2023 American Control Conference (ACC)
San Diego, CA, USA. May 31 - June 2, 2023

Sequence-to-sequence LSTM-based Dynamic System Identification of Piezo-electric Actuators

Ruocheng Yin[1] and Juan Ren[1,†]

Abstract— **During the past few year, recurrent neural network (RNN) has been proposed to model the nonlinear dynamics of various dynamic systems, such as nano positioning systems (e.g, piezo electric actuators (PEAs)). Although high modeling accuracy has been demonstrated using RNNs, it has been found that the conventional RNNs (such as vanilla RNN) are susceptible to gradient vanishing or exploding issue and hence difficult to train. Deep RNNs, such as Long short-term memory (LSTM), have been proposed to address these issues. However, due to the conventional training data construction, the training is susceptible to overfitting and the computation is extensive. In this paper, we propose a new type of LSTM in the application of PEA system identification: a sequence-to-sequence learning approach (namely, LSTMseq2seq). The structure of LSTMseq2seq and its training data construction are presented in detail. The efficacy of LSTMseq2seq in terms of modeling accuracy and computation speed is demonstrated by applying it for PEA system identification and comparing its performance with that of vanilla RNN.**

Keywords: Sequence-to-sequence, LSTM, PEA, System Identification

I. INTRODUCTION

Nano-positioning devices, such as piezoelectric actuators (PEAs), have been widely used in many high-precision industries, products, and systems due to its fast response and mechanical stability. For example, the atomic force microscope (AFM) [1] [2], micro forming [3], and adaptive optics [4] have many applications of PEAs. However, due to the nonlinear dynamics of PEA (e.g., hysteresis and creep), the real-time positioning control of PEA is always challenging at high frequency and/or large motion range. This is because the PEA nonlinearity is more pronounced and the system modeling accuracy in real-time control is limited in these circumstances.

With the development of machine learning, neural networks, such as feedforward neural network (FNN) [5] and recurrent neural network (RNN) [6], have been proposed for dynamic system identification in real-time control of PEAs during recent years to overcome the aforementioned issues. However, applying the neural networks in the control modeling may introduce other problems that originated from the neural networks while taking advantage of machine learning. The issue with FNN is that it does not treat the FNN input as time series during the training process regardless of the time sequence of input and can affect the behavior of PEAs greatly. In the previous work from our team [6], a vanilla RNN model was originally proposed and integrated with a model predictive controller for PEA position tracking control. RNN can recognize the inter-temporal dependencies of a system, and satisfying PEA nonlinear dynamics identification was achieved in the tested frequency and amplitude ranges. However, one limitation of vanilla RNN is that it is not capable of considering and understanding the system's long-term dependencies of the sequential input and hence cannot capture the correct dynamics of the system over the entire band of operating frequency [7], [8], especially with variable frequency drive inputs. Moreover, RNNs are susceptible to gradient vanishing, or exploding issues [9], [10] and hence difficult to avoid overfitting problem.

To overcome these issues in dynamic system identification using RNN, a deep learning RNN approach–the long short-term memory (LSTM)–has been proposed [7], [8]. The LSTM was proposed by Hochreiter and Schmidhuber in 1997 for the purpose of minimizing the negative effects of the Vanishing Gradient Problem [8]. In traditional RNN, the Vanishing Gradient Problem is mostly caused by repetitive multiplication with the recurring weight with the hidden states during the progress of back propagation [8], [11]. To overcome this issue, in the LSTM, a memory cell was added into the network neurons to establish another path to convey the time-relevant information to the next time instant. Moreover, this also makes LSTM able to take into account both long-term memory and short-term memory and recognize the importance of historical data in future prediction [7], [12]. LSTM regards each time step in the sequence of input with the consideration of the long-term temporal dependencies, so that it can possess a better ability to handle more complex nonlinear time series during learning, and thus be more capable for dynamic system identification of complex nonlinear systems, such as PEAs. However, one known issue is that LSTM is prone to overfitting [13], [14], especially when used for nonlinear system modeling [15]. This is because the nonlinear LSTM model is traditionally trained using single (input, output) time sequence pair which tends to be very long if the learning data must cover broad frequency ranges [7], [12]. At the same time, such a training process is extremely time consuming. This limits the application of LSTM in dynamic modeling of the nonlinear systems which are highly frequency-dependent, such as PEAs. Therefore, in order to take the advantages of LSTM in nonlinear system identification, particularly for PEA systems, we aim to develop a new learning mechanism to address these issues.

In this work, we propose a sequence-to-sequence LSTM

[1]R. Yin and [1]J. Ren are with the Department of Mechanical Engineering, Iowa State University, Ames, IA 50011, USA innocen3@iastate.edu,juanren@iastate.edu
[†] Corresponding author.

(namely, LSTMseq2seq) learning approach for accurate PEA system identification. The proposed LSTMseq2seq contains two layers of LSTM expanded time wise: LSTM encoder and decoder, respectively. Sinusoidal waves are used as the building blocks to generate the input (i.e., PEA drive input) training dataset. Instead of concatenating the all the sinusoidal waves together to generate a long time sequence as in [6], [7], LSTMseq2seq takes each single frequency training input as a sample and groups the samples into mini-batches. Then the training process is proceeded batch-by-batch. By adding the LSTM encoder and decoder, LSTM-seq2seq regards the entire input mini-batch as an entity and makes predictions for each sample in the mini-batch. Thus, the training efficiency can be greatly improved, especially when the dataset is large. The efficacy of LSTMseq2seq in terms of modeling accuracy and computation speed is demonstrated by applying it for PEA system identification and comparing its performance with that of vanilla RNN.

II. SYSTEM IDENTIFICATION

In this section, the limitations of traditional RNN (e.g., vanilla RNN) and existing LSTM for PEA system identification are discussed in detail. Then we provide the details of the proposed LSTMseq2seq, including the architecture and the process of constructing the training data set for LSTM sequence-to-sequence learning.

A. Traditional RNN and Its Limitation in PEA System Identification

RNN is a class of neural network in which the connections between nodes can create a cycle (i.e., feedback), allowing output from some nodes to affect subsequent input to the same nodes [2], [12]. This allows RNN to exhibit temporal dynamic behavior and is thus suitable for dynamic system identification.

Fig. 1 is a typical configuration of RNN structure, the vanilla RNN, for modeling a SISO system (e.g., a single axis PEA stage) [6]. It consists of three layers: an input layer, a hidden layer, and an output layer, which are shown in Fig. 1 as the columns of black dots/solid circles, circles, and dashed circles, respectively. The input layer always has one more node than the hidden layer to take in the input signal $u_{(r),k}$, where k is the sampling instant. The remaining nodes in the input layer take in the outputs of the hidden layer from the calculation of the previous sampling instant. y_r is the RNN output and $x_k = [x_{k,1}, x_{k,2}, ..., x_{k,N}]^T$ is the state vector. A fully connected network is applied between the input layer and the hidden layer. For nonlinear system identification, nonlinear functions, such as the hyperbolic tangent (tanh) function, can be used as the activation function in the hidden layer, and the output layer can be defined using a linear state output equation [6]. The entire RNN system in Fig. 1, can be described as:

$$\begin{aligned} x_{k+1} &= \tanh(W_1 x_k + B_2 + B_1 u_{(r),k}) \\ y_{(r),k} &= W_2 x_k + B_3 \end{aligned}, \tag{1}$$

where W_1, W_2, and B_1 are the $N \times N$ states weight, $1 \times N$ output weight, and $N \times 1$ input weight matrices, respectively.

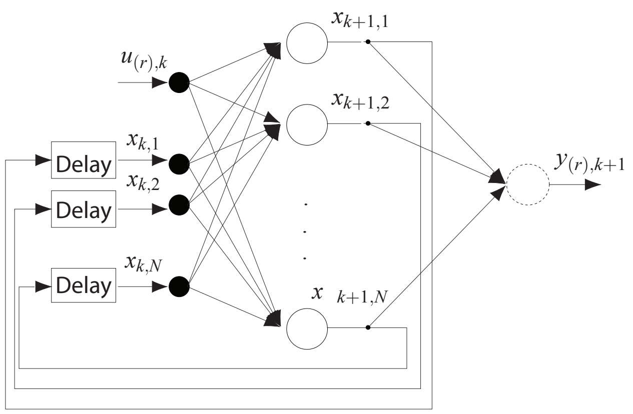

Fig. 1: Vanilla RNN structure.

$B2$ and $B3$ are scalar parameters. It is clear that the output of RNN, $y_{(r),k}$, only depends on the input $u_{(r),k-1}$ as the state vector x_k is computed internally through iterations at each sampling instant.

Suppose $U_{(ts)}$ and $Y_{(ts)}$ are the input and output training data sequences (e.g,. drive input and measured output of a PEA system) of the above RNN, respectively, and $Y_{(rts)}$ is the output generated by the RNN subject to $U_{(ts)}$, training of the RNN is to find the optimal values of the parameters parameters in Eq. 1 (W_1, W_2, B_1, B_2 and B_3) such that the error $Y_{(ts)} - Y_{(rts)}$ is minimized, i.e., $||Y_{(ts)} - Y_{(rts)}|| < \varepsilon$ for any infinitesimal positive ε. The optimization problem can be formulated as

$$\begin{aligned} \min_{W_1,B_2,B_1,W_2,B_3} \quad & J_{(r)} = ||Y_{(ts)} - Y_{(rts)}|| \\ \text{subject to}: \quad & x_{k+1} = \tanh(W_1 x_k + B_2 + B_1 U_{(ts),k}) \cdot \\ & Y_{(rts),k} = W_2 x_k + B_3 \end{aligned} \tag{2}$$

Typically, such an optimzation problem can be solved by the Back Propagation Approach (BPTT) [9]. However, the recurrent property of vanilla RNN makes the current output $y_{(r),k}$ be affected by all of the previous hidden states $\{x_k | k = 0, 1, 2, \cdots, L-1\}$, L is the sequence length. This brings a fundamental problem which is known as the Vanishing Gradient Problem [9]. Because the weights of each hidden layer node (also known as the recurring weights) are the same, the hidden states could vanish by repetitive multiplications with the recurring weights, especially if the weights are small. Similarly, the same mechanism could also affect the RNN in the back-propagation of the cost function ($J_{(r)}$) gradient. In other words, during the training BPTT process, the cost function gradient of errors will be decaying exponentially through hidden layers as time elapses. A mathematical explanation can be presented as follows:

$$\begin{aligned} \frac{\partial J_{(r)}}{\partial u_{(r)}} &= \sum_{1 \le t \le T} \frac{\partial J_{(r),t}}{\partial u_{(r)}} \\ \frac{\partial J_{(r),t}}{\partial u_{(r)}} &= \sum_{1 \le k \le t} \left(\frac{\partial J_{(r),t}}{\partial x_t} \frac{\partial x_t}{\partial x_k} \frac{\partial x_k}{\partial u_{(r)}} \right) \end{aligned}, \tag{3}$$

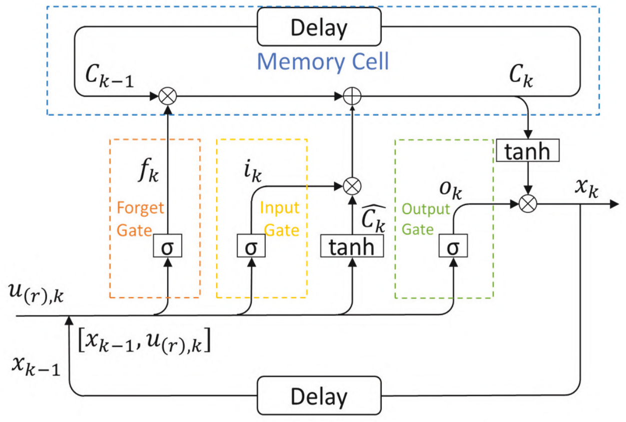

Fig. 2: LSTM layer structure.

where $\frac{\partial J_{(r),t}}{\partial x_t}$, $\frac{\partial x_t}{\partial x_k}$,and $\frac{\partial x_k}{\partial u_{(r)}}$ are regarded as the temporal component or temporal contributions [9]. Any of these variables having an initial value that is close to zero could lead to Vanishing Gradient Problem. Similarly, the gradient may explode if the weights of the hidden layers are big.

B. Long Short-term Memory (LSTM) Structure and Training

To overcome this problem, LSTM [8] was proposed by adding a memory cell to the structure. As shown in Fig. 2, the LSTM has four key elements to control the signal flow of the unit: input gate, output gate, forget gate, and memory cell, where $u_{(r),k}$ and $x_{(r),k}$ respectively denote the input $u_{(r)}$ and the hidden states $x_{(r)}$ at the sampling instant k. σ represents the gate nonlinear activation function. Thus, the gate outputs can be formulated as

Forget Gate:

$$f_k = \sigma(W_f[x_{k-1}, u_{(r),k}] + b_f)$$

Input Gate:

$$i_k = \sigma(W_i[x_{k-1}, u_{(r),k}] + b_i)$$

Output Gate:

$$o_k = \sigma(W_o[x_{k-1}, u_{(r),k}] + b_o)$$

Memory cell input and output:

$$\begin{aligned} \hat{C}_k &= \tanh(W_C[x_{k-1}, u_{(r),k}] + b_C) \\ C_k &= f_k C_{k-1} + i_k \hat{C}_k, \\ x_k &= o_k \cdot \tanh(C_k) \end{aligned} \tag{4}$$

where W_f, W_i, W_o, and W_C are the corresponding weights in different gates and operators, respectively. b_f, b_i, b_o, and b_C are the corresponding bias in each different gates and operators, respectively. In general, the activation function for the memory cell is the hyperbolic tangent, and σ for the three gates are sigmoid function.

The final output of the entire LSTM is computed by adding a fully connect network layer after the LSTM layer, mathematically presented as

$$y_{(r),k} = W_{(r)} \cdot x_k + b_{(r)},$$

where $W_{(r)}$ is the weight of the fully connected network and $b_{(r)}$ is the bias.

According to the equations above, for the gates with sigmoid function, the range of the gates is between 0 and 1. If the forget gate keeps a value of 1, the memory cell fully reserves its stored information. Therefore, LSTM can achieve memorize or discard stored information actively.

Training set generation and limits: In existing work [6], [7], both the vanilla RNN and LSTM constructed the training data using the same approach. The training input (i.e., the drive voltage to PEA), $U_{(ts)}$ was constructed by concatenation the single period sinusoidal signals defined using different (f, A) (frequency, amplitude) pairs, i.e., $S(f, A) = A[sin(2\pi ft + \frac{3\pi}{2}) + 1], t \in [0, \frac{1}{f}]$. Thus, the training input is

$$U_{(ts)} = \bigcup_{(f_i, A_i) \in \Omega} S(f_i, A_i), \tag{5}$$

where Ω is the set of the selected (f_i, A_i) pairs, and $\bigcup$ denotes concatenation. Then the corresponding PEA output is collected by measuring the PEA displacement subject to the input $U_{(ts)}$. The (f_i, A_i) pairs can be selected using the approach developed in [6].

Although both the frequency and amplitude-dependent behavior of PEA is considered in forming $U_{(ts)}$. In this case, one limitation of using the concatenated time series is that it may result in different training results due to the order of sinusoidal signal concatenation. Furthermore, concatenation can result in an extra long training time sequence if the size of Ω is big, and thus, make the training extremely time consuming and lead to possible overfitting issues. Therefore, we propose a sequence-to-sequence regression learning-based LSTM approach for the system identification of PEA for large data set handling and more efficient training.

C. Sequence-to-sequence LSTM System Identification

The architecture of the proposed LSTMseq2seq contains two layers of LSTM expanded time-wise, LSTM encoder and decoder, respectively, as shown in Fig. 3. The input sequence $u_{(r)}$ send its all elements $u_{(r),k}$ at each time step into the encoder layer. The output sequence $y_{(r)}$ is generated by the decoder layer, and all elements $y_{(r),k}$ are the output of the decoder layer for each time step. The basic mathematical logic of the LTSMseq2seq is derived from the aforementioned equations for both RNN and LSTM. Specifically, the LSTMseq2seq2 model can be formulated as

Encoder:

$$x_{e,(k)} = f(x_{e,(k-1)}, u_{(r),k})$$

Decoder

$$x_{d,(k)} = f(x_{d,(k-1)}, y_{(r),k-1}, s)$$

$$y_{(r),k-1} = g(x_{d,(k)})$$

Here, x_d and x_e are the hidden states in the encoder and decoder at sampling instant k, respectively. The unit function f used in both the encoder and decoder layers is a nonlinear activation function. It can be as simple as the RNN with an activation function like Eq. 1, or as complex

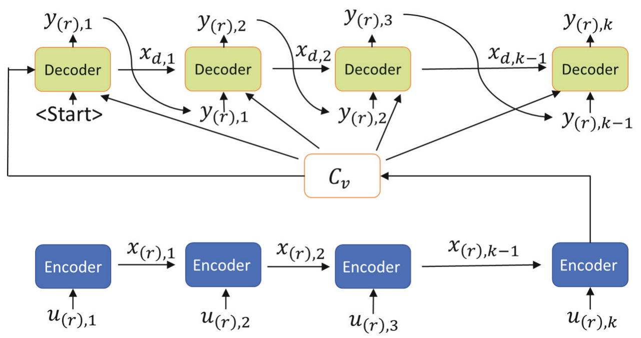

Fig. 3: Sequence-to-sequence LSTM structure.

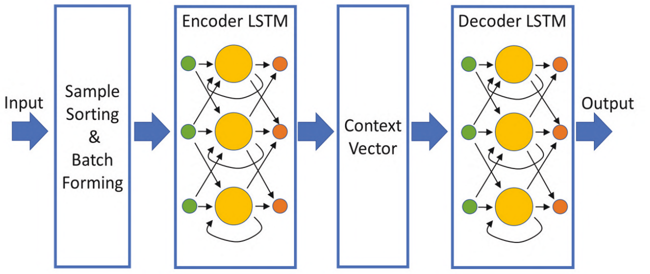

Fig. 4: Schematic architecture of the proposed LSTM-seq2seq.

as the aforementioned LSTM as presented in Eq. 4. It has been selected to use LSTM as the unit function $f(\cdot)$ in this study for better performance. The function $g(\cdot)$ represents the output layer of the decoder unit, and it can be varied with different complexity compare to $f(\cdot)$. C_v is the final state of the encoder layer as well as the initial state of the decoder layer, which is also known as the context vector or the summary information since it memorizes the information of the hidden state and the memory cell for the entire encoder layer [16] [17].

Sequence-to-sequence training: To train the LSTM-seq2seq, we use multi-period sinusoidal wave S_i as the building blocks. Each S_i defined by a $(f_i,\, A_i)$ pair is considered as a sample. The training input $U_{(ts)}$ contains K_s samples with different $(f_i,\, A_i)$ combinations, where the (f_i,A_i) set Ω can be selected using the aforementioned approach [6]. Different from previous work which use concatenated sinusoidal inputs, each sample (i.e., S_i and its corresponding measured PEA output are considered as an independent time sequence in the training input $U_{(ts)}$ and output $Y_{(ts)}$, respectively. Then the samples of the entire training set $(U_{(ts)},\, Y_{(ts)})$ are sorted based on the length and divided into mini-batches evenly with a predefined batch size n (i.e., n samples per mini-batch, thus K_s/n mini-batches in total). The sorting process ensures that the samples with similar lengths are grouped in the same mini-batch such that the amount of padding in each mini-batch is minimized. Then the LSTMseq2seq PEA model is trained batch-by-batch. It can be trained with the back-propagation method [9], [15], which uses a gradient descent iterative optimization algorithm to obtain trainable parameters. A pre-chosen epoch number λ is defined to specify the number of complete pass of each mini-batch to the model. Thus, the iteration number of the entire training is $\lambda K_s/n$. The entire architecture of the proposed LSTMseq2seq for PEA system identification is illustratively shown in Fig. 4.

III. EXPERIMENT RESULT AND DISCUSSION

To validate and demonstrate the proposed system identification approach, the proposed LSTMseq2seq was implemented to model the nonlinear dynamics of a PEA stage (Nano-OP30, Mad City Labs). The voltage input and the corresponding PEA displacement were collected using a data acquisition system (NI PCIe-6353, National Instruments) installed in a desktop workstation (Intel Xeon W-2125, RAM 32GB). The LSTMseq2seq model was generated using MatLab Simulink (MathWorks, Inc) on the same workstation. The experiment setup is shown in Fig. 5. The sampling rate was set to 10 Hz.

Training set $U_{(ts)}$ generation: To generate the training set $U_{(ts)}$, 10000 points were randomly generated in the $f-A$ plain for the frequency range of 20-375 Hz and PEA input amplitude range of 0-1.5 V (blue dots in Fig. 6). Then Ω (the set of 500 $(f_i,\, A_i)$, red dots in Fig. 6) were selected using the k-mean algorithm in [6], i.e., $K_s=500$. To test the proposed LSTMseq2seq in handling large data sets, S_i samples with multiple periods (3 and 10) were generated to form $U_{(ts)}$. The corresponding PEA output of each S_i was measured to form the output training set $Y_{(ts)}$. The entire training set $(U_{(ts)}, Y_{(ts)})$ was randomly split into 90% training and 10% validation. For comparison, the concatenated training set using the same S_i samples were generated for each tested period case (see Fig. 7 as an example for the 3-period case) and applied to train a vanilla RNN with the same training-validation data split ratio.

LSTMseq2seq training: The proposed LSTMseq2seq model was built with 10 hidden units in each encoder/decoder layer, respectively. For the total $K_s=500$ samples, the mini-batch size was set as $n=20$. Thus, 25 mini-batches in total. The epoch number was investigated in terms of validation accuracy. As shown in Fig. 8, for the chosen LSTMseq2seq structure, the validation accuracy is small and steady (i.e., does not show further improvement) when the epoch number is beyond ~ 25. Thus, the epoch number was selected as $\lambda=32$ for the rest of the tests.

LSTMseq2seq vs. RNN training performance: The training performance in terms of validation accuracy and training time of LSTMseq2seq and RNN are compared in Table I. For the two sets of training data in 3 periods and 10 periods, the validation error during training of LSTMseq2seq is small enough for both training sets, and even for the 10-period training set, the training time was still less than 5 min. However, RNN takes more than twice of the time

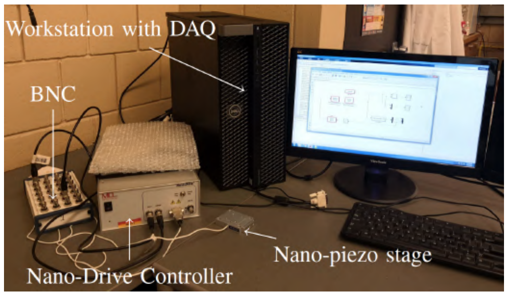

Fig. 5: Experimental setup.

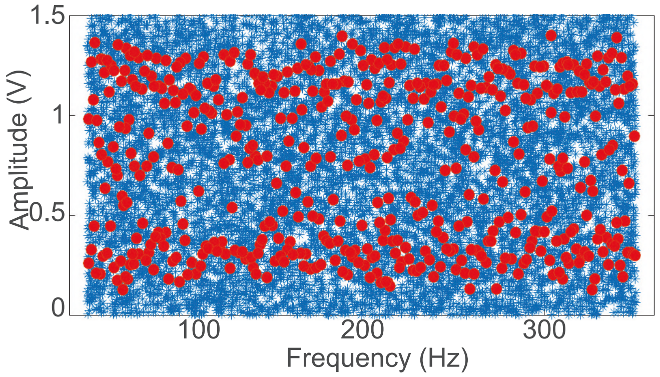

Fig. 6: Ω (set of (f_i, A_i) pairs) selected for generating the training input $U_{(ts)}$. Red dots represent the selected (f_i, A_i) pairs by the k-means algorithm.

TABLE I: Training performance comparison of vanilla RNN and LSTMseq2seq.

$U_{(ts)}$ size	3 periods		10 periods	
Approach	LSTMseq2seq	RNN	LSTMseq2seq	RNN
Validation RMSE	0.027	0.054	0.026	0.045
Training time (min)	2.5	5.5	5	19

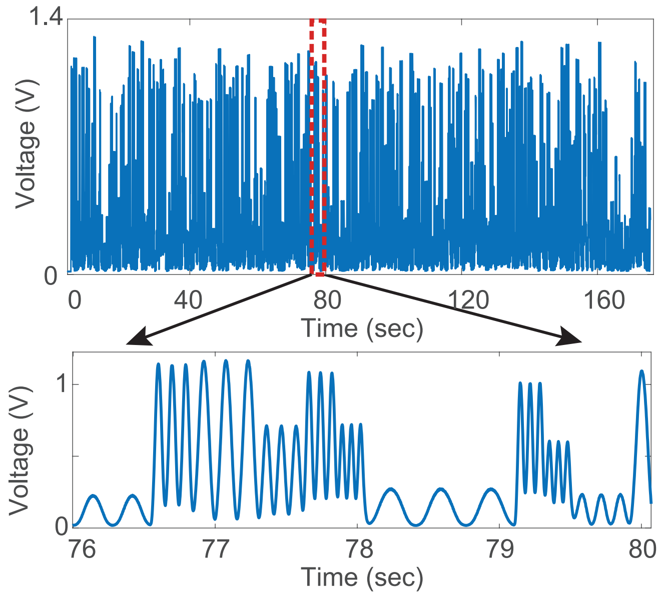

Fig. 7: The concatenated 3-period S_i time series according to the selected (f_i, A_i) pairs in Fig. 6 for RNN training.

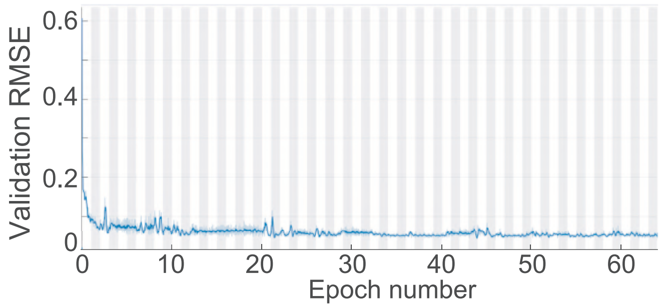

Fig. 8: Training validation accuracy vs. epoch numbers of LSTMseq2seq.

to finish training yet with nearly twice of the validation error, comparing to LSTMseq2seq. Therefore, the comparison demonstrated that the proposed LSTMseq2seq achieves more accurate and efficient training compared to RNN, especially for large data sets.

LSTMseq2seq vs. RNN PEA displacement prediction: To further demonstrate the system identification accuracy, we also tested the accuracy of the LSTMseq2seq and vanilla RNN (both trained with the training sets with 500 3-period S_i samples) in predicting the PEA displacement (i.e., PEA output). Specifically, sinusoidal and triangle signals in 30Hz, 60Hz, and 120Hz were used to measure the actual PEA displacements, respectively. To get convincing results, it was our intention to select these three frequencies because they did not overlap with those in the training set. The prediction performances of the RNN and the LSTMseq2seq are compared in Table II. The RMS prediction error, ξ_{rms} was quantified as

$$\xi_{rms} = \frac{||Y_m - Y_p||_2}{||Y_m||_2} \times 100\%, \tag{6}$$

where Y_p and Y_m are the model predicted and actual measured PEA displacements, respectively.

Clearly, the proposed LSTMseq2seq is more accurate in predicting the PEA displacement for all three frequencies. For detailed comparison, the prediction result for the 30Hz triangle input in the time domain is shown in Fig. 9, in which it can be easily seen that the RNN prediction fluctuates above and below the actual PEA output while the LSTMseq2seq prediction was able to reduce the prediction by more than 50%. Moreover, Fig. 10 shows the comparison of the two neural networks in modeling the nonlinear PEA hysteresis at 30 Hz vs. the experimentally measured result. It is obvious that the LSTMseq2seq is more accurate (closer to the actual PEA curve) than RNN in modeling the PEA nonlinear hysteresis.

Therefore, according to the results, it is convincing that the proposed LSTMseq2seq2 is more accurate and computationally efficient in dynamic system identification and more capable of handling large data sets. However, restricted by the sampling rate of the current hardware, training data size at frequency range is limited; thus the improvement of the

TABLE II: Prediction Performance Comparison between vanilla RNN & LSTMseq2seq.

PEA input waveform	Sinusoidal			Triangle		
Frequency (Hz)	30Hz	60Hz	120Hz	30Hz	60Hz	120Hz
$\xi_{rms-LSTMseq2seq}$ (%)	2.9	2.7	1.4	3.4	3.6	1.9
$\xi_{rms-RNN}$ (%)	6.4	4.9	4.8	7.2	5.9	3.2

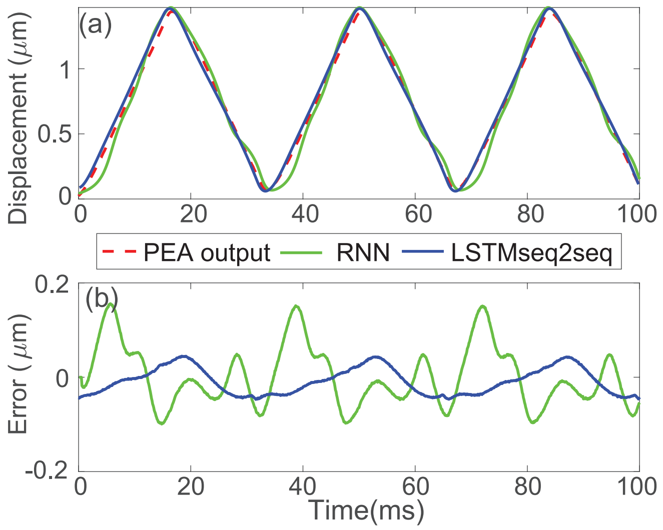

Fig. 9: Comparison of the PEA output prediction for 30 Hz triangle drive input: (a) LSTMseq2seq and RNN predicted PEA output, and (b) the corresponding prediction error (compare to the actual PEA output).

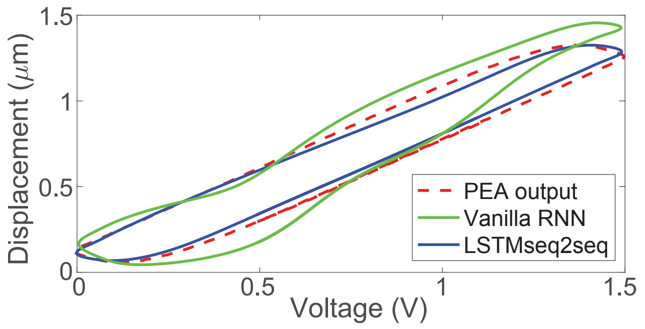

Fig. 10: Comparison of the PEA hysteresis at 30 Hz modeled using LSTMseq2seq and RNN vs. the experimentally measured result.

proposed LSTMseq2seq is not at significant as the lower frequency cases. As for future work, we will investigate the options and use faster hardware, such as the field-programmable gate array (FPGA), and integrate LSTMseq2seq with a real-time controller for high-speed nano-positioning applications.

IV. CONCLUSION

In this paper, a new concept, an LSTM sequence to sequence learning structure for dynamic system identification of PEA has been proposed. This approach takes advantages of both LSTM and sequence-to-sequence learning simultaneously in time series processing: the former can eliminate the common learning issues in traditional RNN and the latter can increase the learning efficiency significantly. Therefore, the proposed system identification model is capable of handling large data set in training with superior accuracy and computation efficiency. This has been demonstrated by the experiment results in system identification and the displacement prediction of a PEA stage. In addition, the proposed LSTMseq2seq architecture can be adapted to improve the system identification accuracy for any dynamic system that can be modelled using RNN.

ACKNOWLEDGMENT

This work was supported by the National Science Foundation (NSF) (CMMI- 1751503) and Iowa State University.

REFERENCES

[1] K. Mollaeian, Y. Liu, S. Bi, and J. Ren, "Atomic force microscopy study revealed velocity-dependence and nonlinearity of nanoscale poroelasticity of eukaryotic cells," *Journal of the mechanical behavior of biomedical materials*, vol. 78, pp. 65–73, 2018.

[2] S. Xie and J. Ren, "Iterative learning-based model predictive control for precise trajectory tracking of piezo nanopositioning stage," in *2018 Annual American Control Conference (ACC)*, pp. 2922–2927, IEEE, 2018.

[3] Y. Tian, D. Zhang, and B. Shirinzadeh, "Dynamic modelling of a flexure-based mechanism for ultra-precision grinding operation," *Precision Engineering*, vol. 35, no. 4, pp. 554–565, 2011.

[4] F. Qin, D. Zhang, D. Xing, D. Xu, and J. Li, "Laser beam pointing control with piezoelectric actuator model learning," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 3, pp. 1024–1034, 2017.

[5] L. Cheng, W. Liu, Z.-G. Hou, J. Yu, and M. Tan, "Neural-network-based nonlinear model predictive control for piezoelectric actuators," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 12, pp. 7717–7727, 2015.

[6] S. Xie and J. Ren, "Recurrent-neural-network-based predictive control of piezo actuators for trajectory tracking," *IEEE/ASME Transactions on Mechatronics*, vol. 24, no. 6, pp. 2885–2896, 2019.

[7] M. S. Patil, B. Charuku, and J. Ren, "Long short-term memory neural network-based system identification and augmented predictive control of piezoelectric actuators for precise trajectory tracking," *IFAC-PapersOnLine*, vol. 54, no. 20, pp. 38–45, 2021.

[8] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[9] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*, pp. 1310–1318, PMLR, 2013.

[10] S. Hochreiter, "Untersuchungen zu dynamischen neuronalen netzen," *Diploma, Technische Universität München*, vol. 91, no. 1, 1991.

[11] A. Rehmer and A. Kroll, "On the vanishing and exploding gradient problem in gated recurrent units," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 1243–1248, 2020.

[12] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.

[13] Y. Wang, "A new concept using lstm neural networks for dynamic system identification," in *2017 American control conference (ACC)*, pp. 5324–5329, IEEE, 2017.

[14] S. Ookura and H. Mori, "An efficient method for wind power generation forecasting by lstm in consideration of overfitting prevention," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 12169–12174, 2020.

[15] J. Gonzalez and W. Yu, "Non-linear system modeling using lstm neural networks," *IFAC-PapersOnLine*, vol. 51, no. 13, pp. 485–489, 2018.

[16] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[17] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in neural information processing systems*, vol. 27, 2014.