Exploring the Predictive Performance of Simple Regression Models and ANN in 2D Truss Analysis

Vagelis Plevris, Alejandro Jiménez Rios, and Usama A. Ebead

Contents

1 Objective and Literature Review

Artificial intelligence (AI) has garnered considerable interest in various scientific domains in recent years, spanning from managing large datasets to aiding in medical diagnostics. Its integration into everyday life is evident through personalized advertisements, virtual assistants, autonomous vehicles, and more. Unsurprisingly, AI techniques have permeated engineering disciplines, including civil and structural engineering [1, 2], showcasing remarkable achievements [3, 4].

Málaga-Chuquitaype [5] authored a thought-provoking review article on the application of machine learning (ML) in structural design. The article raises a compelling query regarding the future necessity of human engineers in the field of

V. Plevris (✉) · U. A. Ebead
Qatar University, Doha, Qatar
e-mail: vplevris@qu.edu.qa; uebead@qu.edu.qa

A. J. Rios
Oslo Metropolitan University, Oslo, Norway
e-mail: alejand@oslomet.no

M. Kioumarsi, B. Shafei (eds.), *The 1st International Conference on Net-Zero Built Environment*, Lecture Notes in Civil Engineering 237,
https://doi.org/10.1007/978-3-031-69626-8_123

structural design. Notably, the paper refrains from providing a direct answer to this question but rather operates on the assumption that the role of human engineers in traditional design practices is steadily diminishing. In another work, Nguyen and Vu [6] undertook a comparative analysis of ML algorithms aimed at predicting the behavior of truss structures. They conducted this comparison through the examination of two numerical examples, evaluating the performance of multiple ML algorithms using three error metrics. The outcomes of their investigation demonstrated that AdaBoost outperformed the other six algorithms that were considered in the study.

Mai et al. [7] introduced a deep unsupervised learning framework for optimizing truss structures under various constraints. They explored several illustrative examples to showcase the effectiveness of their proposed framework, highlighting its ability to deliver high-quality optimal solutions while significantly reducing computational costs compared to traditional methods. Kang and Yoon [8] focused on the configuration and training of a two-layer ANN tailored for truss design applications, highlighting its potential roles in structural design problem-solving. Mai et al. [9] introduced a straightforward and resilient unsupervised ANN framework designed for conducting geometrically nonlinear analyses of inelastic truss structures. The fundamental concept involved utilizing the ANN to directly predict nonlinear structural responses without resorting to time-consuming incremental-iterative algorithms typical in standard finite element (FE) methods.

Khodadadi et al. [10] developed an innovative enhanced ANN model tailored for optimizing the design of truss structures which featured two distinct characteristics. Firstly, an improved initialization mechanism was introduced, leveraging opposite-based learning. Secondly, the algorithm incorporated a small set of tunable parameters to enhance its ability for exploration and exploitation. The efficacy of the method was evaluated in engineering design scenarios.

The primary objective of this study is to assess and compare the performance of various regression models in the context of predicting critical structural parameters within a plane truss model. Specifically, we aim to examine the effectiveness of linear, polynomial (both second and third degrees), and artificial neural network (ANN) regression models in estimating the maximum displacement, maximum (tensile) stress, and minimum (compressive) stress exhibited by the truss structure. By subjecting these diverse regression techniques to a rigorous evaluation, we seek to identify which model excels in capturing the intricate and nonlinear relationships that govern the behavior of the truss under varying conditions. This research objective is crucial for providing valuable insights into the selection of the most suitable regression model for structural analysis and design, with potential applications spanning fields such as civil engineering, architecture, and materials science.

Furthermore, this study also endeavors to investigate the role of model complexity in enhancing predictive performance. By systematically transitioning from simpler linear models to more intricate polynomial and ANN models, we aim to discern how adding complexity to the regression models affects their accuracy and generalization capabilities. Understanding the interplay between model complexity and predictive accuracy is essential, as it can guide practitioners and researchers in selecting the most appropriate modeling approach for diverse structural scenarios.

The remainder of this paper is structured as follows: Section 2 provides an in-depth overview of the truss analysis problem being examined, Sect. 2.2 delves into the framework for data generation, accompanied by a discussion of its primary attributes. Section 3 outlines the various regression models that have been used and assessed, whereas Sect. 4 presents and deliberates upon the findings obtained. Lastly, Sect. 5 encapsulates the conclusions drawn from this study.

2 Truss Structure and Dataset

The truss considered is a standard 10-bar plane truss, depicted in Fig. 1. The model has six nodes and ten elements in total, numbered from 1 to 10, as shown in the figure. The members have a modulus of elasticity (E) equal to 10,000 ksi ($\approx$ 68.95 GPa), while the length L is equal to 360 in (9.144 m). Each of the applied nodal loads is equal to 100 kip ($\approx$ 444.82 kN). This truss structure has been extensively used as a benchmark problem in structural optimization, with the objective being to minimize the weight of the structure under constraints on stresses and displacements [11–13].

The dataset comprises 2000 data points, each featuring five input variables and three output variables, resulting in a tabular format with dimensions of 2000 rows and 8 columns. The outputs of the dataset are computed using an FE analysis code written in MATLAB. The whole dataset is stored as a text file with 253 kB file size. A portion of the data, specifically 15% (or 300 data points), is reserved for testing purposes and remains excluded from the training of any of the models. For comprehensive information about the input and output variables, please refer to the subsequent subsections.

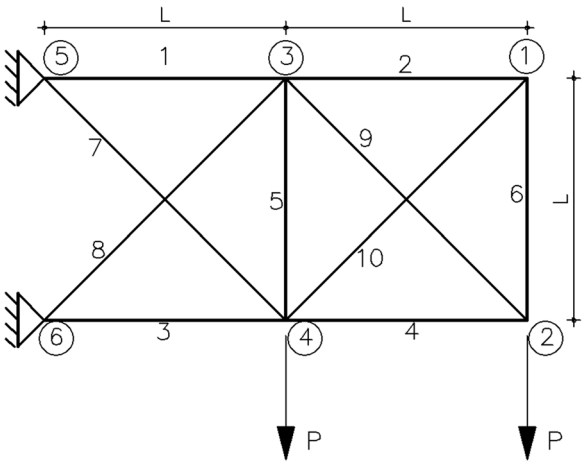

Fig. 1 The 10-bar truss structure considered

2.1 Input and Output Variables

There are five input variables related to the cross-sectional area of individual members. Given that there are a total of ten members, it becomes necessary to organize them into specific groups. Table 1 provides an overview of both the input variables and the grouping arrangement for these elements. For example, input variable I-4 is the section area of members 7 and 9 of the truss (diagonals in the NW-SE direction).

There are three output variables: (1) Maximum absolute vertical displacement (in the y direction), in inches (in), (2) Maximum stress (ksi) of all members (maximum tensile stress), (3) Minimum stress (ksi) of all members, in absolute terms (i.e., maximum compressive stress).

2.2 Generation of the Dataset

The five input variables correspond to section areas, each falling within the specified range of [0.1, 35]. These inputs are created as random values within this range using a uniform distribution and are then randomly combined with each other. Consequently, the first five columns of the dataset consist of random numbers selected from this specified range ($2000 \times 5 = 10{,}000$ random numbers in total for the first five columns). The output variables, on the other hand, are determined through FE analysis of the truss structure. This analysis is carried out 2000 times, resulting in the generation of 2000 distinct output values.

In Fig. 2, box plots depict the input and output variables, highlighting distinct characteristics. It is evident that the input variables exhibit a uniform distribution

Table 1 Input variables and grouping of the truss elements

Input variable	Elements	Range	Units
I-1 (A_1, A_2)	1, 2 (top, horizontal)	[0.1, 35]	in^2
I-2 (A_3, A_4)	3, 4 (bottom, horizontal)	[0.1, 35]	in^2
I-3 (A_5, A_6)	5, 6 (vertical)	[0.1, 35]	in^2
I-4 (A_7, A_9)	7, 9 (diagonal NW-SE)	[0.1, 35]	in^2
I-5 (A_8, A_{10})	8, 10 (diagonal SW-NE)	[0.1, 35]	in^2

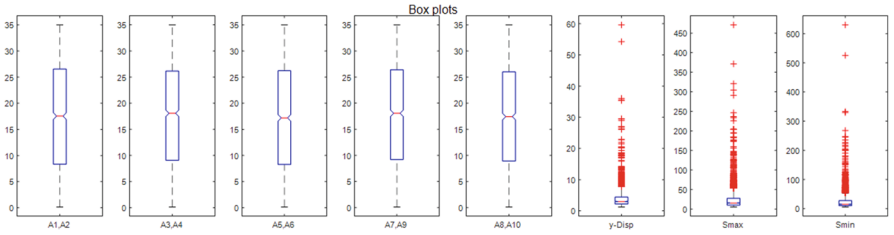

Fig. 2 Box plots of input and output variables

within the [0.1, 35] range. In contrast, the distribution of the output variables shows considerable non-uniformity, marked by the presence of numerous outliers.

3 Regression Models

3.1 Linear (First Order) and Polynomial Regression (Second and Third Order)

Linear regression [14] (LR) is a fundamental statistical method used to model the relationship between a dependent variable and one or more independent variables. This method assumes that the relationship between the variables is linear, i.e., can be represented by a straight line. In its simplest form, known as simple linear regression, the model predicts the dependent variable based on a single independent variable (one input). It calculates the best-fitting line through the data points, where "best-fitting" is defined as minimizing the sum of the squares of the differences between the observed and predicted values. This line is represented by the equation "$y = mx + c$," where y is the dependent variable, x is the independent variable, m is the slope of the line, and c is the y-intercept. Linear regression is widely used in various fields for predictive analysis and inferential statistics, owing to its simplicity and interpretability. In our specific analysis, we deal with a more intricate scenario involving five input variables and three output variables. Consequently, the number of independent variables amounts to five.

Polynomial regression [14] is a versatile statistical method employed to capture nonlinear relationships between a dependent variable and one or more independent variables. Unlike linear regression, which assumes a linear relationship, polynomial regression allows for the modeling of more intricate patterns by introducing polynomial terms of the independent variables into the equation.

In our analysis, we employ two distinct polynomial regression models to better capture complex relationships within our dataset. The first model is a second-degree polynomial regression model, which introduces second-degree terms and interactions between the independent variables. For instance, in a second-degree model featuring two independent variables, x and y, the terms incorporated would include the constant, x, x^2, y, xy, and y^2. This model extends beyond the simplicity of the linear (first degree) model, allowing us to account for nonlinearities and more intricate patterns.

Moreover, we also utilize a third-degree polynomial regression model, which incorporates third-degree terms and further enriches the model by including additional interactions between the independent variables. This third-degree model surpasses the complexity of the second-degree model, enabling us to comprehensively explore and represent the nonlinear relationships and higher-order interactions among our five independent variables, thereby enhancing our understanding of the underlying data dynamics.

To accommodate the three outputs, we run the linear and polynomial regression models three times each, generating three distinct sub-models for each model, each tailored to predict one of the three output variables. This is not the case with the Artificial Neural Network model, which can handle all output variables simultaneously.

3.2 Artificial Neural Network

Artificial Neural Network (ANN) regression is a robust ML technique used to model complex relationships between input variables and predict output values, which has several applications in structural engineering problems [15–19]. In our study, we implement a back-propagation ANN regression model with specific architectural characteristics to suit our research needs. Our ANN model consists of two hidden layers, each containing 25 neurons, making it a deep neural network capable of capturing intricate patterns and nonlinear dependencies within the data. The input layer comprises five neurons, corresponding to our five independent variables, while the output layer consists of three neurons, aligning with the three output variables we aim to predict.

For the training process, we employ the Levenberg–Marquardt training algorithm [20], a widely used optimization method for fine-tuning neural network weights and biases. To ensure the model's robustness and generalization capability, we divide our dataset into three distinct subsets: 15% of the data is used for testing, the same as with the previous models, 70% of the data is allocated for training itself, while another 15% is used (during training) for validation purposes. The validation procedure monitors the performance of the network on the validation data during training to prevent overfitting. This comprehensive approach ensures that our ANN regression model not only learns from the available data but also generalizes well to make accurate predictions on new and unseen data points, ultimately enhancing the reliability of the research findings. It has to be noted that the testing set is exactly the same for all regression models. In all cases, the testing is done after the model has been fully developed and the testing data remains unseen to the model until then.

3.3 Performance Metrics

To assess the performance of each model, we utilize three separate metrics: (i) the Root Mean Squared Error (RMSE), (ii) the Pearson correlation coefficient (R), and (iii) the Mean Normalized Gross Error (MNGE). The formulas for these three metrics are outlined in Eqs. (2)–(4):

$$\overline{p} = \frac{1}{N}\sum_{i=1}^{N} p_i, \quad \overline{r} = \frac{1}{N}\sum_{i=1}^{N} r_i \tag{1}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(p_i - r_i)^2} \tag{2}$$

$$R = \frac{\sum_{i=1}^{N}(p_i - \overline{p})(r_i - \overline{r})}{\sqrt{\sum_{i=1}^{N}(p_i - \overline{p})^2} \cdot \sqrt{\sum_{i=1}^{N}(r_i - \overline{r})^2}} \tag{3}$$

$$\mathrm{MNGE} = \frac{1}{N}\sum_{i=1}^{N}\frac{|p_i - r_i|}{r_i} \tag{4}$$

In these equations, N represents the total count of data points in the entire dataset, or within a specific subset such as the training or testing subset, depending on the subset chosen for evaluating the regression model. In addition, r_i represents the real (target) value, p_i denotes the predicted value, and $\overline{r}$, $\overline{p}$ denote the mean values of the real and predicted values, respectively. RMSE is expressed in the units of the corresponding output, while R and MNGE are unitless quantities. More detailed information about these metrics is available in reference [21].

Additionally, we employ the Taylor diagram, which integrates three statistical parameters—namely, (i) the Centered Root Mean Square Difference (CRMSD), (ii) R, and (iii) the Standard Deviation σ—into a single, easily interpretable diagram. The Taylor diagram proves valuable for summarizing and comparing the relative strengths of various models, as discussed in Ref. [21]. The formula for the calculation of CRMSD is given in Eq. (5).

$$\mathrm{CRMSD} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}[(p_i - \overline{p}) - (r_i - \overline{r})]^2} \tag{5}$$

4 Numerical Results

Figure 3 displays plots depicting the predicted values contrasted with the actual (target) values for each model and output variable. Additionally, the graph features a reference line, $y = x$, representing an ideal match, accompanied by two error boundary lines signifying a 10% prediction error tolerance.

Table 2 presents the error metrics values for each model and each output. The target values, denoting a perfect match, are also shown in the table. Figure 4

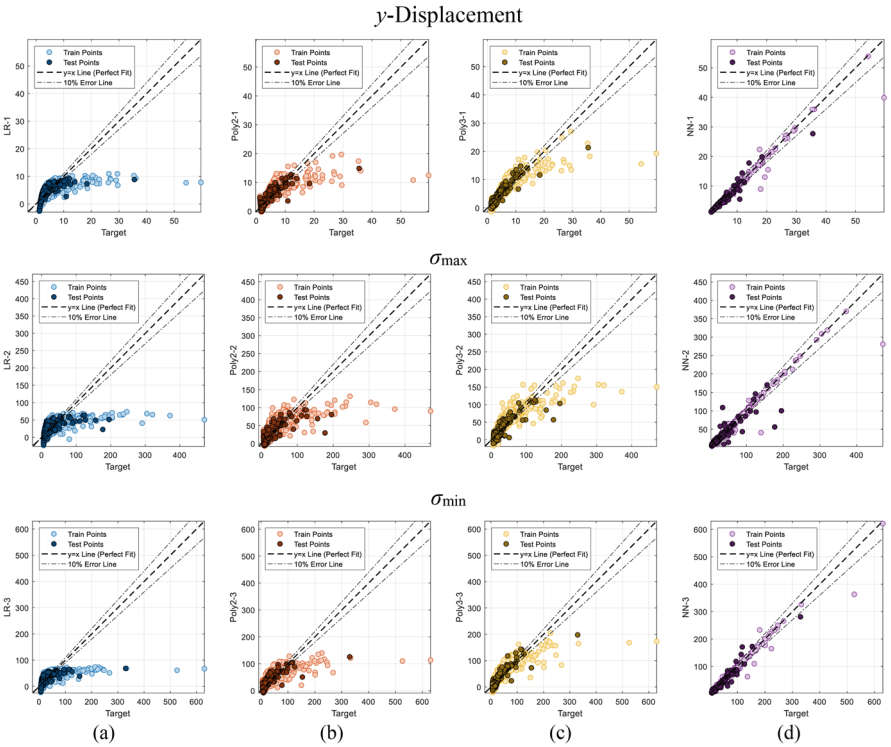

Fig. 3 Predicted vs. actual (target) values for all four models, for all three outputs: (**a**) LR Model, (**b**) Poly2 Model, (**c**) Poly3 Model, (**d**) NN Model

Table 2 Error metrics values

		All data (train and test)			Test data		
	Model	RMSE	R	MNGE	RMSE	R	MNGE
Output variable	Target	0	1	0	0	1	0
1 (y-displacement)	LR	2.96	0.603	0.403	2.32	0.638	0.408
	Poly2	2.41	0.760	0.307	1.81	0.798	0.311
	Poly3	1.96	0.849	0.263	1.39	0.888	0.264
	NN	0.62	0.987	0.030	0.71	0.971	0.043
2 (σ_{max})	LR	25.54	0.571	0.701	20.19	0.570	0.677
	Poly2	21.08	0.735	0.551	16.46	0.740	0.538
	Poly3	17.03	0.837	0.459	13.86	0.826	0.470
	NN	6.80	0.976	0.072	11.39	0.884	0.106
3 (σ_{min})	LR	28.27	0.559	0.737	22.25	0.625	0.693
	Poly2	23.58	0.722	0.572	17.97	0.782	0.580
	Poly3	19.58	0.819	0.484	13.93	0.880	0.489
	NN	5.62	0.987	0.075	6.72	0.972	0.092

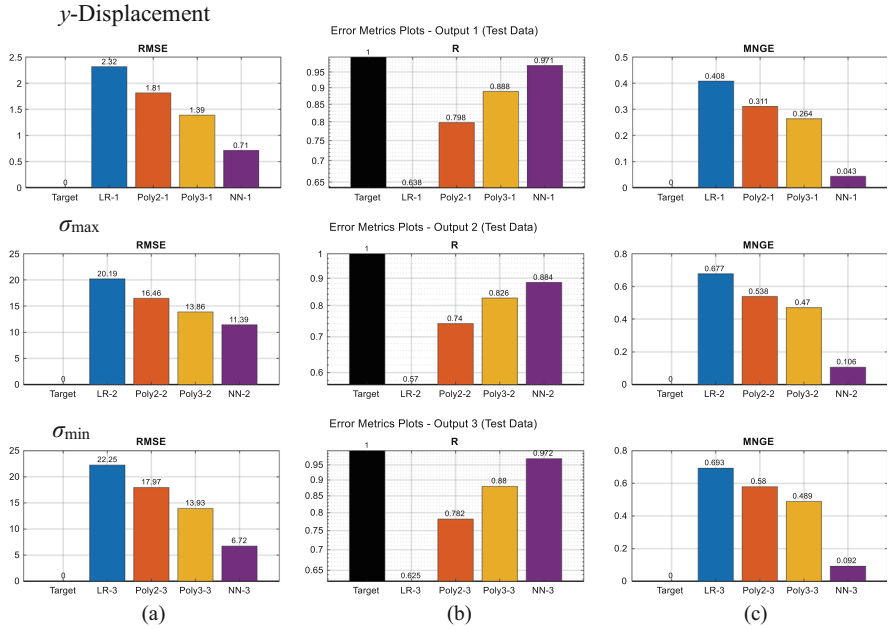

Fig. 4 Error metrics for all models and all outputs (Test data only): (**a**) RMSE, (**b**) *R*, (**c**) MNGE

illustrates the error metrics for all four models across each output variable, exclusively for the test dataset. To facilitate comparison, we have included an initial column that serves as a reference point denoting the "target" value for each metric. In other words, it represents the value associated with a perfect fit, which is zero for RMSE and MNGE and 1 for *R*. Importantly, these diagrams exclusively pertain to test predictions, omitting the training data employed by the models. This deliberate exclusion ensures a fair assessment of the models' generalization capabilities, highlighting their performance on unseen data points. While it is evident that all models perform optimally on the training set, our focus here centers on the test set, where models typically exhibit comparatively lower performance.

Notably, the ANN model outperforms the other models across all considered metrics. Additionally, the introduction of complexity to simpler models proves beneficial, with the second-degree polynomial model demonstrating superior performance compared to the linear model, and the third-degree polynomial model surpassing the second-degree model in terms of predictive accuracy.

The MNGE metric provides valuable insights into the model's real-world performance. Specifically, when analyzing the ANN model's MNGE value of 0.043 for the first output, it signifies an average prediction error of 4.3% across all test data set predictions for the maximum *y*-displacement of the nodes.

Digging deeper into the output data, we find that the median error is even lower than the MNGE, at 2.9%. It is important to note that in this context, MNGE for a single observation represents the bias error (the difference between the predicted

value and the real value) over the real value, in absolute terms. In stark contrast, the MNGE values for the other models consistently exceed 25%, rendering them impractical for real-world applications.

4.1 Taylor Diagrams

In Fig. 5, we display three Taylor diagrams, each corresponding to an individual output variable, and exclusively based on the data of the test set. Within these diagrams, every model is symbolized as a distinct point, with the reference target point positioned at the diagram's bottom. In a Taylor's diagram, the proximity of a model point to this target point serves as an indicator of the model's predictive accuracy—closer points signify better predictions. As before, it remains evident that the ANN model consistently outperforms the other models. Furthermore, the trends

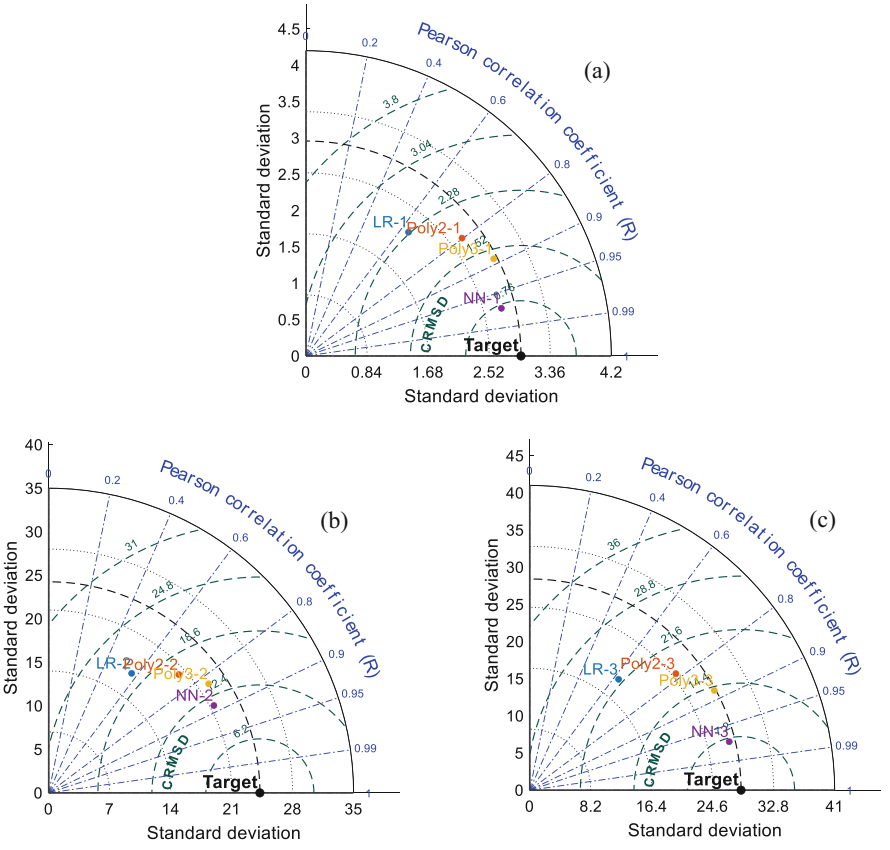

Fig. 5 Taylor diagrams for all models, for all three outputs (test data only): (**a**) output 1 (y-displacement), (**b**) output 2 ($\sigma_{\max}$), (**c**) output 3 ($\sigma_{\min}$)

observed in the previous paragraph, regarding error metrics, are reaffirmed here, underscoring the ANN model's superior performance across these visual representations as well.

5 Conclusions

Our study has undertaken a comprehensive evaluation of various regression models, encompassing linear, second- and third-degree polynomial, and ANN models, to predict crucial parameters within a plane truss model. Our findings unequivocally establish the supremacy of the ANN model, demonstrating its exceptional aptitude for capturing intricate nonlinear relationships within the dataset. Notably, our exploration has shed light on the pivotal role of model complexity. The augmentation of complexity, observed in the transition from linear to polynomial models, has yielded tangible improvements in predictive performance. The superior performance exhibited by the ANN model holds promise for a wide array of practical engineering applications.

While our study represents a significant stride in comprehending the performance of regression models, promising avenues for further investigation persist. Future research might delve deeper into hyperparameter tuning, refined feature selection techniques, and the exploration of model architectures, with the aim of further optimizing the ANN model's capacity. Moreover, the realm of regression modeling extends far beyond the models assessed in this study. The incorporation of additional models, such as Support Vector Machine (SVM), K-nearest neighbors (KNN), and Decision Tree, among others, could offer additional insights into predictive accuracy.

The ANN's predictions of structural behavior serve a dual purpose. They serve as valuable surrogate modeling tools, reducing the computational demands for the analysis of intricate structural models with minimal compromise in accuracy, but also open doors to various practical applications. For instance, they can be integrated into the context of structural optimization, furnishing predictions of structural responses and obviating the need for FE analysis in every optimization iteration. This capability can prove highly beneficial, particularly in the initial phases of optimization.

However, it is imperative to approach such predictions with discernment. While the overall performance of advanced regression models remains commendable, certain cases may yield predictions significantly divergent from actual output values. This is substantiated by the presence of outliers observed in the "Predictions vs. Targets" plots, even in the case of the best-performing model. Hence, while regression models offer remarkable predictive capabilities, their outputs should be interpreted cautiously, with consideration for potential outliers and/or anomalies.

References

1. Lagaros, N.D., Plevris, V.: Artificial intelligence (AI) applied in civil engineering. Appl. Sci. **12**(15), 7595 (2022). https://doi.org/10.3390/app12157595
2. Lagaros, N.D., Plevris, V. (eds.): Artificial Intelligence (AI) Applied in Civil Engineering, p. 698. MDPI, Basel (2022). https://doi.org/10.3390/books978-3-0365-5084-8
3. Lu, X., Plevris, V., Tsiatas, G., De Domenico, D.: Editorial: artificial intelligence-powered methodologies and applications in earthquake and structural engineering. Front. Built Environ. **8**, 876077 (2022). https://doi.org/10.3389/fbuil.2022.876077
4. Lagaros, N.D., Tsompanakis, Y. (eds.): Intelligent Computational Paradigms in Earthquake Engineering. Idea Group Publishing (2006)
5. Málaga-Chuquitaype, C.: Machine learning in structural design: an opinionated review. Front. Built Environ. **8**, 815717 (2022). https://doi.org/10.3389/fbuil.2022.815717
6. Nguyen, T.-H., Vu, A.-T.: A Comparative Study of Machine Learning Algorithms in Predicting the Behavior of Truss Structures. Springer Singapore, Singapore (2021)
7. Mai, H.T., Lieu, Q.X., Kang, J., Lee, J.: A novel deep unsupervised learning-based framework for optimization of truss structures. Eng. Comput. **39**(4), 2585–2608 (2023). https://doi.org/10.1007/s00366-022-01636-3
8. Kang, H.-T., Yoon, C.J.: Neural network approaches to aid simple truss design problems. Comput. Aided Civ. Inf. Eng. **9**(3), 211–218 (1994). https://doi.org/10.1111/j.1467-8667.1994.tb00374.x
9. Mai, H.T., Lieu, Q.X., Kang, J., Lee, J.: A robust unsupervised neural network framework for geometrically nonlinear analysis of inelastic truss structures. Appl. Math. Model. **107**, 332–352 (2022). https://doi.org/10.1016/j.apm.2022.02.036
10. Khodadadi, N., Talatahari, S., Gandomi, A.H.: ANNA: advanced neural network algorithm for optimisation of structures. Proc. Inst. Civ. Eng. Struct. Build. **177**, 1–23 (2023). https://doi.org/10.1680/jstbu.22.00083
11. Ghasemi, M.R., Hinton, E., Wood, R.D.: Optimization of trusses using genetic algorithms for discrete and continuous variables. Eng. Comput. **16**(3), 272–303 (1997). https://doi.org/10.1108/02644409910266403
12. El-Sayed, M.E.M., Jang, T.S.: Structural optimization using unconstrained nonlinear goal programming algorithm. Comput. Struct. **52**(4), 723–727 (1994). https://doi.org/10.1016/0045-7949(94)90353-0
13. Plevris, V.: Innovative Computational Techniques for the Optimum Structural Design Considering Uncertainties, p. 312. National Technical University of Athens, Athens (2009). https://doi.org/10.12681/eadd/17936
14. Montgomery, D.C., Peck, E.A., Vining, G.G.: Introduction to Linear Regression Analysis, 6th edn. Wiley, Hoboken (2021)
15. Solorzano, G., Plevris, V.: An open-source framework for modeling RC shear walls using deep neural networks. Adv. Civ. Eng. **2023**, 7953869 (2023). https://doi.org/10.1155/2023/7953869
16. Solorzano, G., Plevris, V.: DNN-MLVEM: a data-driven macromodel for RC shear walls based on deep neural networks. Mathematics. **11**(10), 2347 (2023). https://doi.org/10.3390/math11102347
17. Solorzano, G., Plevris, V.: ANN-based surrogate model for predicting the lateral load capacity of RC shear walls. In: 8th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS 2022), 5–9 June 2022, Oslo, Norway (2022). https://doi.org/10.23967/eccomas.2022.050
18. Plevris, V., Solorzano, G.: Prediction of the eigenperiods of MDOF shear buildings using neural networks. In: 8th ECCOMAS Thematic Conference on Computational Methods in Structural Dynamics and Earthquake Engineering (COMPDYN 2021), 28–30 June 2021, ECCOMAS: Athens, Greece, pp. 3894–3911. Eccomas Proceedia. https://doi.org/10.7712/120121.8755.20415

19. Sharib, S., Ahmad, N., Plevris, V., Ahmad, A.: Prediction models for load carrying capacity of RC wall through neural network. In: 14th ECCOMAS Thematic Conference on Evolutionary and Deterministic Methods for Design, Optimization and Control (EUROGEN 2021), 28–30 June 2021, ECCOMAS: Streamed from Athens, Greece, pp. 132–142. Eccomas Proceedia. https://doi.org/10.7712/140121.7956.18529
20. Hagan, M.T., Menhaj, M.B.: Training feedforward networks with the Marquardt algorithm. IEEE Trans. Neural Netw. **5**(6), 989–993 (1994)
21. Plevris, V., Solorzano, G., Bakas, N.P., Ben Seghier, M.E.A.: Investigation of performance metrics in regression analysis and machine learning-based prediction models. In: 8th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS 2022), 5–9 June 2022, Oslo, Norway (2022). https://doi.org/10.23967/eccomas.2022.155