Proceedings Paper:
Bruce, I., Wholey, w., Kannan, R. et al. (3 more authors) (2025) Development of an AI tool for predicting the performance of steel columns subjected to a blast load. In: Proceedings of the 27th International Symposium on Military Aspects of Blast and Shock (MABS27). 27th International Symposium on Military Aspects of Blast and Shock (MABS27), 05-10 Oct 2025, Colmar, France. Military Aspects of Blast and Shock (MABS).

DEVELOPMENT OF AN AI TOOL FOR PREDICTING THE PERFORMANCE OF STEEL COLUMNS SUBJECTED TO A BLAST LOAD

Ian Bruce[1], Will Wholey[2], Ramaseshan Kannan[3], Archie Luxton[4], Aliz Fischer[1], Sam Rigby[3]

[1]Arup, San Franciso, USA, [2]Arup, New York, USA
[3]Arup, Manchester, United Kingdom, [4]Arup, London, United Kingdom

Key words: blast, column, AI, analysis

ABSTRACT

Finite Element Analysis (FEA) of building structural columns subject to a blast loading are time consuming and even a simple single column model can take several hours to build, analyze and post process. These types of high-fidelity analyses are typically done during the detailed design phase of a new building when there is sufficient time for them to be carried out. However, by the detailed design phase the design is well established, and it is harder to accommodate design changes. A quick assessment tool that can be used during the concept design phase would be invaluable in allowing architects and engineers to work out the required column size while the design is still flexible.

To this end we are developing an AI tool that can be used for assessing the blast performance of steel columns. The tool uses a neural network surrogate model of the full fidelity FEA analysis, and to date, predicts analysis results for steel columns with I-section profiles. Outputs that are predicted by the surrogate include likelihood of failure, residual column capacity after a blast event, and a basic characterization of damage level from the blast, alongisde confidence levels in each of these predictions.

We used the LS-DYNA finite element analysis software to run a suite of column blast analyses for a range of column sizes, charge masses, standoffs, and column orientations.

The raw results from these models were post-processed to categorize the performance of each column (failure, damage, remaining load capacity). These were then fed into a machine learning algorithm to develop the AI tool. Our surrogate can both predict the performance as well as the accompanying uncertainty in its prediction, thereby overcoming the black-box nature of AI predictions.

The tool is accessed through an API allowing it to be used from a simple web interface or linked to from other software such as Excel or Rhino 3D.

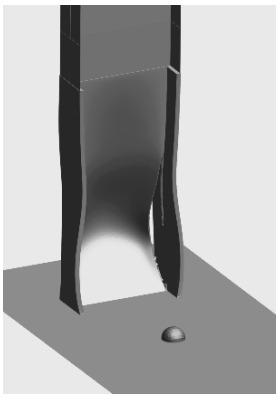

Figure 1: LS-DYNA Column Blast Analysis

BACKGROUND

When designing buildings with blast requirements it is necessary to perform blast analysis on the structural columns to ensure they can survive the loading from a specified charge mass/stand-off pair and still carry the required structural loads. Currently we use the LS-DYNA Finite Element software to perform these dynamic blast analyses. Even with the modeling automation tools we have developed these column blast analyses can still take a day or so to turn around a specific column.

This turn-around time is workable during the Detailed Design phase but does not work during the Concept Design stage when the initial column layout and sizing is still being determined by the structural engineer and architect. Design changes during the detailed design phase can be costly due to the amount of rework that is needed (see Figure 2). There is clear motivation to develop rapid assessment tools that can be used during the Concept Design stage to provide initial insights while the design is still fluid.

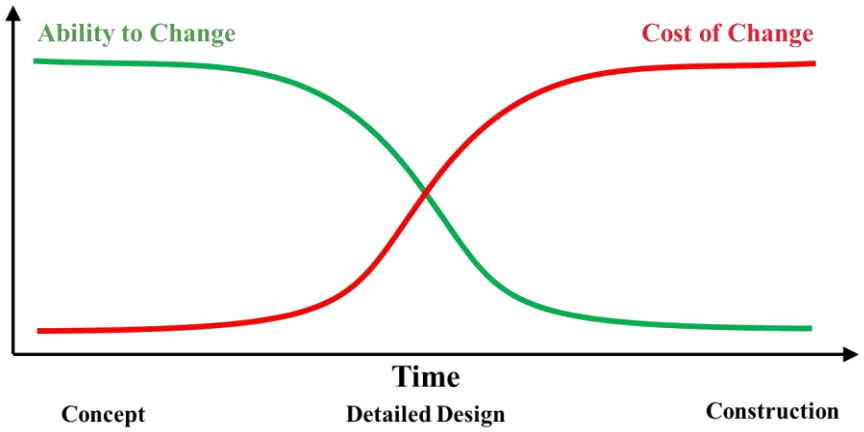

Figure 2: Design process and cost of change

One of the tools we are developing is a quick assessment tool that can be used to assess the blast performance of steel columns.

METHODOLOGY

In developing the proof of concept for this tool we have followed the methodology outlined below.

- Research typical building column sizes, potential charge sizes and stand-offs
- Develop a list of column blast load cases to analyze.
- Construct the analysis models and perform the analysis
- Post-process the models and extract the key performance parameters
- Train the machine learning algorithm on the results from the analysis models
- Deploy the tool using an API to allow access from various software
- Develop an initial user interface for the tool.

Load Case Selection

Following discussion with our Structural Engineering team around typical column types and sizes seen in buildings, we chose for this initial tool to assess the performance of American wide flange steel beams (W-Beams) as specified by ASTM with sizes ranging from W10x60 (10.2in by 10.1in) through to W44x408 (44.8in by 16.1in)

For the charge we considered sizes from small backpack improvised explosive devices (IEDs) up to full-size vehicle IEDs. Table 1 below shows the range of charge parameters considered, with stand-off and angle definitions shown in Figure 3. The minimum standoff analyzed for each charge / stand-off pair was limited so that the scaled distance (z) of the charge from the nearest element of the column was greater than the value given below.

$$z > 0.178 \ \frac{m}{kg^{\frac{1}{3}}}$$

This ensured that the relevant Kingery & Bulmash blast parameters and positive phase durations were within the range of validity for the analyses in question.

Table 1: Charge parameters

Charge Size	Stand-off	Angle to Column
2.5 kg	0.25 m	0 deg
5.0 kg	0.50 m	30 deg
10 kg	1.00 m	45 deg
25 kg	2.00 m	60 deg
50 kg	4.00 m	90 deg
100 kg		
250 kg		
500 kg		

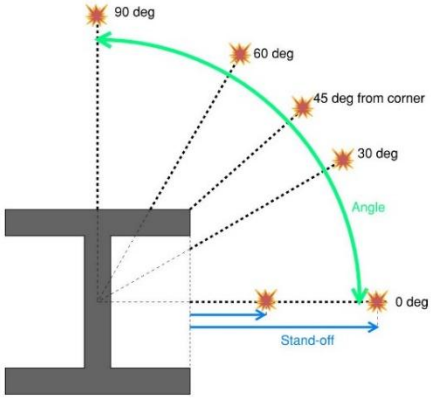

Figure 3: Charge parameter definition

From this overall design space, a total of 793 load cases were chosen to be analyzed.

Modeling, Analysis and Post-Processing

For the analysis we used the commercially available Finite Element Analysis (FEA) software Ansys LS-DYNA. This software includes an explicit (dynamic) solver which is particularly suited for simulating extreme events such as blast and impact load cases.

Column Model

The column is 15ft (4.572m) high. The lower section where the blast impacts the column is modeled using solid elements and the upper section is modeled using beam elements. Figure 4 below shows an example model for a W44x408 column.

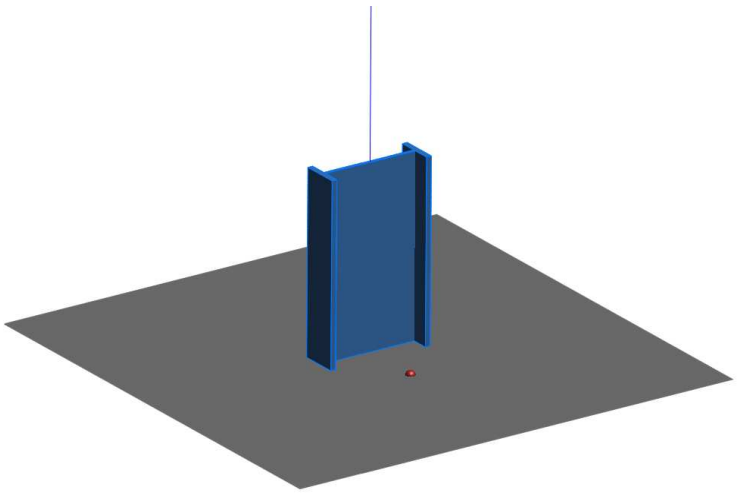

Figure 4: Typical column model

The following material properties were used for the steel.

- Density: 7850 kg/m3
- Young's Modulus: 210GPa
- Poisson's Ratio: 0.3
- Yield Stress: 350MPa
- Failure Strain: 15%
- Strain Rate Effects: Cowper-Symonds model c=1300, p=5

The base of the column is fully fixed, and the top of the column is fixed in the lateral directions but is allowed to move in the vertical direction.

Column Loading

A gravity loading is applied to the whole model, and an initial vertical load that is 20% the vertical elastic buckling capacity of the column is applied to the top of the column.

Blast Loading

The blast loading was applied to the column using the *Load_Blast_Enhanced option in LS-DYNA. This is an implementation of the standard Kingery-Bulmash equations for calculating blast wave properties. The charge is modeled as a hemispherical charge located on the ground at the appropriate stand-off and angle to the column.

Post-Blast Column Loading

After the blast event, the vertical load on the column is increased to determine the remaining vertical capacity of the column after it has been subjected to, and potentially damaged by, the blast. The load was increased until the point of column failure, which was indicated by excessive deflection under minimal additional load.

Analysis

Given the required turnaround for the analyses, a high-performance computing (HPC) cloud provider was used to analyze the models.

Post-Processing

The column models were post-processed utilizing the batch processing options within the Oasys suite of software. This allowed the models to be automatically post-processed when each analysis run finishes. The overall performance of each column was extracted and output in a csv file format that could be easily ingested by the machine learning routines. The following performance parameters were extracted:

- Downward vertical displacement of the top of the column
- Peak vertical force in the column at the base (remaining column capacity)
- Peak lateral displacement of the column at the top of the detailed section
- Eroded element energy

From these values a determination is made as to whether the column is damaged and whether the column has failed.

- If the eroded element energy is greater than zero, then elements have failed and been deleted from the model. This is classed as the column being damaged
- If the top of the column has dropped by more than 0.5% of the original height of the column then the column is classed as failed.

Table 2 below shows a sample of the training data written to the csv file.

Table 2: Sample training data

run_name
W36X135_500kg-2m-30degWKAxis

col_height	sec_width	sec_depth	web_thick	flange_thick
4.572	0.305	0.904	0.0152	0.0201

charge_mass	charge_x	charge_y	charge_z	charge_ang
500	1.885	1.088	0	29.99304

fail	damage	dz_disp_min	col_force	lat_disp
1	1	-1.29118	359978.6	0.808634

The first three rows (eleven entries) specify the analysis name, column dimensions, charge size and charge location. The last row (five entries) details the performance of the column, i.e. the outputs for which we train the machine learning model.

Machine Learning Model Training

Machine learning (ML) is a subfield of artificial intelligence where computer systems learn from data, identify patterns, and make predictions or decisions without being explicitly programmed for every task.

There are many possible choices for ML models. Since the model is trained on the results of a simulation and is intended to learn the behavior of the simulation, our model is a ‘surrogate’ of the simulation.

We note that the results from the FE analysis include:

- “categorical” outputs such as failure or damaged states. These assume discrete values (“labels”) such as ‘failed’ or ‘damaged’.
- continuous values such as column force or lateral displacement.

It is desirable for the surrogate model to not only predict these results but also report a level of confidence in the accuracy of the predictions. Therefore, the surrogate model we wish to train is an ‘uncertainty-aware’ surrogate, i.e. a probabilistic machine learning model that returns both a prediction at an unseen point and the uncertainty in the prediction expressed as a probability distribution.

The starting point for the creation of a ML model is an exploratory data analysis (EDA). EDA is a technique that summarizes the data graphically, allowing us to identify patterns and relationships between variables, and discover anomalies and outliers. As part of the EDA of the training data, we carried out data distribution analysis for the dependent quantities, i.e., analysis results. This revealed a large skew in the distribution of column displacement results as shown in Figure 5. The imbalance stems from the fact that most columns in our training dataset did not undergo failure, resulting in most values of displacement being close to 0.0 m.

As such, we omitted the lateral and vertical displacements from the set of results to use for the model training, focusing only on column forces, failed, and damaged indicators. Obtaining a well-distributed training dataset that has better representation of displacements requires better sampling strategies, an exercise we leave to future work.

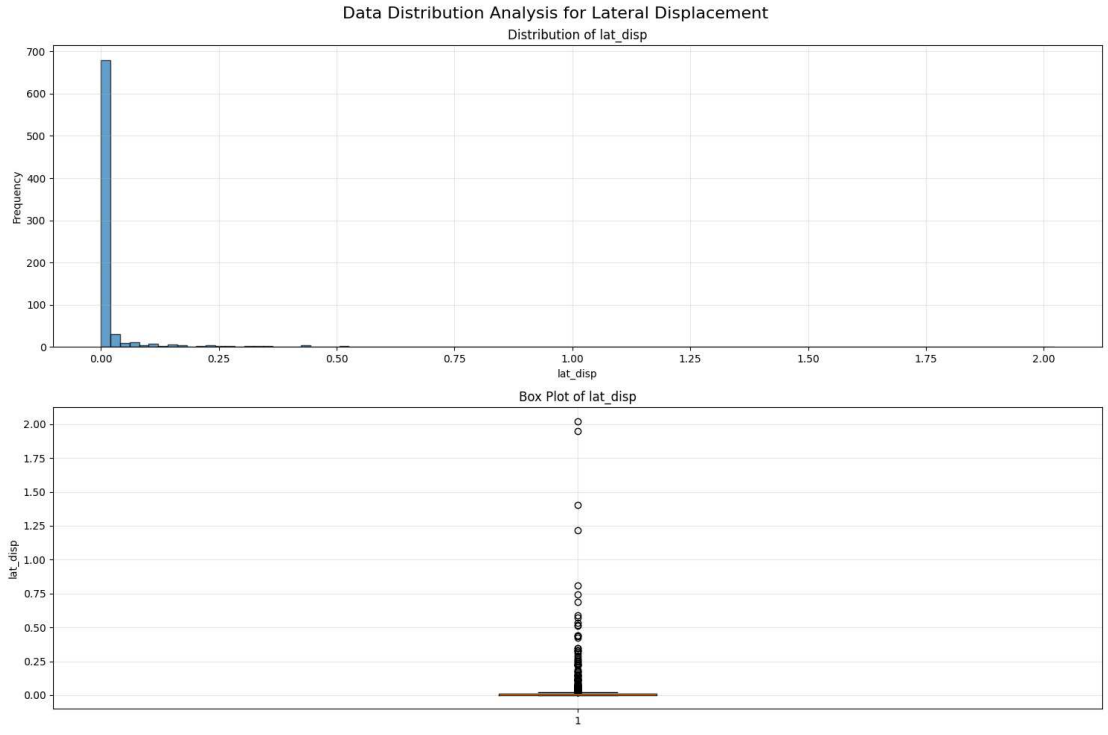

Figure 5: EDA of training data for lateral displacement

Since our surrogate model must predict categorical and continuous values, we require it to work as both a classification and a regression model. Furthermore, as this is intended to be uncertainty-aware, we choose a Bayesian Neural Network built on a multilayer perceptron (MLP) architecture.

We use the PyTorch library to train and evaluate our MLP model. The simulation data was divided into train, validate, and test groups with a split ratio of 80%, 10%, and 10% respectively. For training, we use a loss function that is a composition of losses from classification and regression. For the regression variable, we use a loss defined by Symmetric Mean Absolute Percentage Error, defined as follows

$$\text{SMAPE} = \frac{1}{n} \times \frac{|prediction - target|}{(prediction + target)/2},$$

where $|x|$ is the absolute value of x. Hyperparameters such as the number of hidden layers and the number of neurons per hidden layer are arrived at using hyperparameter optimization (HPO), for which we use the SMAPE as a metric for the objective function on validation set. Training and HPO are carried out in AWS's Sagemaker environment, which allows multiple 'compute' instances to be used in parallel to evaluate the metric. This results in an MLP architecture with 10 layers each with 128 neurons. The network accepts 9 inputs and predicts three outputs, two of which are classification outputs and the remainder regression.

Once trained, testing the tuned model resulted in a MAPE of 19.1% for column force, and accuracies of 97.5% for 'fail' and 95% for 'damage' with f1 scores of 0.975 and 0.95. An f1 score close to 1 indicates the model has low incidence of false positives and false negatives.

Tool Deployment

The current tool is hosted on an internal server and is accessed through an API. A request is sent to the API with the following inputs;

- Column Dimensions
  - sec_width, sec_depth, web_thick, flange_thick
- Charge Details
  - charge_mass, charge_x, charge_y, charge_z, charge_ang

The API replies with the following outputs;

- Post-blast column vertical capacity
- Column failure (yes or no)
- Column damage (yes or no)

Along with each of these outputs is a confidence index. The confidence index is expressed as a *credible interval*. A 95% credible interval is the range within which the true value is expected to lie with 95% probability, given the model and data. The larger this range, the lower confidence in the prediction. The interval is therefore a qualitative measure that must be combined with engineering judgment to assess the reliability of the model's outputs. We currently use a tolerance of 50% of the prediction to compare with the credibility width, i.e., if $CI\ Width \leq (1 \pm 0.5)\ Prediction$, a given prediction is considered trustworthy.

A sample text-based output from the API is shown below.

```
Given an input of:
  sec_width: 0.29
  sec_depth: 0.508
  web_thick: 0.0226
  flange_thick: 0.0404
  charge_mass: 50.0
  charge_x: 0.0
  charge_y: 1.254
  charge_z: 0.0
  charge_ang: 90.0
Predictions:
  Continuous:
         col_force: 12191478     (95% credibility interval: 9633687 to
15964122)
  Categorical:
    fail: 0 (100.0% confidence)
    damage: 0 (100.0% confidence)
```

The advantage of using an API is that the tool can then be accessed from a whole range of applications allowing different front ends to be used depending on the required task. The primary options we are developing are;

- Linking to the API from MS Excel
- An interactive web interface
- A Python script that can be integrated into larger workflows.

Figure 6 below shows the current iterations of the web interface we are developing.

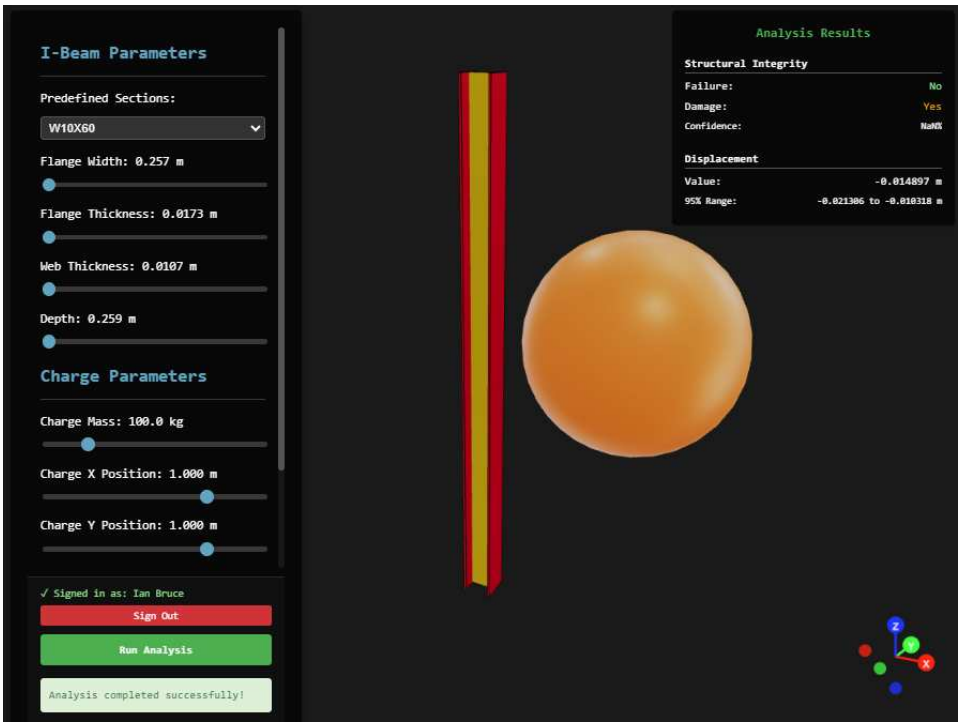

Figure 6: Tool web interface

DISCUSSION

There is a clear need for rapid and accurate blast analysis tools that allow for key performance metrics (e.g. failure, displacement, support reactions, etc.) to be estimated for common structural forms (e.g. columns) at the Concept Design stage. This paper presents the development of one such tool. Overall, we have been successful in demonstrating that we can develop an AI-based predictive tool to provide such rapid estimates of the performance of a column under a blast loading. However, there is more refinement that can be done.

Currently the load cases we analyzed were fairly evenly spread across column size, charge size and stand-off. However, for training the machine learning algorithms our dataset must contain representative spreads of both inputs and outputs. Future work will include using better sampling methods to ensure a surrogate with better generalizability. A two-step approach to the analysis may provide some benefit, in which a first set of load cases are run with a broad spread and then a second set of load cases are run that are more focused on where we see column failure from the first set.

As with any AI tool like this it is important to understand its limitations. For this tool the limitations are primarily around the data set used to train the tool. If column sizes and loading parameters of the use case are within the bounds of the training design space, then there will be reasonable confidence in the results. For use cases outside of the initial training space the accuracy of the tool's predictions will diminish. Note that despite this reduction in accuracy, the tool may well still return results with a high degree of confidence, so the limits/extents of the training dataset should always be made clear in advance. It is important to ensure as tools are deployed out to a wider audience that the understanding of these limitations doesn't get lost.

NEXT STEPS

Some of the next steps we are looking at for this tool include:

Analysis Benchmarking

Benchmarking our analysis models against known test data to confirm that the fidelity of our models is sufficient for the performance metrics we are pulling.

Machine Learning

Evaluating how we are using the data set for training the AI tool and investigating whether there are more appropriate metrics to use.

Validation

We are currently carrying out validation exercises to confirm that the results from the AI tool are in line with the original data set used to train the model.

Additional Steel Column Type

Currently the model has been trained on American Wide-Flanged Steel I-Beams (W-Beams). We would like to expand this to include other steel section types typically used for building columns (e.g. Hollow Structural Section - HSS, RHS, SHS, CHS).

A future goal would be to include concrete columns. However, with all the variability around column shape, size and rebar layout the design space for these column types is orders of magnitude larger.

Front End Development

Several options we are currently working on are discussed in the Tool Deployment section.

A future goal would be to connect the API into the main structural analysis software we use. This would allow the blast performance of the columns to be assessed at the same time as the other standard structural load cases.

Refined Loading Model

*Load_Blast_Enhanced is a simple and well-established fast-running load model based on the ConWep/Kingery & Bulmash curves. In its current form, the loading neglects any form of one-way coupling (i.e. the form and geometry of the column influencing the blast loading through shadowing, stagnation, coalescence at re-entrant corners, etc.) or two-way coupling (load-induced structural deformation influencing the loading through fluid-structure-interaction). Clearly, using a more sophisticated numerical tool to generate higher fidelity training data will, due to enhanced model run-times, generate considerably less training data, thereby hampering the AI-based tool's ability to learn. This is a trade-off which is receiving considerable further attention, with one emerging solution being to develop a "plug-in" fast-running loading model which incorporates the effects discussed above, but at a run-time equivalent to *Load_Blast_Enhanced.