International Journal of Modern Manufacturing Technologies
ISSN 2067–3604, Vol. XVI, No. 1 / 2024
https://doi.org/ 10.54684/ijmmt.2024.16.1.24

THE MODELLING OF A HYSTERESIS GRAPH OF PIEZOELECTRIC ELEMENTS USING DEEP LEARNING BIDIRECTIONAL LSTM

Fawwaz Al-Inizi, Marek Płaczek, Andrzej Wróbel, Jacek Harazin

Silesian University of Technology, Konarskiego 18A, 44-100 Gliwice, Faculty of Mechanical Engineering, Department of Automation of Technological Processes and Integrated Manufacturing Systems

Corresponding author: Fawwaz Al-Inizi, fawaz.salaheldeen@gmail.com

Abstract: The phenomenon of hysteresis is an integral part of dynamic systems in many fields of science such as physics, chemistry, biology and many more. It describes an inherent dependence of a system state based on the history of varying number of its previous states. Hysteresis can manifest as a dynamic lag between an input signal and an output system behaviour, which depends on the degree of that system dy-namics. Modelling systems containing hysteresis is a challenging mathematical task given their highly non-linear behaviour. This paper discusses and develop a deep learning model using bidirectional LSTM (long short-term memory) for predicting voltages necessary to stimulate a piezoelectric element to produce displacements in order to cancel or minimize vibrations. The predicted voltages rely on given displacements and time domain of the initial noise input. This noise input can then be amplified to match the resonance frequency of another piezoelectric element to generate the maximum voltage capable by this later piezoelectric element. This sinusoidal voltage then travels to a piezoelectric actuator to generate displacement that can cancel the initial noise. The model resulted a coefficient of determination score of 0.99983, a loss score of 0.0092 and MSE (mean squared error) of 8.5568e-05. Created model has proven that machine learning is a viable method for hysteresis modelling and can be further improved with increased input data availability and further investigation into different deep learning algorithms.
Key words: deep learning, piezoelectric element, LSTM, displacements, voltages, neural networks, model evaluation, hysteresis graph.

1. INTRODUCTION

Smart materials and structures are very important elements of modern technology, industry as well as common people life. Their development is a parallel process of introducing inventions in material science, technology as well as control systems. They can function as smart elements when integrated with artificial intelligence and further accelerate their development process [1-3]. What is more, such kind of tools are very important elements of the Industry 4.0 idea whose aim is to create smart factories and optimize materials, technology, energy consumption and so on [4-6]. Issues of modelling, research and application of piezoelectric transducers were presented in other authors' publications. Tools for modelling and testing of classical and non-classical piezoelectric transducers were presented among others in [7-10]. Presented tasks are elements of modern devices and systems that could be developed using neural networks both, on designing and application stage.
Recurrent neural networks (RNN) are the main blocks of bidirectional LSTM. They are specialized in processing long range of sequences (vectors) from $x_{(1)},\dots, x_{(t)}$, where t is time-step. This is particularly important in our case since we are dealing with long ranges in the domains of time, voltage, and displacement, and where other neural networks like convolutional networks (CNN) might fail in achieving better results. RNN also allows to share parameters (weights) across the training period (across different time index), which helps generalizing the prediction process rather than building different weights for each time index (time steps). I.e., “Each member of the output is a function of the previous members of the output. Each member of the output is produced using the same update rule applied to the previous outputs” [11]. Because RNN alone process data from the past (along with the present time) to predict the future, implementing bidirectional RNN process past and future data using two RNNs, where one goes forward in time starting from start of the sequence, and another RNN goes backwards starting from end of the sequence.
However, to strengthen the core of RNN further, the LSTM was proposed to processes “long-term dependencies” [12], which describes the pattern interconnections between data, and mimicking a memory model for long sequences. Hence, several applications adopted LSTM including speech recognition, acoustic modelling, trajectory prediction, sentence embedding, and correlation analysis [12]. LSTM has also several variants including

LSTM without a Forget Gate, LSTM with a Forget Gate, LSTM with a Peephole Connection, Gate Recurrent Unit, and Minimal Gated Unit [12].
Because the hysteresis effect in piezoelectric elements is non-linear and has long sequence range, combining both bidirectional technique and LSTM together serve a promising system.
Applying neural networks in modelling PEA (piezoelectric actuators) hysteresis is not entirely new and previous promising attempts have been performed. This includes merging LSTM and NARX neural networks (another RNN-based neuron for non-linearity modelling), where both integrated with an "adaptive weighted hybrid hysteresis mode" [13]. Both LSTM and NARX here used different input data: input voltage frequency and output angle for LSTM, and time sequence for NARX. Although the hybrid system generates good accuracy with small input voltage frequency and gradually declines with high voltage frequency [13], it requires training two separate models (LSTM and NARX), and hence adding complexity and resources. Physics-based method in noise cancellation are also available.
Another technique is using ANN (artificial neural network; a contributor to RNN) to model relationship between frequency and displacement under a range of voltage excitation, where the input layer consists of voltage and frequency, and output consists of displacements [14]. The model performance indicates a consistency from 90.3 to 98.9%, with a regression value of 0.999 [14]. However, we believe ANN is not sufficient for our hysteresis modelling since ANN is not recurrent (the output value of a time-step is not used as input for the next input function), nor weights are shared along the whole sequence, unlike directional LSTM.
Modelling hysteresis with only LSTM has also been performed by Liu, Zhou, and Huo [15], using voltage as input and displacement as output. However, our model uses a bidirectional mechanism, and its input is multi-feature, using displacement and time domains as input.
Apart from hysteresis modelling, deep learning is used in other applications, such as weather forecasting, speech recognition, real time computer vision analysis, chat bots, deep fakes, recommendation systems, robotics, image colouring, and in healthcare.
Consequently, the development of deep learning modelling in relation to piezoelectric transducers and their applications seems to be a promising research area. Research of their applications in structural health monitoring [16], passive or active vibration damping [17-19] or energy harvesting systems [20,21] done so far, proved their wide range of possible applications. The presented work is the result of further research on the modelling of piezoelectric transducers, in which the authors undertook the use of deep learning model for predicting voltages necessary to stimulate a piezoelectric element to produce displacements in order to cancel or minimize vibrations. This idea is a part of the aforementioned development of modern technology.

2. PROCEDURE

Piezoelectric actuators can be used to actively cancel out noise through destructive interference. Depending on the me-dium through which vibrations are transferred, piezoelectric elements can take the form of plates, stacks or foils. The latter can be used to efficiently create interfering audio waves which cancel out with unwanted noise from the background. Such technology exists al-ready in modern headphone applications. New potential implemen-tations could use piezoelectric plates or stacks to cancel out vibra-tions between machines and their foundations or any sensitive equipment, serving as a direct separator.
Vibration cancellation requires precise synchronisation of inter-fering waves with the source to maximise the effect of destructive interference. Hysteresis becomes a very important factor in the proper tuning of active dampers. To illustrate how the model could be implemented loosely, a simple noise cancelation system (Fig. 1) could be visualized for general purpose applications.

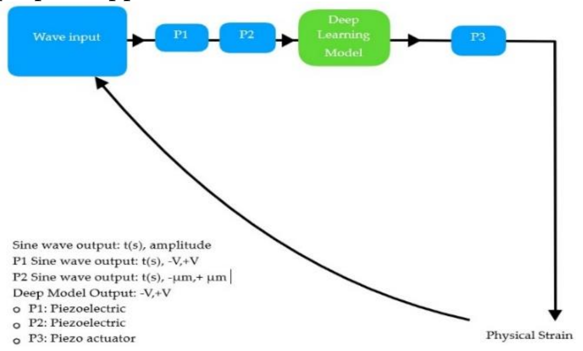

Fig. 1. Simplified noise cancelation system with deep learning model integrated

The wave input could be any vibrational wave of varying frequency and amplitude, such as background noise or machinery vibration. The time domain and amplitude of the input wave is recorded using sensors. The wave signal then acts as input to the first piezoelectric element (P1). Such physical wave will stimulate
P1 as a mechanical force, inducing P1 to generate a sinusoidal voltage due to the hysteresis effect piezoelectric elements possess. This voltage will then travel to piezoelectric element P2, stimulating it to produce mechanical force this time.
The new displacement wave will then act as an input, along with the time domain this displacement changes with (since time domain is part of the compression and expansion mechanism a piezoelectric element experience, hence hysteresis is generated), will act as input to the deep learning model, and predict
the next voltages based on both time domain and displacement. These predicted voltages will act as a charge on the piezo actuator P3, generating physical strain to cancel the first initial wave input. A simple diagram of the deep learning model and its architecture can be illustrated as in (Fig. 2).

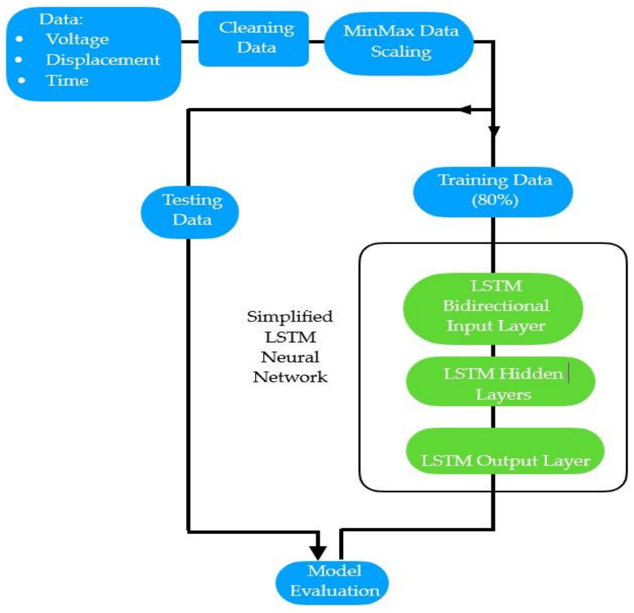

Fig. 2. Simplified deep learning model architecture

Raw experimental data is first recorded using suitable sets, where voltage, displacements, and time domain are the primary parameters (The more parameters are needed for a deep learning model, the more complex and dimensionally rich the model becomes). At this point, outlier regions are expected in the dataset, and hence cleaning it is required. In other cases, with different datasets of many dimensions, only suitable features that are distinct in the dimension-sphere are selected and others are discarded (dimensionality reduction algorithms can generate such outcome).
Next, data is scaled in order set default range where deep learning algorithms can work on. This is important so as not to put more weights into certain measurement unit than another. The min-max scaling algorithm is applied on the whole dataset.
The new scaled data is then divided into training and testing sets. The percentage for the test set vary from model to model, yet the recommended is about 20% for test data. This test data will not be included during the training process and will be used only to evaluate the model later. The training set is used for the training process, and a validation set is taken from the training set for better evaluation performance.
The training set is later fed into the model architecture, where the training capacity of the model is predefined. Model architecture development is not an easy process, and several trial and error of model parameters need to be tweaked in order to hit the best performance. Hence model iterations, or hyper-parameter tuning might be required (This model did not utilize hyper-parameter tuning due to lacking hardware capacity). After training completes, the model is saved along with its weights. The model can be then used for evaluation purposes to measure its performance, using accuracy, mean squared error, and loss. If the model does not meet sufficient requirements, then restructuring the architecture is needed, or perhaps to gather more data.
A hybrid model can be designed as well, where multiple models built on top of different deep learning algorithms are integrated together to form one model. This can be complex task and resource consuming and far away what this project aims: to develop a single efficient model with the least complexity, and able to be integrated into other system if possible (the integration into real-world systems like machinery has not been investigated in this project).

3. LONG SHORT-TERM MEMORY ARCHITECTURE

An experimental dataset of 50,001 samples [22] has been obtained in order to develop an LSTM Bidirectional model using deep learning. The LSTM neural network structure is presented in (Fig. 3). Long short-term memory (LSTM) neural networks are a special kind of recurrent neural networks (RNN) that have memory-based neurons able to remember large sequences that represent complex non-linear patterns [23].
The long-term memory is called the cell state, where previous information is stored within it due to the recursive nature of the cells. The forget gate placed below the cell state is used to modify the cell states. The forget gate output determines which information is stored or forgotten by outputting 1 to keep information or multiplying 0 to a position in the matrix.
The cell stores the data and processed by three gates, where each has its own neural network and is trained according to its task [24]. The input gates determine which information should enter the cell states, and the output gate tells which information should be passed on to the next hidden state [23].

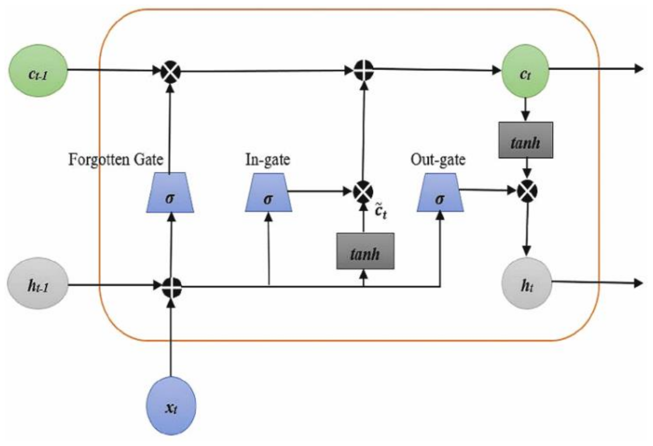

Fig. 3. LSTM neural network structure [13]

In the input gate, new data (along with previous predictions) enter where a vector of values is generated representing the possibilities. The vector values will range between [0,1] due to the logistic sigmoid activation function in the input gate, where the highest value is more probable to be the next prediction. “The generated vector is then added with the viable possibilities, previously stored in the cell, to produce a collection of possibilities; their values may range from less than -1 to more than 1” [24].

$$i_{t-1} = W_i x_t + R_i y_{t-1\,undefined} + p_i \odot c_{t-1\,\text{undefinded}} + b_i \tag{1}$$

$$i_t = \sigma\,(i_{t-1}) \tag{2}$$

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{3}$$

Where i_t is input gate variable, W_i is the input weights, R_i is the recurrent weight, p_i is the peephole weights, b_i is the bias weight and σ is the logistic sigmoid. The symbol ⊙ is for the multiplication of two-vector.
The forget gate task is the removal of nonessential data in the cell state. This is done when the gate’s two inputs receive new data at time t and previous predictions to be then multiplied by eight matrices. The output then goes through an activation function that gives 0 to forget data, or output 1 to store data in the future. “The result will be multiplied with the possibilities that are collected from the input gate and the cell. The useful possibilities will be stored in the cell” [24].

$$f_{t-1} = W_f x_t + R_f Y_{t-1} + p_f \odot c_{t-1} + b_f \tag{4}$$

$$f_t = \sigma(f_{t-1}) \tag{5}$$

Where f_t is output gate variable, W_f is the input weights, R_f is the recurrent weight, p_f is the peephole weights, b_f is the bias weight and σ is the sigmoid function. The symbol ⊙ is for the multiplication of two-vector.

Output Gate (selection gate): The result of the output gate is based on the new data along with previous predictions. Then multiplying the output gate result and normalized possibilities given by the cell and input gate will produce the final prediction. The tanh activation function will then normalize the stored possibilities which may range from -1 to 1 [24].

$$o_{t-1} = W_o x_t + R_0 Y_{t-1} + p_o \odot c_t + b_o \tag{6}$$

$$o_t = \sigma(o_{t-1}) \tag{7}$$

Where o_t is output gate variable, W_o is the input weights, R_0 is the recurrent weight, p_o is the peephole weights, b_o is the bias weight and σ is the logistic sigmoid. The symbol $\odot$ is for the multiplication of two-vector.
The cell formula is represented as follows:

$$c_t = z_t \odot i_t + c_{t-1} \odot f_t \tag{8}$$

The input block for cell state (Fig. 3) is represented as follows:

$$z_{t-1} = W_z x_t + R_z y_{t-1} + b_z \tag{9}$$

$$z_t = g(z_{t-1}) \tag{10}$$

Where z_t is the input block variable, W_z is the input weights, R_z is the recurrent weight, b_z is the bias weight, and $g(z)$ is the hyperbolic tangent activation function:

$$g(x) = \tanh(x) \tag{11}$$

The output block for cell state is represented as follows:

$$y_t = h(c_t) \odot o_t \tag{12}$$

Where y_t is the output block variable, and h is the hidden state as $h_t = o_t \odot \tanh(c_t)$.
Bidirectional LSTMs is based on bidirectional RNN [25], processing long data sequences in both forward and backward directions with two separate hidden layers. In BDLSTMs, the output layer is connected to both hidden layers. Bidirectional networks outperform unidirectional networks in fields such as phoneme classification and speech recognition [25]. (Fig. 4) illustrates an28nfoldding BDLSTM structure containing a forward LSTM layer and backward LSTM layer.

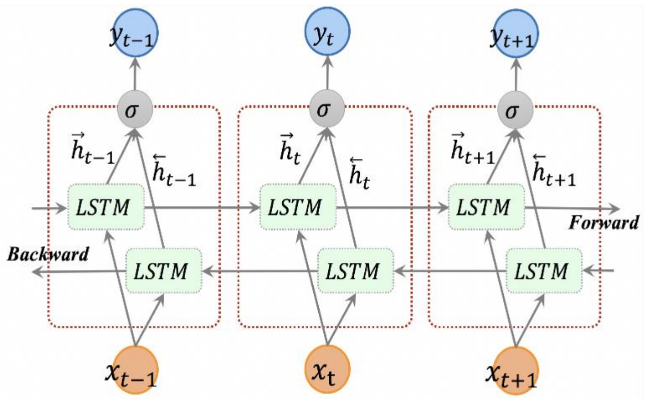

Fig. 4. Simplified bidirectional LSTM System [25].

$\overrightarrow{h_t}$ is the forward layer output such that its input sequence is time $t-1$ to time $t+1$, while $\overleftarrow{h_t}$ is the backward layer output where its input is time $t+1$ to $t-1$ [13]. y_t is the output vector produced by BDLSTM layer [9]:

$$y_t = \sigma(\overrightarrow{h_t}, \overleftarrow{h_t}) \tag{13}$$

where σ is an activation function. The final output of BDLSTM layer's can be represented by vector Yt = [Yt-n, … , Yt-1], in which the predicted voltage (Yt-1) is the last element [25].

4. MODEL IMPLEMENTATION

After understanding how the LSTM and BILSTM algorithms work, we can explain how the model was built (see appendix 1 for the Python source code). Data is cleaned (if necessary) to remove outliers or noise. For example, we can observe a chunk of data that is a noise for the overall shape of hysteresis plot on (Fig. 5).
The dataset is cleaned by taking only the measurements recorded after time of 5s.
Then data is scaled by taking MinMax for time domain, voltage, and displacement vectors, where the values for each vector will be scaled within [-1,1]:

$$x\ scaled = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{14}$$

After cleaning and scaling data, we obtain the following plot presented on (Fig. 6).

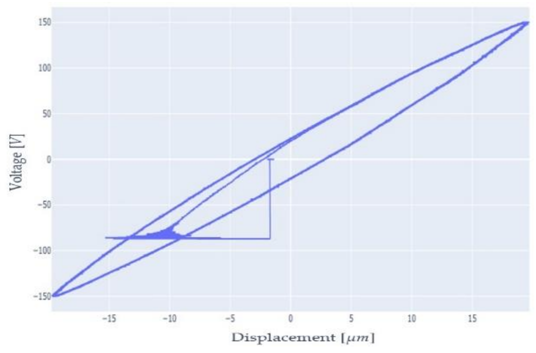

Fig. 5. Raw collected data at 150V (pre-clean).

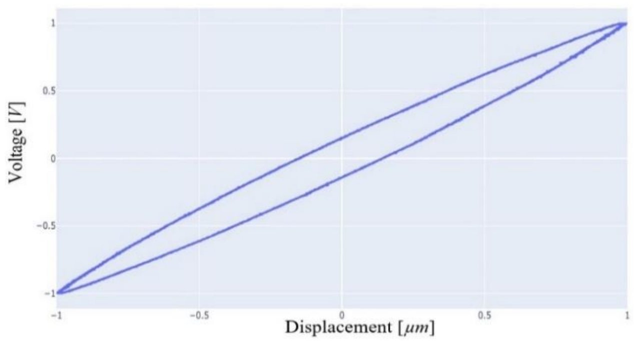

Fig. 6. Cleaned and scaled data.

Next step is creating sequences for our training input, which is time and displacement in this case. Here we consider many-to-one, multi-feature relationship of input and output, where sequences contain time and displacements are used to predict voltage. In our case, a time-series of 10 sequences will be chosen. For example, let us assume we have two vectors each represent a feature (dimension):
#two vectors representing two features.
[3 6 9 12 15 18 21 24 27 30 33 36 39 42 45 48 51 5457 60 63 66 69 72 75 78 81 84 87 90 93 96 99 102 105 108 111 114 117 120 123 126 129 132 135]
[5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 105 110 115 120 125 130 135 140 145 150 155 160 165 170 175 180 185 190 195 200 205 210 215 220 225]
Now, the two vectors will be reshaped into 15 samples, 3 time-steps, and two features (Table 1).

Table 1. Multi-feature samples example

[3 5] [6 10] [9 15] [12 20] [15 25] [18 30] [21 35] [24 40] [27 45] [30 50] [33 55] [36 60]	[39 65] [42 70] [45 75] [48 80] [51 85] [54 90] [57 95] [60 100] [63 105] [66 110] [69 115] [72 120]
[75 125] [78 130] [81 135] [84 140] [87 145] [90 150] [93 155] [96 160] [99 165] [102 170] [105 175] [108 180]	[111 185] [114 190] [117 195] [120 200] [123 205] [126 210] [129 215] [132 220] [135 225]

In this example the n_steps are 10, meaning time steps of 10 will be used to predict a voltage. Next, the data is split into training and test datasets, where 80% of the dataset goes to the training set and remaining 20% goes to the test set.

The next step is building the stacked BILSTM architecture. In the input layer of the BILSTM the Leaky Relu activation function was used, and 300 cells were chosen. Then the hidden layers were added, followed by full dense connected layer. Finally, an output layer with tanh activation function is implemented.
Once the last line is executed, the training session start starting from epoch 1 and gradually increasing till epoch 300, unless it is terminated if found there is no model improvement based on the early stopping technique. The training time varies based on several factors such as the complexity of the model architecture, number of epochs, and computer hardware. The model was trained on cloud GPU (graphical processing unit) rather on CPU (central processing unit) provided by the cloud based Kaggle service.

5. MODEL EVALUATION

There are several methods to evaluate the model, starting from Coefficient of determination R2. The standards for using Coefficient of determination R2 are:

1. R2 = 1 (best value)
2. R2 is between (0 - 1) closer to 1 the better
3. R2 = 0 (the model is mean based model)
4. R2 < 0 (Not good)

Coefficient of determination was used on the test data and its prediction result. This resulted in a score of 0.99983. We can evaluate the model further via MSE (mean squared error) and loss metrics (root mean squared error) on the test data.
This resulted into loss: 0.0092 and MSE (mean squared error): 8.5568e-05. In plotting our model and comparing it to the real values we can observe a slight difference on some regions. No model is fully accurate and hence this model is naturally prone to errors shifts.

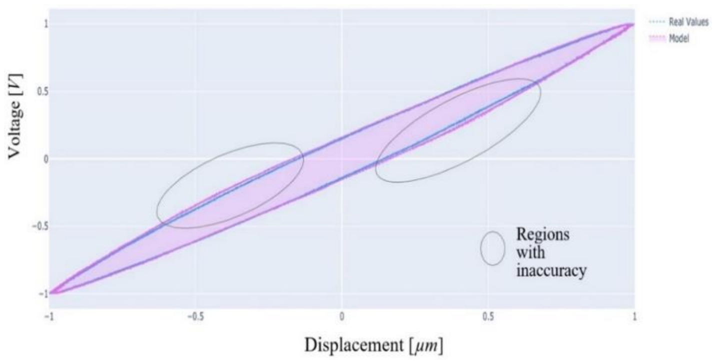

Fig. 7. Real values model vs model plot

In fact, obtaining a fully 100% accurate model may fail in data generalization when encountering new sets of data that differ from the training data. Hence, small errors can enhance the model to deter the model from overfitting (strict learning) to the training dataset. To plot, first some additional data processing is required.
The dotted regions in (Fig. 7) indicate some inaccuracy in the model when compared with the experimental model. This is shown in the two regions in the x-axis ≈[-0.6 ,- 0.2]µm, and ≈[0.2, 0.6] µm, with respect to their y-axis points. The rest of the model varies between either overlapping with the experimental model or is slightly offset in small margin. The error difference can also be plotted between experimental voltage values and predicted voltages. The straight line in the experimental data (Fig. 8) is the plot of experimental voltage on x and y axes simultaneously, which creates a straight line considering the difference between any value on x-axis and its matching pint on y-axis is 0. The model plot is for predictions (ŷ) against experimental voltage. The gap in the model plot indicates the difference between real and predicted values, hence it is error.
This will produce (Fig. 8) below with equal axis scale range [-1, 1]. This error gap is noticeable in the ŷ-axis range of ≈[-0.5, 0.5], with respect to y-axis coordinates, where this range overlaps also with (Fig. 7) error range.

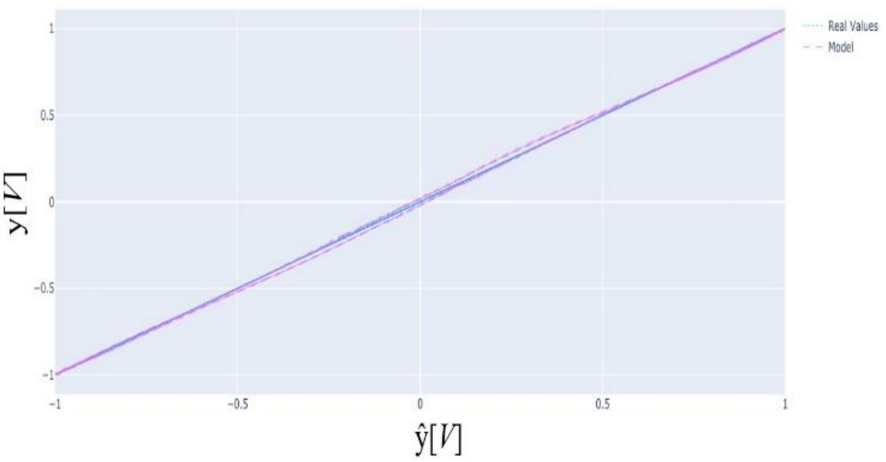

Fig. 8. Error difference between real voltage values and predicted voltages

Finally, we can plot the learning rate curve along with the validation curve:

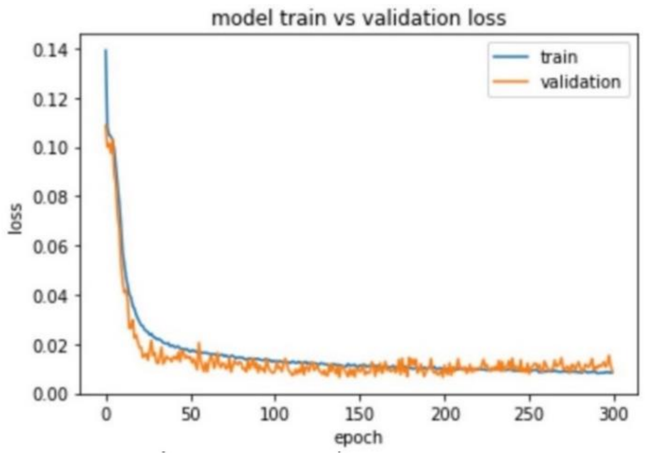

Fig. 9. Main model learning rate curve

Figure 9 gives an important indicator if the model is overfitting or undercutting by plotting the learning curve for both training set and validation set. The learning curve for the training set helps to indicate how well is the model is training, while for the valuation set helps to indicate how well is the model able to generalize. The Loss metric has been used to indicate error value along the progressive epoch-axis. As the model trains till the specified epoch value (300 epochs in this case), it is expected the loss value to decline. As a note, under-fitting model achieves a high error rate (loss). Figure 9 indicates a good fit for our model and can be compared with several examples of over-fit, under-fit curves below that were generated during the experimental development of the model.

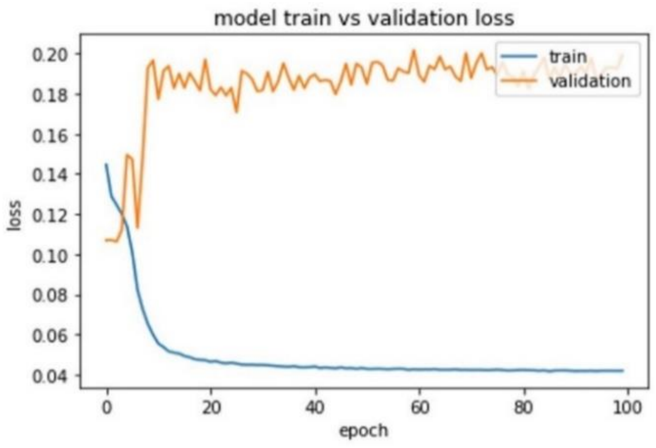

Fig. 10. Over-fit model example (Progress model 1)

Figure 10) (see section 6 for more) is an example of a model performing generally well on the training set yet it is not generalizing sufficiently on the validation set, as the loss rate is increasing along the epochs. A possible solution for this is to decrease number of epochs (training capacity).
Figure 11 and Figure 12 show under-fit models, where the training and validation curves remain almost flat as in Figure 11, or when the training loss is further declining along the epochs indicating the model has still the potential to improve but not provided with sufficient complexity to train further.

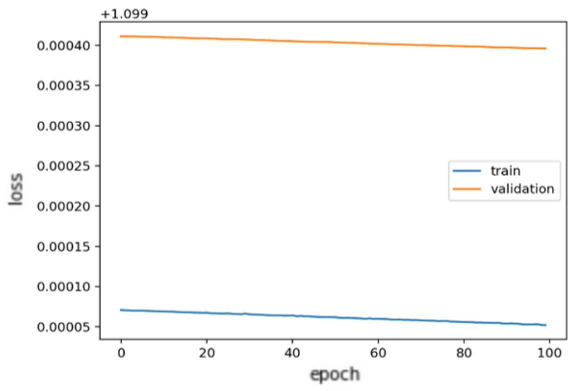

Fig. 11. The under-fit model example 1 [26]

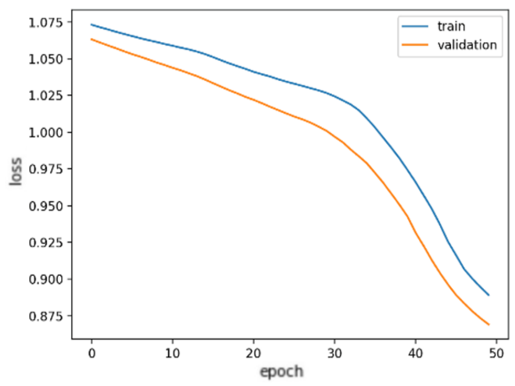

Fig. 12. The under-fit model example 2 [26]

In Figure 13 both curves intersect at some point along the epochs range, approximately somewhere between 200 and 250 epochs. This indicates the training was sufficient at that intersection and training is no longer needed even though training loss is declining, and the valuation curve started to increase slightly. However, to further confirm this case, increasing the training capacity (by increasing epochs) will indicate if both curves split off is consistent and will diverge further, or will keep adjacent to each other and improve the loss.

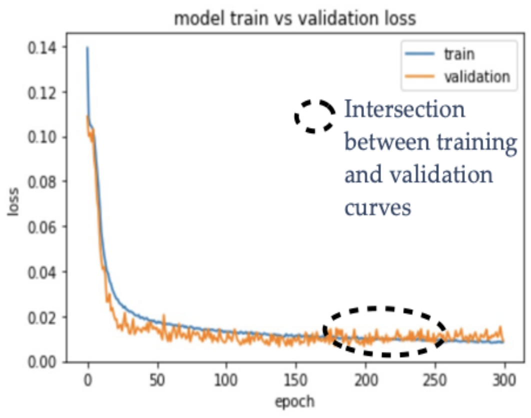

Fig. 13. Main model learning curve

The first 20 samples of predictions can be observed and compared with its experimental values as follow (Table 2). The predicted voltages are slightly more than experimental voltages consistently.

6. USING MODEL TO PREDICT DISPLACEMENTS

To check if the model is versatile enough, we can test the model to predict displacements based on voltage input (objective reversal). However, it appears results are not consistent nor better than when used for its original purpose. A plausible reason is that LSTM is not efficient to be applied on not what it trained on; that is, predicting displacements. This will require a separate yet same model architecture but with different dataset processing to achieve this goal.
Figure 14 demonstrates two inaccurate regions, specifically in the regions in the x-axis ≈ [-0.6 ,0], and ≈ [0.6 , 1], with respect to the y-axis coordinates. Figure 14 has more inaccurate regions than Figure 7, and the error difference is expected, as in Figure 15. Similar to the methodology explained in obtaining this error difference as explained for Figure 8, the gap in Figure 15 indicates large error difference, and hence the model is not valid for such purpose to predict displacements. The coefficient of determination R2 results into 0.9276, while loss: 0.1914, and MSE

(mean squared error): 0.0369. Hence, it might be wiser to create new model using time domain and voltage as training data to predict displacements.

Table 2. Comparing first 20 samples for predicted and experimental voltages

Predicted voltage [V]	Experimental voltage [V]
-0.7977	-0.7878
-0.7985	-0.7886
-0.7994	-0.7894
-0.8005	-0.7902
-0.8009	-0.7909
-0.8012	-0.7917
-0.8020	-0.7925
-0.8025	-0.7932
-0.8031	-0.7940
-0.8029	-0.7948
-0.8032	-0.7955
-0.8047	-0.7963
-0.8053	-0.7970
-0.8059	-0.7978
-0.8069	-0.7983
-0.8080	-0.7993
-0.8079	-0.8001
-0.8096	-0.8008
-0.8101	-0.8016
-0.8106	-0.8023

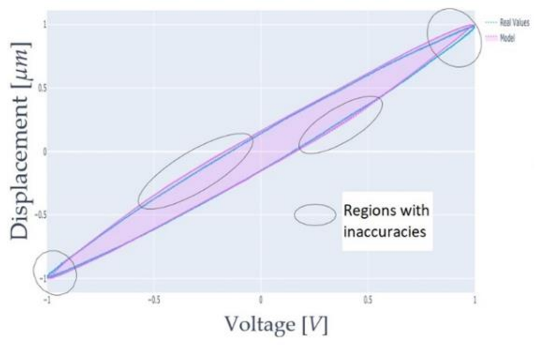

Fig. 14. Experimental model vs predicted model (predicting displacement)

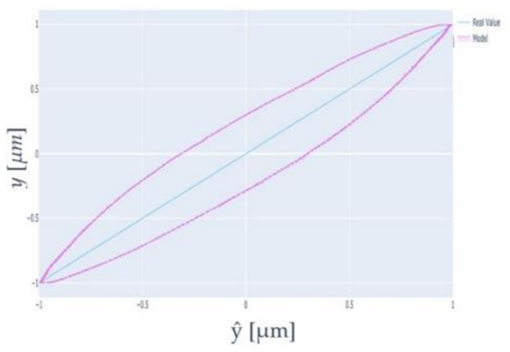

Fig. 15. Error difference between experimental displacement and predicted displacement

The first 20 samples of predictions can be observed and compared with its experimental values as in (Table 3).

Table 3. Comparing first 20 samples for predicted and experimental displacements

Predicted displacement	Experimental displacement
-0.8592	-0.7084
-0.8598	-0.7103
-0.8604	-0.7103
-0.8610	-0.7103
-0.8616	-0.7138
-0.8622	-0.7134
-0.8628	-0.7153
-0.8633	-0.7119
-0.8639	-0.7122
-0.8645	-0.7211

-0.8651	-0.7180
-0.8657	-0.7204
-0.8663	-0.7227
-0.8669	-0.7180
-0.8675	-0.7196
-0.8681	-0.7235
-0.8687	-0.7235
-0.8692	-0.7239
-0.8698	-0.7180
-0.8704	-0.7242

The predicted displacements are significantly larger than experimental displacements consistently, which does account the large error gap in Figure 15.

7. THE PRE-EXPERIMENTAL MODELS BEFORE THE FINAL

Training deep learning models take considerable amount of resource namely hardware and model complexity. This section demonstrates the progress the author has taken in order to reach the final model. The last progressive model is particularly competitive with the final one, yet reasons will be established why they were dismissed in favour of the final one.

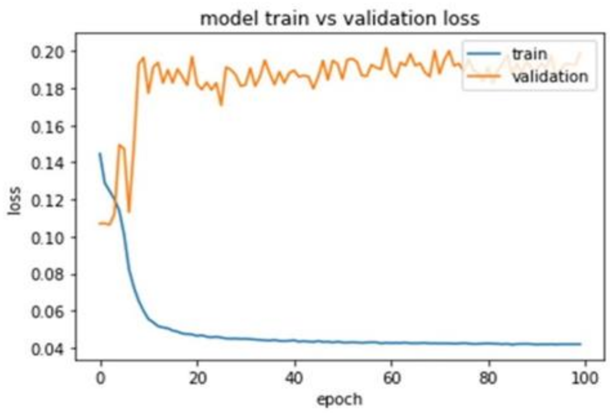

Fig. 16. Learning curve (Progressive model 1)

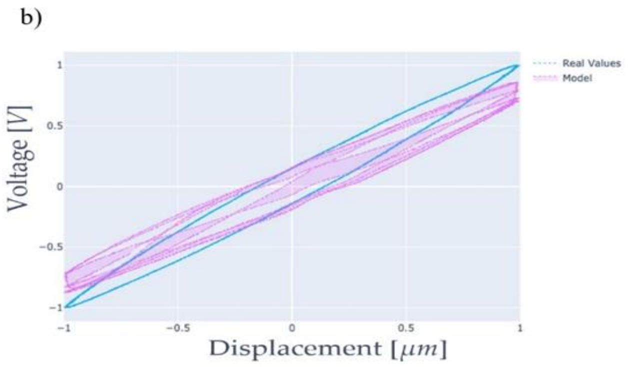

Fig. 17. Experimental model vs predicted model (progressive model 1).

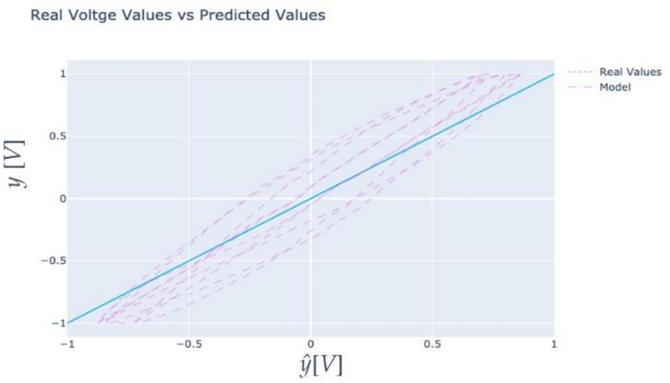

Fig. 18. Over-fit model (progress model 1).

Table 4. Progressive model 1 results

R^2 Score	Loss	MSE
0.8745	0.1731	0.0409

Figure 18 shows over-fitting model, meaning it fails at generalizing data as indicated by the valuation learning curve. Figure 16 shows insufficient model and has large range skews at the outer ranges. This insufficiency is also reflected in Figure 17 error difference. Hence progressive model 1 is poor.

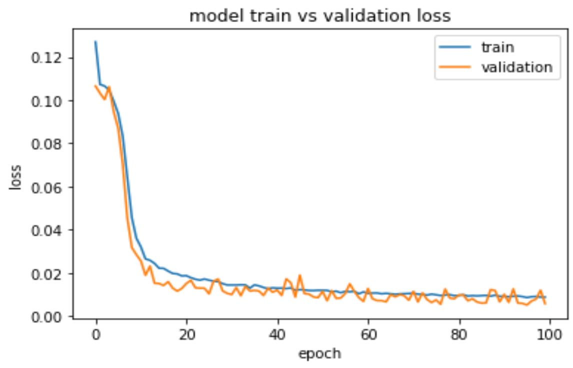

Fig. 19. Learning curve (progressive model 2)

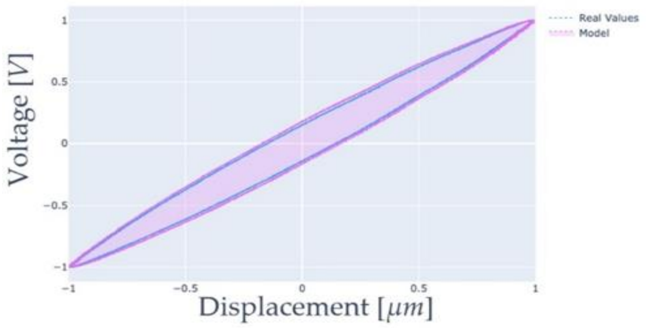

Fig. 20. Experimental model vs predicted model (progressive model 2)

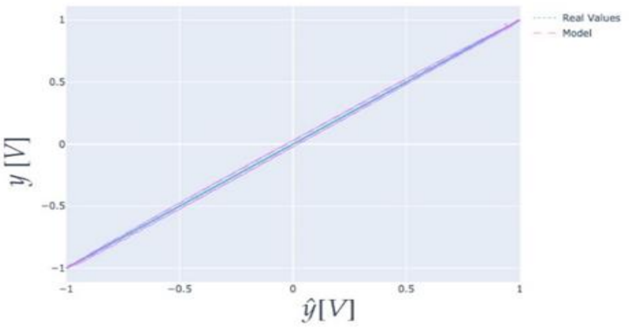

Fig. 21. Error difference (progressive model 2)

Table 5. Progressive model 2 results

R^2 Score	Loss	MSE
0.9999	0.0050	0.00004

Progressive model 2 is noticeably completive with the final model, showing promising results as in Figure 19. However, the model graph shows some disturbing offset from the experimental model as if engulfing the experimental model. This is resulting in a slight error as documented in Table 5. Nevertheless, Figure 20 final model could arguably deliver better generalization performance for hysteresis prediction, since some regions align well with the experimental model, and others are offset slightly.

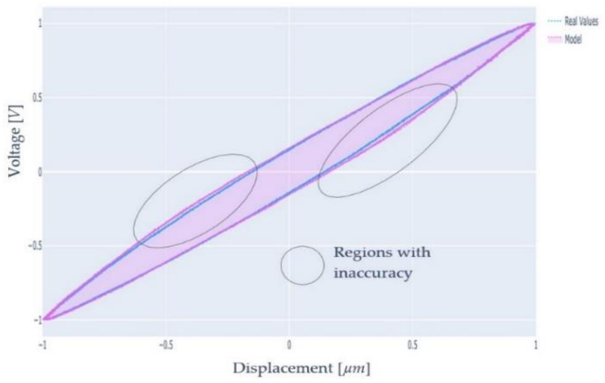

Fig. 22. Real values model vs model plot (final model)

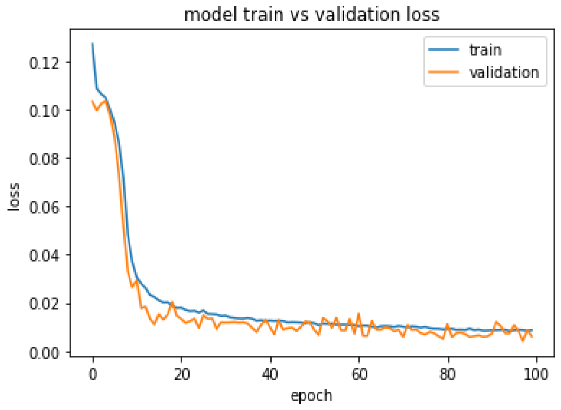

Fig. 23. Learning curve (progressive model 3)

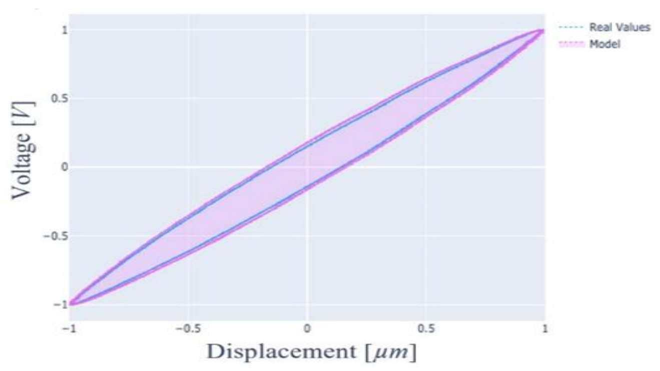

Fig. 24. Experimental model vs predicted model (progressive model 3)

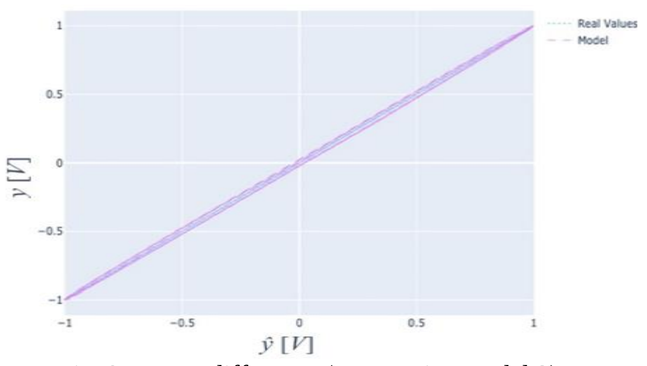

Fig. 25. Error difference (progressive model 3)

Table 6. Progressive model 3 results

R^2 Score	Loss	MSE
0.9999	0.0054	0.00004

Progressive model 3 is also dismissed due to similar reasons as for progressive model 2.

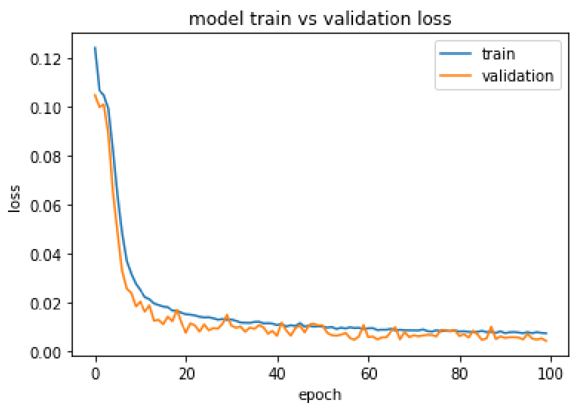

Fig. 26. Learning curve (progressive model 4)

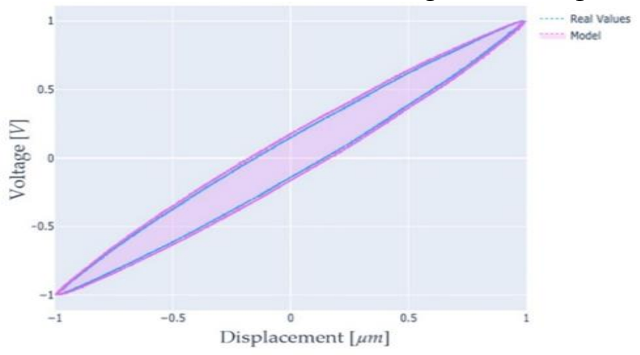

Fig. 27. Experimental model vs predicted model (progressive model 4)

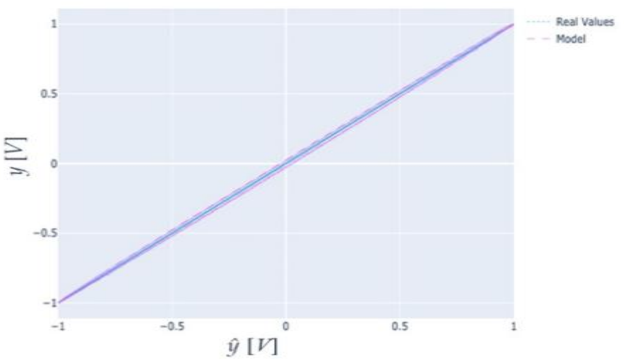

Fig. 28. Error difference (progressive model 4)

Table 7. Progressive model 3 results

R^2 Score	Loss	MSE
1.000	0.0040	0.00001

Progressive model 4 is also dismissed due to similar reasons as for progressive model 2.

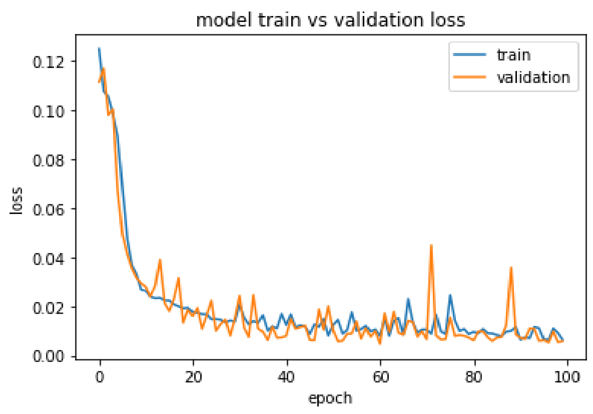

Fig. 29. Learning curve (progressive model 5)

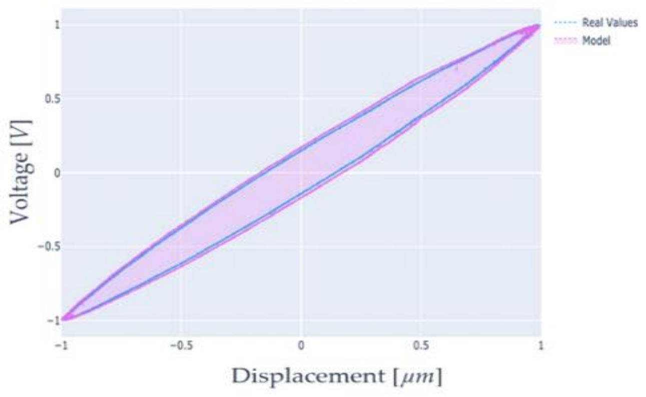

Fig. 30. Experimental model vs predicted model (progressive model 5)

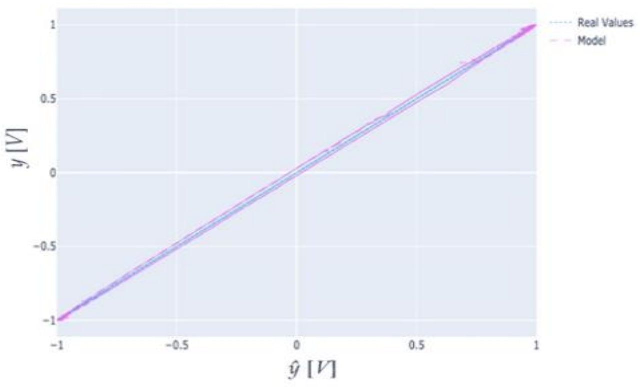

Fig. 31. Error difference (progressive model 5)

Table 8. Progressive model 3 results.

R^2 Score	Loss	MSE
0.9999	0.0048	0.00003

Progressive model 5 is also dismissed due to similar reasons as for progressive model 2. A summary of the progressive models can be reviewed below. It is worth to note that running the code several time can generate varying results than what is documented below, including the final accepted model.

8. CONCLUSION

In section 3 the hysteresis main model has been developed as the final model for piezoelectric hysteresis at 150Hz with good performance overall. Section 5 demonstrated the progressive models in developing the main model, yet these progressive models were dismissed. However, the main model (section 3) has some inaccuracies and changing several parameters, or the neural network architecture can lead to better improvements. A hyper-parameter tuning technique can be applied where all possible ranges of parameters are provided (e.g., number of hidden network neurons, loss functions) and the training phase will cover these ranges with all possible combinations. However, this technique can be expensive and resource demanding. Also, the main model has not been tested with different frequencies, yet this is worth the experiment. There are other deep learning algorithms that are worth experimenting with or developing hybrid system of several models. LSTM was chosen particularly for its long sequences for both past and future data processing. The main model can be stored in any compatible smart computers or systems that can use the model format .h5, and in python run-time environment. Other smart control systems like Arduino microcontroller or Raspberry Pi can be integrated with the model to provide real-time (or semi real-time) data input to the model and predictions output. A larger data set is desirable than the proposed one for this model. A model can be versatile and high performant if it is trained on large data coming from different type of piezoelectric elements and condition, and this can make it easy integrable into any system on demand with the least configurations requirements. Large data is also desirable to reduce over-fitting and provide sufficient ratio of training: validation datasets.

Conflicts of Interest: There is no conflict of interest.

REFERENCES

1. Sitek W. Trzaska J., (2021). *Practical Aspects of the Design and Use of the Artificial Neural Networks in Materials Engineering*. Metals 2021, 11, 1832.
2. Sitek W. Irla A., (2016). *The Use of Fuzzy Systems for Forecasting the Har-denability of Steel*. Archives of Metallurgy and Materials, 61(2A), 797 - 802.
3. Timofejczuk A., (2008). *Identification of diagnostic rules with the application of an evolutionary algorithm.* Eksploatacja i Niezawodnosc - Mainte-nance and Reliability,1(37), 11-16.
4. Pollak A. Hilarowicz A. Walczak M. Gąsiorek D., (2020). *A Framework of Action for Implementation of Industry 4.0*. An Empirically Based Re-search. Sustainability, 12, 5789.
5. Pollak A. Temich S. Ptasiński W. Kucharczyk J. Gąsiorek D. (2021). *Prediction of Belt Drive Faults in Case of Predictive Maintenance in Industry 4.0 Platform*. Appl. Sci. 11, 10307.
6. Mostyn W. Huczala D. Moczulski W. Timofiejczuk A., (2020). *Dimensional optimization of the robotic arm to reduce energy consumption*. Mm science journal, 1, 3745 – 3753.
7. Buchacz A. Płaczek M., (2012). *The analysis of a composite beam with piezoelectric actuator based on the approximate method*. Journal of Vibroengineering, 14(1), 111 - 116.
8. Buchacz A. Płaczek M. Wróbel A., (2014). *Modelling and analysis of systems with cylindrical piezoelectric transducers*. Mechanika., 20(1), 87 - 91.
9. Degefa T.G. Wróbel A. Płaczek M., (2021). *Modelling and Study of the Effect of Geometrical Parameters of Piezoelectric Plate and Stack*. Appl. Sci., 11, 11872.
10. Molla D. Płaczek M. Wróbel A., (2021). *Multiphysics Modeling and Material Selection Methods to Develop Optimal Piezoelectric Plate Actuators for Active Noise Cancellation*. Appl. Sci., 11, 11746.
11. Goodfellow I. Bengio Y. Courville A., (2016). *Deep learning*. 2016 MIT Press.
12. Yu Yong, et al., (2019). *A review of recurrent neural networks: LSTM cells and network architectures*. Neural computation., 31(7), 1235 - 1270.
13. Wang G. Yao X. Cui J. Yan Y. Dai J. & Zhao, W., (2020). *A novel piezoelectric Hysteresis modeling method Combining LSTM and narx neural networks*. Modern Physics Letters B., 34(28), 2050306.
14. Paralı L. et al., (2017). *The artificial neural network modelling of the piezoelectric actuator vibrations using laser displacement sensor.* Journal of Electrical Engineering., 68(5), 371.
15. Liu Y. Zhou R. Huo M., (2019). *Long short term memory network is capable of capturing complex hysteretic dynamics in piezoelectric actuators*. Electronics Letters., 55(2), 80 - 82.
16. Płaczek M. Buchacz A. Wróbel A., (2015). *Use of piezoelectric foils as tools for structural health monitoring of freight cars during exploitation*. Eksploatacja i Niezawodnosc - Maintenance and Reliability., 17(3), 443 - 449.
17. Buchacz A. Płaczek M. Wróbel A., (2013). *Control of characteristics of mechatronic systems using piezoelectric materials*. Journal of Theoretical and Applied Mechanics., 51, 225 - 234.
18. Buchacz A. Płaczek M. Wróbel A., (2014). *Modelling of passive vibration damping using piezoelectric transducers - the mathematical model*. Eksploatacja i Niezawodnosc - Maintenance and Reliability., 16(2), 301 - 306.
19. Płaczek M., (2015), *Modelling and investigation of a piezo composite actuator application*. Int. J. Materials and Product Technology., 50(3/4), 244 - 258.
20. Płaczek M., (2020). *Modelling and production process of the energy harvesting system based on MFC piezoelectric transducers*. International Journal of Modern Manufacturing Technologies., 13(3), 106 - 114
21. Płaczek M. Kokot G., (2019). *Modelling and Laboratory Tests of the Temperature Influence on the Efficiency of the Energy Harvesting System Based on MFC Piezoelectric Transducers*. Sensors., 19(7), 1558.
22. Ayala H. V. H. Rakotondrabe, M. dos Santos Coelho L., (2020). *Piezoelectric micromanipulator dataset for hysteresis identification*. Data in Brief., 29, 105175.
23. Roshan S. et al., (2020). *Violence Detection in Automated Video Surveillance: Recent Trends and Comparative Studies*. The Cognitive Approach in Cloud Computing and Internet of Things Technologies for Surveillance Tracking Systems., 157 – 171.
24. Al-jabery K., et al., (2020). *Selected Approaches to Supervised Learning*. Computational Learning Approaches to Data Analytics in Biomedical Applications., 101 - 123.
25. Cui Z. et al., (2018). *Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction*. arXiv preprint arXiv., 1801, 02143.
26. Brownlee J., (2018). *Deep learning for time series forecasting: predict the future with MLPs, CNNs and LSTMs in Python*. Machine Learning Mastery.

Received: February 15th, 2024 / Accepted: June 15th, 2024 / Paper available online: June 20, 2024 © International Journal of Modern Manufacturing Technologies.