, for loss functions with longer projection lengths, the forward pass and backpropagation steps take longer and these computations cannot be distributed over the processing resources (due to the sequential nature of the network inference). In addition, by comparing results from projection length = 5 and projection length = 25, it is observed that the former performs better initially (i.e., in lower iterations) while the latter shows its advantage later on. From this observation, we adopt a sequentially increasing projection length model in this

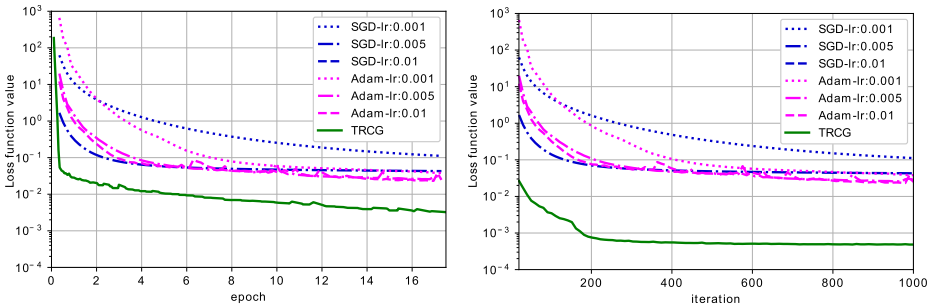


**Fig. 3.** Optimization trends using different optimizers: the numerical values in the legend indicate the learning rates where applicable.

study. In the following section, the models are trained by using loss functions with projection lengths equal to 5, 10, and 25, respectively; for each, the models are trained for a fixed number of iterations.

### 3.3. Importance sampling technique

For learning highly nonlinear systems, samples may be distributed extremely unevenly in different behavioral regions. For instance, elastoplastic systems normally respond linearly to the major portion of a ground motion, regardless of the intensity of the motion. In other words, the system undergoes nonlinear deformations occasionally only when a large impact occurs in the input. As a result, the fraction of the one-step ahead response transitions that are within the elastic region is dramatically larger than the inelastic region. This induces a severe imbalance in the training data distribution, which turns out to be detrimental for model robustness. Importance sampling is a technique for online batch selection that is used to circumvent the problem with unevenly distributed data.

A review of more common batch selection methods are given in Section 7 of [44]. One of the simplest and most effective approaches for adaptive batch selection is rank-based selection [45,44]. In this method, during the training phase, samples of each batch are sorted in descending order based on their function value, and then, their probability of re-selection is updated based on their ranking. The idea was first employed for reinforcement learning using temporal difference (TD) as the reference for sample sorting, and later was adopted for deep learning applications and based on loss function values. In this study, a similar approach is introduced which is inspired by the notion of ranked-based batch selection.

In the implemented hardsampling technique, a hardsampling rate $r$ is defined which is the ratio of samples in each batch that are eventually selected from the hardsamples. The model starts with randomly selected samples in the first iteration. At the end of the iteration, $k$ (is a user-defined hyperparameter) samples with the maximum contributions in the total batch loss are added to a list of hardsamples. In fact, the list of hardsamples is a bag of samples that are not learned well by the model yet. In the next iteration, the new batch samples are selected such that $b_1$ samples are randomly selected from the entire training samples and $b_2$ samples are randomly selected from the the list of hardsamples and $b2 = \lfloor r \times (b_1 + b_2) \rfloor$ ($\lfloor\rfloor$ is a rounding function that sets the value to the largest integer smaller than the actual value of the argument). At the end of the iteration, the list of hardsamples is updated and passed to the next iteration. The process continues accordingly.

To evaluate the effectiveness of the technique, the optimization process is performed with and without hardsampling technique and results are compared in Fig. 5. In this example, the rate of hardsampling $r$ is 50%. The result clearly confirms the advantage of hardsampling technique in fast and better learning of the model. Therefore, in this study this technique is also used in the training process of the models. The approach is adaptive, meaning that the algorithm automatically picks hardsamples throughout the training process. In engineering problems, we may have an *a priori* hypothesis about the hardsamples. For instance, in the elastoplastic models, it is expected that one-step ahead response transitions that go beyond the elastic limit are hardsamples. In the next section, we will confirm that our adaptive hardsampling technique automatically detects these samples with no need for externally imposed constraints.

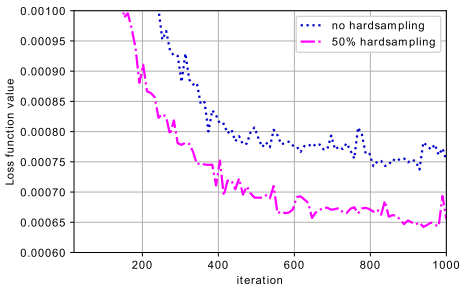


**Fig. 5.** Optimization trends using different hardsampling ratios.

## 4. Numerical case studies

In this section, two case studies are considered to validate the strengths of DynNet in response prediction of different nonlinear (NL) systems. These case studies differ in terms of the type of introduced nonlinearity to the systems. The first case is a four-DOF shear building system with elastic perfectly plastic springs. The second model consists of a four-DOF shear building system equipped with nonlinear ($3^{rd}$ order) elastic stiffeners (schematics of the force displacement behaviors are shown in Fig. 6). It is expected that DynNet performs equally well for different structural types and therefore shear buildings are selected with no loss of generality. The governing equations of motion (EOM) for these two nonlinear systems are shown in Eqs. 2 and 3.

$$m\ddot{x} + c\dot{x} + f(x) = -m\Gamma\ddot{x}_g. \tag{2}$$

$$f_1(x) = \begin{cases} K_i x & x \leqslant \Delta_y, \\ F_y & x > \Delta_y. \end{cases} \qquad f_2(x) = K_i(\alpha_1 x + \alpha_2 x^3). \tag{3}$$

For the numerical simulation, Newmark's method for nonlinear systems is used in MATLAB. For this purpose, 20 strong ground motions are randomly selected from Center for Engineering Strong Motion Database (CESMD) [46]. In addition to that, 10 band limited random white time series are synthesized and added to the the library of the input signals. The earthquake ground motions are scaled using the wavelet algorithm proposed by Hancock et al. [47]. The target matched spectra for twenty earthquake ground motions as well as the mean matched and target spectra are shown in Fig. 7. The algorithm scales the time histories in a way that the response spectrum optimally matches with the target spectrum within the range of $0.2T_1$ to $1.5T_1$ ($T_1$ is the fundamental period of the structure).

For each case study, the scaled earthquake ground motions as well as random time histories are analyzed to predict structure's responses (i.e., displacement, velocity, and acceleration) at all four DOFs. This data include both training and testing datasets. From 30 simulated ground motions, eight ground motions are randomly picked to be used as the training dataset and the rest for testing. Note that since DynNet is heavily constrained by the physics of the problem and enjoys low training variable space, it is expected that the model is easily trainable with small amount of training data and also is desirably generalized for a wide range of testing data.

### 4.1. Case 1: Elastic-perfectly plastic model (NL type 1)

In this section, the results on the first test case - a four-DOF shear building with elastic perfectly plastic stiffness - are presented. The mechanical properties of the structure is presented in Table 1. In this table, $M1 - M4$ and $K1 - K4$ stand for mass and elastic stiffness values of DOF1 to DOF4, respectively. *Fy* shows the stories' yielding force. To consider

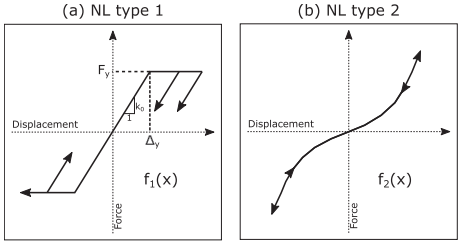


**Fig. 6.** Force–displacement relationships of two nonlinear cases.

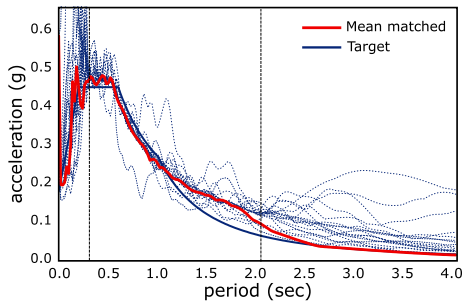


**Fig. 7.** Earthquake response spectra matched with respect to the target spectrum and the mean spectrum.

**Table 1**
Mechanical properties for NL type 1.

| Mechanical props. | Values | Units |
|---|---|---|
| M1 | 0.259 | $kip.s^2/in$ |
| M2/M1 | 1 | - |
| M3/M1 | 0.75 | - |
| M4/M1 | 0.5 | - |
| Fy | 50 | *kips* |
| K1 | 168 | *kips*/*in* |
| K2/K1 | 7/9 | - |
| K3/K1 | 1/3 | - |
| K4/K1 | 1/4 | - |

the robustness of DynNet, three levels of noise are also considered (0%, 5%, and 10% noise levels). For each noise case, the additional noise is imposed proportionally with respect to the maximum amplitude of the signal (e.g., 5% noise is associated with an additional Gaussian white noise with maximum amplitude equal to 5% of the maximum amplitude of the signal). The network is trained to predict the full response at all DOFs including displacement, velocity, and acceleration time histories given the earthquake ground motion.

As concluded in the previous section, the network is trained in a multi-level manner: 1000 iterations with 10-step projection loss, then 1000 iterations with 25-step projection loss, and finally, 1000 iterations with 50-step projection loss. During the training process, batch size was set fixed at 1024 (i.e., 1024 one-step ahead transitions). In total, the network is trained for less than 100 epochs using TRCG optimizer. The learning curve is presented in Fig. 8 (NL type 1). The figure demonstrates that by increasing the length of projection in the custom loss function, a sharp drop in the loss function occurs.

As previously explained, the training phase incorporates the proposed hardsampling technique. To evaluate the physical interpretation of the automatically selected hardsamples, Fig. 9 is presented. In this figure, the entire training dataset (including eight signals) are shown and divided by vertical lines. The signal sections that are labelled as hardsamples are color coded in red. Interestingly, hardsamples are mostly found when a sudden drop (due to a severe nonlinear behavior) has happened. This observation confirms that the algorithm reuses highly nonlinear samples to reinforce its learning ability.

To evaluate the prediction performance of the trained network, the prediction results on one randomly picked testing signal with 5% noise for short and long trajectories are presented in Figs. 10 and 11. The predictions are compared with the reference simulated signals in both time and frequency domains (velocity predictions are neglected for brevity). Note that in all performance plots throughout the paper, the quantities are normalized with reference values and therefore are unitless.. For short trajectories (i.e., five seconds prediction in Fig. 10), the performance is promising. Note that the nonlinear baseline variations are accurately predicted in the displacement time signals. In terms of frequency, the accuracy of the predicted signal is very high. For longer trajectories (i.e., 40 seconds prediction in Fig. 11), the prediction accuracy is as high. The modal peaks in the frequency domain are captured accurately. Notably, all the baseline variations in the displacement time signal are predicted accurately using the trained network. Note that such high accuracies for predicting severely nonlinear responses are unprecedented in the literature.

To further quantify the accuracy of the predictions in all the testing signals, Pearson correlation coefficients (PCC) are calculated between predicted and ground truth signals (40 seconds predictions) and presented in Fig. 12. PCC is a measure to quantify the fitness of predicted trajectories with respect to the ground truth signals [48]. The results for all three noise levels are presented. The histograms demonstrate the distribution of different prediction accuracy values. In general, for all predicted quantities (i.e., displacement, acceleration, and internal force) and all considered noise levels, more than 90% of DynNet's predictions have PCC above 0.8. Particularly, force and acceleration predictions are exceptionally accurate. Intuitively, in those subsets of the full state that oscillate around zero and do not have baseline shifts (i.e., velocity, acceleration, and internal force), the predictions turned out to be more accurate due to their more predictable temporal behavior. Note that in the noisy cases, the likelihood of having very high PCC is inevitably low due to the irreducible noise. Still, DynNet shows a very good performance in response predictions of these highly nonlinear signals with strong noise.

In general, recurrent networks are prone to instability in longer trajectories [49]. Error accumulation due to feeding back the output of the network is reported as the main source of this instability [50]. In this study, by physically constraining the network as well as utilizing the projection loss for training, the model enjoys stability for longer trajectory predictions. Fig. 13 shows prediction errors for different noise levels with respect to the length of the estimated signal (the projection length). For the noiseless case, the mean squared error (MSE) gradually increases as the trajectory lengthens. However, the error is still very low for very long trajectories (i.e., 10,000 one-step ahead predictions

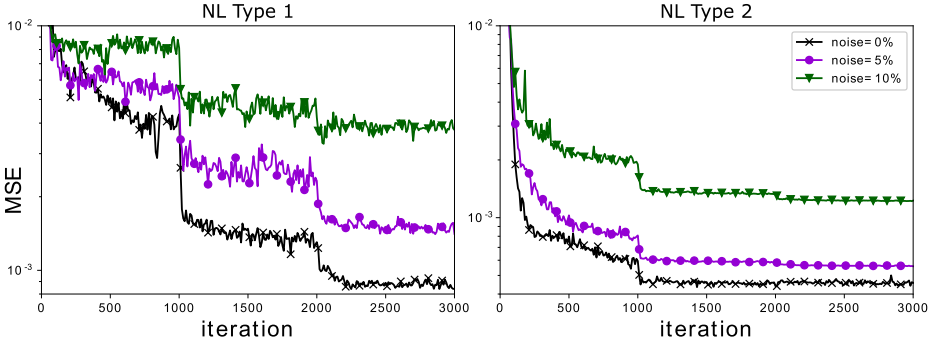


**Fig. 8.** Loss function profile versus iteration: the projection length for loss function calculation changes at iteration 1000 and 2000 (projection length equals to 10, 25, and 50 for each portion). The sudden drops in the loss function values at those transitions show the effectiveness of the proposed training technique.

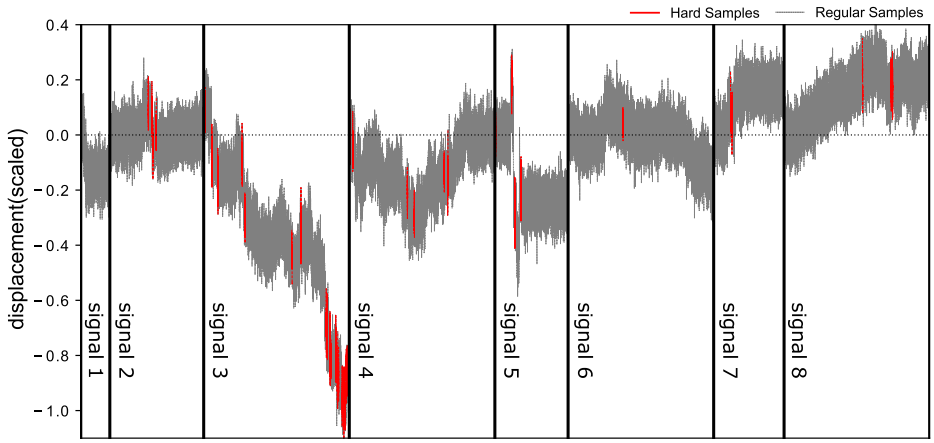


**Fig. 9.** Locations of the hardsamples by using adaptive sampling in the NL type 1: as expected, the majority of hardsamples are located where large residual displacements occur.

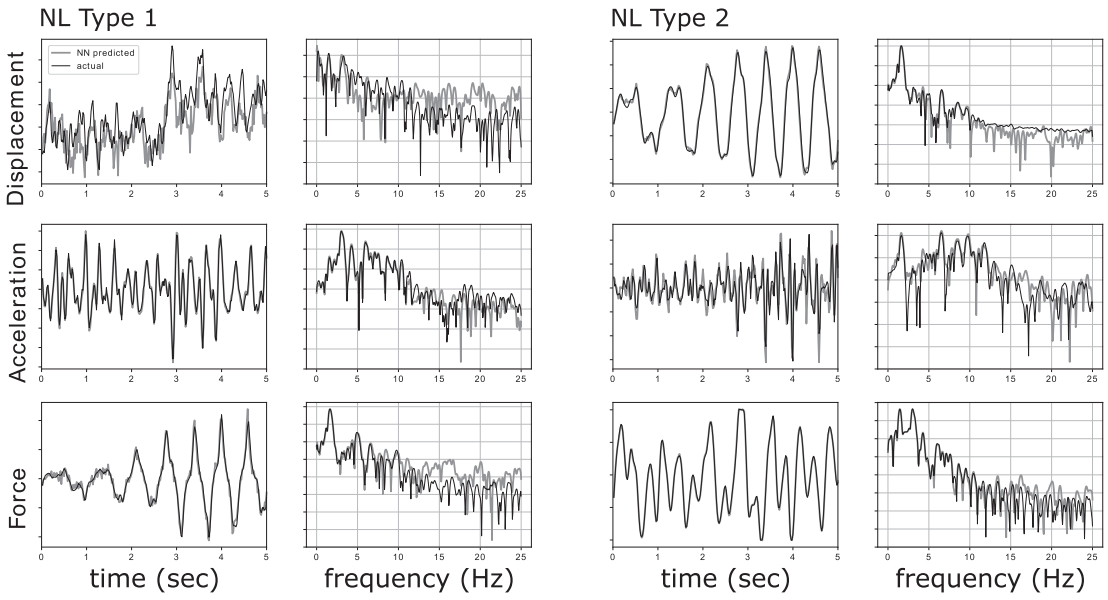


**Fig. 10.** Predicted signals for five seconds with 5% noise. The plots show that the network is very accurate in predicting responses for a short future. A high level of accuracy is visible in both time and frequency representations of the signals with more consistency in the time signals.

equivalent to 200 seconds). Interestingly, for the two noisy cases, except for the lower range of trajectories, the error remains nearly constant for longer trajectories. This implies that: (1) DynNet is quite stable regardless of the trajectory length; and (2) noisier data tends to discount the increasing error issue for longer trajectories.

Finally, to verify the strength of DynNet in identifying the nonlinear behavior, hysteresis diagrams for a randomly picked signal and different noise levels are shown in Fig. 14. The DynNet estimated signals could very accurately capture the linear tangent of the spring force. In addition, the transition to nonlinear region is learned very accurately (normalized force values are exactly bounded within −1 to 1). The same level of accuracy is noticeable in all noise cases.

To further investigate the scalability and generalization of the trained DynNet, the nonlinear responses of the structure subjected to different magnitudes of a selected earthquake ground motion are inferred and compared with the numerical solutions. Four levels of magnitude are considered in this analysis: 0.50x, 0.85x, 1.00x, and 1.20x (compared to the normalized ground motion). The results are presented in Fig. 15. In this plot, dotted lines show exact simulation results while solid lines represent DynNet predictions. Results of internal forces and displacements for the $1^{st}$ DOF are shown for brevity. Internal forces are very accurately predicted in all four levels of magnitude of the ground motion. The accuracy is lower in the displacement predictions, however, the overall trends and amplitudes are carefully captured by DynNet. Note that the selected ground motion contains a strong shockwave at ~$380^{th}$ time step which causes severe nonlinear response and a baseline shift (residual deformation) in the displacement predictions. The model, however, is still successful in following the exact variations of the building responses.

### 4.2. Case 2: Nonlinear elastic model (NL type 2)

In the second case study, a four-DOF structure with nonlinear elastic springs is studied. For the nonlinear springs, a $3^{rd}$ order polynomial behavior is introduced that models a hardening after initial pseudo-linear phase (see Fig. 6). Due to the elasticity of the model, no residual displacements are expected here. Mechanical properties of the building are presented in Table 2. In this table, M's and K's are defined as explained before. $\alpha_1$ and $\alpha_2$ are coefficients of the $3^{rd}$ order restoring force equation (Eq. 3). The training process is identical to the previous case study. DynNet requires no pre-processing or special

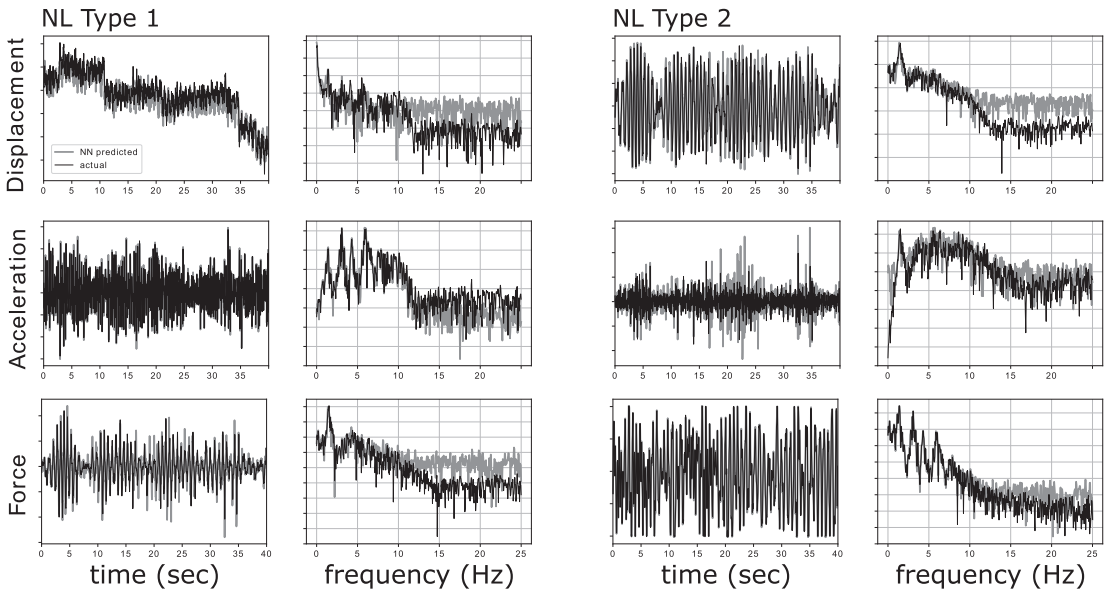


**Fig. 11.** Predicted signals for 40 seconds with 5% noise. The plots show that the network is still accurate in predicting responses for a longer time. A high level of accuracy is visible in both time and frequency representations of the signals with more consistency in the time signals. Notice that the displacement signal for the NL type 1 is strongly nonlinear. However, the network successfully estimates it.

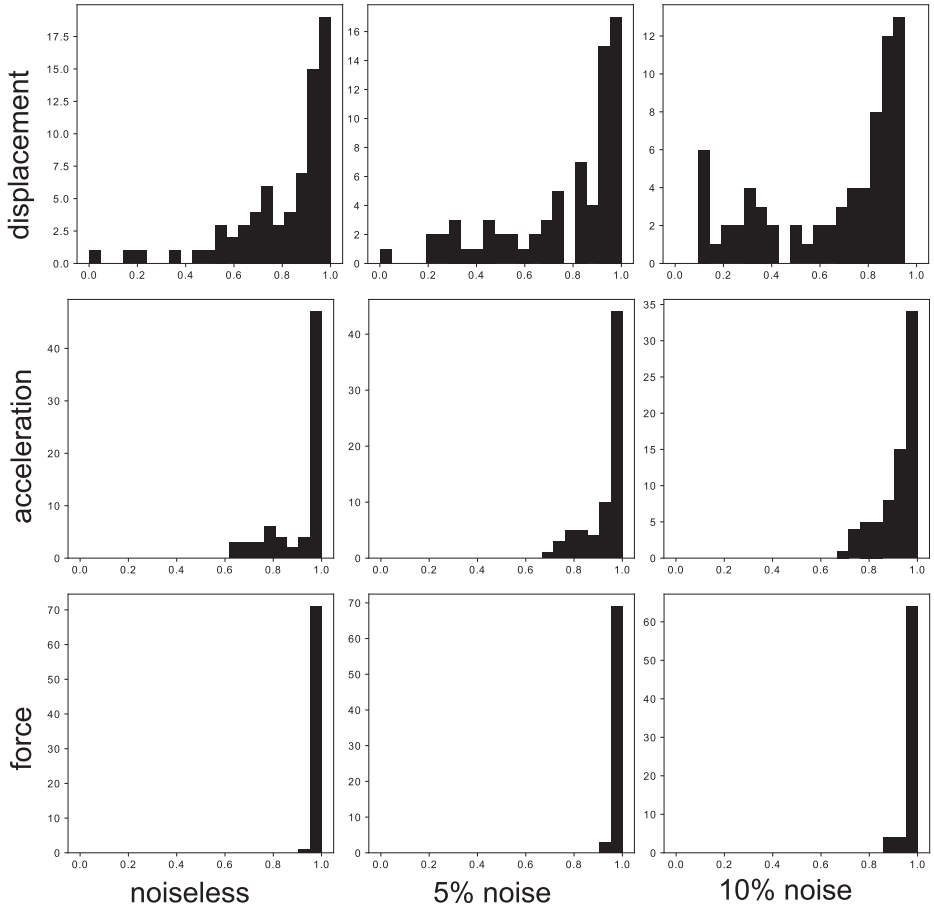


**Fig. 12.** Pearson correlation coefficient histograms for the predicted responses - NL type 1.

accommodation for different nonlinear models. The model is trained for the same number of iterations and epochs as the previous test case. Loss function variation versus the number of iterations is shown in Fig. 8. Again, sudden drops in the loss values are observed when the projection length of the custom loss increases.

The nonlinear response predictions for a randomly picked ground motion from testing data are presented in Figs. 10 and 11 (short and long trajectories, respectively). As before, DynNet shows a promising

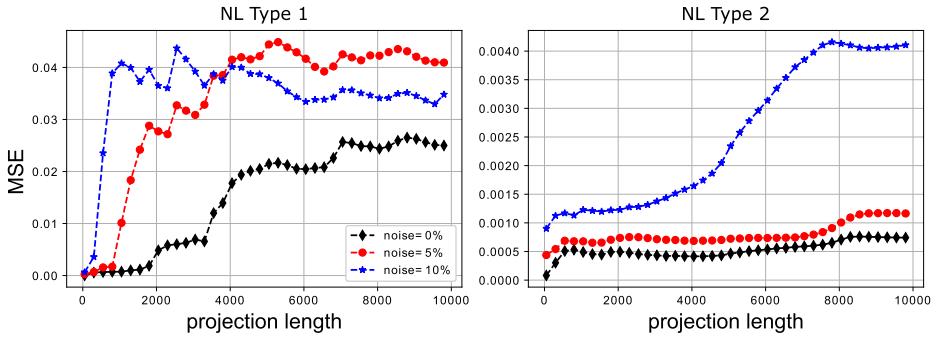


**Fig. 13.** MSE values of the predicted signals vs. the length of projection. As expected, the error increases as the projection length becomes longer. However, in most cases after a gain in the error at the beginning, the error flattens for longer projections. Notice that as expected, noisier signals have higher MSE errors.

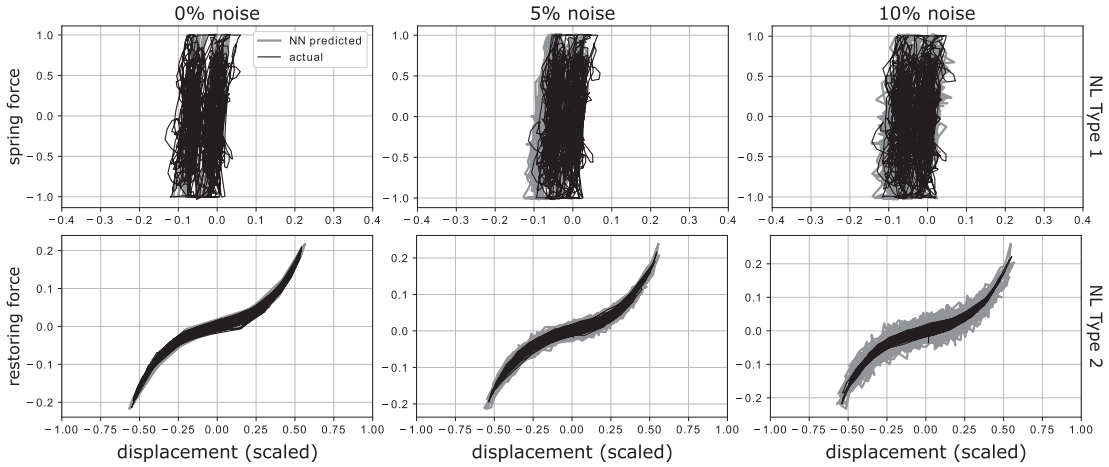


**Fig. 14.** Hysteresis diagrams in two NL cases at the first floor and for different noise levels. Both sets of results confirm the promising performance of the network in learning different nonlinear behaviors.

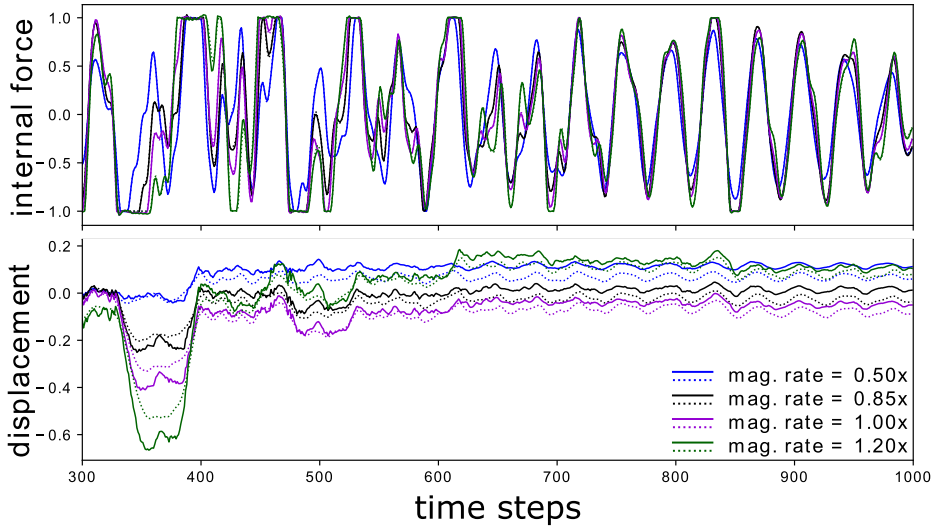


**Fig. 15.** Response predictions for a ground motion with different magnitudes (NL type 1). This figure demonstrates the generalization strength of the trained NN model. Note that dashed lines show the actual responses from the numerical simulation. Despite strong nonlinear behavior, all four different magnitudes are predicted very accurately.

performance in nonlinear response predictions, both in time and frequency domains, regardless of the length of the trajectory. To evaluate the performance of the trained neural network on the entire testing data, PCC coefficients are calculated and the histograms are shown in Fig. 16. Note that similar to the previous test case, three levels of measurement noise are considered for both training and evaluation phases of the network.

In Fig. 16, the general note is that the number of very high accuracy predictions (i.e., with PCC above 0.8) is not as high as the previous case, especially when measurement noise is introduced. However, for

**Table 2**
Mechanical properties for NL type 2.

| Mechanical props. | Values | Units |
|---|---|---|
| M1 | 0.340 | $kip.s^2/in$ |
| M2/M1 | 0.8 | - |
| M3/M1 | 0.75 | - |
| M4/M1 | 0.6 | - |
| K1 | 100 | $kips/in$ |
| K2/K1 | 3/4 | - |
| K3/K1 | 1/2 | - |
| K4/K1 | 1/4 | - |
| $\alpha_1$ | 1 | - |
| $\alpha_2$ | 10 | $in^2$ |

noiseless and 5% additive noise cases, the results still show high accuracy. Histograms of the displacement and internal force predictions show a unimodel distribution with the statistical mode at PCC $\in [0.95, 1.0]$. In terms of the prediction stability for longer trajectories, MSE errors with respect to the length of the prediction trajectory is presented in Fig. 13. Again, as observed in the PCC histograms, two lower noise cases show a steady and downbeat trend of the MSE loss progression as the trajectory length increases while the 10% noise case is not as stable. Notice that the values of the MSE errors generally are significantly lower in the NL type 2 (nonlinear elastic case) compared to the NL type 1 (elastic-perfectly plastic) while histograms show higher accuracy for prediction of the latter model. This observation is explained by the inelastic behavior of the NL type 1 model which can cause baseline shifts (i.e., residual deformations). We showed that DynNet is successful in capturing baseline variations, even though a small discrepancy causes much larger MSE errors for these response predictions. The baseline variations are not expected in the elastic model.

Finally, in order to validate the ability of the neural network to predict nonlinear elastic behavior of the spring forces, hysteresis diagrams are plotted in Fig. 14. The restoring force here includes both the elastic spring force and the damping force (i.e., $c\dot{x} + f(x)$ in Eq. 2). According to these plots, DynNet predictions very accurately match with the simulation results. The $3^{rd}$ order behavior of the spring as well as the small energy dissipation area caused by the damper force is identified and correctly predicted. In higher noise levels, the prediction shows higher fluctuations around the exact plots which can be simply explained by the high level of noise.

## 5. Conclusion

In this study, we proposed a data-driven approach for comprehensive prediction of nonlinear dynamic responses of multi degrees of freedom (DOF) systems using Recurrent Neural Networks. In particular, inspired by common implicit dynamic analysis algorithms, DynNet block is designed as a one-step ahead response predictor. By repeatedly inferring the block, long response trajectories are predicted. Compared to the most advanced data-driven methods, DynNet has significantly smaller variable space, resulting less computational effort per iteration. Due to physics-based constraints of the proposed architecture, the network requires more advanced optimizers for a smooth and efficient learning process. With this regard, trust-region approach using CG-Steihaug (TRCG) algorithm was implemented. In addition, for more efficient learning, a simple importance sampling technique as well as a trajectory loss function was developed and implemented which resulted in faster learning of the severely nonlinear transitions.

For verification, DynNet was tested in two nonlinear case studies: a

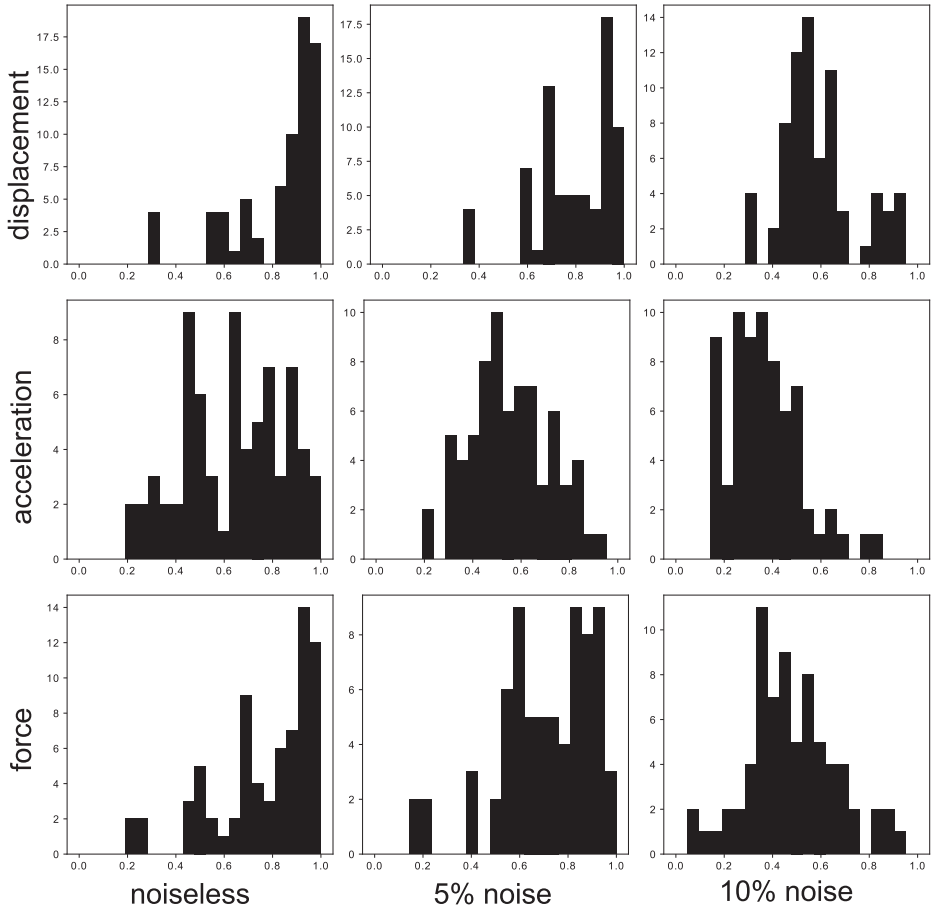


**Fig. 16.** Pearson correlation coefficient histograms for the predicted responses - NL type 2.

four-DOF shear building (1) with elastic perfectly plastic stiffness, and (2) with nonlinear elastic ($3^{rd}$ order) stiffness. For each test case, three levels of measurement noise were included to evaluate the noise propagation characteristics of the proposed network. The networks were trained using less than 30% of the available data and evaluated using the remaining 70%. In both test cases, we showed that the network quite successfully was able to predict a complete set of nonlinear responses including displacement, velocity, acceleration, and internal force time histories at all DOFs given the applied ground motions only. The stability of the predictions for longer trajectories was analyzed and concluded that for the majority of cases, DynNet holds the error level stably as the trajectory length grows. Due to the recurrent nature of this approach, the inference computational time increases linearly with the length of the ground motion. In addition, using hysteresis diagrams, we showed that the performance of DynNet in capturing nonlinear behaviors of the systems is promising.

Data-driven function approximators are extremely popular in science and technology, however, in engineering applications due to the availability of accurate governing equations and numerical solutions, fully black-box function approximators are less accepted. This study tries to bridge the gap between black-box models and available exact solutions to create a fast learner function approximator. It is believed that DynNet creates a great potential for faster and scalable regional sustainability and health monitoring analyses.

## CRediT authorship contribution statement

**Soheil Sadeghi Eshkevari:** Conceptualization, Methodology, Software, Validation, Formal analysis. **Martin Takáč:** Conceptualization, Software, Resources, Funding acquisition. **Shamim N. Pakzad:** Supervision, Funding acquisition. **Majid Jahani:** Methodology, Validation.

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

Research funding is partially provided by a grant from the U.S. Department of Transportation's University Transportation Centers Program, and National Science Foundation through Grants CMMI-1351537, CCF-1618717, and CCF-1740796, and by a grant from the Commonwealth of Pennsylvania, Department of Community and Economic Development, through the Pennsylvania Infrastructure Technology Alliance (PITA).

## References

[1] Yuen Ka-Veng, Beck James L, Katafygiotis Lambros S. Efficient model updating and health monitoring methodology using incomplete modal data without mode matching. Struct Control Health Monitor: Off J Int Assoc Struct Control Monitor Eur Assoc Control Struct 2006;13(1):91–107.
[2] Ching Jianye, Beck James L. New bayesian model updating algorithm applied to a structural health monitoring benchmark. Struct Health Monitor 2004;3(4):313–32.
[3] Johnson Erik A, Lam Heung-Fai, Katafygiotis Lambros S, Beck James L. Phase i iasc-asce structural health monitoring benchmark problem using simulated data. J Eng Mech 2004;130(1):3–15.
[4] Golnaz Shahidi S, Pakzad Shamim N. Generalized response surface model updating using time domain data. J Struct Eng 2014;140(8):A4014001.
[5] Christenson Richard, Lin Yi Zhong, Emmons Andrew, Bass Brent. Large-scale experimental verification of semiactive control through real-time hybrid simulation. J Struct Eng 2008;134(4):522–34.
[6] Ahmadizadeh M, Mosqueda G, Reinhorn AM. Compensation of actuator delay and dynamics for real-time hybrid structural simulation. Earthq Eng Struct Dyn 2008; 37(1):21–42.
[7] Al-Subaihawi Safwan, Kolay Chinmoy, Marullo Thomas, Ricles James M, Quiel Spencer E. Assessment of wind-induced vibration mitigation in a tall building with damped outriggers using real-time hybrid simulations. Eng Struct 2020;205: 110044.
[8] Deng Li, Yu Dong, et al. Deep learning: methods and applications. Found Trends Signal Process 2014;7(3–4):197–387.
[9] Cybenko George. Approximation by superpositions of a sigmoidal function. Mathe Control Signals Syst 1989;2(4):303–14.
[10] Leshno Moshe, Lin Vladimir Ya, Pinkus Allan, Schocken Shimon. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. Neural Networks 1993;6(6):861–7.
[11] Towell Geoffrey G, Shavlik Jude W, Noordewier Michiel O. Refinement of approximate domain theories by knowledge-based neural networks. In: Proceedings of the eighth National conference on Artificial intelligence, Boston, MA, vol. 861866; 1990.
[12] Worden Keith, Manson Graeme. The application of machine learning to structural health monitoring. Philosoph Trans Roy Soc A: Mathe Phys Eng Sci 2007;365 (1851):515–37.
[13] Ying Yujie, Garrett Jr James H, Oppenheim Irving J, Soibelman Lucio, Harley Joel B. Jun Shi, and Yuanwei Jin. Toward data-driven structural health monitoring: application of machine learning and signal processing to damage detection. J Comput Civil Eng 2013;27(6):667–80.
[14] Yi Ting-Hua, Li Hong-Nan, Sun Hong-Min. Multi-stage structural damage diagnosis method based on. Smart Struct Syst 2013;12(3_4):345–61.
[15] Gui Guoqing, Pan Hong, Lin Zhibin, Li Yonghua, Yuan Zhijun. Data-driven support vector machine with optimization techniques for structural health monitoring and damage detection. KSCE J Civil Eng 2017;21(2):523–34.
[16] Abdeljaber Osama, Avci Onur, Kiranyaz Serkan, Gabbouj Moncef, Inman Daniel J. Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks. J Sound Vib 2017;388:154–70.
[17] Gulgec Nur Sila, Takáč Martin, Pakzad Shamim N. Convolutional neural network approach for robust structural damage detection and localization. J Comput Civil Eng 2019;33(3):04019005.
[18] Brunton Steven L, Proctor Joshua L, Kutz J Nathan. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. Proc National Acad Sci 2016;113(15):3932–7.
[19] Raissi Maziar, Karniadakis George Em. Hidden physics models: Machine learning of nonlinear partial differential equations. J Comput Phys 2018;357:125–41.
[20] Eshkevari Soheil Sadeghi, Pakzad Shamim N, Takáč Martin, Matarazzo Thomas J. Modal identification of bridges using mobile sensors with sparse vibration data. J Eng Mech 2020;146(4):04020011.
[21] Song Junho, Ok Seung-Yong. Multi-scale system reliability analysis of lifeline networks under earthquake hazards. Earthq Eng Struct Dyn 2010;39(3):259–79.
[22] Mahsuli M, Haukaas T. Seismic risk analysis with reliability methods, Part I: Models. Struct Saf 2013;42:54–62.
[23] Roohi Milad, Hernandez Eric M, Rosowsky David. Nonlinear seismic response reconstruction and performance assessment of instrumented wood-frame buildings—validation using neeswood capstone full-scale tests. Struct Control Health Monitor 2019;26(9):e2373.
[24] Mattson Steven G, Pandit Sudhakar M. Statistical moments of autoregressive model residuals for damage localisation. Mech Syst Signal Process 2006;20(3):627–45.
[25] Bornn Luke, Farrar Charles R, Park Gyuhae, Farinholt Kevin. Structural health monitoring with autoregressive support vector machines. J Vib Acoust 2009;131 (2).
[26] Lightbody Gordon, Irwin George W. Multi-layer perceptron based modelling of nonlinear systems. Fuzzy Sets Syst 1996;79(1):93–112.
[27] Lagaros Nikos D, Papadrakakis Manolis. Neural network based prediction schemes of the non-linear seismic response of 3d buildings. Adv Eng Softw 2012;44(1): 92–115.
[28] Sainath Tara N, Kingsbury Brian, Saon George, Soltau Hagen, Mohamed Abdel-rahman, Dahl George, Ramabhadran Bhuvana. Deep convolutional neural networks for large-scale speech tasks. Neural Networks 2015;64:39–48.
[29] Sun Shan-Bin, He Yuan-Yuan, Zhou Si-Da, Yue Zhen-Jiang. A data-driven response virtual sensor technique with partial vibration measurements using convolutional neural network. Sensors 2017;17(12):2888.
[30] Wu Rih-Teng, Jahanshahi Mohammad R. Deep convolutional neural network for structural dynamic response estimation and system identification. J Eng Mech 2019;145(1):04018125.
[31] Zhang Ruiyang, Liu Yang, Sun Hao. Physics-guided convolutional neural network (phycnn) for data-driven seismic response modeling. arXiv preprint arXiv: 1909.08118, 2019a.
[32] Zhang Ruiyang, Chen Zhao, Chen Su, Zheng Jingwei, Büyüköztürk Oral, Sun Hao. Deep long short-term memory networks for nonlinear structural seismic response prediction. Comput Struct 2019;220:55–68.
[33] Zhang Ruiyang, Liu Yang, Sun Hao. Physics-informed multi-lstm networks for metamodeling of nonlinear structures. arXiv preprint arXiv:2002.10253, 2020.
[34] He Kaiming, Zhang Xiangyu, Ren Shaoqing, Sun Jian. Deep residual learning for image recognition. corr abs/1512.03385 (2015), 2015.
[35] Riddell Rafael, Newmark Nathan Mortimore. Statistical analysis of the response of nonlinear systems subjected to earthquakes. Technical report, University of Illinois Engineering Experiment Station. College of..., 1979.
[36] Chopra Anil K, et al. Dynamics of structures. Upper Saddle River, NJ: Pearson Education; 2012.
[37] Lu Yiping, Zhong Aoxiao, Li Quanzheng, Dong Bin. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. arXiv preprint arXiv:1710.10121, 2017.
[38] Qi Chen Tian, Rubanova Yulia, Bettencourt Jesse, Duvenaud David K. Neural ordinary differential equations. In: Advances in neural information processing systems, 2018, p. 6571–83.

[39] Robbins Herbert, Monro Sutton. A stochastic approximation method. The Annals of Mathematical Statistics, 1951, p. 400–407.
[40] Kingma Diederik P, Ba Jimmy. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
[41] Berahas Albert S, Jahani Majid, Takáč Martin. Quasi-newton methods for deep learning: Forget the past, just sample. arXiv preprint arXiv:1901.09997, 2019.
[42] Peng Xu, Roosta Fred, Mahoney Michael W. Second-order optimization for non-convex machine learning: An empirical study. In: Proceedings of the 2020 SIAM International Conference on Data Mining. SIAM; 2020. p. 199–207.
[43] Nocedal Jorge, Wright Stephen J. Numerical Optimization. Springer Series in Operations Research. 2nd ed. Springer; 2006.
[44] Loshchilov Ilya, Hutter Frank. Online batch selection for faster training of neural networks. arXiv preprint arXiv:1511.06343, 2015.
[45] Schaul Tom, Quan John, Antonoglou Ioannis, Silver David. Prioritized experience replay. arXiv preprint arXiv:1511.05952, 2015.
[46] Haddadi Hamid, Shakal A, Stephens C, Savage W, Huang M, Leith W, Parrish J. Center for engineering strong-motion data (cesmd), 2008.
[47] Hancock Jonathan, Watson-Lamprey Jennie, Abrahamson Norman A, Bommer Julian J, Markatis Alexandros, McCOY EMMA, Mendis Rishmila. An improved method of matching response spectra of recorded earthquake ground motion using wavelets. J Earthq Eng 2006;10(spec01):67–89.
[48] Weisstein Eric W. Correlation coefficient, 2006.
[49] Salehinejad Hojjat, Sankar Sharan, Barfett Joseph, Colak Errol, Valaee Shahrokh. Recent advances in recurrent neural networks. arXiv preprint arXiv:1801.01078, 2017.
[50] Holden Daniel, Komura Taku, Saito Jun. Phase-functioned neural networks for character control. ACM Trans Graphics (TOG) 2017;36(4):1–13.