
Proceedings Paper:
Rigby, S. orcid.org/0000-0001-6844-3797, Smyl, D., Rhodes, T. et al. (2 more authors) (2025) On the use of artificial neural networks for inverse analysis of single degree of freedom response of blast loaded structures. In: Syngellakis, S. and Teixeira-Dias, F., (eds.) WIT Transactions on The Built Environment. 17th International Conference on Structures under Shock and Impact (SUSI 2025), 09-11 Jun 2025, Edinburgh, UK. WIT Press , pp. 213-225. ISBN: 9781784664954 ISSN: 1743-3509 EISSN: 1743-3509

https://doi.org/10.2495/SUSI250191

ON THE USE OF ARTIFICIAL NEURAL NETWORKS FOR INVERSE ANALYSIS OF SINGLE DEGREE OF FREEDOM RESPONSE OF BLAST LOADED STRUCTURES

SAMUEL E. RIGBY[1,2], DANNY SMYL[3], TYLER RHODES[2], ANQI CHEN[2] & GAUTHIER STIEGLER[2]
[1]School of Mechanical, Aerospace and Civil Engineering, University of Sheffield, Sheffield, UK
[2]Arup Resilience, Security and Risk, London, UK
[3]School of Civil and Environmental Engineering, Georgia Institute of Technology, Georgia, USA

ABSTRACT

The Single Degree of Freedom (SDoF) method is widely used to evaluate the response of structures or structural elements under blast loading. Here, the structure in question is transformed into a single-point equivalent for which a single equation of motion can be solved. While the SDoF method is orders-of-magnitude faster than alternatives such as the finite element method, both are focussed only on solving the forward problem (inputs → outputs). In practise, however, a required performance limit is known (peak displacement, support rotation, etc.) and an adequate structure should be provided so as to not exceed that limit. This necessitates some form of iteration as the SDoF equation of motion cannot be simply inverted. Alternatively, machine learning techniques such as artificial neural networks (ANNs) may be used. ANNs are agnostic to input data type and therefore can just as easily learn patterns between input and output data as they can between output and input data. This paper presents a novel application of ANNs to rapidly solve the inverse problem (outputs → inputs) for SDoF structures subjected to blast loads. 180,000 SDoF analyses were run for 36 British Steel Universal Column sections (5,000 runs for each). The subsequent data was used to train an ANN classification network to suggest a section size deemed to meet a target value of support rotation, for a given set of basic inputs (span, peak force, impulse). The ANN is able to perform to a high degree of accuracy (> 90% correct classification of the test data set) and performs well in unseen 'design cases', suggesting that machine learning could be a highly valuable tool to aid in solving inverse problems relating to blast loading.
Keywords: artificial neural network, blast, classification, inverse modelling, SDoF.

1 INTRODUCTION

Protection of critical national infrastructure is predicated on the ability of engineers to accurately model the effects of blast on structures. Whether from large-scale industrial explosions such as Tianjin (2015) and Beirut (2020), acts of terror, or during ongoing conflicts such as those in Ukraine and the Middle East, explosions impart loads on structures that are several orders of magnitude larger than those which civil infrastructure is typically designed to resist. Furthermore, blast loading parameters are highly sensitive to the exact composition, size, shape, and location of the explosive, and are significantly influenced by non-linear physical processes which occur when a blast wave reflects off and diffracts around obstacles (e.g. in an urban environment), or when two wavefronts coalesce [1].

Whilst tools such as computational fluid dynamics (CFD) and the finite element method (FEM) provide solutions to the underlying conservation equations (e.g. they are 'physics-based') with a potentially very high degree of spatiotemporal resolution, they cannot easily incorporate the underlying uncertainties in explosion events. That is, whilst they may provide a near-exact answer to a given set of well-defined inputs, they provide only *one* answer (for, say, a few hours of computational time), and thus their use in probabilistic, risk-based design is currently limited. CFD/FEM therefore finds a natural place more in the detailed design stage, rather than the scheme design stage, and there remains a very clear need for a tool which provides 'good enough and quick enough' results.

WIT Transactions on The Built Environment, Vol 215, © 2025 WIT Press
www.witpress.com, ISSN 1743-3509 (on-line)
doi:10.2495/SUSI250191

This is where we find the natural habitat of the Single Degree of Freedom (SDoF) method; at the stage where engineers need to rapidly assess a large number of potential designs and quickly ascertain which would work in principle, which would be prohibitively expensive (in terms of cost, or carbon) to implement, which simply would not work, etc. The SDoF method works by transforming a 'real life' system with distributed mass, stiffness, loading, and resultant displacements (and stresses, strains, etc.) into an 'equivalent' single point [2], with one value of mass, stiffness, applied load, and damping coefficient (although this is typically ignored in blast applications due to the fact that its influence on first peak displacement is negligible [3]). This single point normally represents the point of maximum displacement of the 'real life' structure (e.g. midspan of a beam or column), and it is allowed to translate through one degree of freedom only, hence the name.

Transforming a structure into an equivalent single point has the effect of reducing a large system of simultaneous equations:

$$\boldsymbol{M}\ddot{\boldsymbol{z}}(t) + \boldsymbol{K}\boldsymbol{z}(t) = \boldsymbol{F}(t), \tag{1}$$

into a single equation:

$$K_M m\ddot{z}(t) + K_L k z(t) = K_L F(t), \tag{2}$$

where $\boldsymbol{M}$ and $\boldsymbol{K}$ are mass and stiffness matrices, and $\ddot{\boldsymbol{z}}$, $\boldsymbol{z}$ and $\boldsymbol{F}$ are vectors of time-varying acceleration, displacement and applied force at each node, whereas m, k, $\ddot{z}$ and z are scalar values of the above. Mass and load transformation factors, K_M and K_L, are employed in the SDoF method to ensure that kinetic energy, internal strain energy, and work done are conserved when moving from 'real life' to 'equivalent' systems [2]. The SDoF method offers orders-of-magnitude reductions in run-time compared to FEM and has demonstrated high levels of agreement with experimental data (see Rigby et al. [4] as an example).

2 THE DESIGN PROCESS AS AN INVERSE METHOD

The methods described previously (CFD, FEM, SDoF), although offering substantial differences in terms of fidelity of solution and computational expense, all still solve the problem in a 'forward model' sense. That is, for a given set of inputs (load, span, support conditions, etc.) and design variables (material type, depth, width, section modulus, etc.), the response of that system can be calculated; for the SDoF method this is peak displacement or support rotation.

In terms of protective *design*, however, the aim is to limit response to some acceptable value, and whilst the inputs are normally constrained (e.g. the element *has* to span a certain distance), the design variables are chosen such that the response limit is not exceeded. These response limits are given for support rotation and ductility ratio (peak deflection divided by elastic deflection limit), for either Category 1 (for the protection of personnel and equipment from the effects of primary and secondary fragments and falling portions of the structure under the action of blast loading) or Category 2 (for the protection of structural elements themselves from collapse under the action of blast loading) protection levels (see Table 1).

This means that the design process can effectively be viewed as an inverse problem (outputs → inputs), with *analysis* being the corresponding forward problem (inputs → outputs), a conceptual point introduced in Gallet et al. [6] and expanded upon in Gallet et al. [7]. The aim of this paper is to explore this idea in the context of blast response of structural elements. In particular, we devise a machine learning model to rapidly solve the

Table 1: Rotation and deformation limits for common structural components for Category 1 and Category 2 protection levels. *(Source: Adapted from Cormie et al. [5].)*

Element	Protection category	Support rotation	Ductility ratio
Structural steel beams/plates	1	2°	10
	2	12°	20
Reinforced concrete beams/ slabs	1	2°	–
	2	4°	–

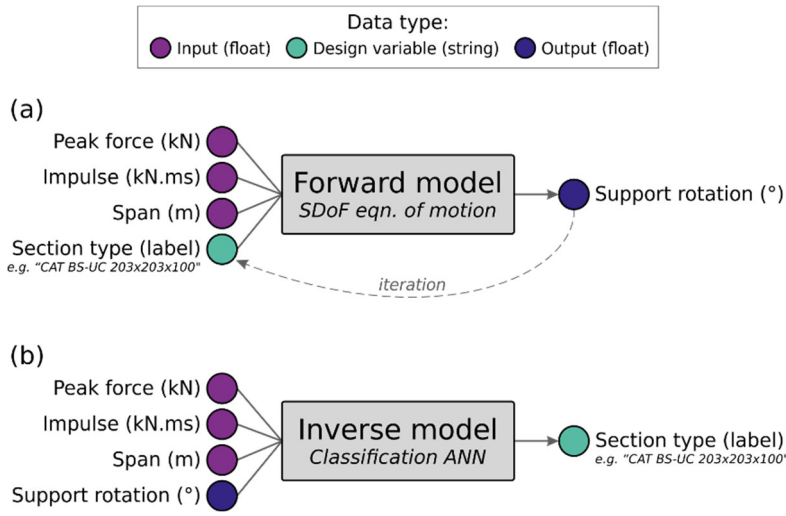

Figure 1: (a) Conceptual overview of the standard 'forward model' approach; and (b) the 'inverse model' approach adopted in this study.

inverse/design problem, as in Fig. 1, meaning that the design process can be undertaken in a single step, rather than through iteration of the forward model, which would otherwise be the approach adopted.

When considering steel columns specifically, as can be seen in Fig. 1, the forward model requires an initial guess of section type, e.g. a choice of one of the sections available from the BS-UC (British Steel Universal Column) catalogue [8] (see the table in the Appendix for the full list). This section is then run through the forward model (which, although typically inexpensive when solving the SDoF equation of motion, may still require several thousand timesteps to be computed), the results extracted, compared against the response limit, and a new section chosen if necessary. Clearly, this may require several iterations before the most suitable section is chosen, and the required computational steps could number in the many thousands. Alternatively, an inverse model will 'jump' straight to section type for a given response limit in a *single* computation.

3 OVERVIEW OF DATASET

A total of 180,000 individual SDoF analyses were performed, corresponding to 5,000 runs for each of the 36 BS-UC column types. Analyses were performed using *Ergo Compute*, Arup's in-house SDoF solver. The columns were simply supported, and the applied loading was a linearly-decaying reverse-ramp load, uniformly distributed along the column's length,

acting on the column's major axis. The steel was modelled with a yield strength of 355 MPa, elastic modulus of 200 GPa, and a *dynamic increase factor* of 1.17 to incorporate strain rate sensitivity of the steel [5]. The columns had a bilinear elastic-perfectly-plastic resistance function, i.e. they performed elastically (linearly increasing resistance) until the plastic moment capacity was reached (plastic modulus, Z_{pl}, multiplied by yield stress, σ_y), and then deformed plastically with a constant resistance thereafter. No damping was specified, and analyses were terminated when velocity of the single degree of freedom reached zero (and therefore displacement had reached a maximum). The forcing function was input as a peak pressure and positive phase duration, which can simply be converted to peak force (*Ergo* allows the user to specific a *loaded length*, with force then given by the multiple of pressure, span, and loaded length. For simplicity, in this study the loaded length was set to 1m, and the peak force was controlled by peak pressure and span only) and impulse (integral of force with respect to time. Since the loading decays linearly, this is simply half the peak force multiplied by positive phase duration) in post-processing. An example output is shown in Fig. 2.

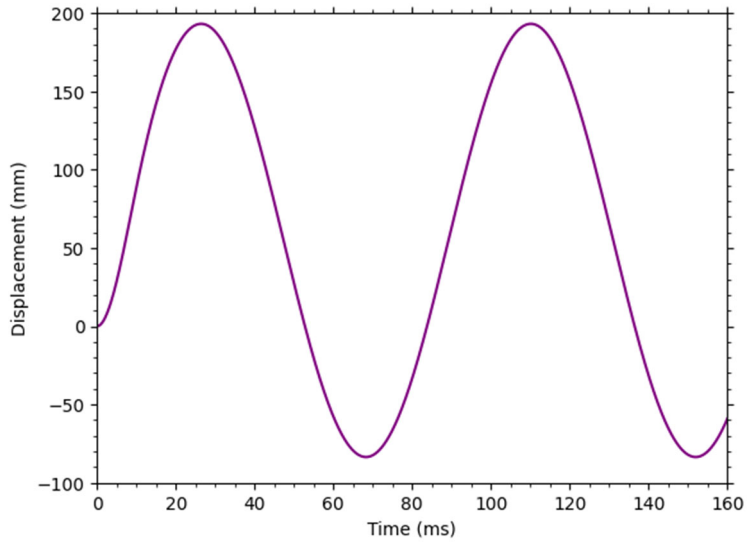

Figure 2: Example *Ergo* output for an 8 m spanning 203 × 203 × 113 column with 1 MN peak force and 10 ms duration. Peak displacement = 193.1 mm, peak support rotation = 2.76°. Note: analysis was run beyond first zero velocity (~30 ms) for clarity of presentation.

Monte Carlo sampling was used to initialise each simulation. Span was randomly selected from the interval 2–10 m (uniform probability). Peak pressures were randomly selected from the interval 9–5000 kPa, and positive phase durations from the interval 0.1–40 ms (both with a uniform probability in *log*-space), loosely corresponding to Hopkinson-Cranz scaled distances of 1–20 kg/m$^{1/3}$ and charge masses of 0.1–1,000 kg [9]. Basic logic was used to reject any results where peak rotation exceeded 45°, and several 'passes' were undertaken to ensure the outputs were reasonably well distributed between 0–45°, with a slight preference towards rotations of 2°, 4°, and 12°, as these are common limits used in design (e.g. as in

Table 1). This naturally resulted in the tendency for higher values of force and impulse to be associated with stiffer columns (larger plastic modulus).

In summary, each independent *Ergo* analysis ultimately returned five pieces of information: peak force (kN), impulse (kN.ms), span (m), section classification (in the form 'CAT BS-UC XXXxYYYxZZZ'; one of 36 available BS-UC sections), and maximum support rotation (°). This comprises the entire 'labelled' dataset which was used to train the machine learning inverse model described in the next section. Examples of the labelled data for the columns with largest and smallest section modulus are shown in Fig. 3.

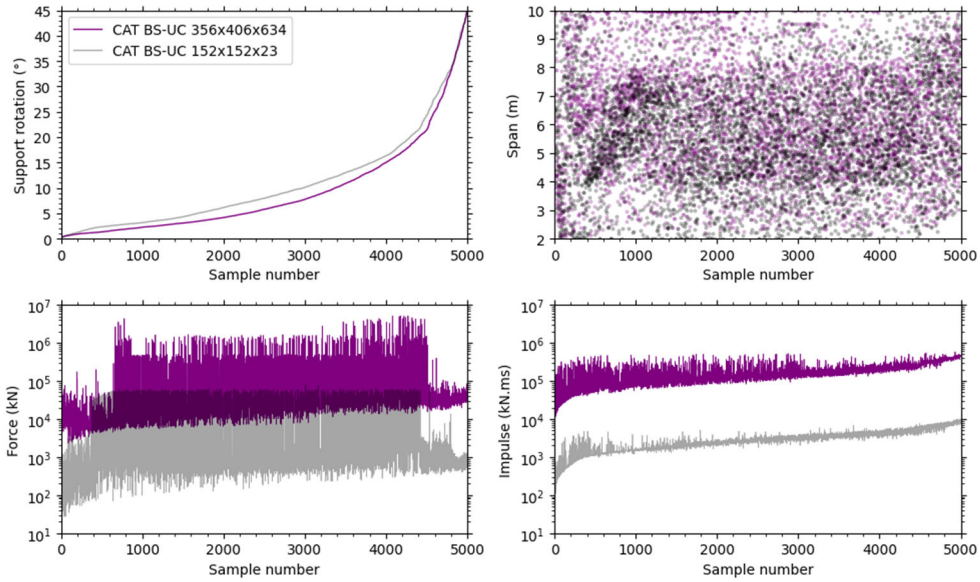

Figure 3: Labelled data for columns with largest (356 × 406 × 634) and smallest (152 × 152 × 23) plastic modulus, sorted by increasing support rotation.

4 CLASSIFICATION ARTIFICIAL NEURAL NETWORK

Classification artificial neural networks (ANNs) are a type of supervised machine learning model used to classify labelled data into a number of discrete categories and have recently been used to predict failure mode of reinforced concrete walls under near-field explosions [10].

In general, ANNs are designed to 'learn' the connections between input data and output data through successive 'training'. A network is formed of three main structures: input, hidden, and output layers (see Fig. 4). Each of these layers is formed of connected nodes, or 'neurons', and an input signal (in the form of numerical values fed into the input layer) is then propagated through the network until, ultimately, the ANN produces a prediction in the output layer. Each neuron will send a signal along a pathway to every (if the network is what is known as 'fully connected') node in the next layer, and whether this node then sends the signal onwards, and how strong the resulting signal is, is a function the combined strengths of the incoming signals. This strength is adjusted using tuneable parameters – the 'weights' and 'biases' – along the pathways between neurons.

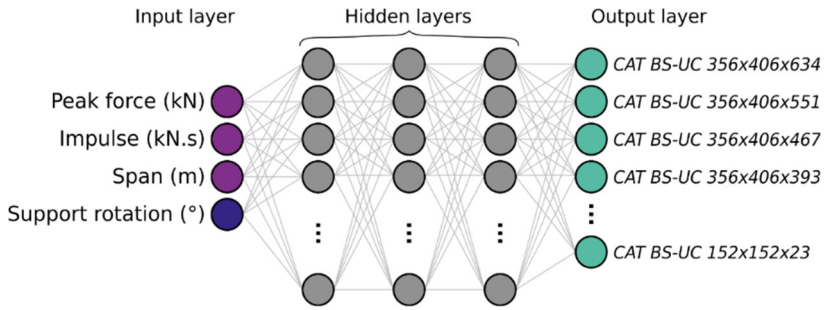

Figure 4: Architecture of the classification neural network adopted in this study.

Mathematically, the output signal of a given node, y, is given as [11]:

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{3}$$

where x_i is the signal received from the ith node on the previous layer, w_i is the weight, b is the bias, n is the number of nodes in the previous layer, and $f(\cdots)$ indicates that the composite signal is passed through an 'activation' function before going on to the next layer. The activation function (with ReLU, the 'rectified linear unit', used in this study) introduces non-linearity to the network (allowing complex relations between input and output to be learned), as well as suppressing low magnitude, potentially irrelevant connections and also ensuring computational efficiency.

First, the labelled data is separated into training, validation, and test data. The training dataset is available to the model to learn from throughout, whereas the validation and test datasets are kept aside and do not form part of the model's training. The model is sequentially asked to predict the validation dataset during training, so its performance can be monitored, whereas the test dataset is used only after training to assess how well the final model can predict *unseen* data, i.e. 'generalise'.

The network is then trained through successive passes of labelled data, where for a given set of inputs the network is asked to produce an output (or 'prediction'), and its performance compared against the known output (or 'target'). Training is typically done in 'batches'; the number of predictions the network is asked to make before weights and biases are adjusted. Once the network has attempted to predict the entire training dataset, one 'epoch' is said to have passed, and training can continue for a specified number of epochs or until some performance criteria has been met (hence the need for the validation dataset).

In regression networks, the output is typically numerical, with the output predictions readily assessed against targets using measures such as mean absolute error or root mean square error. In classification tasks, however, the network is asked to assign a *confidence value* to each of the neurons in the output layer. The 'true' target, therefore, would be a series of zeros for each incorrect label, and a single 1.00 for the correct label. The classification network therefore outputs a series of numbers which sum to 1.00 (achieved using the *softmax* function on the output layer).

In this study, column classes were listed in order of decreasing plastic section modulus (see Appendix) in a slight reworking of the order in which they appear in the BS-UC catalogue. This was deemed necessary to ensure that similar predictions (in terms of input)

propagated through the network in a similar manner and ultimately arrived at predictions in close proximity to one another. In that sense, the index of each class has some physical significance, i.e. it is the rank order of plastic section modulus.

Imagine the converse, where two datapoints share all but one input, say span, yet their classes occupy the top and bottom nodes of the output layer (with the inputs associated with the shorter span requiring a slightly less stiff section for a given support rotation). Here, we are asking one set of inputs to return [1, 0, 0, … 0] and the other only marginally different set to return [0, 0, 0, … 1], requiring each signal to be sent to drastically different parts of the network; a difficult task that is sure to hamper the training process. As another example, if we trained a network to classify types of pet from a photograph of an animal, we would hope that connections representing 'Springer Spaniel' and 'Cocker Spaniel' occupy similar areas of the network, whereas those representing 'Goldfish' and 'Horse' should occupy wildly different areas.

ANN training is achieved by minimising the 'loss function', effectively a measure of how close the predictions are to the targets, with a loss of zero suggesting a network that makes perfect predictions every time. Here, the categorical cross entropy loss function is used:

$$L = -\frac{1}{N}\sum_{j=1}^{N}\sum_{c=1}^{C} y_{j,c}\ln(\hat{y}_{j,c}) \tag{4}$$

where N and C are the number of samples (batch size) and the number of classes (36, in this study), $y_{j,c}$ is a binary indicator that represents whether class c is the correct class for sample j, and $\hat{y}_{j,c}$ is the predicted probability of class c, sample j. The term 'accuracy' refers to the number of correct classifications (the neuron in the output layer with the highest numerical value is deemed the 'predicted' class, and the numerical value of this output (from 0 to 1) is called 'confidence'. The aim is for the network to predict with 100% accuracy *and* 100% confidence) divided by the total number of samples.

The machine learning model was generated in Python using TensorFlow and the Keras Sequential Model. Input features were scaled to have zero mean and unit variance to prevent excessively large inputs from dominating the network's learning. The loss function was minimised using the Adam optimizer with a learning rate of 0.001 and a batch size of 64. The model was trained for 200 epochs as this was deemed sufficient to ensure proper training whilst avoiding overfitting (manifesting as a divergence in the loss/accuracy curves of the training and validation datasets). Hyperparameter tuning determined that three hidden layers of 256 neurons in each layer achieved the best balance between accuracy and computational time, whilst also avoiding overfitting. Dropout, which is commonly adopted to reduce a network's tendency to overfit, was not required. L1 and L2 regularisation, which encourage sparsity in the network and penalise very large weights respectively, were found to have a detrimental effect on model accuracy.

The train/test split was 80/20, with a further 10% of the training data set aside for validation, giving 129,600/14,400/36,000 datapoints in the training/validation/test sets. Fig. 5 shows the loss and accuracy of the network with increasing epochs, with modest improvement in performance beyond epoch 100, but importantly with overfitting having been avoided. The final accuracy of the test dataset was reported as 90.41% (loss: 0.1899 (training), 0.2093 (validation), 0.2047 (test); accuracy: 91.53% (training), 90.55% (validation), 90.41% (test)).

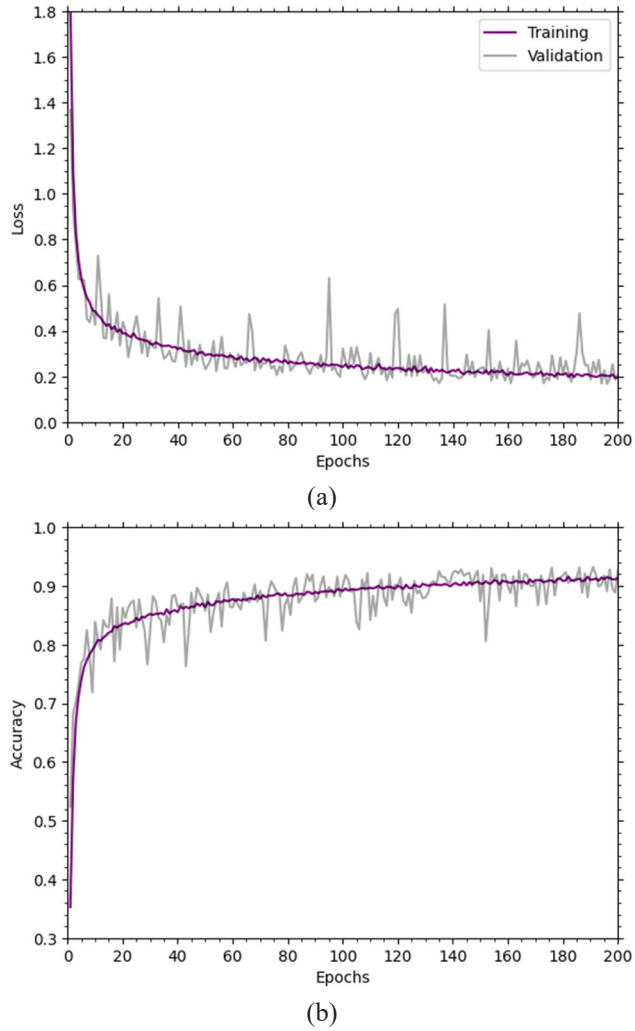

Figure 5: Training progress of the classification network. (a) Loss; and (b) Accuracy.

5 PERFORMANCE ASSESSMENT

The trained network was evaluated against the test dataset (approximately 1,000 datapoints per class), the results of which are compiled in the 'confusion matrix' shown in Fig. 6. Here, true classes are shown on the *y*-axis, and predicted classes are shown on the *x*-axis. A model with 100% predictive accuracy would occupy only the diagonal spaces.

Generally, the model is able to correctly classify the test data around 90% of the time, however this is not consistent among the classes. Of the 36 section types, 13 are predicted with >95% accuracy, and 21 are predicted with >91% accuracy. There are five classes that score an accuracy of <80%, however on closer inspection of Fig. 6 it appears as though these are formed of two main clusters, each containing three sections with similar plastic modulus.

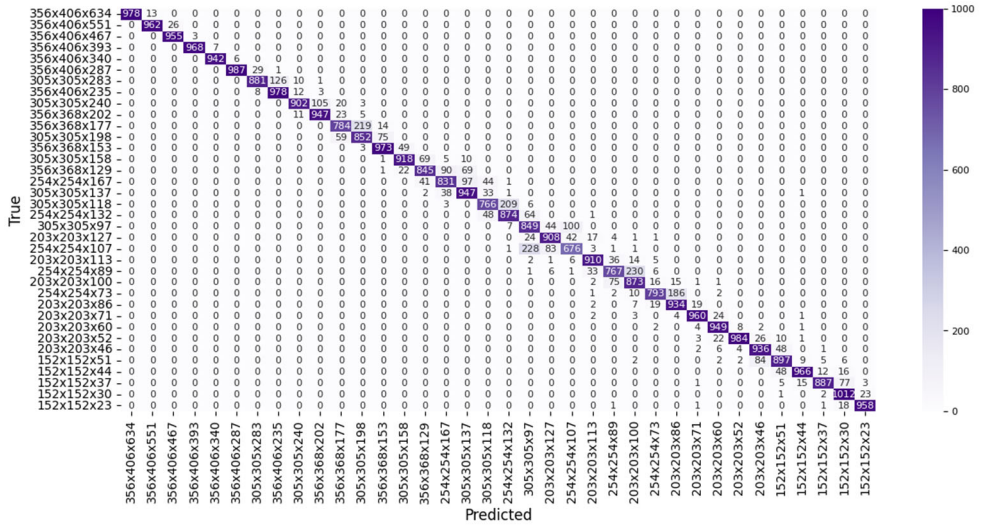

Figure 6: Confusion matrix of the trained model against the test dataset.

For the two lowest-accuracy sections (254 × 254 × 107 and 254 × 254 × 89), confidence in the true class was calculated separately for cases where the model predicted correctly and incorrectly (see Table 2). Even when the model misclassifies, it still has around a 30% confidence in the true classification, meaning that in these cases it would almost certainly have picked the correct class as its second choice (second highest confidence value). Therefore, whilst the accuracy of the model's *first* guess may be around 90% overall, the accuracy of its *top two* guesses would be approaching 100%

Table 2: Low-accuracy section classifications and corresponding misclassifications.

Section	Plastic modulus, Z_{pl} (cm^4)	Mass per metre (kg/m)	Accuracy (%)	Confidence in true class (%)		Most frequent misclassification
				When correct	When incorrect	
254 × 254 × 107	1480	107.1	68.08	78.62	33.11	305 × 305 × 97
305 × 305 × 97	1590	96.9	84.90	80.60	36.83	203 × 203 × 127
203 × 203 × 127	1520	127.5	91.07	91.75	29.06	254 × 254 × 107
254 × 254 × 89	1220	88.9	73.47	88.29	26.21	203 × 203 × 100
203 × 203 × 100	1150	99.6	88.81	88.81	29.81	203 × 203 × 113
203 × 203 × 113	1330	113.5	93.43	97.20	23.23	254 × 254 × 89

Each time the model misclassified, the most frequent incorrect class was also stored, again see Table 2. The two lowest accuracy classes are each joined by two sections with remarkably similar properties, and the model understandably has some difficulty in distinguishing between them. Currently, the ANN is not given these properties (plastic modulus and mass) in advance, but ongoing work is exploring whether further improvements in accuracy can be achieved if the model is allowed to incorporate this information into the training process.

6 THE MODEL IN USE

Whilst the model has shown good accuracy against the test dataset, its intended use will be subtly different. Rather than predicting a section that is known to *exactly* match a set of inputs, the model will be given span, loading, and a *target* rotation, and asked to provide a section which, to its best knowledge, matches that rotation most closely. For example, running the model for the inputs [8, 1000, 5000, 2], in the order of span (m), peak force (kN), impulse (kN.ms), and support rotation (°) returns: CAT BS-UC 254 × 254 × 73. Running this section back through the *Ergo* forward model returns a peak support rotation of 1.85°, within 7.5% of the target value. These are the same inputs as in Fig. 2, but with a larger section.

In order to assess how close the rotation of the predicted section is to the target rotation, in a more general sense, a grid search was performed with peak force ranging from 5e3–5e5kN (20 values spaced logarithmically), impulse ranging from 5e3–1e5kN.ms (25 values

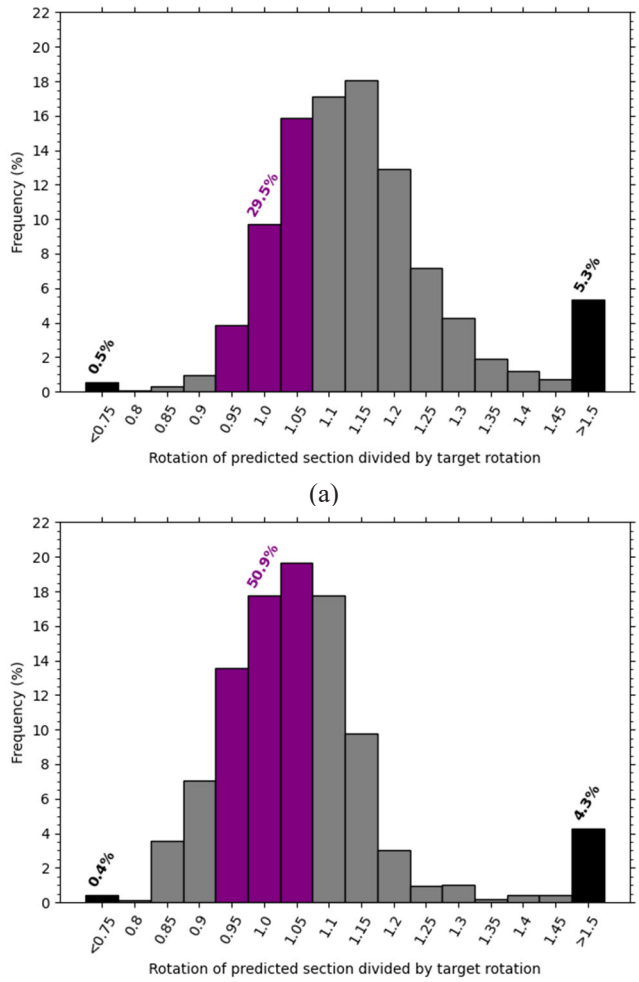

Figure 7: Histograms of peak rotation of predicted section divided by target peak rotation. (a) First choice only; and (b) Closest match from the model's top two choices.

spaced logarithmically), span ranging from 4–8 m (increments of 1 m), and two target rotations: 2° and 12°, as per Table 1, resulting in 5,000 total unseen design cases. For each case, the top two sections from the predictor (in terms of % confidence) were run through the *Ergo* forward model. The results from this grid search are presented in Fig. 7, where (a) shows the output from only the top classification, whereas (b) shows the closest match from either of the top two classifications. It can be seen that in 29.5% of the design cases, the top choice section results in a rotation that is within ±7.5% of the target, and in 50.9% of the design cases the top two choices return a rotation within ±7.5% of the target. These are both indicated by the purple regions of each histogram.

Excluding the underflow and overflow bins, the mean deviation is +13.0% when considering only the top choice, and +4.2% when considering the top two choices. These are associated with mean confidence values of 95.95% (top choice only) and 99.79% (top two choices). In total, only 5.8% of the design cases sit within the under/overflow bins (representing >25% underprediction or >50% overprediction respectively) when considering the top choice, which reduces to 4.7% when considering the top two choices.

7 SUMMARY AND OUTLOOK

This paper presents the development of a classification artificial neural network to perform rapid inverse analysis of blast loaded steel columns. 180,000 individual SDoF analyses are leveraged to train the network, which takes inputs of span, force, impulse, and target support rotation, and outputs one of 36 BS-UC section classifications. The trained model is able to correctly classify the test dataset 90.41% of the time. The model was further tested on 5,000 unseen design cases, where it was able to pick a section – which resulted in an output rotation within 7.5% of the target – in 29.5% of the cases if a single output was taken from the inverse model, which increased to 50.9% of the cases if the top two outputs were taken from the model and the closest match selected.

As a proof-of-concept, this work can be considered highly successful, and it is hoped that future studies will benefit by reframing the design process as an inverse method and leveraging machine learning tools to aid in this. This will particularly benefit scenarios where the forward model is much more computationally expensive than the SDoF method used herein (e.g. CFD/FEM), and therefore any computational steps saved by reducing or eliminating the need for iteration, will have considerable savings.

The model itself could be further improved by: (a) incorporating prior knowledge of relevant section properties such as plastic modulus and mass per unit length; (b) training the model on more section types; and (c) developing a non-symmetrical loss function to attempt to correct the inherent conservativism of the model, which showed average deviations in the design case of >1.00. Work is ongoing in this regard.

ACKNOWLEDGEMENT

The first author gratefully acknowledges financial support from the Royal Academy of Engineering under the *Industrial Fellowships Round 17* programme.

APPENDIX

Full list of BS-UC sections sorted by decreasing plastic modulus [8].

Section	Plastic modulus, Z_{pl} (cm^4)	Mass per metre (kg/m)	Section	Plastic modulus, Z_{pl} (cm^4)	Mass per metre (kg/m)
356 × 406 × 634	14,200	633.9	254 × 254 × 132	1,870	132
356 × 406 × 551	12,100	551	305 × 305 × 97	1,590	96.9
356 × 406 × 467	10,000	467	203 × 203 × 127	1,520	127.5
356 × 406 × 393	8,220	393	254 × 254 × 107	1,480	107.1
356 × 406 × 340	7,000	339.9	203 × 203 × 113	1,330	113.5
356 × 406 × 287	5,810	287.1	254 × 254 × 89	1,220	88.9
305 × 305 × 283	5,110	282.9	203 × 203 × 100	1,150	99.6
356 × 406 × 235	4,690	235.1	254 × 254 × 73	992	73.1
305 × 305 × 240	4,250	240	203 × 203 × 86	977	86.1
356 × 368 × 202	3,970	201.9	203 × 203 × 71	799	71
356 × 368 × 177	3,460	177	203 × 203 × 60	656	60
305 × 305 × 198	3,440	198.1	203 × 203 × 52	567	52
356 × 368 × 153	2,960	152.9	203 × 203 × 46	497	46.1
305 × 305 × 158	2,680	158.1	152 × 152 × 51	438	51.2
356 × 368 × 129	2,480	129	152 × 152 × 44	372	44
254 × 254 × 167	2,420	167.1	152 × 152 × 37	309	37
305 × 305 × 137	2,300	136.9	152 × 152 × 30	248	30
305 × 305 × 118	1,960	117.9	152 × 152 × 23	182	23

REFERENCES

[1] Larcher, M. & Casadeia, F., Explosions in complex geometries – A comparison of several approaches. *International Journal of Protective Structures*, **1**(2), pp. 169–196, 2010.

[2] Biggs, J., *Introduction to Structural Dynamics*, McGraw-Hill: New York, 1964.

[3] Krauthammer, T. & Altenberg, A., Negative phase blast effects on glass panels. *International Journal of Impact Engineering*, **24**(1), pp. 1–17, 2000.

[4] Rigby, S.E., Tyas, A., Bennett, T., Warren, J.A. & Fay, S., Clearing effects on plates subjected to blast loads. *Engineering and Computational Mechanics*, **166**(EM3), pp. 140–148, 2013.

[5] Cormie, D., Mays, G. & Smith, P., Blast *Effects on Buildings*, 3rd ed. Thomas Telford: London, 2019.

[6] Gallet, A., Rigby, S., Tallman, T.N., Kong, X., Hajirasouliha, I., Liew, A., Liu, D., Chen, L., Hauptmann, A. & Smyl, D., Structural engineering from an inverse problems perspective. *Proceedings of the Royal Society A*, **478**, 20210526, 2021.

[7] Gallet, A., Liew, A., Hajirasouliha, I. & Smyl, D., Machine learning for structural design models of continuous beam systems via influence zones. *Inverse Problems*, **40**(5), 055011.

[8] British Steel, Sections: Universal columns (UC) datasheet. https://britishsteel.co.uk/media/nm1kuplv/british-steel-universal-columns-datasheet-100723.pdf. Accessed on: 21 Jan. 2025.

[9] Kingery, C.N. & Bulmash, G., Airblast parameters from TNT spherical air burst and hemispherical surface burst. Technical Report ARBL-TR-02555. Aberdeen Proving Ground, MD, USA: U.S Army BRL, 1984
[10] Holgado, D., Mourão, R., Montalva, A. & Florek, J., On the assessment of reinforced concrete (RC) walls under contact/near-contact explosive charges: A deep neural network approach. *Buildings*, **14**(9), 2683, 2024.
[11] Dennis, A., Pannell, J.J., Smyl, D.J. & Rigby, S.E., Prediction of blast loading in an internal environment using artificial neural networks. *International Journal of Protective Structures*, **12**(3), pp. 287–314, 2021.