Mechanical Systems and Signal Processing 164 (2022) 108201

Contents lists available at ScienceDirect

Mechanical Systems and Signal Processing

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

Ensemble of recurrent neural networks with long short-term memory cells for high-rate structural health monitoring

Vahid Barzegar [a,*], Simon Laflamme [a,c], Chao Hu [b,c], Jacob Dodson [d]

[a] Department of Civil, Construction, and Environmental Engineering, Iowa State University, Ames, IA, USA
[b] Department of Mechanical Engineering, Iowa State University, Ames, IA, USA
[c] Department of Electrical and Computer Engineering, Iowa State University, Ames, IA, USA
[d] Air Force Research Laboratory, Munitions Directorate, Fuzes Branch, Eglin Air Force Base, FL, USA

ARTICLE INFO

Communicated by S. Fassois

Keywords:
Deep learning
Long short-term memory
Recurrent neural network
High-rate systems
Structural health monitoring
Transfer learning
Nonlinear time series

ABSTRACT

The deployment of systems experiencing high-rate dynamic events, such as hypersonic vehicles, advanced weaponry, and active blast mitigation systems, require high-rate structural health monitoring (HRSHM) capabilities in the sub-millisecond realm to ensure continuous operations and safety. However, the development of high-rate feedback systems is a complex task because these dynamic systems are uniquely characterized by (1) large uncertainties in their external loads, (2) high levels of non-stationarity and heavy disturbance, and (3) unmodeled dynamics from changes in system configuration. In this paper, we present a deep learning algorithm specifically engineered for HRSHM applications. It consists of an ensemble of recurrent neural networks (RNNs) constructed with long short-term memory (LSTM) cells with transfer learning capabilities to cope with the highly limited availability of training data as it is typical for high-rate systems. The use of an ensemble of RNNs empowers multi-rate sampling capability to capture multi-temporal features of the time series, thus enabling modeling of non-stationarities. Also, because the RNNs use short-sequence LSTMs and are arranged in parallel, the computation time is substantially reduced to the sub-millisecond range. The performance of the algorithm is investigated on experimental high-rate dynamic data produced from accelerated drop tower tests and benchmarked against results from a prior investigation using a purely on-the-edge algorithm termed variable input space observer (VIO) and its hybrid architecture (hybrid VIO) that incorporated physical knowledge and is considered as an upper bound on the algorithm performance. Numerical results show that the ensemble of RNNs, composed of five RNNs, significantly outperformed the VIO, with performance close to that of the hybrid VIO when it comes to the estimation error metrics, with an average computation time of 25 μs per prediction step. Yet, the ensemble of RNNs exhibits chattering for low-amplitude excitations, likely attributable to the behavior of RNNs sampling at small time delays. An examination of the RNN weights and hidden states confirms that the algorithm can capture multi-temporal features, and an investigation with respect to noise in the training dataset showed that the algorithm is robust for signal-to-noise ratios up to 20 dB.

1. Introduction

High-rate dynamic systems are defined as systems experiencing high amplitude events, often higher than 100 g, over durations less than 100 ms. Their dynamics are uniquely characterized by (1) large uncertainties in their external loads, (2) high levels of

* Corresponding author.
E-mail address: barzegar@iastate.edu (V. Barzegar).

https://doi.org/10.1016/j.ymssp.2021.108201
Received 16 September 2020; Received in revised form 8 April 2021; Accepted 29 June 2021
Available online 19 July 2021

non-stationarity and heavy disturbance, and (3) unmodeled dynamics from changes in system configuration [1]. Examples of high-rate systems include hypersonic vehicles, advanced weaponry, and blast mitigation systems. The integration of real-time feedback capabilities for these systems is challenging, yet has the potential to empower field deployment by ensuring continuous operations and safety. Successful closed-loop applications will require the development of high-rate state estimation and system identification capabilities, also known as high-rate structural health monitoring (HRSHM), with an objective real-time performance under 1 ms [2].

Targeting HRSHM applications, Joyce et al. [3] developed a sliding mode observer-based algorithm for real-time identification of fundamental frequencies. Downey et al. [4] proposed fast state estimation by matching the measured frequency response to pre-simulated physical representations with experimentally demonstrated sub-millisecond capabilities. Yan et al. [5] introduced a time-based hybrid approach based on model reference adaptive systems with numerically demonstrated sub-millisecond capabilities. Hong et al. [6] proposed a path to on-the-edge data-based time series estimation through an adaptive wavelet network observer. While the algorithm had a computation time unapplicable to HRSHM, it showed great convergence capabilities without necessitating any pre-training, in particular for non-stationary time series.

The lack of available data for pre-training is an important obstacle in developing HRSHM algorithms, where it is assumed that input–output datasets are highly limited or unavailable. This is attributable to both the large costs associated with running high-rate experiments, and non-repeatability of events inherent to the systems' strong non-stationarities. When available input–output data only represents a limited dynamic behavior of the system, a solution is to use transfer learning [7]. Developed to specifically address the problem of limited labeled data for training a classifier, transfer learning has been successfully applied in image classification and computer vision [8–11]. Transfer learning has also been applied to one-dimensional time series datasets for various tasks, including demand [12–14] and supply [14,15] prediction, and language modeling [16–18].

In this paper, we propose a purely data-based algorithm inspired by work from Hong et al. [6], but with sub-millisecond capabilities. The algorithm is based on an ensemble of recurrent neural networks (RNNs) constructed with long short-term memory (LSTM) cells trained using transfer learning. Its novelty resides in its unique architecture engineered to cope with HRSHM-specific challenges, including sub-millisecond computation capabilities, fast convergence under very limited training, and estimation capabilities in a highly non-stationary environment. There has been extensive research on the use of neural networks for SHM applications [19–21]. Among the diverse neuro-representations, RNNs show particular promise at modeling time series due to their sequential organization and lend themselves to leveraging delay vectors as inputs [22]. Nevertheless, a known challenge with RNNs is the vanishing gradient, whereas long-term dependencies in time series are difficult to capture. LSTM cells have been proposed as a solution [23,24], where they use nonlinear gating functions to control the flow of information to only use relevant input [25].

RNNs with LSTM cells have been successfully implemented in nonlinear time-series modeling [15,22,26,27]. Zhu and Laptev [28] presented an encoder–decoder RNN for prediction of trip demands and quantified prediction uncertainties. Shih et al. [29] incorporated temporal pattern filters in the frequency domain with RNNs to enhance multi-step-ahead prediction performance. Che et al. [30] proposed an RNN-based method for datasets with randomly missing values and showed significant improvement over the state-of-the-art imputation methods. Other variations of RNNs, such as echo state networks [31–33] and gated recurrent units [34–36] have been applied to the time-series prediction problems.

While RNNs have been well studied, their applications to highly non-stationary time series is still challenging. An important aspect of the wavelet network developed in [6] was the utilization of an adaptive input space, where time series inputs were sampled at different rates to account for changing local dynamics. The construction of the adaptive input space was based on Takens' embedding theorem [37], and on the assumption that a non-stationary system could be sub-divided in small stationary sections [38]. Under each sequential sampling event, the algorithm looked at the last few time series entries and estimated the required embedding dimension and time delay of the delay vector. This idea of multi-rate sampling, also used in other work to extract temporally localized time series features [39,40], to cope with the non-stationarity of high-rate systems is integrated into our proposed algorithm by leveraging multiple RNNs arranged in parallel and sampling at different rates, also known as ensemble of RNNs, with their outputs combined through an attention layer and mapped to the estimation through a linear neuron. Using ensemble of RNNs for HRSHM is advantageous by enabling the extraction of multi-temporal dependencies. In addition, the use of LSTM cells empowers short-sequence architectures due to the shorter delay vectors used by individual RNNs [22,26,28], which combined with a parallel RNN architecture, results in substantially enhanced computation speed.

The rest of the paper is organized as follows. Section 2 presents the proposed RNN architecture along with its transfer learning and ensemble processes. Section 3 described the methodology used in validating the algorithm, including a description of the experimental tests used in collecting realistic data, pre-training process, and performance evaluation metrics. Section 4 presents and discusses results from the numerical investigation. Section 5 concludes the paper.

2. RNN architecture for HRSHM

In this section, the proposed algorithm for high-rate prediction is presented. It includes the background on RNNs with LSTM cells, the use of transfer learning for training in conducting predictions with limited knowledge, and the use of an ensemble of RNNs to cope with non-stationarity.

2.1. Recurrent neural networks with LSTM cells

Given a temporal sequence of measurements from a dynamic system, $\mathbf{x}_k = \{x_1, x_2, \ldots, x_k\}$, taken at a discrete time step k, an RNN maps x_k to x_{k+1} through the sequential modification of internal hidden states $\mathbf{h}$, as illustrated in Fig. 1(a) for an unfolded

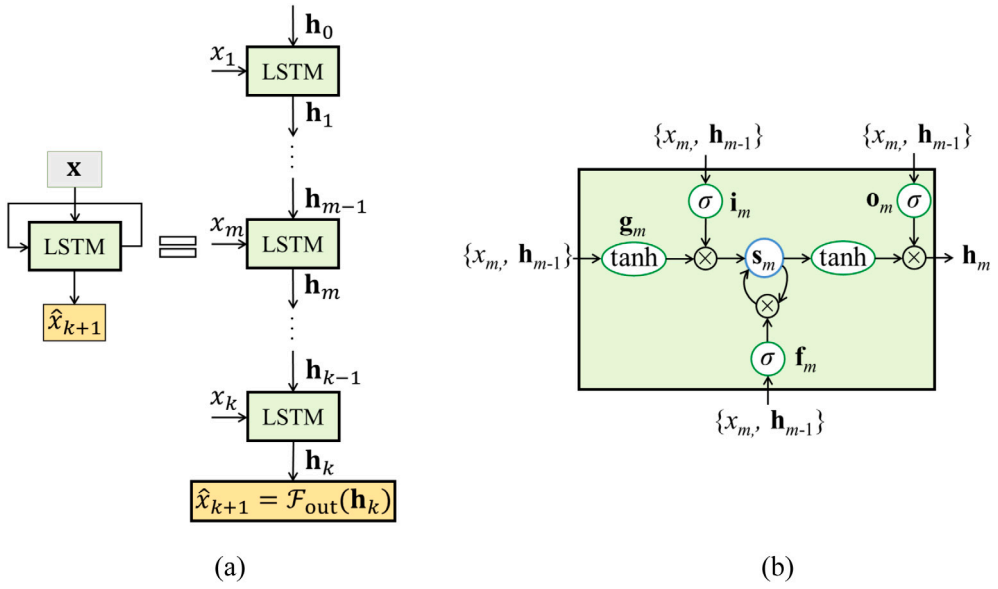

Fig. 1. (a) Unfolded basic RNN; (b) LSTM cell architecture.

RNN layer with LSTM cells. Each LSTM cell receives the previous hidden state and current measurement as inputs, and outputs the current hidden state to be used as the input for the next LSTM cell. The role of the hidden states are to model and remember the dependence among the input sequence of measurements. Mathematically, for each new measurement x_k, the LSTM cell updates the hidden state through a nonlinear function r

$$\mathbf{h}_k = r(\mathbf{h}_{k-1}, x_k) \tag{1}$$

with $\mathbf{h}_0$ commonly initialized at zero [41]. The output of the LSTM cell $\mathbf{h}_k$ is fed into a feed-forward network $\mathcal{F}_{out}(\mathbf{h}_k)$ to compute the estimation $\hat{x}_{k+1}$.

$$\hat{x}_{k+1} = \mathcal{F}_{out}(\mathbf{h}_k) \tag{2}$$

LSTM cells are used to augment memory capabilities, particularly useful in modeling sequences of data. A single LSTM cell is illustrated in Fig. 1(b). It is composed of three internal gates (Eqs. (3)–(6)), i.e., an input gate $\mathbf{i}$, a forget gate $\mathbf{f}$, and an output gate $\mathbf{o}$. These gates act as smooth switches to modulate the internal flow of information. The inputs to the cell are first passed through gate $\mathbf{g}$ and normalized within range $(-1, 1)$ (Eq. (3)). All the other gates output values within range $(0, 1)$ in order to reflect the relevance of the information, where the outputs closer to zero are less relevant than the outputs closer to 1. The input gate decides whether the information should be written into the internal memory $\mathbf{s}$ ($\mathbf{i}_k \odot \mathbf{g}_k$ in Eq. (7)), the forget gate decides when to reset the internal memory ($\mathbf{f}_k \odot \mathbf{s}_{k-1}$ in Eq. (7)), and the output gate takes the required information from the internal memory, discards irrelevant data, and produces the output (Eq. (8)).

The LSTM cell is defined by the following equations [41]

$$\mathbf{g}_k = \tanh(\mathbf{W}_x^g x_k + \mathbf{W}_h^g \mathbf{h}_{k-1} + \mathbf{b}_g) \tag{3}$$

$$\mathbf{i}_k = \sigma(\mathbf{W}_x^i x_k + \mathbf{W}_h^i \mathbf{h}_{k-1} + \mathbf{b}_i) \tag{4}$$

$$\mathbf{f}_k = \sigma(\mathbf{W}_x^f x_k + \mathbf{W}_h^f \mathbf{h}_{k-1} + \mathbf{b}_f) \tag{5}$$

$$\mathbf{o}_k = \sigma(\mathbf{W}_x^o x_k + \mathbf{W}_h^o \mathbf{h}_{k-1} + \mathbf{b}_o) \tag{6}$$

$$\mathbf{s}_k = \mathbf{f}_k \odot \mathbf{s}_{k-1} + \mathbf{i}_k \odot \mathbf{g}_k \tag{7}$$

$$\mathbf{h}_k = \mathbf{o}_k \odot \tanh(\mathbf{s}_k) \tag{8}$$

where $\tanh(\cdot)$, and $\sigma(\cdot)$ represent hyperbolic tangent and the logistic sigmoid functions, respectively, $\mathbf{W}_x$ and $\mathbf{W}_h$ the input and output weights, respectively, $\mathbf{b}$ the bias vector associated with the gate in subscript, and $\odot$ an element-wise multiplication. Both weights and biases are shared through all time steps.

2.2. Transfer learning

A common assumption in machine learning methods is that training and test datasets follow the same distribution and are drawn from the same feature space [7]. In our application domain of interest, this assumption does not hold, because it is assumed that

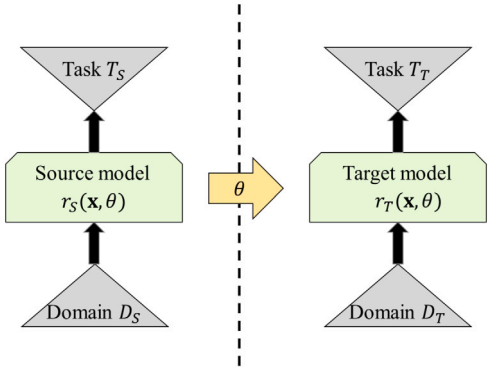

Fig. 2. Schematic representation of transfer learning.

most data cannot be collected a priori, let alone the non-stationarity of the dynamics. In other words, it is assumed that only limited information about the dynamics is available. We leverage transfer learning to transfer knowledge between task domains.

A domain $\mathcal{D}$ is defined with its feature space and its probability distribution $\{\chi, P(\mathbf{x})\}$, and a learning task $\mathcal{T}$ is defined as an input space with a corresponding predictive function $\{\mathbf{x}, r(\cdot)\}$. The objective of transfer learning, illustrated in Fig. 2, is to accelerate learning in a target domain $\mathcal{D}_T$ with learning task $\mathcal{T}_T$ by exploiting the knowledge gained from $\mathcal{D}_S$ [7] given a dynamic event source domain $\mathcal{D}_S$ with learning task $\mathcal{T}_S$, is . This is done by finding a set of optimal parameters θ^* for $\mathcal{D}_S$ such that [42]

$$\theta^* = \underset{\theta\in\Theta}{\operatorname{argmin}} \sum_{\mathbf{x}\in D_T} P(\mathbf{x}) L(\hat{x}, r(\mathbf{x},\theta)) \tag{9}$$

where θ consists of the weights and biases of the RNNs, Θ is the parameter space, $L(\cdot)$ is the loss function, and $\hat{x}$ is the target value, for example, the estimation $\hat{x}_{k+1}$. However, the target domain data is assumed as unavailable for training, and the minimization (Eq. (9)) is conducted instead in D_S, and because the probability distribution is unknown, the minimization problem is further modified by evaluating the average value of the loss function

$$\theta^* = \underset{\theta\in\Theta}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} L(\hat{x}, r(\mathbf{x}_i,\theta)) \tag{10}$$

The transfer learning process is shown in Fig. 2 .

2.3. Ensemble of RNNs

The algorithm consists of a number of RNNs running in parallel, each sampling the time series at a different rate in order to extract multi-temporal features. Because our algorithm is developed for real-time applications, its training is performed sequentially. This involves multi-rate sampling of data at each time step (i.e., multi-rate sequential sampling), conducted as data becomes available. At time step k, each RNN i has an input vector $\mathbf{x}_k^i$ characterized by a sampling delay τ and input sequence length d, also known as embedding dimension. Introducing different sampling delays and embedding dimensions allows exposing each LSTM to a different time resolution of the underlying system and yielding the extraction of a different feature, thus decomposing the input time series into less complex sub-systems that are easier to model and predict. Fig. 3 illustrates the multi-rate sampling sequences process for an arbitrary time series with J samples. The solid lines and the circles represent the underlying continuous system and the discrete sampling, respectively. A multi-rate sampling at time step k for a window of size Δk_i is extracted from the values in $[k-\Delta k_i, k]$. The resulting sample is the input for RNN_i and is a delayed vector with length d_i of the form

$$\mathbf{x}_k^i = \left\{ x_{k+1-d\tau_i}, x_{k+1-(d_i-1)\tau_i}, \dots, x_{k+1-2\tau_i}, x_{k+1-\tau_i} \right\} \tag{11}$$

with τ defined as a multiple of discrete steps. Remark that, regardless of the selected values $\{\tau_i, d_i\}$, the target estimation for each RNN is x_{k+1}, corresponding to the next delay value of the input sequence (Eq. (11)).

The number of RNNs and their corresponding time delays τ are selected based on investigating the topology of phase-space of the source domain time series, and the corresponding sequence length d are selected based on a parametric study. The hyper-parameter selection will be discussed in detail in Section 3. Each RNN is trained independently using the sequentially sampled data from the source domain corresponding to their selected input delay vector. The sampling window slides with a stride of S in number of discrete steps, and samples are fed into the RNNs as the sampling window moves forward.

The overall algorithm for high-rate time series estimation is schematized in Fig. 4. First, the number of RNNs and their respective input parameters τ and d are selected, and the RNNs are pre-trained using the available data in a source domain. This procedure is explained in detail for the experimental validation in Section 3.2. After, the pre-trained RNNs, as illustrated in Fig. 5, with their feed-forward layer removed, are used to extract features of the target estimation $\hat{x}_{k+1}$ by sampling the target domain time series according to each individual delay vector characteristics (τ_i, d_i). The outputs of the RNNs are fed into an attention layer to

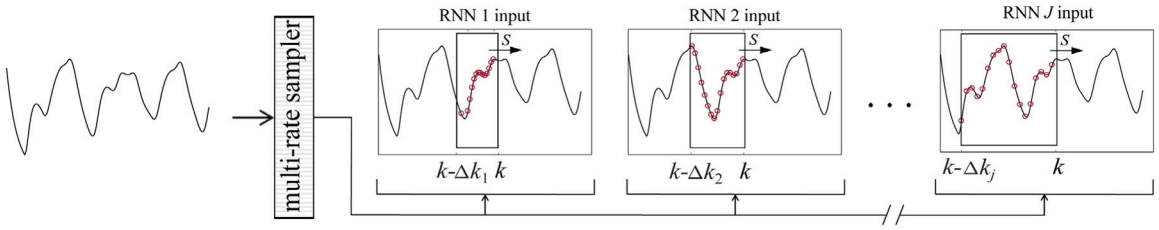

Fig. 3. Multi-rate sequential sampling illustration.

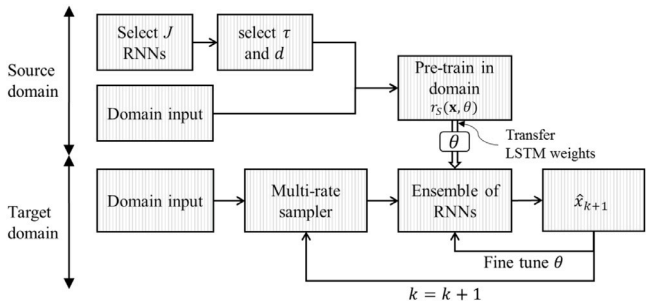

Fig. 4. Flowchart of the proposed high-rate time series estimation algorithm.

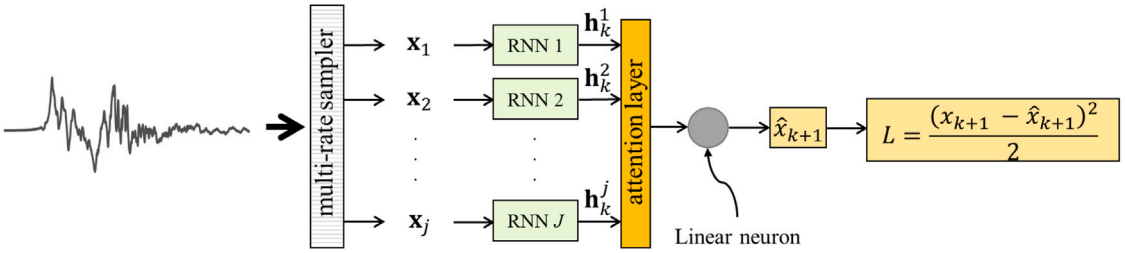

Fig. 5. Schematics of the high-rate time series estimation algorithm.

identify the relative importance of the features extracted by each of them, therefore allowing the model to switch its focus based on local temporal features. In this layer, the output of the i^{th} RNN, $\mathbf{h}_k^i$, is multiplied by an attention weight factor $\alpha_i \in (0, 1]$. The attention mechanism with only a single weight is intentionally selected to have a short memory because of the expected fast-changing dynamics of high-rate systems requiring fast adaptability of the representation.

It is assumed that the target value is a linear combination of the RNN outputs, whereas the output of the attention layer is fed into a single neuron with linear activation to compute the next step value $\hat{x}_{k+1}$. The least square loss function

$$L = \frac{(x_{k+1} - \hat{x}_{k+1})^2}{2} \tag{12}$$

is then defined as the objective function to optimize neuro-parameters using back-propagation in time [43].

3. Validation methodology

3.1. Drop tower tests

The proposed algorithm was validated on experimental high-rate dynamic data produced from accelerated drop tower tests [6]. The experimental configuration is shown in Fig. 6 and described in details in [6]. Briefly, an electronics package consisting of four circuit boards, each equipped with one accelerometer, was placed inside a canister. Each accelerometer was capable of measuring accelerations up to 120,000 g-force (or 120 kg) at a sampling rate of 1 MHz. The canister was dropped five consecutive times and responses were recorded. Deceleration time series (TS) of the bottom accelerometer (TS_1) and top accelerometer (TS_2) were used for the validation process.

The recorded time series responses for the five consecutive tests (test 1 to test 5) are plotted in Fig. 7 and Fig. 8 for accelerometers TS_1 and TS_2, respectively. The following observations can be made from the figures: (1) the impact is very high, reaching almost

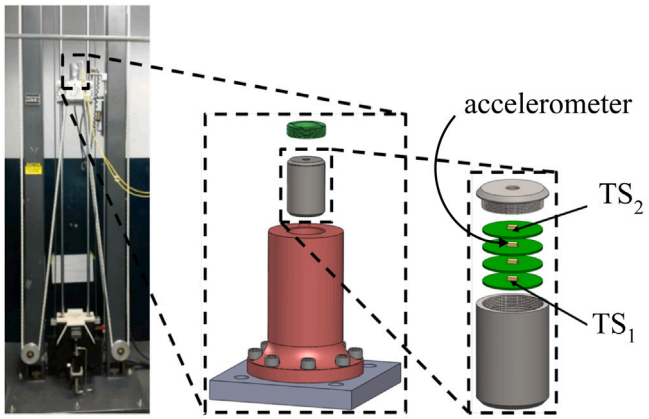

Fig. 6. Drop tower experimental setup [6].

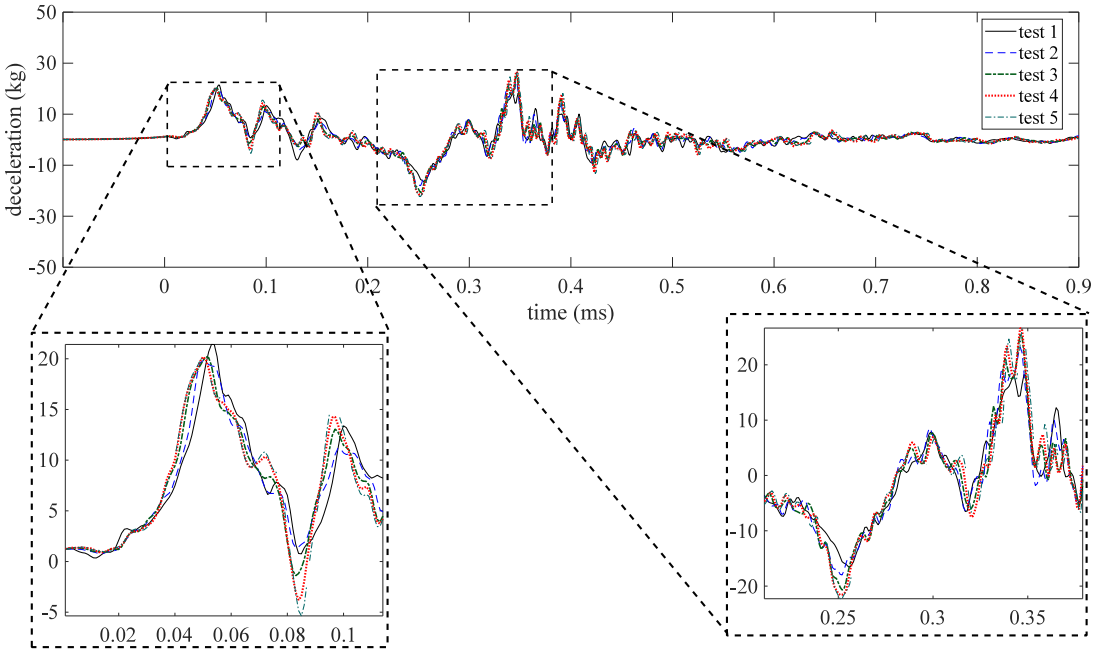

Fig. 7. Deceleration time history of TS_1 for tests 1–5.

85 kg; (2) responses are non-stationary; and (3) a cross-comparison among tests reveals a change in dynamics after each hit, which is hypothesized to be the result of cracks in the epoxy holding the accelerometers and loosening of connections. It follows that the physical modeling of such high-rate dynamic system is difficult, where the necessity to leverage machine learning tools.

3.2. Pre-training

In our validation, we take the time series of test 1 from TS_1 as the source domain (prior knowledge) and take the time series of tests 1–5 from TS_2 as the target domain (real-time data). By doing so, we simulate that prior knowledge was obtained through a first set of test, and that a different, yet dynamically related response was occurring real-time. Thus, TS_1 (test 1) is used for the pre-training of the RNNs, and TS_2 (tests 1–5) are estimated using the proposed algorithm, that is by transferring knowledge gained from the TS_1 time series.

The ensemble is constructed using five RNNs to accelerate computation time. Performance of the algorithm as a function of different numbers of RNNs will also be investigated. Values for τ and d used in constructing the delay vector of each RNN are based on pre-processing of the TS_1 time series and estimating their maximum values over the entire time series based on the topology of phase space and optimal sequence length [44].

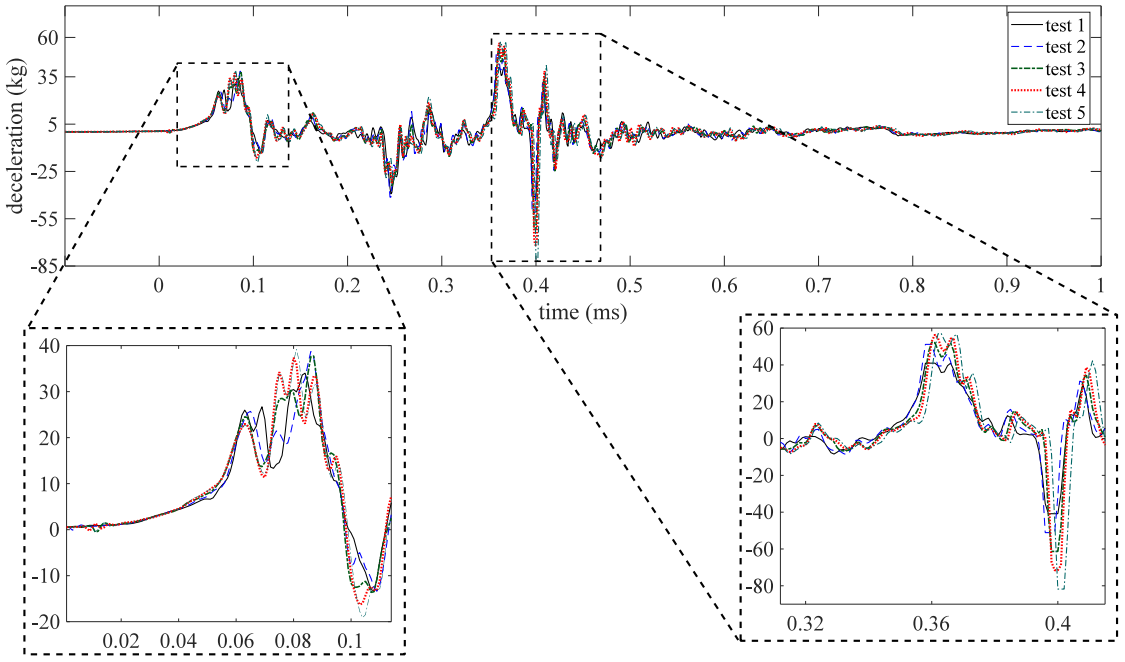

Fig. 8. Deceleration time history of TS_2 for tests 1–5.

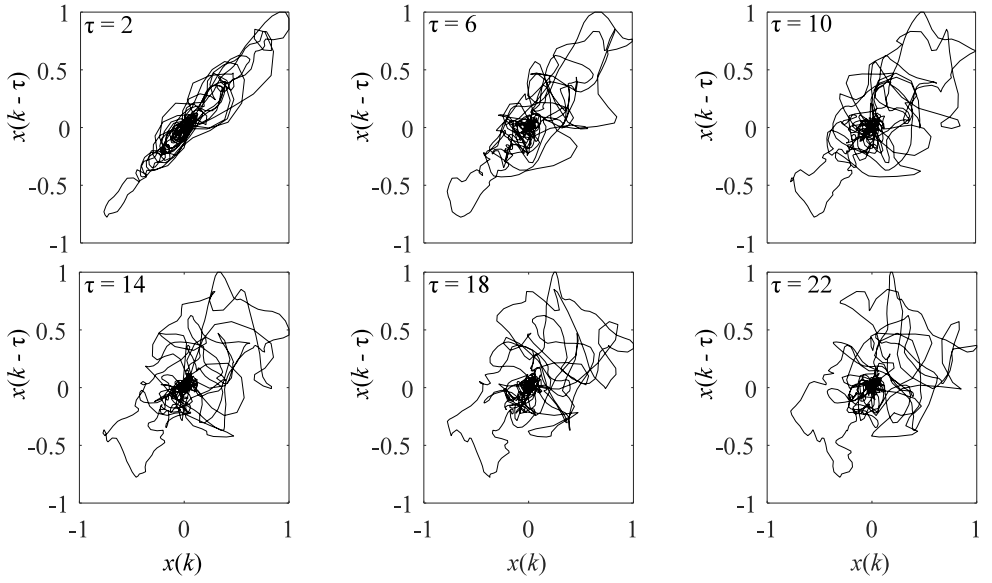

Fig. 9. Topology of the phase space for various delay values (τ): $\tau = 2, 6, 10, 14, 18,$ and 22.

First, the choice of the maximum value for τ, τ_{max}, is based on mutual information and here conducted visually by investigating the topology of the phase-space, plotted in Fig. 9 for various values of τ. For a small delay ($\tau = 2$), the topology is closer to a straight line. As τ increases, the phase-space unfolds up to $\tau = 14$, after which the topology appears to fold back on itself, indicating a loss of information. Thus, $\tau_{max} = 14$, and each RNN will be assigned a different value of τ between $\tau = 1$ and $\tau_{max} = 14$, with values equally spaced. For five RNNs used in an ensemble, we assign $\tau_i = \{2, 5, 8, 11, 14\}$. Remark that this technique assumes stationarity of the entire time series, but yet is considered as a good starting point. More advanced investigations on algorithms for selecting $\tau_{\max}$, and in particular, assigning more targeted values for each RNN, is left to future work.

Second, the sequence length corresponding to each delay is selected by minimizing the root-mean-square error (RMSE) on the estimation, where for that purpose an RNN is trained for each new choice of τ and d. Results from this investigation are plotted in Fig. 10. Based on these results, the following parameters $\{\tau_i, d_i\}$ are selected to construct the delay vector of the five RNNs:

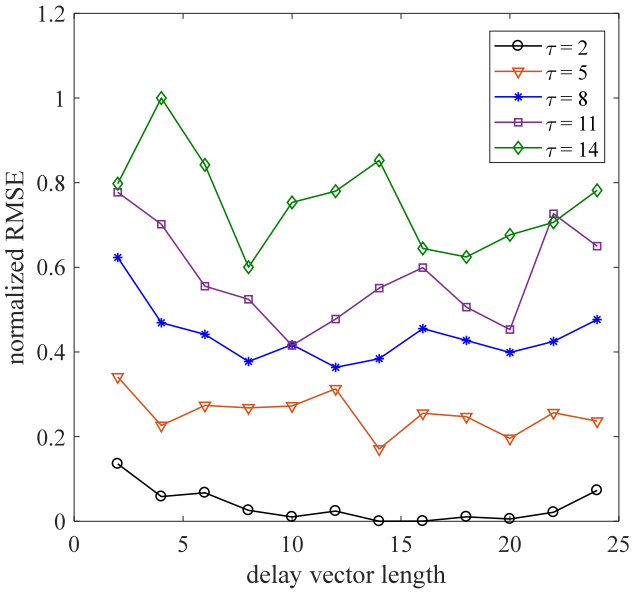

Fig. 10. Normalized RMSE versus delay vector length under each selected time delays.

Table 1
Input vector parameters as a function of the number of RNNs in the ensemble.

1 RNN	3 RNNs	5 RNNs	7 RNNs	9 RNNs	
$\{\tau_1, d_1\}$	$\{2, 15\}$	$\{2, 15\}$	$\{2, 15\}$	$\{2, 15\}$	$\{2, 15\}$
$\{\tau_2, d_2\}$	–	$\{5, 14\}$	$\{5, 14\}$	$\{5, 14\}$	$\{5, 14\}$
$\{\tau_3, d_3\}$	–	$\{8, 12\}$	$\{8, 12\}$	$\{8, 12\}$	$\{8, 12\}$
$\{\tau_4, d_4\}$	–	–	$\{11, 10\}$	$\{11, 10\}$	$\{11, 10\}$
$\{\tau_5, d_5\}$	–	–	$\{14, 8\}$	$\{14, 8\}$	$\{14, 8\}$
$\{\tau_6, d_6\}$	–	–	–	$\{17, 12\}$	$\{17, 12\}$
$\{\tau_7, d_7\}$	–	–	–	$\{20, 6\}$	$\{20, 6\}$
$\{\tau_8, d_8\}$	–	–	–	–	$\{23, 8\}$
$\{\tau_9, d_9\}$	–	–	–	–	$\{26, 6\}$

Table 2
Hyper-parameters used for pre-training.

Parameter	Value
Window size	$\tau(d+5)$
Stride	τ
Samples/window	3
# of epochs	2
Learning rate	0.006
LSTM hidden dimension	$2d$

{2,15}, {5,14}, {8,12}, {11,10}, {14,8}. Remark that the numerical investigation will investigate the performance of the algorithm as a function of different number of RNNs in the ensemble, including 1, 3, 5, 7, and 9 RNNs. The parameters used in constructing their input vectors are listed in Table 1. In the case of 7 and 9 RNNs, values for τ larger than $\tau_{\max}$ were selected to avoid using near-redundant information in the input.

After the construction of the delay vectors, each RNN is trained using standard back-propagation in time with the least square loss function, as shown in Fig. 5. All of the numerical simulations were performed in Python using a self-developed code and ran with an Intel Core i7-4770 processor. Table 2 lists the hyper-parameters used for training. The one-step-ahead prediction results of the RNNs on the training time series TS_1 are plotted in Fig. 11. As the delay increases, the prediction moves from capturing local details to capturing global trends, which is also evident in the frequency plots of Fig. 12 whereas higher time delays do not capture higher frequencies.

3.3. Performance evaluation

The validation of the algorithm is conducted using six performance metrics, listed in Table 3. Metric J1 measures the average computation time per step. Metrics J2 and J3 are the mean absolute error (MAE) and RMSE, both being measures of average

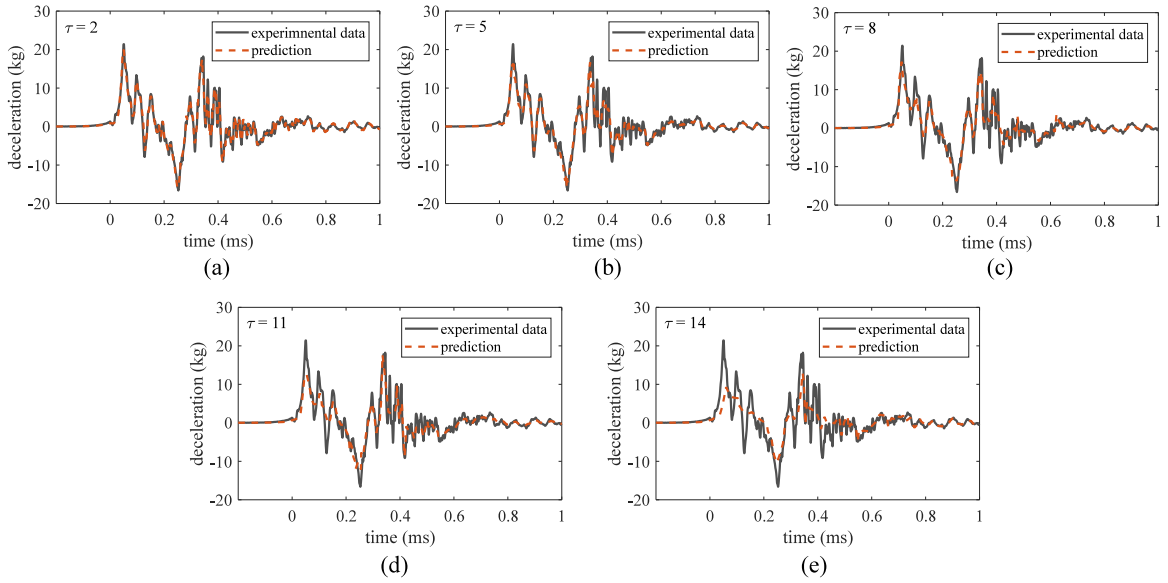

Fig. 11. One-step-ahead prediction performance of the trained RNNs on TS_1: (a) $\tau = 2$; (b) $\tau = 5$, (c) $\tau = 8$; (d) $\tau = 11$; (e) $\tau = 14$.

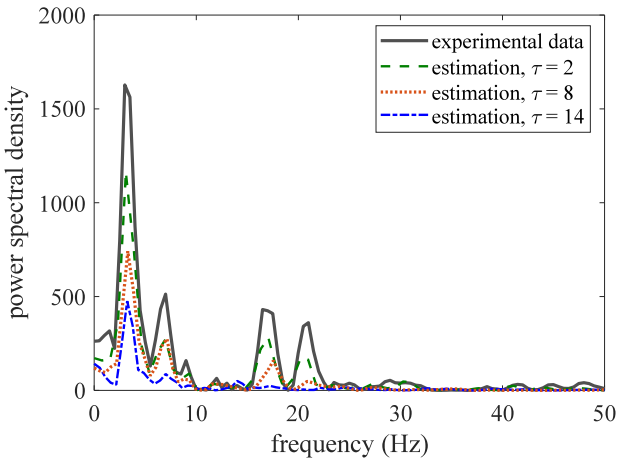

Fig. 12. Power spectral density of the experimental data versus estimated values for $\tau = 2, 8$, and 14.

estimation errors but with the RMSE giving a higher weight to larger errors, written as

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}\left|x_{k+1} - \hat{x}_{k+1}\right| \tag{13}$$

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(x_{k+1} - \hat{x}_{k+1}\right)^2} \tag{14}$$

where N is the number of samples. Metric J4 is a measure of convergence of the estimation to the real value, and is taken as the MAE of the estimations up to the first peak of the measurements occurring at 0.06 ms, where a lower value for J4 indicates faster convergence. Metric J5 is the absolute error of the prediction in the entire trace. Metric J6 is the delay ratio and takes the following form

$$\text{delay ratio} = \frac{\sum_{i=1}^{N}\left|x_{k+1} - \hat{x}_{k+1}\right|}{\sum_{i=1}^{N}\left|x_k - \hat{x}_{k+1}\right|} \tag{15}$$

Table 3
Performance metrics.

Metric	Description (units)
J1	Computation time/step (μs)
J2	MAE (kg)
J3	RMSE (kg)
J4	MAE of the first 0.05 ms (kg)
J5	Absolute error (kg)
J6	Delay ratio

Table 4
Hyper-parameters used for one-step-ahead prediction.

Parameter	Value
Window size	τd
Stride	1
Samples/window	1
Learning rate	0.006

The above metric compares the average delay of the prediction with respect to the previous data point. As the prediction improves, $\hat{x}_{k+1} \to x_{k+1}$ and J6 $\to 0$. This metric allows applying a penalty for performing a naive prediction with J6 $\to \infty$ for $\hat{x}_{k+1} \to x_k$. For all performance metrics, a smaller value represents better performance.

Numerical results are benchmarked against results reported in Hong et al. [6], where a purely on-the-edge learning observer, termed variable input observer (VIO), was developed and showed promising performance on TS_2 data. The authors also augmented their VIO with physical knowledge to improve performance (hybrid VIO). Using the authors' data, performance metrics J2–J6 were computed for both the VIO and hybrid VIO and will be used for benchmarking, with "hybrid VIO" considered as an upper bound on performance. Remark that performance metric J1 is not used, because it was concluded in [6] that the algorithm was not capable of sub-millisecond performance.

4. Numerical results

The pre-trained RNNs are used to conduct one step-ahead prediction of TS_2 time series in the ensemble configuration schematized in Fig. 5. The hyper-parameters of the ensemble are listed in Table 4. Sampling is conducted for each data point as it becomes available, and the one-step-ahead value is predicted.

Fig. 13 plots a typical result obtained on TS_2, test 1. Overall, all techniques appear to yield an acceptable match of experimental data, with the ensemble of RNNs exhibiting more chattering for low-amplitude excitations occurring after the high-rate event (observable after 0.7 ms). A zoom on the start of the excitation (bottom left figure) shows that the ensemble of RNNs converges slightly faster than the VIO, with the hybrid VIO converging immediately which is reflected in the estimation error. A zoom on the high-amplitude event (bottom-right figure) shows that the ensemble of RNNs significantly outperforms the VIO by being capable of capturing rapid changes in dynamics, with a performance close to that of the hybrid VIO. Hybrid VIO shows consistent estimation errors throughout the time series with occasional error spikes. VIO yields large estimation errors mainly due to a lag in capturing the changing dynamics while the ensemble of RNNs shows smaller errors because of faster convergence.

Fig. 14 compares the performance metrics of the three methods. The ensemble of RNNs resulted in a significant gain in the estimation accuracy compared to the VIO, with improvements of 65%, 64%, and 52% for J2, J3, and J5, respectively. It also exhibited 22% improvement in convergence (J4) and 23% less naive estimation (J6); however, it underperformed the hybrid VIO by 55% and 24% over J2 and J3, but showed an improvement of 12% over J5. Also, the hybrid VIO yielded 66% faster convergence and was 56% less naive compared with the ensemble of RNNs.

Fig. 15 presents the evolution of the attention weights for each of the RNNs in the ensemble. The non-stationary nature of the high-rate dynamic event requires a rapid and continuous change of each attention weight. It can be observed that the attention weights change more importantly when the system experiences a more chaotic response. Another observable feature is that RNN # 4 and #5, which have the largest delay parameters, attain relatively higher weights than those of the other RNNs, with average values of 0.22 and 0.23, respectively, compared to 0.2, 0.18, and 0.17 for RNNs #1 to #3, respectively. This feature contribute to fast-adapting capability of the estimations. Since the signal fluctuates significantly, the trend of the signal becomes important in accurate prediction of the next step whereas the small delay RNNs provide the high-frequency contents the prediction. It can also be observed that the variations in weights of RNN #1 and #2 are higher, whereas the RNNs are more aggressive at tracking high frequency details in the signal, with standard deviations in weights of 0.028 and 0.18 for RNN #1 and #2, respectively, compared to 0.010, 0.011, and 0.013 for RNN #3, #4, and #5, respectively.

This observation can be confirmed by studying the behavior of the hidden states in Fig. 16, which plots a typical scaled hidden states from RNN #1 (HS 1), and one from RNN #4 (HS 4). HS4 tends to capture peaks, noticeable by its time series adapting faster to higher peaks, but not the lower frequency behaviors, noticeable by the hidden state remaining constant particularly after approximately 0.5 ms, whereas HS1 tends to adapt faster. This faster adaptation of HS1 to low-amplitude excitations can be attributed to its small time delay ($\tau = 2$) and be responsible for the high chattering at the end of the excitation observable in

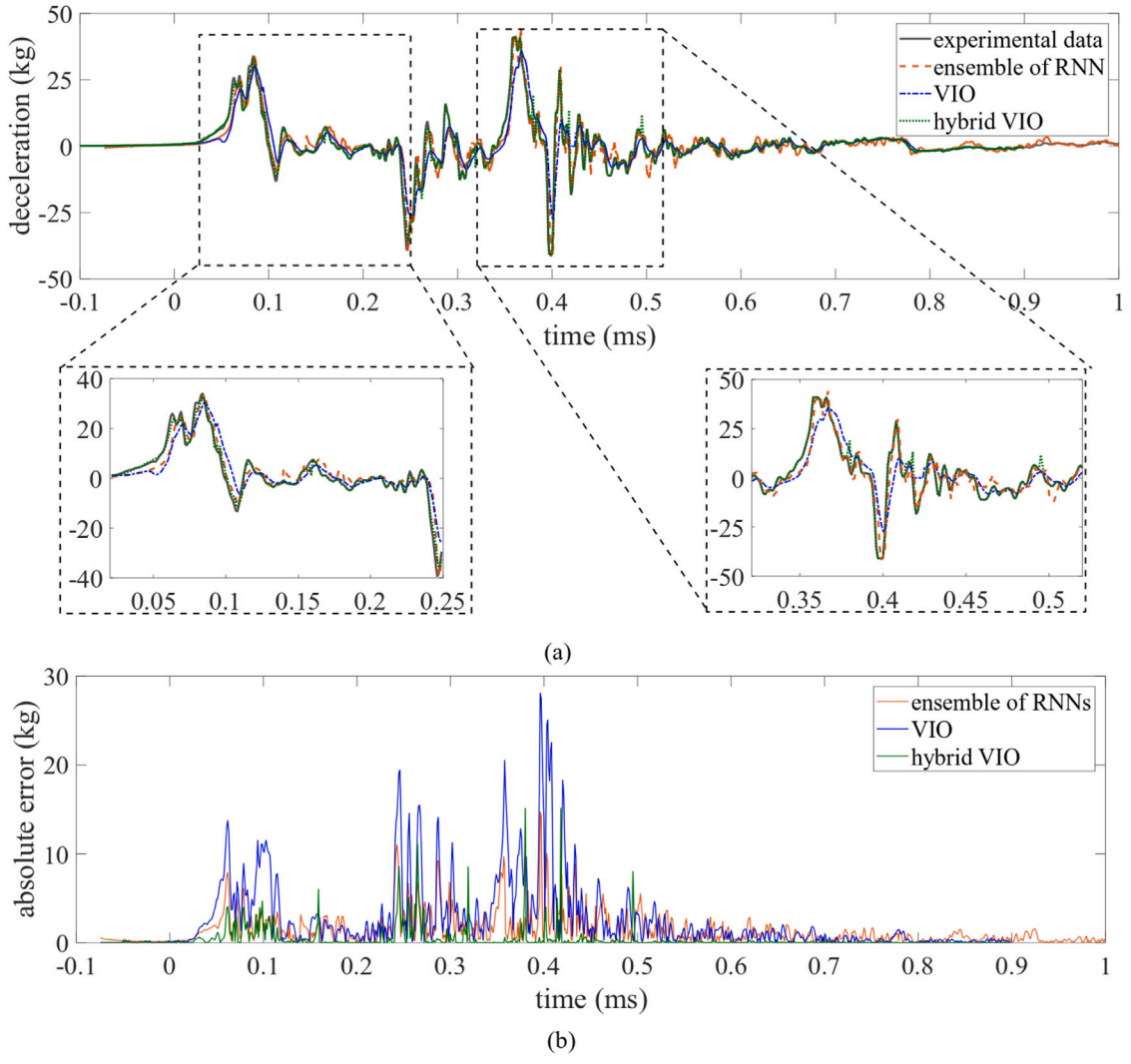

Fig. 13. Prediction time histories over TS_2 — test 1; (a) comparing the estimations, and (b) comparing the evolution of absolute errors of ensemble of RNNs, VIO, and hybrid VIO.

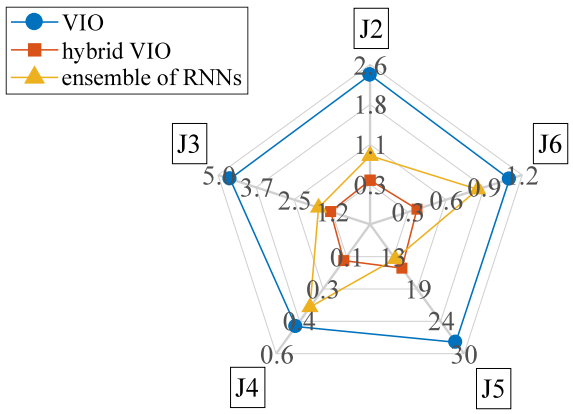

Fig. 14. Spider plot of performance metrics J2–J6 for the ensemble of RNNs, VIO, and hybrid VIO, over TS_2 — test 1.

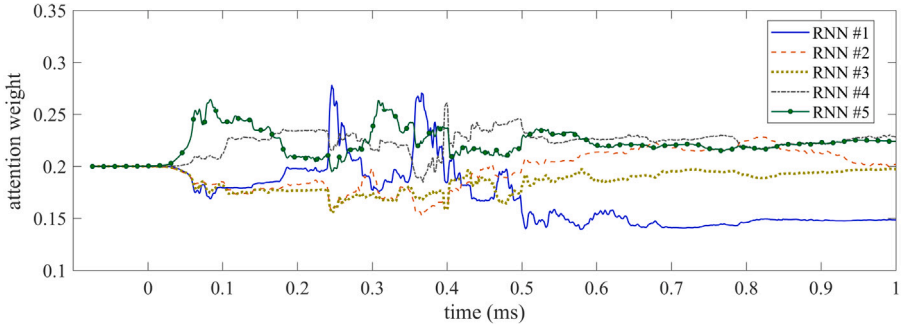

Fig. 15. Evolution of the RNN attention weights over TS_2 — test 1.

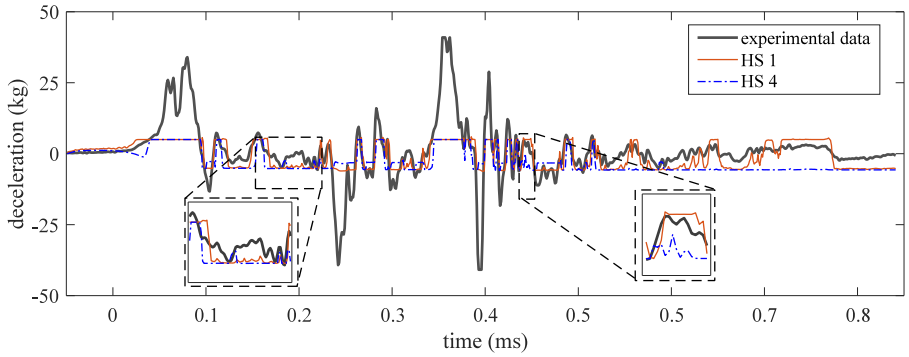

Fig. 16. Typical hidden state from RNN 1 (HS 1) and RNN 4 (HS 4) time history compared to TS_2 — test 1.

Fig. 13. Remark that, here, only some levels of constraints on the dynamics learned by the LSTM cells were enforced through the multi-rate sampler. The incorporation of additional physical constraints would require additional physical knowledge, for example, knowledge of modal properties [45], which was taken as not accessible for the drop tower experiment.

Fig. 17 shows, for each of the five TS_2 tests, the performance of the algorithm as a function of the number of RNNs used in the ensemble. Using a higher number of RNNs significantly increases of the computation time of the algorithm, yet with five RNNs remaining under 25 μs. Metrics J2 to J5 decrease with the addition of new RNNs, but with a decreasing marginal gain, where the increase in performance becomes insignificant compared to the addition cost in computation time. In particular, averaging over the five tests, the ensemble with five RNNs outperforms the ensemble with one RNN by 21%, 35%, 45%, 61% and 13% and the ensemble with three RNNs by 8%, 12%, 37%, 35%, and 3% over J2 to J6, respectively, while the ensemble with nine RNNs only outperformed the ensemble with five RNNs by 4%, 5%, 8%, 2%, and 1.3% over J2 to J5, respectively, yet increasing the average computation time from 25 μs to 58.6 μs.

Lastly, the robustness of the algorithm with respect to noise in the training dataset TS_1 test 1 is investigated. To do so, white Gaussian noise with zero mean was introduced in the time series with different signal-to-noise (SNR) ratios. The RNNs were re-trained on the noise dataset and the J2 to J5 performance metrics of the predictions for all test were calculated in the presence of noise for different values of SNR and plotted in Fig. 18. It can be observed that, overall, the performance decreases when adding noise below 20 dB, in particular over metric J5 (maximum absolute error), attributable to error spikes. Metric J6 (naivety of the estimation) appears less sensitive to noise.

5. Conclusions

In this paper, an ensemble of recurrent neural networks (RNNs) was proposed for high-rate structural health monitoring (HRSHM) applications, where the algorithm was used for one-step-ahead predictions of high-rate time series. The novelty of this algorithm was in its design enabling pre-training on very limited datasets, performance on highly non-stationary time series, and sub-millisecond computing time. This was achieved by utilizing short-sequence long short-term memory (LSTM) cells, a parallel computing architecture, and with multi-rate sampling where each RNN was constructed with a different input delay vector in order to capture different temporal dynamics within the time series. Transfer learning was used to extrapolate the learned dynamics to the new time series.

The algorithm was built as follows. First, characteristics of the different delay vectors were selected by investigating the topology of the training data's phase space. Second, the optimal embedding dimension for each RNN was established by minimizing the root-mean-square error of the prediction during pre-training. Third, the ensemble of RNNs was constructed by assembling these RNNs

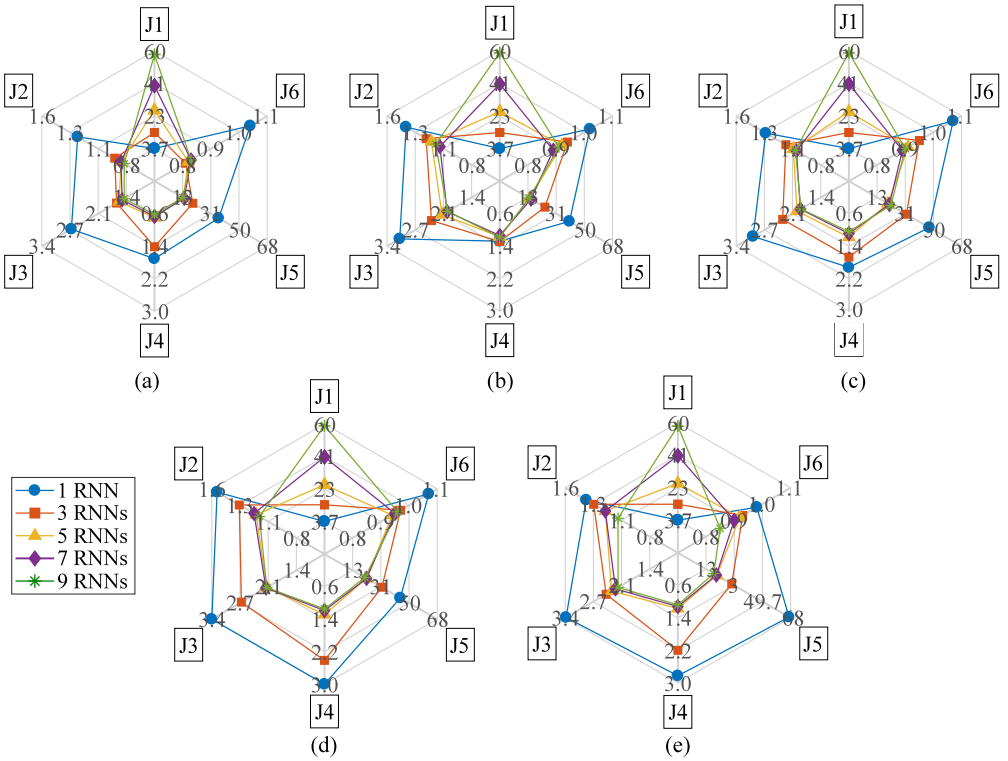

Fig. 17. Spider plots of the performance metrics J1–J6: (a) test 1; (b) test 2; (c) test 3; (d) test 4; and (e) test 5.

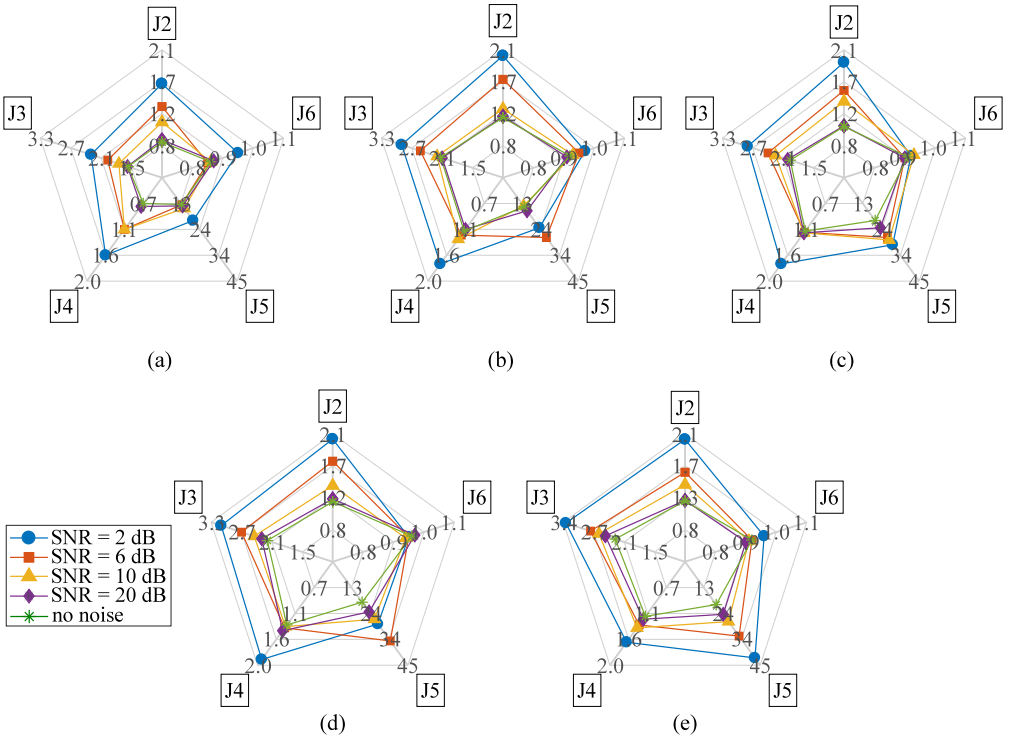

Fig. 18. Effect of noise in training dataset on the performance algorithm in terms of J2–J5 performance metrics: (a) test 1; (b) test 2; (c) test 3; (d) test 4; (e) test 5.

in parallel. In the estimation phase, each RNN sampled the new time series sequentially at its pre-determined sampling rate and conducted an individual estimation. These estimations were passed into an attention layer, and the combined output passed into a linear neuron to produce the final estimation.

The performance of the algorithm was evaluated on experimental high-rate dynamic data produced from accelerated drop tower tests, where time series data were collected from two accelerometers over five consecutive tests. Time series data from one accelerometer and a single test (TS_1 test 1) were used for pre-training, while time series data form the other accelerometers were used for validation (TS_2 tests 1–5). Results were benchmarked against those obtained from a previous study conducted on the same dataset conducted using a purely on-the-edge algorithm termed variable input observer (VIO) along with its hybrid version (hybrid VIO) that included some physical knowledge of the system and was considered as an upper bounder on performance.

Results showed that, using five RNNs, the proposed algorithm significantly outperformed the VIO and performed near the hybrid VIO for the estimation error metrics. It also outperformed the VIO for the convergence and delay ratio metrics, but still underperformed the hybrid VIO for these metrics. Overall, the algorithm showed good convergence for high-amplitude events, but chattering for low-amplitude excitations likely attributable to the performance of RNNs using low time delays. A study on the number of RNNs used in the ensemble showed that augmenting the number of RNNs has a positive impact on all performance metrics, but with a significant decrease in marginal gain beyond five RNNs and with a high increase in computation time. For this specific dataset, using five RNNs performed very well at the estimation task with an average computation time of 25 μs per step. An investigation of the RNN weights and time series of two typical hidden states confirmed the capability of the ensemble of RNNs to capture multi-temporal features. Lastly, a study on the robustness of the algorithm with respect to noise in the pre-training time series (TS_1 test 1) showed sensitivity in performance for noise levels under signal-to-noise ratios of 20 dB.

Overall, this study showed the potential of the proposed ensemble of RNNs at conducting very fast estimations for highly non-stationary time series, a critical step toward HRSHM applications. A simple, direct application of the proposed algorithm would be for high-rate damage detection through the study of the prediction error, where a notable change in the prediction error would correlate with a change in the system state. More advanced implementations can integrate predictions with simplified adaptive physics-based methods to directly obtain actionable quantities.

CRediT authorship contribution statement

Vahid Barzegar: Conceptualization, Formal analysis, Investigation, Methodology, Validation, Writing-original draft, Writing - review & editing. **Simon Laflamme:** Conceptualization, Methodology, Writing-original draft, Writing - review & editing, Supervision, Funding acquisition, Project administration. **Chao Hu:** Writing-original draft, Writing - review & editing, Supervision, Funding acquisition. **Jacob Dodson:** Resources, Data curation, Writing - review & editing.

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

The work presented in this paper is funded by the National Science Foundation under award number CISE-1937460. Their support is gratefully acknowledged. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsor.

References

[1] Jonathan Hong, Simon Laflamme, Jacob Dodson, Bryan Joyce, Introduction to state estimation of high-rate system dynamics, Sensors 18 (2) (2018) 217.
[2] Jonathan Hong, Simon Laflamme, Jacob Dodson, Study of input space for state estimation of high-rate dynamics, Struct. Control Health Monit. 25 (6) (2018) e2159.
[3] Bryan Joyce, Jacob Dodson, Simon Laflamme, Jonathan Hong, An experimental test bed for developing high-rate structural health monitoring methods, Shock Vib. 2018 (2018).
[4] Austin Downey, Jonathan Hong, Jacob Dodson, Michael Carroll, James Scheppegrell, Millisecond model updating for structures experiencing unmodeled high-rate dynamic events, Mech. Syst. Signal Process. 138 (2020) 106551.
[5] Jin Yan, Simon Laflamme, Jonathan Hong, Jacob Dodson, Online parameter estimation under non-persistent excitations for high-rate dynamic systems, Mech. Syst. Signal Process. (2020).
[6] Jonathan Hong, Simon Laflamme, Liang Cao, Jacob Dodson, Bryan Joyce, Variable input observer for nonstationary high-rate dynamic systems, Neural Comput. Appl. 32 (9) (2018) 5015–5026.
[7] Sinno Jialin Pan, Qiang Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1345–1359.
[8] Yoshua Bengio, Deep learning of representations for unsupervised and transfer learning, in: Proceedings of ICML Workshop on Unsupervised and Transfer Learning, 2012, pp. 17–36.
[9] Gabriela Csurka, Domain adaptation for visual applications: A comprehensive survey, 2017, arXiv preprint arXiv:1702.05374.
[10] Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, Silvio Savarese, Taskonomy: Disentangling task transfer learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 3712–3722.
[11] Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, Samy Bengio, Transfusion: Understanding transfer learning for medical imaging, in: Advances in Neural Information Processing Systems, 2019, pp. 3347–3357.

[12] Huaxiu Yao, Yiding Liu, Ying Wei, Xianfeng Tang, Zhenhui Li, Learning from multiple cities: A meta-learning approach for spatial-temporal prediction, in: The World Wide Web Conference, 2019, pp. 2181–2191.
[13] Hussain Kazmi, Johan Driesen, Automated demand side management in buildings, in: Artificial Intelligence Techniques for a Scalable Energy Transition, Springer International Publishing, 2020, pp. 45–76.
[14] Yujiao Chen, Zheming Tong, Yang Zheng, Holly Samuelson, Leslie Norford, Transfer learning with deep neural networks for model predictive control of HVAC and natural ventilation in smart buildings, J. Cleaner Prod. 254 (2020) 119866.
[15] Anna X. Wang, Caelin Tran, Nikhil Desai, David Lobell, Stefano Ermon, Deep transfer learning for crop yield prediction with remote sensing data, in: Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies, Vol. 18, ACM Press, 2018.
[16] Sanjay Purushotham, Wilka Carvalho, Tanachat Nilanon, Yan Liu, Variational recurrent adversarial deep domain adaptation, 2016.
[17] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, Antoine Bordes, Supervised learning of universal sentence representations from natural language inference data, 2017, arXiv preprint arXiv:1705.02364.
[18] Yifan Peng, Shankai Yan, Zhiyong Lu, Transfer learning in biomedical natural language processing: An evaluation of bert and elmo on ten benchmarking datasets, 2019, arXiv preprint arXiv:1906.05474.
[19] Yujie Ying, James H. Garrett, Irving J. Oppenheim, Lucio Soibelman, Joel B. Harley, Jun Shi, Yuanwei Jin, Toward data-driven structural health monitoring: Application of machine learning and signal processing to damage detection, J. Comput. Civ. Eng. 27 (6) (2013) 667–680.
[20] Rih-Teng Wu, Mohammad R. Jahanshahi, Deep convolutional neural network for structural dynamic response estimation and system identification, J. Eng. Mech. 145 (1) (2019) 04018125.
[21] David García, Dmitri Tcherniak, An experimental study on the data-driven structural health monitoring of large wind turbine blades using a single accelerometer and actuator, Mech. Syst. Signal Process. 127 (2019) 102–119.
[22] Kyongmin Yeo, Igor Melnyk, Deep learning algorithm for data-driven simulation of noisy dynamical system, J. Comput. Phys. 376 (2019) 1212–1231.
[23] Sepp Hochreiter, Jürgen Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[24] Felix A. Gers, Jürgen Schmidhuber, Fred Cummins, Learning to forget: Continual prediction with LSTM, Neural Comput. 12 (10) (2000) 2451–2471.
[25] Klaus Greff, Rupesh K. Srivastava, Jan Koutnik, Bas R. Steunebrink, Jurgen Schmidhuber, LSTM: A search space odyssey, IEEE Trans. Neural Netw. Learn. Syst. 28 (10) (2017) 2222–2232.
[26] Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, Garrison Cottrell, A dual-stage attention-based recurrent neural network for time series prediction, 2017, arXiv preprint arXiv:1704.02971.
[27] Jun Ma, Jack C.P. Cheng, Changqing Lin, Yi Tan, Jingcheng Zhang, Improving air quality prediction accuracy at larger temporal resolutions using deep learning and transfer learning techniques, Atmos. Environ. 214 (2019) 116885.
[28] Lingxue Zhu, Nikolay Laptev, Deep and confident prediction for time series at uber, in: 2017 IEEE International Conference on Data Mining Workshops, ICDMW, IEEE, 2017.
[29] Shun-Yao Shih, Fan-Keng Sun, Hung yi Lee, Temporal pattern attention for multivariate time series forecasting, Mach. Learn. 108 (8–9) (2019) 1421–1441.
[30] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, Yan Liu, Recurrent neural networks for multivariate time series with missing values, Sci. Rep. 8 (1) (2018).
[31] Jaideep Pathak, Brian Hunt, Michelle Girvan, Zhixin Lu, Edward Ott, Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach, Phys. Rev. Lett. 120 (2) (2018).
[32] Min Han, Meiling Xu, Laplacian echo state network for multivariate time series prediction, IEEE Trans. Neural Netw. Learn. Syst. 29 (1) (2018) 238–244.
[33] Qianli Ma, Lifeng Shen, Garrison W. Cottrell, DeePr-ESN: A deep projection-encoding echo-state network, Inform. Sci. 511 (2020) 152–171.
[34] Rui Fu, Zuo Zhang, Li Li, Using LSTM and GRU neural network methods for traffic flow prediction, in: 2016 31st Youth Academic Annual Conference of Chinese Association of Automation, YAC, IEEE, 2016.
[35] Edward De Brouwer, Jaak Simm, Adam Arany, Yves Moreau, GRU-ODE-Bayes: Continuous modeling of sporadically-observed time series, in: Advances in Neural Information Processing Systems, 2019, pp. 7379–7390.
[36] Daizong Ding, Mi Zhang, Xudong Pan, Min Yang, Xiangnan He, Modeling extreme events in time series prediction, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 1114–1122.
[37] Floris Takens, Detecting strange attractors in turbulence, in: Dynamical Systems and Turbulence, Warwick 1980, Springer, 1981, pp. 366–381.
[38] Simon Laflamme, J.J. E. Slotine, J.J. Connor, Self-organizing input space for control of structures, Smart Mater. Struct. 21 (11) (2012) 115015.
[39] Alex Graves, Navdeep Jaitly, Abdel rahman Mohamed, Hybrid speech recognition with Deep Bidirectional LSTM, in: 2013 IEEE Workshop on Automatic Speech Recognition and Understanding, IEEE, 2013.
[40] Ilya Sutskever, Oriol Vinyals, Quoc V. Le, Sequence to sequence learning with neural networks, in: Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems, Vol. 27, Curran Associates, Inc., 2014, pp. 3104–3112.
[41] Alex Graves, Supervised sequence labelling, in: Studies in Computational Intelligence, Springer Berlin Heidelberg, 2012, pp. 5–13.
[42] Vladimir Vapnik, Principles of risk minimization for learning theory, in: Advances in Neural Information Processing Systems, 1992, pp. 831–838.
[43] Ronald J. Williams, David Zipser, Gradient-based learning algorithms for recurrent, in: Backpropagation: Theory, Architectures, and Applications, Vol. 433, 1995.
[44] Thomas Schreiber Holger Kantz, Nonlinear Time Series Analysis, Cambridge University Press, 2006.
[45] Vahid Barzegar, Simon Laflamme, Chao Hu, Jacob Dodson, Multi-time resolution ensemble LSTMs for enhanced feature extraction in high-rate time series, Sensors 21 (6) (2021) 1954.