An approach for displacement prediction in truss structures combining the Finite Element Method and Deep Learning Techniques

Mateus de Paula Ferreira[1], Elisa Dominguez Sotelino[1],

[1]*Dept. of Civil and Environmental Engineering, Pontifícia Universidade Católica do Rio de Janeiro*
Rua Marquês de São Vicente, 225, 22451-900, Rio de Janeiro, Brazil
mdepaulaferreira@aluno.puc-rio.br, sotelino@puc-rio.br

Abstract. Recent advancements in machine learning have facilitated groundbreaking applications across various knowledge domains. This paper introduces a promising application of Deep Learning Techniques (DL) in conjunction with the Finite Element Method (FEM). The aim of this study is to evaluate a convolutional neural networks´s (CNN) ability to predict 2D truss displacement fields. This assessment leverages prior knowledge of the structure's global stiffness matrix (**K**) and applied external load vector (**f**). The technique's advantage lies in circumventing complex numerical approaches for solving linear and nonlinear systems, which often entails extended processing times and substantial computational effort depending on model complexity. The employed methodology involves constructing a dataset comprising varied structural configurations, processed through a finite element solver. This dataset then trains a CNN composed of residual, dense, and fully connected blocks. The outcomes underscore the significance of this approach; the proposed model demonstrates commendable performance in this task, exhibiting a maximum absolute mean error of approximately 2% and a maximum mean squared error of 0.2%, when contrasted with classical FEM solutions.

Keywords: Deep Learning, Computational Mechanics, Finite Element Methods.

1 Introduction

With the emergence of the Finite Element Method (FEM), Structural Engineering has undergone significant advancements in terms of modeling and simulating problems involving the mechanical – and even multiphysics – behavior of structural systems (Zienkiewicz et al. [1]). It has proven efficient due to its versatility – the ability to adapt to solving problems of different natures – and the guarantee of convergent solutions. However, these potentials establish a proportional relationship with the computational cost of its application in a given simulation: the more complex the processed models, the more time and processing resources are consumed (Basarinath et al. [2]). Such a characteristic yields significant influence over the method's use, to the point where, based on the sophistication of the model being employed, its application might become infeasible.

In this context, novel artificial intelligence techniques capable of establishing non-analytical relationships between data have been used and promise to become prominent tools in the field of Computational Mechanics, offering a crucial contribution with significant relevance: processing speed and model simplicity.

In this vein, Bolandi et al. [3][4] proposed coupling FEM and a deep neural network model to predict stress fields in unit-sized planar elements. Their study proposed an analogy between semantic image segmentation and stress distribution in rectangular plates. The proposed model proved capable of reproducing such distribution under varying boundary conditions and applied loadings on the element surface. The maximum observed prediction error was on the order of 0.45%, demonstrating the pertinence of this approach for solving mechanical problems.

Similarly, Dong, Y et al. [5] presented an approach using Deep Learning combined with FEM for solving nonlinear problems. In their work, the shape functions of elements were approximated by deep networks trained to replace the usual polynomial functions. Their results not only showcased the functionality of this approach, but also its efficiency. Models that required extensive processing time – in the order of hours and days – could be processed instantaneously with high accuracy.

In this study, another application of DNN for predicting nodal displacements in plane trusses is introduced in combination with the Finite Element Method. In this approach, FEM is applied to obtain the stiffness matrix of a structural system and its corresponding vector of nodal loads. However, the solution of the resulting linear system foregoes the use of classical numerical methods, making way for a DL model capable of correlating applied loads and stiffness to displacement fields. Thus, the model acts as an alternative solver for the discrete problem, with the main benefit being the instantaneous acquisition of mechanical analysis results.

2 Combining FEM and Deep Neural Networks

The construction of the Finite Element Method can be summarized in three main steps: the assembly of the global stiffness matrix of the structure ($\mathbf{K}$), the cataloging of acting forces upon it ($\mathbf{f}$) and the determination of the displacement field $\mathbf{u}$ by the solution of the linear system $\mathbf{f} = \mathbf{Ku}$. In other words, a structure is fully characterized once its set $\{\mathbf{K}, \mathbf{f}, \mathbf{u}\}$ is known.
When analyzing each of these "components" of the method individually, a certain invariance of two of these terms becomes evident. Indeed, once the geometry, material, and element mesh of the structure are defined, its stiffness matrix is known and it will not change independent of the loads and displacements that may potentially act upon the structure. Similarly, this property is also enjoyed by the field of acting forces, once determined, it too becomes a known component of the system. As a result, the vector $\mathbf{u}$ is merely a function of its respective stiffness matrix and load vector:

$$\boldsymbol{u} = \boldsymbol{\varphi}(\boldsymbol{K}, \boldsymbol{f}) \tag{1}$$

Given that both the assembly of the stiffness matrix and the load vector are not tasks demanding significant processing effort and considering the availability of substantial volumes of data generated by FEM simulations – in this case, sets of triples in the form of $\{\mathbf{K}, \mathbf{f}, \mathbf{u}\}$– the goal of this work is to approximate the φ function by a deep neural network model.
In this way, the Finite Element Method with Deep Learning techniques can be combined, where the former is responsible for generating the stiffness matrices and load vectors of the structure, while the latter acts as a solver for obtaining the vector of nodal displacements. This results in a novel approach to solving mechanical problems, devoid of the necessity to perform stiffness matrix inversion or some numerical method application for solving the linear system.

3 Application: predicting displacement field of 2D trusses under vertical nodal loads

3.1 Methodology

The overall methodology for this study is shown in Figure 1. It involved generating a dataset of sufficient size for training a deep neural network model. Once trained and validated, this model was tested on a separate dataset to assess its performance in predicting nodal displacements.

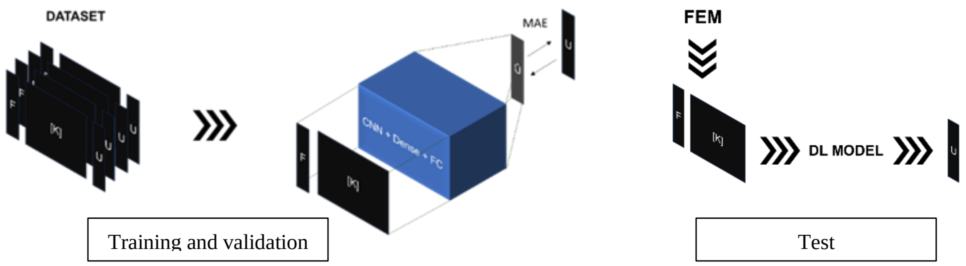

Figure 1. General methodology for combining FEM and DL.

To ensure the dataset exhibited the necessary degree of variability for effective network learning, certain geometric and material parameters of the structure were randomly selected, while adhering to specific value limits representative of real-world applications – as detailed in Table 1.

Table 1. Global parameters for varying in dataset.

Property	Range
Length of horizontal elements	0,3 – 1,0 m
Truss height	0,3 – 1,0 m
Cross section area	$1 \cdot 10^{-4} - 2 \cdot 10^{-4}$ m
Diagonal shapes	up to 6
Young modulus	$1 \cdot 10^{9} - 2 \cdot 10^{9}$ N/m²
Load value	0 – 1000 N
Load position	up to 4 nodes

In the employed methodology, three distinct tests were conduct in order to evaluate the performance of the model including its capacity of generalization. As follows, three databases were created, each designated for a different p of model evaluation testing, as shown in Figure 2. The first one, which have been used for training the model and to Test 1, was composed by trusses with only increasing or decreasing diagonal members. The second one, used in the Test 2, was created with mixed diagonal types, half increasing or decreasing and half decreasing or increasing. The last one, used in Test 3, was composed by all preceding diagonal shapes including other 2 types of alternated diagonal (increasing/decreasing and decreasing/increasing).

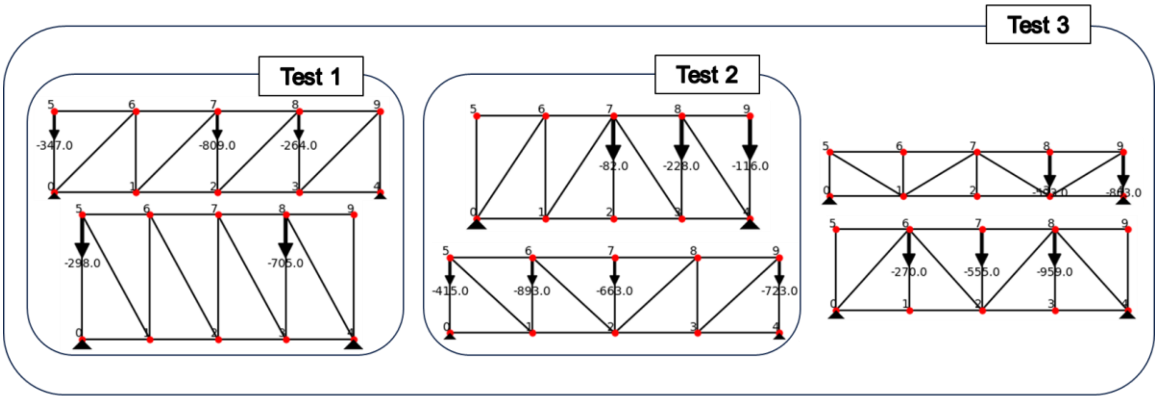

Figure 2. Trusses diagonal types used in the model evaluation.

3.2 Data regularization

Due to the behavior of the functions within the neurons in each layer of the network, to achieve good performance in its learning, it is preferable that the input data provided exhibit a certain regularity, especially regarding the numerical range they fall into. In this context, several transformations, and adjustments (Figure 3) were applied to the dataset to ensure a formatting conducive to the model's learning.

- **Normalization of stiffness matrix:** Due to the high value of the elastic modulus (order of magnitude of 10^9), which is not suitable for neural network learning, a normalization was applied to the matrices in the database, mapping their values to the interval [0,1].
- **Resizing of Load Vectors:** Given that the original database contains objects of different formats, the decision was made to transform the load vectors into matrices with the same dimensions as the stiffness matrices. This process involved mapping the entries of the F vectors onto the main diagonal of the resulting matrix, with the other entries being zero. Normalization of values was also carried

out for the same reasons mentioned earlier.

- **Concatenation of Matrices K and F:** Considering the application of convolutions on the network's input data to enable the model to correlate stiffness data with load data, a three-dimensional tensor data architecture was adopted. This was achieved by concatenating matrices K and F.

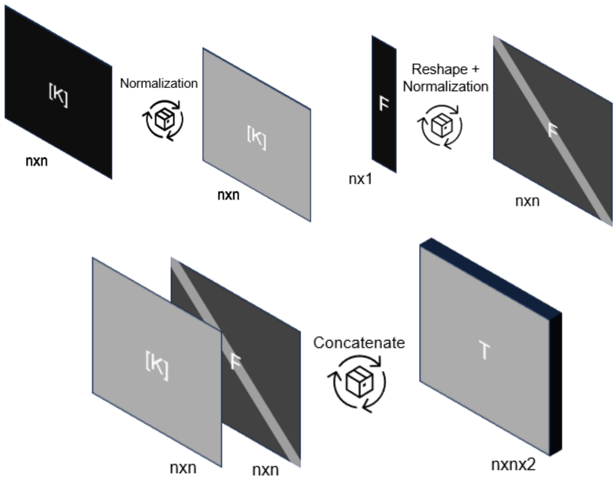

Figure 3. Data regularization for training and test.

3.3 Network architecture

The design of the network to be trained was inspired on the RestNet architecture, taking into consideration the nature of the data and the expected type of outcome. Thus, the network was structured with a backbone, which consists in a convolutional block used to extract the main features of the input data, creating an hierarchical representation of the data and where low-level information is gradually combined and refined with high-level information as it passes through the layers. Subsequently three residual blocks are interleaved with convolutional blocks. This technique of skip connections is very efficient in avoiding the gradient vanishing during the training process as it introduces parts of the original information at4 the end of each convolutional block. Finally, it's performed two dense layers (fully connected), one with 128 parameters, and another with 20 parameters (Figure 4), which, for this study, corresponds to the dimension of the output data.

The entire creation of the model, its training, and usage were done using the TensorFlow library within the Python environment.

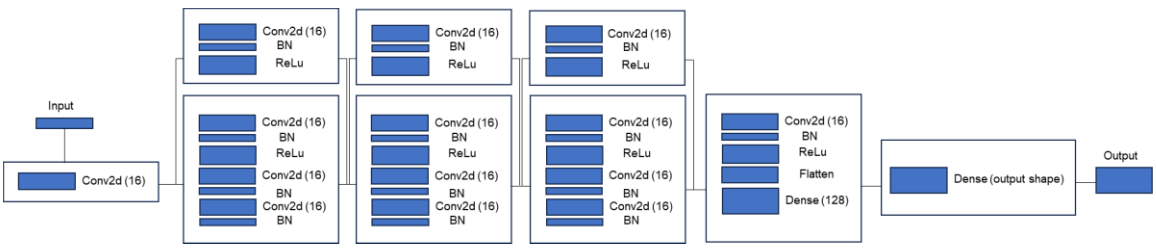

Figure 4. Network architecture.

3.4 Loss function

In the present study, the chosen cost function for the adopted deep neural network was the Mean Absolute Error (MAE), which allows the evaluation between a predicted data p_i and its real value y_i in set with N samples:

$$\boldsymbol{MAE} = \frac{1}{N}\sum_{i=1}^{N} |y_i - p_i| \tag{2}$$

By opting for MAE, the aim is to directly assess the magnitude of prediction errors, providing an objective and understandable measure of the deep neural network's performance in the context of displacement prediction.

4 Results

The results in terms of loss and average error for the training phases are presented in the graphs of Figure 4 below.

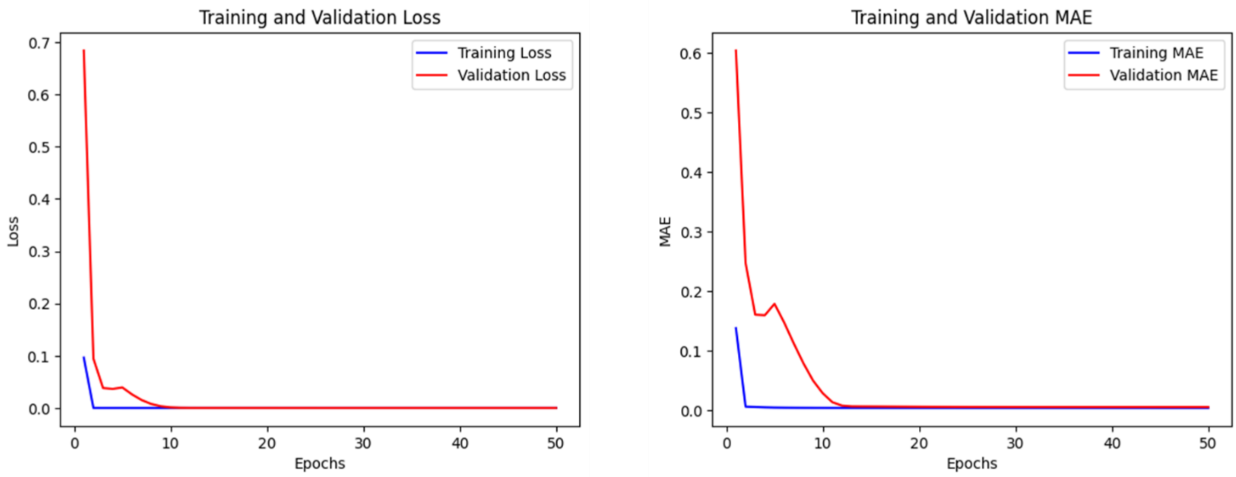

Figure 5. Training and validation curves for the CNN model.

It's noticeable that convergence of the model occurs exponentially even in the early epochs of training, demonstrating the model's effective learning of relationships between input and output data. From the validation curve, the model's performance was deemed satisfactory as, starting from the tenth epoch, no increase in value was observed throughout subsequent epochs. This signifies its adeptness in terms of generalization, without overfitting to the data. The curves of mean absolute error further confirm this successful behavior of the model, as its value, both in training and validation, remained around zero for later epochs.

5 Model testing

5.1 Test 1

In this initial test, the aim was to evaluate the model's performance in predicting nodal displacements in trusses with diagonal arrangements similar to those in the test and validation datasets – that is, trusses with either only increasing or only decreasing diagonal members. It is evident that the model performed well in predicting nodal displacements for the majority of samples, with only a few outliers observed in both MAE and MSE metrics. The values ranged between 0% and 20% for MAE, and 0% and 5% for MSE, as shown in Figure 6.

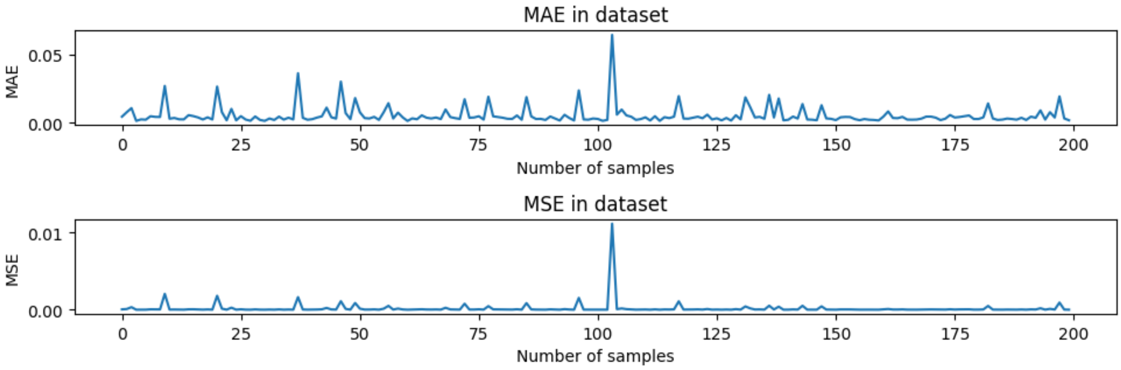

Figure 6. MAE and MSE results for Test 1.

5.2 Test 2

In the second test, the model's performance was evaluated on trusses with diagonal arrangements different from those present in the test and validation datasets: trusses with combinations of both increasing and decreasing diagonal members. It is observed that the model performed even better in predicting nodal displacements for most samples, with values ranging between 0% and 5% for MAE, and 0% and 0.5% for MSE, as shown in Figure 7.

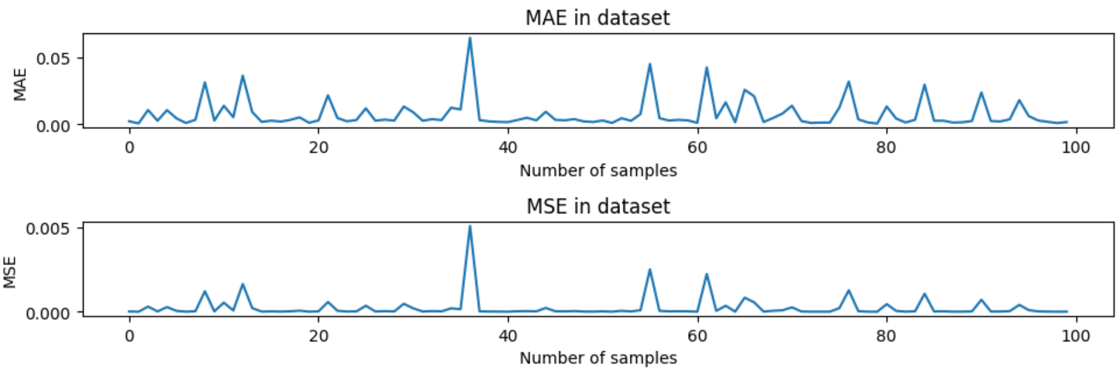

Figure 7. MAE and MSE results for Test 2.

5.3 Test 3

In the third test, the generated database included all possible diagonal arrangements for the trusses, resulting in a broader range of variability. The graphs in Figure 7 display the results for the MAE and MSE metrics in this stage. It is notable that the model sustained its good performance in predicting displacements for most samples, even with the increased variability in the database. The values ranged between 0% and 2% for MAE, and 0% and 0.2% for MSE.

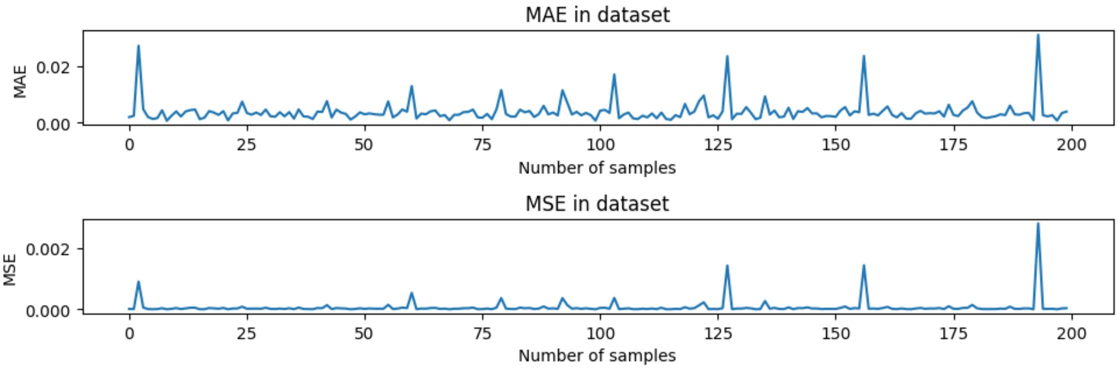

Figure 8. MAE and MSE results for Test 3.

6 Conclusions

The present study aimed to utilize deep neural networks for predicting nodal displacements in a truss structure, utilizing stiffness matrix and nodal load vector data obtained from the Finite Element Method (FEM) as input. Following the completion of model training and validation, the performance of the trained model was assessed through three tests involving different truss arrangements. The model exhibited strong performance across all evaluations, with maximum absolute mean error values of approximately 2% and maximum mean squared error values of around 0.2%. The combined approach, integrating the two techniques, proved to be efficient in predicting displacements without directly solving the system of equations.

Authorship statement. The authors hereby confirm that they are the sole liable persons responsible for the authorship of this work, and that all material that has been herein included as part of the present paper is either the property (and authorship) of the authors, or has the permission of the owners to be included here.

References

[1] Zienkiewicz, O. C. and Taylor, R. L., "The Finite Element Method", 4th Edition, vol. 1: "*The Finite Element Method: From Intuition to Generality*",Appl. Mech. Rev., 23, no. 23, pp. 249-256, March, 1970.
[2] Badarinath, P. V.; Chierichetti, M.; Kahkhi, F. D. "A machine learning approach as a surrogate for a finite element analysis: Status of research and application to one dimensional systems". *Sensors*, vol. 21, pp. 1–18, 2021.
[3] Bolandi, H. et al. "Bridging finite element and deep learning: High-resolution stress distribution prediction in structural components". *Frontiers of Structural and Civil Engineering*, vol. 16, pp. 1365–1377, 2022b
[4] Bolandi, H. et al. "Deep learning paradigm for prediction of stress distribution in damaged structural components with stress concentrations". *Advances in Engineering Software*, vol. 173, 2022a.
[5] Dong, Y. et al. "DeepFEM: A novel element-based deep learning approach for solving nonlinear partial differential equations in computational solid mechanics". *Journal of Engineering Mechanics*, vol. 149, 2023.