**1. Introduction**

Various papers have already been published regarding the vibration analysis of a certain class of truncated tapered beams using Bessel functions (Conway & Dubil, 1964; D. Conway, H & C. H. Becker, E & F. Dubil, 1965; He et al., 2015; Sanger, 1968; Ward, 1913; Wu & Chiang, 2004; Zhou & Cheung, 2000). Cantilever beam structure has numerous applications, thus estimation of natural frequencies and node locations is crucial (Mituletu et al., 2019; Shafiei et al., 2019). In this paper artificial neural networks were used to predict natural frequency parameters and node locations for first three modes of the truncated cantilever beams with varying cross-section properties. There some research papers that talk about using artificial intelligence for predicting natural frequencies of beams (Avcar & Saplioglu, 2015), but those dealt with very special cases compared to more general case considered in this paper. The main objective of the following work is to use the theoretical solutions to generate training data and then train a deep learning model using that data to predict the natural frequency parameters and node locations of first three mode shapes over seen and unseen data ranges. Analysis of cantilever beams was done with larger end clamped and smaller end free. This work deals with the class of beams satisfying the variation in sectional properties as following:

|  |  |  |
| --- | --- | --- |
|  | , | (1) |

where, AC and IC are the cross-sectional area and moment of inertia about flexural axis at the clamped end, δ=x/L ϵ [δ0, 1], δ0=LT/L, LT is the length of truncated portion, L is the total length of beam without truncation (Figure 1), η ϵ [0, 10] is a real number specifying the variation in the area of cross-section.

**2. Theory**

For the cantilever beam subjected to free vibrations the equation of motion is given by the Euler-Bernoulli equation (Eq. 1),

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

where, y(x, t) is deflection at position x in yz-plane perpendicular to flexural axis at time t, E is the Young’s modulus of the material, I(x) is the moment of inertia about flexural axis at position x, is density of the material and A(x) is the area of cross section at position x.

Since variable separation is applicable here, therefore, we can assume solution of (Eq. 1) in the following form,

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

General solution for in Eq. (2) can be given as,

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

, ,

where, Jη and Yη are ηth order Bessel Functions of first and second kind, respectively. Iη and Kη are modified ηth order Bessel Function of first and second kind, respectively.

Following are the Boundary conditions for cantilever beam shown in Fig. (1),

|  |  |  |
| --- | --- | --- |
|  |  | (5a) |
|  |  | (5b) |

After applying boundary conditions (Eq. 4a-4b) on Eq. (3), following set of linear equations was obtained,

|  |  |  |
| --- | --- | --- |
|  |  | (6a) |
|  |  | (6b) |
|  |  | (6c) |
|  |  | (6d) |

where, ,

Now, this linear system of equations (Eq. 5a-5d) to have non-trivial solution following condition should satisfy,

|  |  |  |
| --- | --- | --- |
|  |  | (7) |

After solving Δ = 0 (Eq. 6), we can get values of β = βr, where, r is the mode number. From which, we can compute and using following relations,

|  |  |  |
| --- | --- | --- |
|  | , | (8) |

Finally, mode shapes can be obtained using Eq. 7 as follows,

|  |  |  |
| --- | --- | --- |
|  |  | (9) |

**3. Deep learning model**

*3.1 Defining the input and output variables*

Input variables - **η,** **δ0**.

Output variables - **β1, β2, β3**, **δ21, δ31, δ32**.

where, **η** is exponent defining the variation of Area of cross-section of the cantilever along x-axis and **δ0** is fraction of truncated portion of cantilever. **β1**, **β2**, **β3** are parameters defining first three natural frequencies and **δ21**, **δ31**, **δ32** are non-dimensionalized numbers defining node locations of first 3 mode shapes (Figure 2).

Based on the theory, following plots (Figure 3-4) were generated using MATLAB (The MathWorks, 2019) showing theoretical variation in output variables with respect to input variables. For the combination higher values of **η** and very low values **δ0** numerical solver (Platte & Trefethen, 2010) was not able to solve the equations with enough accuracy, therefore, those values were skipped as can be seen in figure 3.

*3.2. Data generation*

Data was generated for 1,001 uniformly distributed values of **η** є [0, 10] over the entire range and for each **η**, 15000 uniformly distributed data points were collected by varying **δ0** between [2e-3, 0.56] for **η** є [0, 3] and [7e-2, 0.56] for **η** є (3, 10], values of **δ0** were chosen such that entire mentioned range was covered. Reason for this split in the range of **η** being insufficient accuracy in the frequency parameters calculated by the algorithm for small values of **δ0** and large values of **η**.

So, overall distribution of data points was as following:

Dataset 1: For **η** є [0, 3], **δ0** є [2e-3, 0.56], 301\*15,000 datapoints.

Dataset 2: For **η** є [3, 10], **δ0** є [7e-2, 0.56], 701\*15,000 datapoints.

Dataset 3: For **η** є [0, 10], data for above two cases was merged, 1,001\*15,000 datapoints.

Apart from the these datapoints a separate test dataset of 50\*100=5,000 examples was also created for each interval (**η** є [0,3] and **η** є [3, 10]). Models that performed best on the validation set were evaluated on the test dataset for the final evaluation and presentation purposes. These datapoints were generated by sampling 50 random values of **η** from each interval and for each value of **η**, 100 values of **δ0** were also chosen randomly. For generating the data MATLAB (The MathWorks, 2019) was used.

*3.3. Defining learning model*

This section describes the main features of the model. The subsequent choices of activation function (section 5.3), loss function (section 5.4) and optimizer (section 5.5) was made after training and evaluating some simple neural networks on small dataset. It was observed that PReLU, mean squared percent error and Adam gave best results. Layer normalisation did not significantly improve the accuracy; however, it still gave positive results.

*3.3.1. Neural network architecture*

[Input Layer, 2 units] 🡪 [Fully connected Hidden layer without activation function, with 6\*N2 units] 🡪 L1\*[(Fully connected Hidden layer 🡪 Layer normalisation 🡪 activation function), all three layers having N1 units each] 🡪 [Fully connected Hidden layer without activation function, with 6\*N2 units] 🡪 [Output layer, 6 units]

Pictorial representation of the neural network architecture is given in Figure 5.

*3.3.2. Layer Normalisation* (Ba et al., 2016)

Output of normalisation layer is given by :

|  |  |  |
| --- | --- | --- |
|  |  | (10) |

,

where, is the value at ith hidden unit of lth layer and H is the number of hidden units in a layer. All the hidden units in a layer share the same normalization terms and , but different training datapoints have different normalization terms. γ and β are learnable parameters.

*3.3.3. Activation function* – PReLU (Ba et al., 2016)

|  |  |  |
| --- | --- | --- |
|  | PReLU() = *max*(0,)+a0∗*min*(0,) | (11) |

where, a0 is a learnable parameter.

*3.3.4. Loss function* - Mean squared percent error.

|  |  |  |
| --- | --- | --- |
|  |  | (12) |

Where, N is the number training examples, actij is actual value of ith output of jth training example and predij is predicted value of ith output of jth training example.

*3.3.5. Optimizer*

Adam optimizer (Kingma & Ba, 2015)was used for minimising the cumulative loss of the model, along with mini-batch gradient descent. Default value of the parameters β**1** and β2 was used, i.e. {β**1** = 0.9, β2 = 0.999},

*3.4. Training the neural network*

Models were separately trained over Dataset 1 and Dataset 2 as described in section 3.2. Various models were trained on each of these two datasets by tuning the hyperparameters described in Table 1. For tuning the hyperparameters, several models were trained by randomly sampling the values of each hyperparameter from a certain interval best suited for that hyperparameter. For finer tuning on subsequent trials, sampling interval for each hyperparameter was narrowed down to around the region giving best performance. Dataset generated in section 3.2 was shuffled and then datapoints were randomly selected to generate training and validation datasets. Models were trained over the training dataset and then validation dataset was used to pick the best model. Finally, the model that performed best on the validation dataset was evaluated on the test dataset (section 3.2) and results are tabulated in section 4. After training few simple models, values of learning rate parameters, mini-batch size, number of mini-batches, number of iterations, etc. that worked best are given in Table 2 and these values were not changed in further analysis until or unless explicitly mentioned. Models were trained using pytorch (Paszke et al., 2019) on NVIDIA GeForce GTX 1060 6GB GPU.

**4. Results**

After training several models, it was observed that there is no single combination of the hyperparameter values that performed better than all the other combinations over both the datasets (Dataset 1 and Dataset 2). Therefore, out of the models trained and evaluated, two models with significantly different configurations were selected and analysed further (Table 3). After training the neural networks mentioned in Table 3 separately on both the datasets, final evaluation on the test dataset of each data range (section 3.2) was done and the results are shown in Tables 4-5.It can be easily observed from these results that a bigger network can be trained using even more training data to further reduce errors. Since the errors are considerably low for all the practical purposes, therefore reducing them further is unnecessary.

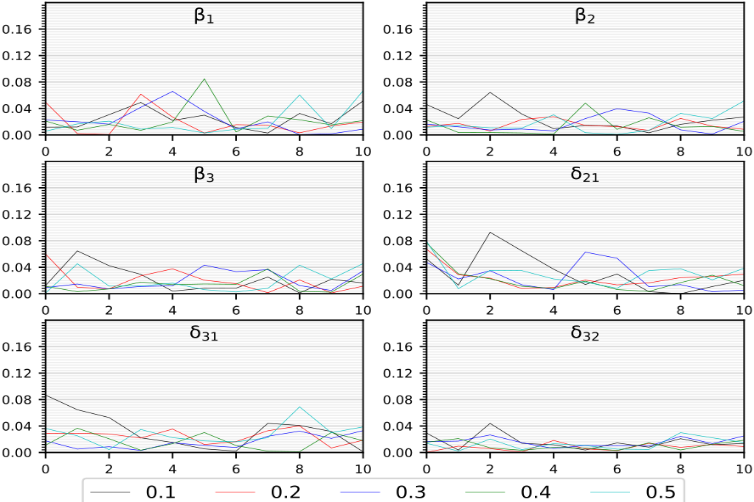
*4.1 Observations*

From Tables 4-5 it can be inferred that larger and more complex neural network improves accuracy in predications in both intervals, but the margin of improvement when trained with bigger network and more data (Model 2), is slightly more significant for smaller values of **η**. This observation is in sync with the fact thatthe mapping function from input to output is more complex for smaller values of **η** compared to larger values.

Models were finally trained over Dataset 3 and evaluated on the test dataset obtained by concatenating test datasets from both data ranges (section 3.2). As per expectations similar trend in percent deviations was observed as shown in Tables 4-5.

*4.2. Graphical comparison between actual and predicated values*

Plots in Figures 6-11 were obtained after training Model 2 (Table 3) over the entire range of **η.** For validation 10,000 datapoints were used and only 3,000 iterations were performed with the Updated Period (Table 1) equal to 500 iterations, all other training parameters were kept same as described in Table 2.

**5. Extrapolation analysis**

In section 4, we evaluated the models over the same data range as they were trained on. Now in this section we check the ability of models to predict over values of **η** and **δ0** taken from outside the range of values used for training. For this analysis, models given in Table 3 were trained over the Dataset 1 and evaluated on the test dataset generated from data range of Dataset 2 and vice-versa.

*5.1 Observations*

From the results shown in Tables 6-7 and Figures 12-15, we can observe that when trained on Dataset 1, Model 1 gives better extrapolation over the unseen data range compared to Model 2. Although, when training was done on Dataset 2, Model 2 does better over unseen data range compared to Model 1. Reasons for these differences are discussed in section 6.

Mean percent deviations are smaller in Table 7 compared Table 6 for both the models. When training over Dataset 2 and evaluating on the values from Dataset 1, we get fairly small accuracy for most of the values, but the deviations shoot up really high for few values in the corner (Figures 14-15).

**6. Conclusion**

From the results presented in sections 4, we can conclude that when the values of input variables (**η** and **δ0)** are small (**η** є [0, 3], **δ0** є [2e-3, .56]), a bigger neural network (Model 2) is required to attain the accuracy similar to what a smaller neural network (Model 1) can attain for relatively large values (**η** є [3, 10], **δ0** є [7e-2, .56]). Reason for lesser accuracy in smaller regime while using a smaller neural network can be attributed to the mapping function from input to output being more complicated for smaller values of **η**, which can be seen in Figures 3-4, as the curvature of plotted curves is more near the smaller values of input variables compared to the larger values.

Also, as observed through the cross-evaluation done in section 5, any model when trained over small values of **η** performed better on average over bigger values of **η** compared to when models were trained over bigger values of **η** and evaluated on smaller values of **η**, because of the smaller mean percent deviations obtained when trained on Dataset 2 and evaluated on Dataset 1. Again, this can also be ascribed to the more complex mapping function in smaller regime. Since using bigger neural network (Model 2) for training over smaller values of **η** (Dataset 1)learns its complex underlying mapping function, therefore when we use it to predict over the values outside the range of training data where the mapping function is relatively smoother, accuracy drops faster compared to smaller neural network (Model 1), which was unable to learn these complex details of the training data, as the input variables get bigger (Figures 12-13). On the other hand, if we train on larger values of **η** (Dataset 2), we are better-off using bigger network (Model 2) as seen in Table 7.

It can be concluded from these results that for extrapolating over unseen data ranges, it is better to train smaller and less complex neural network rather than a complex one, if no prior knowledge about the complexity of unseen data is present. Using simpler neural network helps in generalizing as it does not learn every fine detail of the data used in training. Although, if you know that unseen data is more complex than training data, then using a complex data for training will improve accuracy of predictions over unseen data, because we want our model to learn whatever little information contained in this less complex training data. So, our baseline conclusion is that always use simpler neural network for extrapolation purposes in absence of any knowledge about unseen data.

**7. Further applications**

Models mentioned above can be trained on the data generated for beams under different boundary conditions other than cantilever structure, such as sliding, simply supported, etc. Since the underlying mapping function between input and output is similar for different boundary conditions, therefore, model with architecture similar to that used for analysis of cantilever beams should work reasonably well on other boundary conditions as well. Also, better models could be developed to improve the extrapolation accuracy.

**Acknowledgements**

This research did not receive any specific grant from funding agencies in the public, commercial, or

not-for-profit sectors.

**References**

Avcar, M., & Saplioglu, K. (2015). An Artificial Neural Network Application for Estimation of Natural Frequencies of Beams. *International Journal of Advanced Computer Science and Applications*, *6*(6), 94–102. https://doi.org/10.14569/ijacsa.2015.060614

Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). *Layer Normalisation*.

Conway, H. D., & Dubil, J. F. (1964). Vibration frequencies of truncated-cone and wedge beams. *Journal of Applied Mechanics, Transactions ASME*, *31*(4), 932–934. https://doi.org/10.1115/1.3627338

D. Conway, H & C. H. Becker, E & F. Dubil, J. (1965). Vibration Frequencies of Tapered Bars and Circular Plates. *Journal of Applied Mechanics, Transactions ASME*, *32*, 234–235. https://doi.org/32.10.1115/1.3625766

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *Proceedings of the IEEE International Conference on Computer Vision*. https://doi.org/10.1109/ICCV.2015.123

Kingma, D. P., & Ba, J. L. (2015). Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*.

Mituletu, I. C., Gillich, G. R., & Maia, N. M. M. (2019). A method for an accurate estimation of natural frequencies using swept-sine acoustic excitation. *Mechanical Systems and Signal Processing*. https://doi.org/10.1016/j.ymssp.2018.07.018

Paszke, A., Gross, S., Chintala, S., & Chanan, G. (2019). *pytorch* (1.3.0). https://github.com/pytorch/pytorch

Platte, R. B., & Trefethen, L. N. (2010). *Chebfun: A New Kind of Numerical Computing*. https://doi.org/10.1007/978-3-642-12110-4\_5

Sanger, D. J. (1968). TRANSVERSE VIBRATION OF A CLASS OF NON-UNIFORM BEAMS. *Journal of Mechanical Engineering Science*, *10*(2), 111–120.

Shafiei, N., Ghadiri, M., & Mahinzare, M. (2019). Flapwise bending vibration analysis of rotary tapered functionally graded nanobeam in thermal environment. *Mechanics of Advanced Materials and Structures*. https://doi.org/10.1080/15376494.2017.1365982

The MathWorks, I. (2019). *MATLAB* (No. 2019a). https://www.mathworks.com/products/matlab.html

Ward, P. F. (1913). IX. The transverse vibrations of a rod of varying cross-section . *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *25*(145), 85–106. https://doi.org/10.1080/14786440108634312

Wu, J. S., & Chiang, L. K. (2004). Free vibrations of solid and hollow wedge beams with rectangular or circular cross-sections and carrying any number of point masses. *International Journal for Numerical Methods in Engineering*, *60*(3), 695–718. https://doi.org/10.1002/nme.981

Zhou, D., & Cheung, Y. K. (2000). Free vibration of a type of tapered beams. *Computer Methods in Applied Mechanics and Engineering*. https://doi.org/10.1016/S0045-7825(99)00148-6