This is a preprint draft. The published article can be found at: https://doi.org/10.1016/j.measurement.2021.110098

Please cite this paper as:

**Naser M.Z.** (2021). "Mapping Functions: A Physics-guided, Data-driven and Algorithm-agnostic Machine Learning Approach to Discover Causal and Descriptive Expressions of Engineering Phenomena." *Measurement.* https://doi.org/10.1016/j.measurement.2021.110098

# *Mapping Functions:* A Physics-guided, Data-driven and Algorithm-agnostic Machine Learning Approach to Discover Causal and Descriptive Expressions of Engineering Phenomena

M.Z. Naser, PhD, PE
Glenn Department of Civil Engineering, Clemson University, Clemson, SC 29634, USA
AI Research Institute for Science and Engineering (AIRISE), Clemson University, Clemson, SC 29634, USA
E-mail: mznaser@clemson.edu, Website: www.mznaser.com

## Abstract

This paper presents; *mapping functions*, a machine learning (ML) and simulation-free approach to enable physics-guided and data-driven derivation of expressions that describe engineering phenomena. In this approach, a series of ML models are first developed to examine a given phenomenon, and insights from their analysis, together with those obtained from physics principles, are then used to identify key features governing the noted phenomenon while satisfying the Law of Parsimony of *Occam's Razor*. The identified features are subsequently explored via a search space to *map* the causality of the problem on hand into compact descriptive expressions which can be applied directly to examine such phenomenon, thereby negating the need for subsequent modeling. The proposed approach overcomes some limitations associated with traditional means of arriving at descriptive expressions as examined against structural and fire engineering problems. This approach offers an alternative method that is cognitive, instantaneous, and affordable.

*Keywords:* Machine learning; Mapping functions; Feature selection; Surrogate modeling; Strucutral engineering, Fire engineering.

## Introduction

Engineering problems are often tackled through physical tests, or numerical simulations. The primary goal of such examination is to arrive at insights that tie a cause(s) to an effect(s). Oftentimes, the outcome of the noted exercise is molded into a representation, or series of representations, that capture the mechanisms at which a phenomenon occurs or develops. In all cases, an experiment is conducted under a certain level of control to minimize noise and ensure reliable findings [1]. To maintain control, one parameter at a time is often varied to observe how a particular parameter influences the outcome of an experiment. The same procedure can also be undertaken through a numerical investigation. In such an exercise, a numerical model (say a finite element (FE) model) is first developed and then validated against a benchmark. A benchmark is likely to be of an analytical nature, an experimental nature (i.e., measurements taken during a test), or possibly through a comparison against previously developed numerical models [2].

Ultimately, a holistic analysis is applied to arrive at an *understanding* of the cause-and-effect governing the phenomenon on hand [3]. From an engineering perspective, this *understanding* is often converted into a function, formula, or design expression to convey simplicity. Arriving at such a function can be undertaken via a statistical or a mathematical approach wherein observations from tests or simulations are fitted into expressions that deliver our "understanding" of the examined problem [4]. Such expressions come in handy as they: 1) convert a phenomenon into a meaningful set, 2) articulate the relations between critical parameters, as well as these parameters and the outcome/response observed during the noted investigation, 3) can be easily applied by various stakeholders, 4) can serve as a blueprint (e.g., can be extended) to parallel phenomena, and most of all 5) overcomes the need to carry out additional tests/simulations to analyze already established parameters/relations [5,6]. We must note that arriving at such expressions, while helpful to describe our *understanding* of a problem, is also confined by the space of parameters examined during tests or simulations.

The procedure to develop a design expression also involves additional steps, such as those related to ensuring: 1) reliability, 2) generality, and 3) wide-acceptance of the proposed expression(s). Much of the aforenoted steps require further stipulation and passing of relevant requirements (i.e., cross-examination by independent authorities such as building code committees, etc.). However, these steps occur once an expression is deemed fit for usage, and in order to arrive at this step, an expression (or set of expressions) must first be founded.

Devising a comprehensive experimental campaign is often complex as such campaigns are restricted by the availability of funds, facilities, time allocated for investigation, etc. As such, it is commonly accepted that a few tests are first undertaken – wherein such tests are read by sensor measurements – and then augmented with numerical models to extend the space of the experimental program. In some instances, researchers or building code committees may opt to combine findings from multiple test campaigns to arrive at design expressions [7,8]. These are primarily arrived at via rigorous statistical analysis [9,10]. Given the emphasis of engineering curricula upon such methods, engineers become naturally comfortable with statistical means of investigation.

A deep dive into statistical methods reveals that these methods were designed to operate on data with a "relatively" small number of parameters. These methods draw inferences from a sample of population supplemented with a quantifiable measure of confidence that associates a discovered relationship to be, in fact, "true" – one that is unlikely to be due to noise. In a typical statistical analysis, a model with accompanying statistical distribution is adopted and applied to fit the parameters of interest to the outcome of a given phenomenon [11]. In the instance wherein the number of parameters grows, or the relationship between these parameters turns complex, or the quality of data does not satisfy predetermined conditions set by subject/human knowledge tied with statistical methods, such methods become less effective [12].

With the rise of ML in parallel fields, ML can also be used to arrive at an "understanding" of phenomena [13,14]. Thus, one can also think of ML as a mean to derive descriptive expressions. Unlike statistical methods, ML directly learns from data in search of patterns and makes minimal assumptions about the data type, origin, etc. (thereby becoming useful even if data was collected from unstandardized/homogenous methods or when the data contains highly nonlinear interactions). In addition, ML becomes useful in scenarios where data is wide (i.e., with the number of parameters (or simply features) exceeds the number of observations) [15]. A key distinction between ML and statistical methods is that ML algorithms are often designed to satisfy a

This is a preprint draft. The published article can be found at: https://doi.org/10.1016/j.measurement.2021.110098

Please cite this paper as:

**Naser M.Z.** (2021). "Mapping Functions: A Physics-guided, Data-driven and Algorithm-agnostic Machine Learning Approach to Discover Causal and Descriptive Expressions of Engineering Phenomena." *Measurement.* https://doi.org/10.1016/j.measurement.2021.110098

penalization (or cost function) to overcome issues such as overfitting and poor generalization to new data [16]. According to Bzdok et al. [12], "*statistics draws population inferences from a sample, and machine learning finds generalizable predictive patterns*."

The integration of ML into engineering problems has significantly risen over the past few years. For example, ML algorithms have been applied to structural engineering problems (i.e., property prediction and response prediction) [17,18], material discovery [19,20], robotics [21,22] etc. However, the bulk of the commonly used ML algorithms can be classified under *Blackboxes*. Such algorithms have complex inner working structure, and as such, provide the user with a *tool* to *map* the variables to the outcome of a given problem. Such tools are the *opposite* of what engineers are familiar with. The lack of transparency and figurative constructs associated with ML negates engineers from adopting ML tools. A question then arises, how to use ML to arrive at a representable *understanding* of phenomena – one that resembles commonly used forms of engineering expressions?

Thus, this work fosters the use of ML surrogates that can augment complex ML models into formulae to allow users to create new physics-guided and data-driven descriptive expressions for complex engineering problems. In this work, Extreme Gradient Boosted Trees (ExGBT), Adaboost Regressor (AdaBoost), Extra Trees (ET), and TensorFlow Deep Learning (TFDL) are applied in three case studies to derive expressions that can be used to attain deformation history of beams under fire, ultimate shear strength of cold-formed steel channels, and cyclic response of shear deficient of CFRP-strengthened beams. The proposed approach efficiently reduces the search space to be tackled in a ML analysis and comprises of two steps: 1) physics principles and ML algorithms are applied to identify *features of high importance* within a dataset, and then 2) high-fidelity features are utilized to derive compact expressions via a surrogate. Thus, this work starts with a discussion on feature selection techniques and then dives into the rationale of mapping functions and their application to engineering problems.

## Feature Selection Techniques

This section builds upon the Law of Parsimony of *Occam's Razor* (*Nunquam ponenda est pluralitas sin necesitate*), which is often translated to "*Entities should not be multiplied beyond necessity*". This law implies that simplicity is a goal in itself, and hence it is thought of as the best explanation to a problem is one that involves the fewest possible assumptions whenever possible [23,24]. Thus, to arrive at compact descriptive expressions, one must identify the key features governing the phenomenon on hand. The above infers that the user needs to find "*the optimal feature subset, as there is no guarantee that the optimal parameters [features] for the full feature set are equally optimal for the optimal feature subset*." [25]. Therefore, by identifying key features, a ML analysis avoids overfitting, provides faster and more cost-effective models, and allows a deeper insight into the underlying processes that generated the predictions – all of which indicate an improved performance.

From this view, this section describes commonly used feature selection techniques that can be applied via ML models. The primary goal of such techniques is to identify features in terms of their relevance and redundancy towards the outcome (response or target variable) of a phenomenon. For example, in a space of features, some can be classified as relevant (by varying degrees, i.e., strongly, weakly, or irrelevant) and/or redundant/not redundant. A proper analysis identifies relevant and unique features to realize optimal derivation of design expressions since a model with fewer unnecessary features can be more interpretable and less computationally expensive [26]. In general, feature selection techniques in supervised ML can be grouped into three classes, filter, wrapper, and embedded (intrinsic) methods, and these methods will be discussed herein in detail. The reader is reminded that information with regard to the history and the background of each technique can be found in their perspective references, as well as in [25,27–30].

*Filter methods*

Filter methods select features according to their relationship with the response (target variable) by means of statistical analysis or feature importance methods. These methods operate prior to the ML analysis and hence reduce the number of features to be used in the analysis. The bulk of filter methods evaluate each feature individually to comply with two inherent assumptions; 1) features have a degree of independence, or 2) are entirely independent of each other – both of which may or may not be always true [25]. In most cases, a "relevance" score is calculated for all features, and features with low scores are removed from the analysis. Finally, the leftover features are presented as inputs to the ML model.

A number of techniques can be grouped under filtering methods. These techniques often follow the types of inputs and targets (i.e., whether numerical or categorial) – see Table 1. For regression problems where the target is numerical, correlation-based methods can be applied, such as Pearson's correlation coefficient, Spearman's rank coefficient, alternating conditional expectations (ACE), etc. On the other hand, in classification problems where the target is categorial, the following techniques can be used: ANOVA correlation coefficient, Kendall's rank coefficient, and Chi-Squared test. Some methods, such as mutual information metric, and Cramer's V, can be used for regression or classification problems.

Table 1 Common techniques for filter methods

| Target | Input features | |
|---|---|---|
| | Continuous | Categorial |
| Continuous | • Correlation metrics (Pearson's correlation coefficient, Spearman's rank coefficient)<br>• Mutual information<br>• F-test<br>• Neighborhood component analysis<br>• ReliefF<br>• Sequential feature selection | • Cramer's V<br>• Mutual information<br>• ANOVA correlation coefficient<br>• Kendall's rank coefficient<br>• Linear discriminant analysis<br>• F-test<br>• ReliefF<br>• Sequential feature selection |
| Categorial | • Mutual information<br>• Cramer's V<br>• One Way ANOVA<br>• Kendall's rank coefficient | • Cramer's V<br>• Chi-Squared test<br>• Mutual Information<br>• Fisher score |

| | | |
|---|---|---|
| | • Minimum Redundancy Maximum Relevance<br>• Neighborhood component analysis<br>• Logistic regression | • Minimum Redundancy Maximum Relevance<br>• Neighborhood component analysis |

Correlation methods are widely used in engineering problems [31,32]. One should note that the Pearson correlation evaluates a linear relationship between two continuous features. In other words, such a relationship occurs when a change in one feature is associated with a proportional change in the other feature. Pearson correlation assumes both features to be normally distributed and to satisfy homoscedasticity (i.e., data is equally distributed about the regression line). On the other hand, the Spearman correlation evaluates a monotonic relationship between two ranked features. A monotonic relationship is one that describes an increase or decrease in one feature as the other feature increases. The Spearman correlation does not carry any assumptions with regard to the distribution of the data [33]. Despite their usefulness, traditional correlation metrics may not be useful in practical scenarios, as 1) linear correlation may not guarantee a causal relationship, and 2) data obtained may not fit into prescribed assumptions [34].

On the other hand, mutual information is an entropy-based metric between two random features that measures how much knowing the value of one feature reduces the uncertainty on the other feature in a range between 0 to 1 (with higher values indicating higher dependency) [35]. This measure can identify linear or nonlinear associations and is invariant under transformation [36]. Cramer's V measures association between two categorical variables in a range between 0 to 1 based on the chi-square statistic (with a score of unity inferring that one variable being entirely determined by the other).

Some of the advantages of filter methods include, 1) they are computationally simple and hence can be easily scaled to high-dimensional datasets, 2) they are independent of the ML model to be used in the analysis, and 3) feature selection needs to be performed only once, and prior to the start of the ML analysis. On the other hand, filter methods are associated with a user preference or subjective nature. For example, the user must select the confidence level to be applied in the selection filtering analysis. Thus, feature relevance scores do not have obvious/agreed upon cut-off points or metrics to declare which features are of relevance to the phenomenon on hand. A common disadvantage of some filter methods is that they tend to disregard feature dependencies, as well as any interaction with the target variable. It is worth noting that there exist a few solutions to the aforenoted problems (e.g., multivariate search, etc.), as shown in [25,30].

*Wrapper methods*

Unlike filter methods, wrapper methods search for well-performing features by evaluating all the possible combinations of features against an evaluation criterion (or a performance metric) belonging to a given ML algorithm [37,38]. In a way, wrapper methods encompass search algorithms that manipulate features by adding or removing them in pursuit of identifying a combination that maximizes the ML performance (predictive capability). Despite their superiority and taking feature dependencies into account, it is due to their extensive search space and reliance on specific algorithms that wrapper methods can be vulnerable to high computational cost and bias.

Three primary techniques can be employed in wrapper methods: forward selection, backward selection, and stepwise selection. In the first technique, the analysis starts with a null set of features which expands by iteratively adding relative features to the ML model providing the model's performance continues to improve. The opposite happens in the second technique, wherein the analysis starts with all features first, and then worst-performing features (based on a predefined significance level) are removed from the feature space. This iterative procedure ends once no improvement is observed by the removal of lasting features [25]. Both of these techniques fall under sequential feature selection methods. A stepwise selection combines both forward selection with the addition of checking the significance of the newly added feature, and if such significance is found to be minor, then the newly added feature is removed in a similar manner to backward selection.

Commonly used wrapper algorithms include Recursive feature elimination (RFE) [39], and Simulated annealing (SA) [40], to name a few, and these can be used in regression and classification problems. Finally, it is worth noting that wrapper methods can employ a greedy or non-greedy approach to selecting features. In the former, a feature search path always follows the direction that seems favorable to realizing a solution at the time of the iteration (which may lead to a quick solution but can also lead to being stuck at a local optimal as opposed to a global one). On the other hand, a non-greedy approach (i.e., SA) re-evaluates previous combinations of features and is flexible enough to dive into an unfavorable direction for space search if it appears to have a potential benefit within a particular iteration [41].

*Embedded (intrinsic) methods*

Embedded methods learn feature importance during the model training process, and hence it has a built-in capability to identify features of merit to a particular phenomenon via the implementation of regularizers (L1, L2, etc.), constraints, or objective functions. This turns into two positive advantages: 1) accounting for feature interactions, and 2) requiring less computational resources. On another front, embedded methods share some similarities with wrapper methods in which selection techniques are only specific to the used algorithm, which may also cause bias. Commonly used embedded methods include tree ensemble derivatives (Extra Trees, Random Forest), as well as regularized regressions (those which include a penalty: to reduce over-fitting such as Adaboost, LASSO, or to features that do not contribute to the target variable, etc.) [30].

Some of the advantages of embedded methods include quickness resulting from the selection process being embedded within the model fitting process, which negates the need for external selection tools. Also, the intrinsic nature of these methods enables the model from attaining a direct connection that yields informed decisions on selecting the right features that best satisfy the objective (or optimization) function employed by the model. Conversely, a major limitation to embedded methods is that they are model-specific (which implies that some ML models might perform better than others on the same dataset). In retrospect, some of the techniques that employ a greedy search approach might also experience the same limitations as wrappers.

Noting the above and given the nature of most structural and fire engineering problems and similarity between wrappers and embedded methods, it is then thought of best to showcase the general use of filter methods (based on correlation and mutual information) and embedded methods (through four different algorithms ExGBT, AdaBoost, ET, and TFDL). This stems from observations that most engineers are likely to either: 1) filter out unwarranted features before a ML analysis (in a similar manner to carrying out an experimental or numerical investigation), or 2) directly apply a ML model to examine a phenomenon. In the event that wrappers are to be used, then additional information can be found elsewhere [42].

## Rationale to *Mapping Functions*

Deriving a *mapping function* requires an understanding of the physical phenomenon that such a function aims to map, or tie. Mapping a phenomenon infers that the relationship between governing factors (i.e., features) is attained, or perhaps can be approximated close enough, and with sufficient consistency, that the derived function can be used with confidence [43]. Simply put, a *mapping function* is an expression that intelligently ties the input(s) of a phenomenon to the output(s) of a phenomenon. To maximize the effectiveness of a *mapping function*, such a function is to be compact, reliable, and easy to use. While simple/compact functions are preferable, complex phenomena may sometimes yield intricate functions.

To better showcase the concept of *mapping functions*, a visual engineering example can come in handy. In practice, beams are load bearing members in structural systems. As such, beams are designed to satisfy strength (i.e., to have a load bearing capacity that exceeds the magnitude of applied loading) and serviceability (e.g., should not deflect beyond a certain limit) criteria [44]. From a practical perspective, load bearing capacity and degree of deformation in a beam can be easily calculated following engineering and mechanics principles. However, under certain conditions (say, when a fire breakout), assessing load bearing capacity and deformation history turns into a highly multifaceted problem [45].

Given that the magnitude of deformation a beam undergoes under fire conditions primarily reflects the degree of degradation within its load bearing capacity, then it is quite possible to associate these two phenomena together [46]. The above is also true, noting how most fire-based evaluations rely on the degree of deformation a beam experiences to identify the point in time that declares failure [47,48]. Thus, under fire conditions, a primary interest to engineers would be to trace the time- or temperature-deformation history of a fire-exposed beam. Attaining such history experimentally is an involved process due to the harsh nature of fire tests and the need for specialized equipment and experienced personnel [49].

Similarly, to model such history, a user can develop a thermo-mechanical FE model that captures the interaction between fire, beam geometry, and materials properties, as well as applied loads. Realizing the aforenoted interaction requires deep knowledge on the response of construction materials as a function of elevated temperatures, together with other aspects (i.e., temperature-

This is a preprint draft. The published article can be found at: https://doi.org/10.1016/j.measurement.2021.110098

Please cite this paper as:

**Naser M.Z.** (2021). "Mapping Functions: A Physics-guided, Data-driven and Algorithm-agnostic Machine Learning Approach to Discover Causal and Descriptive Expressions of Engineering Phenomena." *Measurement.* https://doi.org/10.1016/j.measurement.2021.110098

induced forces, restraint conditions, and so on) [50]. Both testing and FE approaches are well accepted and have been proven effective, yet continue to suffer on a few fronts (e.g., cost/time associated with setting up fire tests, need for dedicated software/workstations, etc.) [51]. Thus, an opportunity to develop a new approach to tracing the deformation history of beams (or any structural elements for that matter) under fire conditions presents itself.

Physics principles show that the deformation of a fire-exposed beam results from stresses generated due to applied loading, *P*, and degradation to the beam's sectional capacity (a function of temperature rise, geometric features, material properties, restraints, etc.) [46]. Since *P* remains virtually constant during a fire, then the extent of deformation reflects the cumulative degradation in material properties and any possible losses in cross-section size. To better articulate the deformation of beams under fire, Fig. 1 illustrates the deformation history of two identical beams, *Beam 1* and *Beam 2*. *Beam 1* is loaded with $P_1$ (where $P_1>P_2$ and $P_2$ is applied to *Beam 2*). Since $P_1>P_2$, then *Beam 1* will undergo higher levels of deformation under fire. Arriving at this notion is trivial since only one feature (*P*) is varied between the two beams. However, if other features were to be varied as well, then the problem on hand substantially grows.

Noting the above, a hypothesis can then be formulated; "*in order to obtain deformation history of a beam under fire, all that is needed is to identify the primary features and the <u>governing relationship</u> that ties these features to deformation patterns*". Simply put, the so-called governing relationship is a *mapping function* that maps the aforenoted relationship. Arriving at such a function would allow engineers to predict beams' deformation and, by extension, other members with ease and without relying on complex tests or simulations.

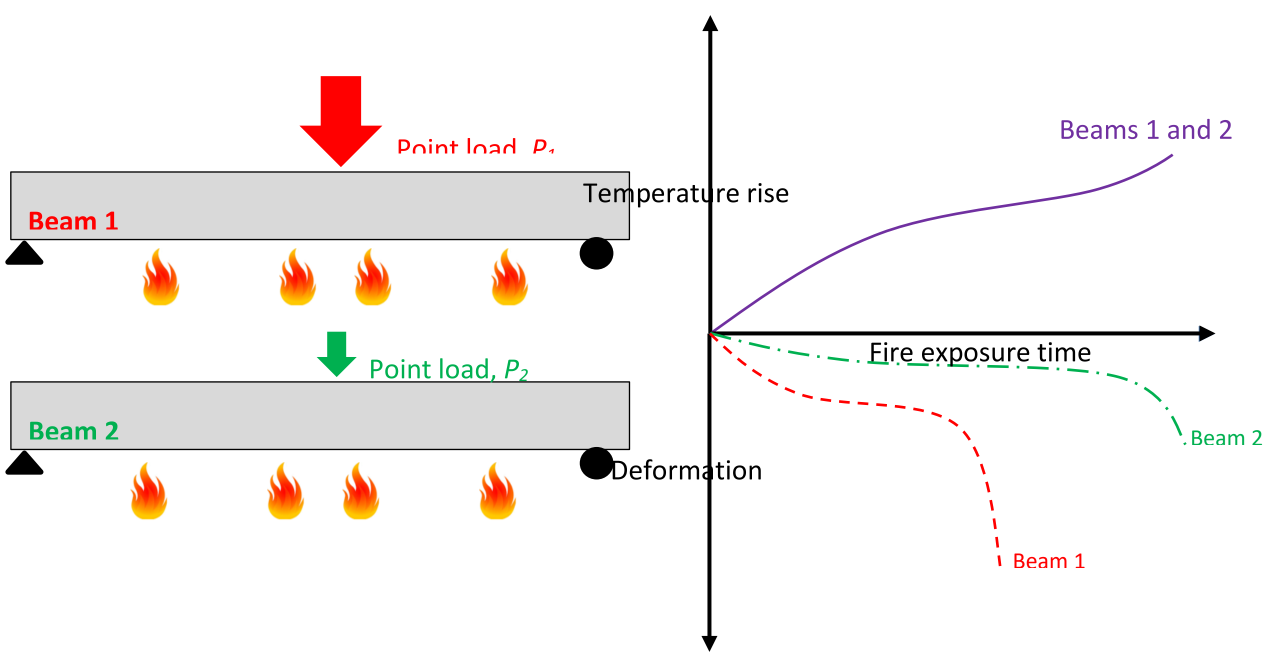


Fig. 1 Typical response of beams under fire conditions

The concept of mapping functions is envisioned to yield models (say design equations) that stem from a data driven nature. Such models are a bit different than those often obtained from structural engineering or fire engineering experiments. In such experiments, specimens are designed given the constraints of the available testing equipment and funds available for tests. Thus, it is rare to test all parameters in a given campaign as structural and fire tests are quite specialized and expensive (in a way, we do not have labs that contain all equipment to tests all combinations of parameters in a single campaign). Therefore, researchers often pick a selection of parameters (primarily identified by expert judgment and knowledge domain) and build specimens to examine such parameters using the available equipment they have. While this often leads to good design expression, given the above limitations, it may also not lead to accounting for all parameters.

On the other hand, a mapping function aims to bridge the limitation above by combining data from different experiments (as opposed to one) then identify the key parameters (from a data point of view) to derive generalized functions (equations). In this methodology, there is not guarantee that the mapping function will be similar to one obtained in the traditional manner. However, the mapping function is expected to be more encompassment of the examined phenomenon given that is built from a larger number of observations that contain an extensive range of the examined parameters (as opposed to a much smaller range in real tests – given the limitation mentioned above).

Figure 2 demonstrates an approach to deriving *mapping functions*. To realize a *mapping function*, observations are to be collected first (from tests or simulations). Data from such observations is then treated to identify key features via one or a combination of the noted selection feature methods described earlier. Feature selection analysis starts by examining all features via filter methods. If such methods prove useful, then the identified features will be treated as inputs. In the event that such methods do not prove useful, then a ML algorithm (or group of algorithms as adopted herein ExGBT, AdaBoost, ET, and TFDL) is applied to examine the importance of all features, and only the features of high fidelity are selected as input. In this work, fidelity refers to two concepts: 1) reoccurring features having an importance score of 10%[1] or higher, 2) as calculated by at least three of the four algorithms listed above. Once the key features are identified, then a ML model/ensemble is trained on the reduced features to understand the problem on hand and satisfy a set of performance metrics. In this work, this ensemble is made by blending all of the four used

[1] This arbitrary score was selected after a series of preliminary studies that were conducted as part of this work which were not shown for brevity. Please note the recent works also agree with the notion that we still lack guidance on setting standardized scores for feature importance [89–91]. Thus, the reader is reminded that the noted score can be revised as per user's preference. In lieu of the presented two concepts, other methods described in the previous section can also be used (i.e., Recursive feature elimination). However, one should also remember that such methods still require a user preference component to assign a score for feature importanc, or the number of features to be selected.

ML algorithms into a Light Gradient Boosted Trees Regressor ensemble as described by Delgado et al. [52].

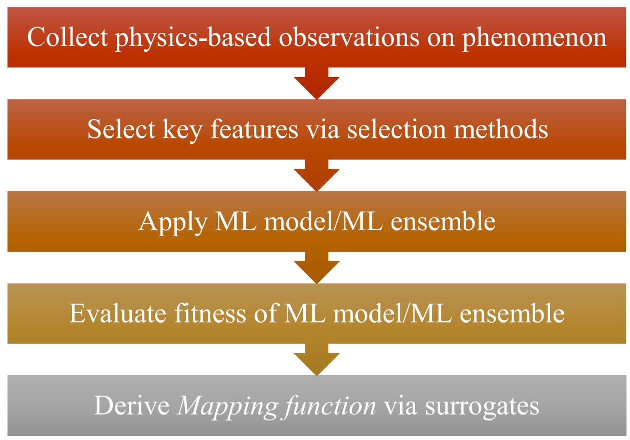



Fig. 2 Flowchart of the proposed approach[2]

The ensemble is trained and validated on randomly shuffled sets of the observations on hand. Three sets are created (T: training, V: validation, and S: testing). The ensemble is trained and validated on the T and V sets, respectively, and then independently cross-checked through assessing the S (hold-out) set. In all cases, 10-fold cross-validation is employed. In each set, performance metrics intended to measure test measurements' closeness to that predicted by the ensemble are applied c. In this work, three regression metrics, including Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Coefficient of Determination ($R^2$) – see Eqs. 1-3 [53–56], are adopted.

$$MAE = \frac{\sum_{i=1}^{n}|E_i|}{n} \tag{1}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} {E_i}^2}{n}} \tag{2}$$

$$R^2 = 1 - \sum_{i=1}^{n}(P_i - A_i)^2 \,/\, \sum_{i=1}^{n}(A_i - A_{mean})^2 \tag{3}$$

where, $A$: actual measurements, $P$: predictions, n: number of data points, $E = A\text{-}P$.

Finally, the ensemble is augmented with a surrogate[3] that translates the understanding of the model into a mathematical function, thereby a *mapping function*. This surrogate is also examined via the three metrics above with the addition of two new tests. Those tests are recommended by Smith [57] (correlation coefficient ($R$) > 0.8 with low errors metrics (e.g., MAE) indicates a strong

[2] Note: this approach can be augmented by referring to the last two sections for a discussion on some of the limitations that may arise.

[3]This work applies Genetic Algorithms (GA) as a surrogate technique to derive mapping functions. GA has been thoroughly examined in the open literature and a more formal and complete discussion on GA can be found at [92,93]. Additional techniques can also be used such as CARTs [94].

correlation between the predictions and actual measurements exists)), and Roy and Roy's [58] external predictability indicator ($R_m > 0.5$). The correlation coefficient and external predictability indicator are calculated as:

$$R = \frac{\sum_{i=1}^{n}(A_i-\bar{A}_i)(P_i-\bar{P}_i)}{\sqrt{\sum_{i=1}^{n}(A_i-\bar{A}_i)^2\sum_{i=1}^{n}(P_i-\bar{P}_i)^2}} \quad (4)$$

$$R_{\mathrm{m}} = R^2 \times (1-\sqrt{|R^2-Ro^2|}) \quad (5)$$

where

$$Ro^2 = 1-\frac{\sum_{i=1}^{n}(predicted_i-updated_i^o)^2}{\sum_{i=1}^{n}(predicted_i-\text{mean of predictions})^2}, updated_i^o = k \times predicted_i \quad (6)$$

And $k$ is the slope of regression lines between the regressions of actuals against predictions.

For completion, the user may opt to use additional performance metrics than that described above given the notion that we do not have a standardized procedure of selecting and assigning metrics. This work opted to showcase the presented metrics as they have been widely used by researchers in the area of structural and fie engineering [38,59,60].

**Selected Machine Learning Algorithms**

This section briefly describes the four algorithms showcased herein (ExGBT, AdaBoost, ET, and TFDL) since the full description is found in their respective references, as well as in [61–65]. To maintain harmony, all algorithms were primarily used in their default settings and then applied to all three case studies. The reader is also reminded that the proposed approach is algorithm-agnostic and is applicable by using other algorithms as well. The selected algorithms are shown herein for illustration purposes.

*Extreme Gradient Boosted Trees (ExGBT)*

The ExGBT is a sequential model that generates predictions from weaker tree-like models by optimizing an arbitrary differentiable loss function [66]. Notably, ExGBT aligns successive trees to previously obtained residual errors to concentrate training on the most challenging targets to predict, as seen here:

$$Y = \sum_{k=1}^{M} f_k(x_i), f_k \in F = \{f_x = w_{q(x)}, q: R^p \to T, w \in R^T\} \quad (7)$$

where, $M$ is additive functions, $T$ is the number of leaves in the tree, $w$ is a leaf weights vector, $w_i$ is a score on $i$-th leaf, and $q(x)$ represents the structure of each tree that maps an observation to the corresponding leaf index [67]. The code of the used ExGBT can be found online at [68,69]. This algorithm incorporates the following pre-tuned settings of learning rate of 0.02, "least squares regression loss" function, maximum tree depth of 7, subsample feature of 0.8, and 1000 for the number of boosting stages.

*AdaBoost Regressor (AdaBoost)*

The Adaptive Boosting algorithm fits a regressor to the original dataset and then fits additional copies of the same regressor with weights adjusted according to the error of the current prediction [66]. In AdaBoost, the notion is that a committee of regressors will behave in a superior manner to a single regressor. The code script for this algorithm can be found at [70], which has a typical loss function of:

$$L = \sum_{i=1}^{m} L_t(i) D_t(i) \quad (8)$$

where $L_t$ is a loss function (i.e., linear, exponential, etc.) constrained within [0, 1], $m$ is the number of examples, and $D_t$ is for data distribution. The adopted algorithm used a "linear" loss function and a learning rate of 0.1.

*Extra Trees (ET)*

The ET algorithm is one that comprises of a large number of decision trees (DTs) compiled into one algorithm to examine the whole dataset. Given the nature of such trees, a prediction from an ET follows the majority vote principle (e.g., arithmetic mean of all DTs) [63]. A typical formulation of ET is similar to a Random Forest algorithm, as can be seen herein:

$$Y = \frac{1}{J}\sum_{j=1}^{J} C_{j,full} + \sum_{k=1}^{K}\left(\frac{1}{J}\sum_{j=1}^{J} contribution_j(x,k)\right) \quad (9)$$

where, $J$ is the number of trees, $k$ represents a given feature, $K$ is the total number of features, $c_{full}$ is the average of the entire dataset (initial node). The used algorithm can be found herein [71] and has the following default settings; number of trees and leaves = 500 and 50, respectively, and a maximum depth of "none".

*TensorFlow Deep Learning (TFDL)*

TensorFlow is an open-source library developed by Google Brain to support Deep Learning [72]. TFDL imitates the topology of the brain and comprises of three layers. These layers use a "*relu*" activation function which enables the algorithm of generating an approximation form that permits gradient-based optimization. The used algorithm in its default settings (neurons in each layer = 44, number of training examples = 128, optimizer = *Adam*, learning rate = 0.001) can be found at [65].

## Case Studies

This section describes three case studies to be used in this work. These case studies will be examined via filter and embedded methods (by using four different algorithms ExGBT, AdaBoost, ET, and TFDL, and ensemble) to identify critical features and then derive mapping functions corresponding to each phenomenon.

### *Case study 1: mapping function for deformation of reinforced concrete beams under fire conditions*

As discussed earlier, deriving *mapping functions* to enable physics-guided and simulation-free assessment of deformation in reinforced concrete beams using ML requires information with regard to governing features covering geometric characteristics, material properties, level of loading, etc. These features are to be collected from physical fire tests [73–83]. In this study, observations from 20 simply supported reinforced concrete (RC) beams tested under standard fire conditions were collected in an earlier work [46] with the following features: duration of exposure under standard fire ($t$), compressive strength of concrete ($f_c$), yield strength of steel ($f_y$), steel reinforcement ratio ($\rho_s$), span length ($L$), percentage of load ratio ($P$), concrete cover ($V$), and deformation history ($\Delta$) as a function of fire exposure time.

Table 2 and Fig. 3 show further details into the selected features and their ranges. In this database, all beams are of a rectangular cross-section with a steel reinforcement ratio ranging between 0.5-1.1%. The range of compressive strength of concrete and yield strength of steel is from 15 MPa to 59 MPa, and 240 MPa to 591 MPa. The reported concrete cover varies between 20-50 mm, and the beams were 1.75 m to 6.50 m long. The applied loading level ranges between 0.30-0.60 of that applied at ambient conditions. The time of fire exposure extends to 219 min.

This is a preprint draft. The published article can be found at: https://doi.org/10.1016/j.measurement.2021.110098

Please cite this paper as:

**Naser M.Z.** (2021). "Mapping Functions: A Physics-guided, Data-driven and Algorithm-agnostic Machine Learning Approach to Discover Causal and Descriptive Expressions of Engineering Phenomena." *Measurement.* https://doi.org/10.1016/j.measurement.2021.110098

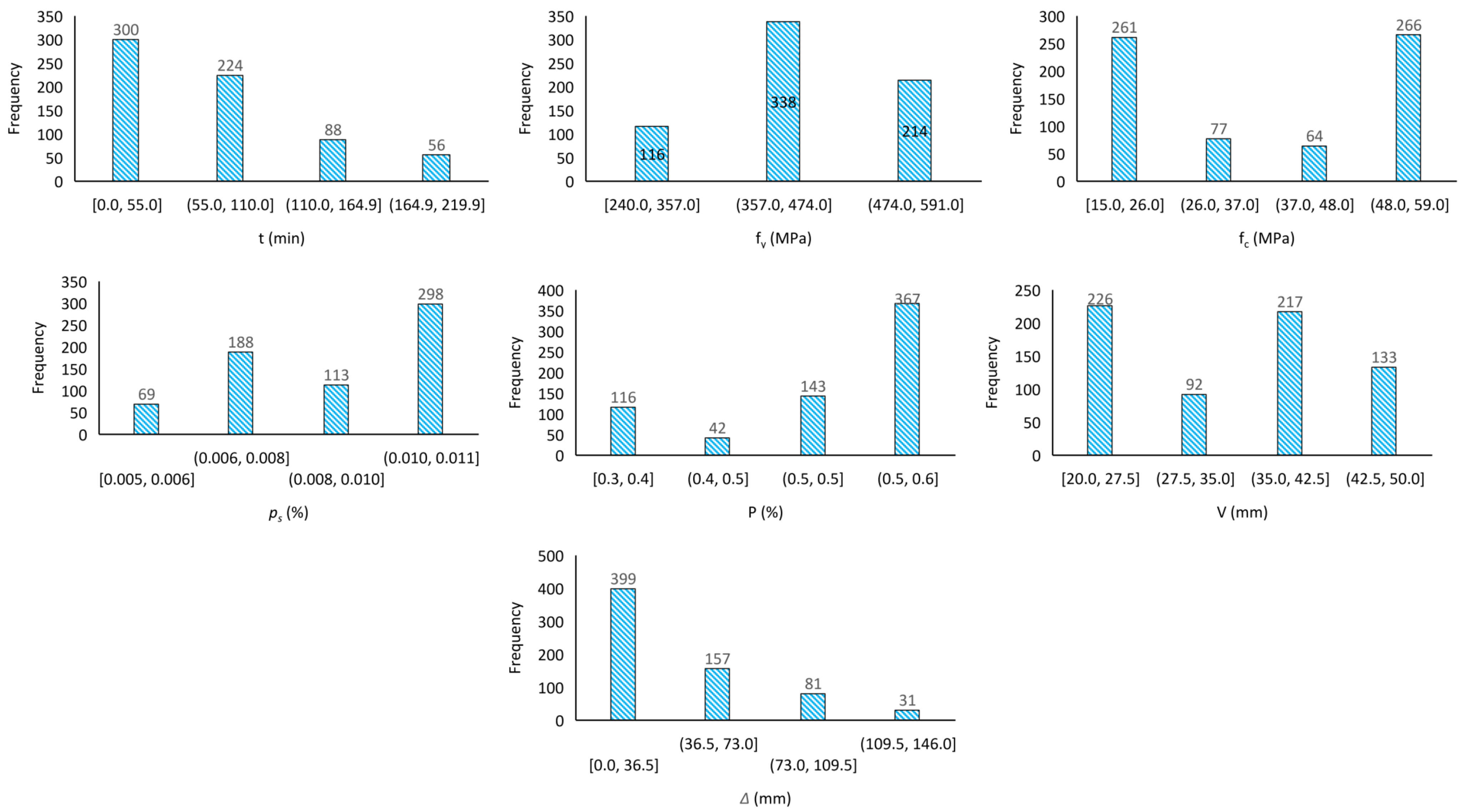



Fig. 3 Details on complied database

Table 2 Statistical insights from the collected database

| *Section* | *Features* | *t (min)* | *$f_y$ (MPa)* | *$f_c$ (MPa)* | *$\rho_s$ (%)* | *L (mm)* | *P (%)* | *V (mm)* | *Δ (mm)* |
|---|---|---|---|---|---|---|---|---|---|
| Fire tests on RC beams | Min | 0.0 | 240.0 | 15.0 | 0.005 | 1750.0 | 0.3 | 20.0 | 0.0 |
| | Max | 219.9 | 591.0 | 59.0 | 0.011 | 6500.0 | 0.6 | 50.0 | 146.0 |
| | Average | 72.5 | 446.2 | 38.1 | 0.009 | 4186.4 | 0.5 | 35.1 | 37.9 |
| | Standard deviation | 54.0 | 103.7 | 16.4 | 0.002 | 1205.4 | 0.1 | 9.9 | 32.7 |
| | Median | 61.1 | 439.0 | 30.5 | 0.010 | 4500.0 | 0.5 | 38.0 | 27.8 |
| | Skewness | 0.9 | -0.5 | 0.1 | -0.3 | 0.0 | -1.0 | 0.2 | 1.1 |
| *Pearson correlation* | | | | | | | | | |
| *Parameter* | | *t* | *$f_y$* | *$f_c$* | *$\rho_s$* | *L* | *P* | *V* | *Δ* |
| *t* | | 1.000 | | | | | | | |
| *$f_y$* | | -0.038 | 1.000 | | | | | | |
| *$f_c$* | | -0.124 | 0.579 | 1.000 | | | | | |
| *$\rho_s$* | | 0.465 | -0.017 | -0.127 | 1.000 | | | | |
| L | | 0.113 | -0.783 | -0.515 | 0.404 | 1.000 | | | |
| *P* | | 0.343 | 0.510 | 0.355 | 0.659 | -0.218 | 1.000 | | |
| *V* | | 0.482 | -0.166 | -0.074 | 0.804 | 0.370 | 0.486 | 1.000 | |
| *Δ* | | **0.665** | -0.240 | -0.169 | 0.140 | **0.319** | -0.041 | 0.058 | 1.000 |
| *Mutual Information* | | | | | | | | | |
| *Parameter* | | *t* | *$f_y$* | *$f_c$* | *$\rho_s$* | *L* | *P* | *V* | *Δ* |
| *t* | | 0.283 | | | | | | | |
| *$f_y$* | | 0.135 | 0.045 | | | | | | |
| *$f_c$* | | 0.151 | 0.870 | 1.000 | | | | | |
| *$\rho_s$* | | 0.135 | 0.966 | 0.838 | 0.027 | | | | |
| L | | 0.135 | 0.966 | 0.838 | 1.000 | 0.035 | | | |
| *P* | | 0.122 | 0.912 | 0.788 | 0.869 | 0.869 | 0.043 | | |
| *V* | | 0.119 | 0.718 | 0.671 | 0.725 | 0.725 | 0.591 | 0.030 | |
| *Δ* | | **0.323** | 0.084 | 0.095 | 0.084 | 0.084 | 0.079 | 0.058 | 1.000 |

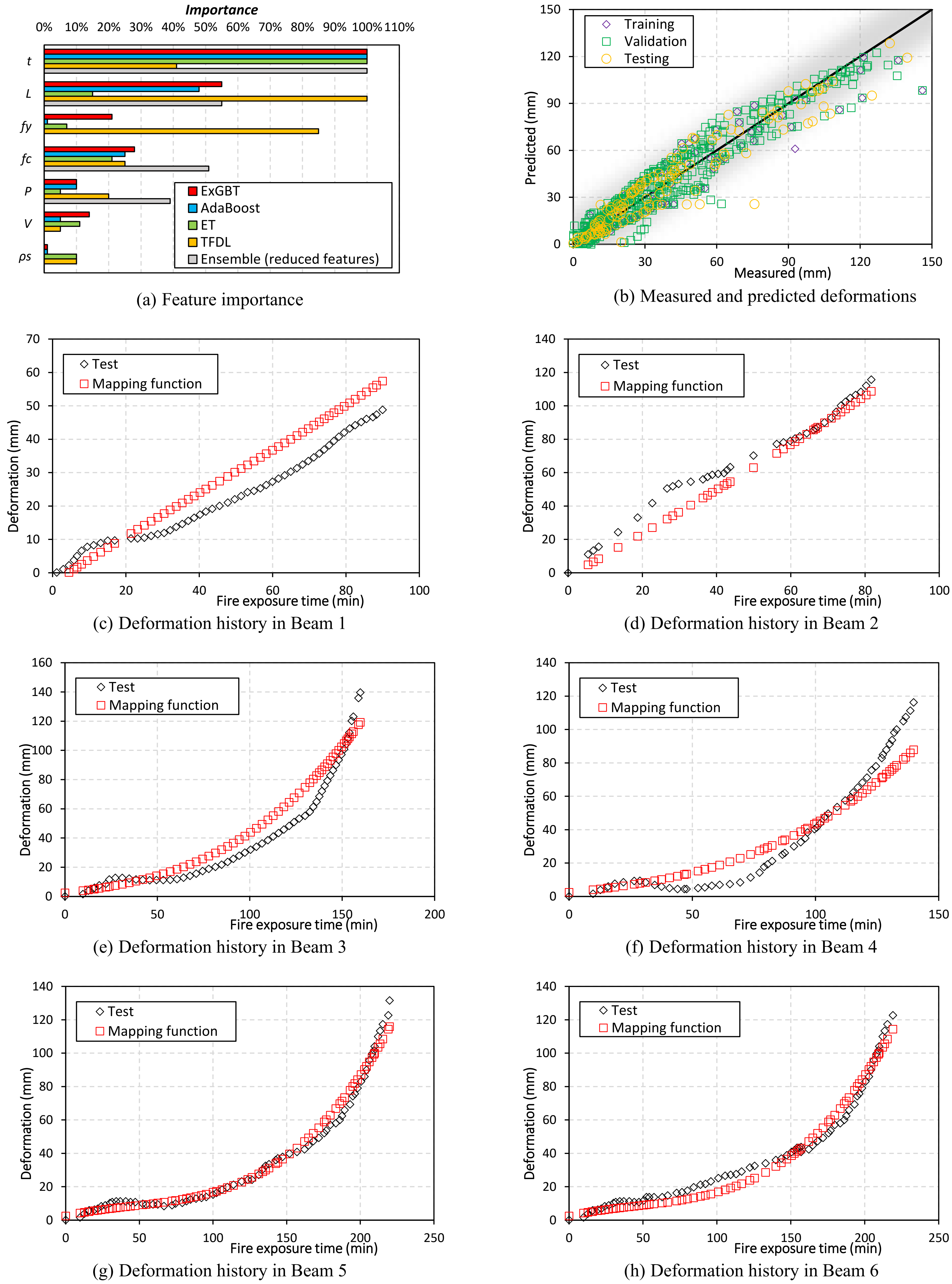
Looking at Fig. 4 shows the feature importance as measured by all algorithms. As one can see, only four features were shown to have an importance score of 10% or more across at least three of the four algorithms used. Thus, these features are $t$, $L$, $f_c$, and $P$, and hence these features were only input to the ensemble to train this ensemble to learn the deformation history phenomenon of RC beams under fire conditions. The performance of this ensemble in training/validation/testing is listed in Table 3. As one can see, the ensemble performs well wherein its error metrics represent small differences (about 5-8 mm) and high $R^2$ scores exceeding 94%. Therefore, this performance is deemed acceptable given that the metrics are similar to those commonly reported by other researchers [17,84], as well as our previous study (which utilized all features and did not examine the importance of features) [46].

Table 3 Performance metrics for training/validation/testing regimes.

| ***Metric*** | ***Ensemble*** | | | ***Mapping function*** | | |
|---|---|---|---|---|---|---|
| MAE | 5.16 | 4.32 | 4.03 | 7.82 | 7.05 | 7.13 |
| RMSE | 8.02 | 6.36 | 6.39 | 10.58 | 9.45 | 9.99 |
| $R^2$ | 94.30 | 96.03 | 96.11 | 90.07 | 91.33 | 90.49 |

Thus, this ensemble is augmented with a GA that yields the following *mapping function*. Metrics for this function are also listed in Table 3, and a cross-comparison is also shown in Fig. 4. It is clear that this reduced-ordered mapping function has good prediction capability. In addition, the *mapping function* derived herein also satisfies Smith [57] ($R$ = 0.96 > 0.8) and Roy and Roy’s [58] ($R_m$ = 0.65 > 0.5) recommendations. As such, this expression can be used directly to evaluate deformation history in RC beams under fire conditions with ease as opposed to carrying out complex FE simulations. A sample of validation plots for induvial beams used in this case study is shown in Fig. 4. Overall, the derived *mapping function* seems to capture the deformation history of all presented beams and across the full duration of fire exposure time.

$$\Delta = 0.326tf_c + 0.0049LP + 0.0005tL + 9.533\times10^{-5}t^2f_c + 1.679\times10^{-5}t^3 + 2.088\times10^{-6}t^2L - 9.77 - 5.49t - 0.0137t^2 - 1.177\times10^{-5}tf_cL - 0.0029tf_c^2 - 0.0937tf_cP \quad (10)$$




(a) Feature importance

(b) Measured and predicted deformations

(c) Deformation history in Beam 1

(d) Deformation history in Beam 2

(e) Deformation history in Beam 3

(f) Deformation history in Beam 4

(g) Deformation history in Beam 5

(h) Deformation history in Beam 6

Fig. 4 Evaluation of feature importance and *mapping function*

*Case study 2: mapping function for ultimate shear strength of cold-formed steel channels*

Due to the complexity of cold-formed steel (CFS) channels with slotted webs, the literature does not have accepted design expressions that can be applied to calculate their ultimate shear strength. Thus, Degtyarev & Degtyareva [55,85–87] carried out a comprehensive numerical campaign examining 3,512 FE simulations to investigate the ultimate shear strength of CFS channels with slotted webs. In this campaign, the ultimate shear strength was numerically obtained by accounting for material and geometric nonlinearities, as well as initial geometric imperfections. As such, this database makes a suitable candidate to explore the potential of the proposed *mapping function* approach.

Overall, this database accounts for 14 features for channels with realistic boundary conditions: channel depth ($D$), channel flange width ($B$), channel flange stiffener length ($B_1$), channel thickness ($t$), length of slots ($L_{sl}$), height of slots ($W_{sl}$), spacing of slots in the longitudinal direction ($S_{sl}$), spacing of slots in the transverse direction ($B_{sl}$), number of perforated regions ($N$), number of slot rows ($n$), yield stress of steel ($f_y$), inside bend radius ($r$), the aspect ratio ($a/h$), and height of the longitudinal stiffener ($h_{st}$), to predict the ultimate shear strength, $V_n$ (see Fig. 5). The outcome of the Pearson correlation and mutual information analyses is listed in Table 4 and shows a strong correlation between channel thickness and inside bend radius, and ultimate shear buckling load. From a practical consideration, Degtyarev & Degtyareva [55,85–87] took the inside bend radius as $2t$ in all of their FE models, and hence the strong association between $r$ and $t$. As one can see, filter methods do not seem to provide good insights to feature selection; thereby, the four ML algorithms are then applied.

This is a preprint draft. The published article can be found at: https://doi.org/10.1016/j.measurement.2021.110098

Please cite this paper as:

**Naser M.Z.** (2021). "Mapping Functions: A Physics-guided, Data-driven and Algorithm-agnostic Machine Learning Approach to Discover Causal and Descriptive Expressions of Engineering Phenomena." *Measurement.* https://doi.org/10.1016/j.measurement.2021.110098

Table 4 Statistical insights from the collected database.

| | *D (mm)* | *B (mm)* | *$B_1$ (mm)* | *t (mm)* | *$L_{sl}$ (mm)* | *$W_{sl}$ (mm)* | *$S_{sl}$ (mm)* | *$B_{sl}$ (mm)* | *N* | *n* | *$f_y$ (MPa)* | *r (mm)* | *a/h* | *$h_{st}$ (mm)* | *$V_n$ (N)* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Minimum | 150.0 | 20.0 | 0.0 | 1.0 | 60.0 | 3.0 | 85.0 | 7.5 | 1.0 | 6.0 | 250.0 | 2.0 | 0.5 | 0.0 | 1199.4 |
| Maximum | 250.0 | 95.0 | 26.0 | 3.0 | 90.0 | 7.0 | 115.0 | 11.5 | 2.0 | 12.0 | 500.0 | 6.0 | 1.5 | 60.0 | 99535.5 |
| Average | 225.8 | 57.8 | 13.0 | 2.0 | 75.0 | 5.0 | 100.0 | 9.5 | 1.7 | 8.0 | 490.9 | 4.0 | 1.0 | 19.6 | 21155.7 |
| Standard deviation | 35.4 | 13.5 | 4.3 | 0.8 | 11.0 | 1.5 | 7.3 | 1.0 | 0.4 | 2.4 | 46.9 | 1.6 | 0.1 | 22.0 | 13557.1 |
| Skewness | -1.1 | 0.2 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | -1.0 | 0.8 | -4.9 | 0.0 | 0.0 | 0.7 | 1.2 |
| *Pearson correlation* | | | | | | | | | | | | | | | |
| *Parameter* | *D* | *B* | *$B_1$* | *t* | *$L_{sl}$* | *$W_{sl}$* | *$S_{sl}$* | *$B_{sl}$* | *N* | *n* | *$f_y$* | *r* | *a/h* | *$h_{st}$* | *$V_n$* |
| *D* | 1.000 | | | | | | | | | | | | | | |
| *B* | 0.649 | 1.000 | | | | | | | | | | | | | |
| *$B_1$* | 0.000 | 0.000 | 1.000 | | | | | | | | | | | | |
| *t* | 0.000 | 0.000 | 0.000 | 1.000 | | | | | | | | | | | |
| *$L_{sl}$* | 0.000 | 0.000 | 0.000 | 0.000 | 1.000 | | | | | | | | | | |
| *$W_{sl}$* | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 1.000 | | | | | | | | | |
| *$S_{sl}$* | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 1.000 | | | | | | | | |
| *$B_{sl}$* | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 1.000 | | | | | | | |
| *N* | 0.218 | 0.132 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 1.000 | | | | | | |
| *n* | 0.371 | 0.249 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.012 | 1.000 | | | | | |
| *$f_y$* | -0.133 | -0.103 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | -0.008 | -0.080 | 1.000 | | | | |
| *r* | 0.000 | 0.000 | 0.000 | 1.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 1.000 | | | |
| *a/h* | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 1.000 | | |
| *$h_{st}$* | 0.312 | 0.200 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.540 | 0.131 | -0.003 | 0.000 | 0.000 | 1.000 | |
| *$V_{cr}$* | 0.115 | 0.084 | 0.018 | 0.798 | -0.183 | -0.194 | 0.037 | 0.025 | -0.141 | -0.168 | 0.121 | 0.798 | -0.171 | -0.156 | 1.000 |
| *Mutual Information* | | | | | | | | | | | | | | | |
| *Parameter* | *D* | *B* | *$B_1$* | *t* | *$L_{sl}$* | *$W_{sl}$* | *$S_{sl}$* | *$B_{sl}$* | *N* | *n* | *$f_y$* | *r* | *a/h* | *$h_{st}$* | *$V_n$* |
| *D* | 0.015 | | | | | | | | | | | | | | |
| *B* | 0.681 | 0.009 | | | | | | | | | | | | | |
| *$B_1$* | 0.056 | 0.188 | 0.001 | | | | | | | | | | | | |
| *t* | 0.007 | 0.107 | 0.166 | 0.683 | | | | | | | | | | | |
| *$L_{sl}$* | 0.014 | 0.072 | 0.108 | 0.006 | 0.047 | | | | | | | | | | |
| *$W_{sl}$* | 0.014 | 0.072 | 0.108 | 0.006 | 0.190 | 0.0555 | | | | | | | | | |
| *$S_{sl}$* | 0.007 | 0.037 | 0.057 | 0.003 | 0.263 | 0.263 | 0.003 | | | | | | | | |
| *$B_{sl}$* | 0.007 | 0.037 | 0.188 | 0.107 | 00072 | 0.263 | 0.201 | 0.001 | | | | | | | |
| *N* | 0.032 | 0.020 | 0.000 | 0.000 | 0.000 | 0.000 | 0.057 | 0.000 | 0.015 | | | | | | |
| *n* | 0.108 | 0.073 | 0.015 | 0.001 | 0.008 | 0.008 | 0.005 | 0.005 | 0.002 | 0.015 | | | | | |
| *$f_y$* | 0.045 | 0.054 | 0.017 | 0.045 | 0.056 | 0.056 | 0.030 | 0.030 | 0.038 | 0.038 | 0.0109 | | | | |
| *r* | 0.007 | 0.107 | 0.166 | 1.000 | 0.006 | 0.006 | 0.003 | 0.003 | 0.001 | 0.001 | 0.045 | 0.683 | | | |
| *a/h* | 0.045 | 0.054 | 0.188 | 0.166 | 0.108 | 0.056 | 0.030 | 0.030 | 0.038 | 0.038 | 0.045 | 0.045 | 0.017 | | |
| *$h_{st}$* | 0.101 | 0.084 | 0.028 | 0.001 | 0.012 | 0.012 | 0.008 | 0.008 | 0.0117 | 0.017 | 0.001 | 0.001 | 0.015 | 0.016 | |
| *$V_{cr}$* | 0.018 | 0.032 | 0.031 | 0.333 | 0.024 | 0.027 | 0.006 | 0.007 | 0.012 | 0.020 | 0.025 | 0.333 | 0.011 | 0.015 | 1.000 |

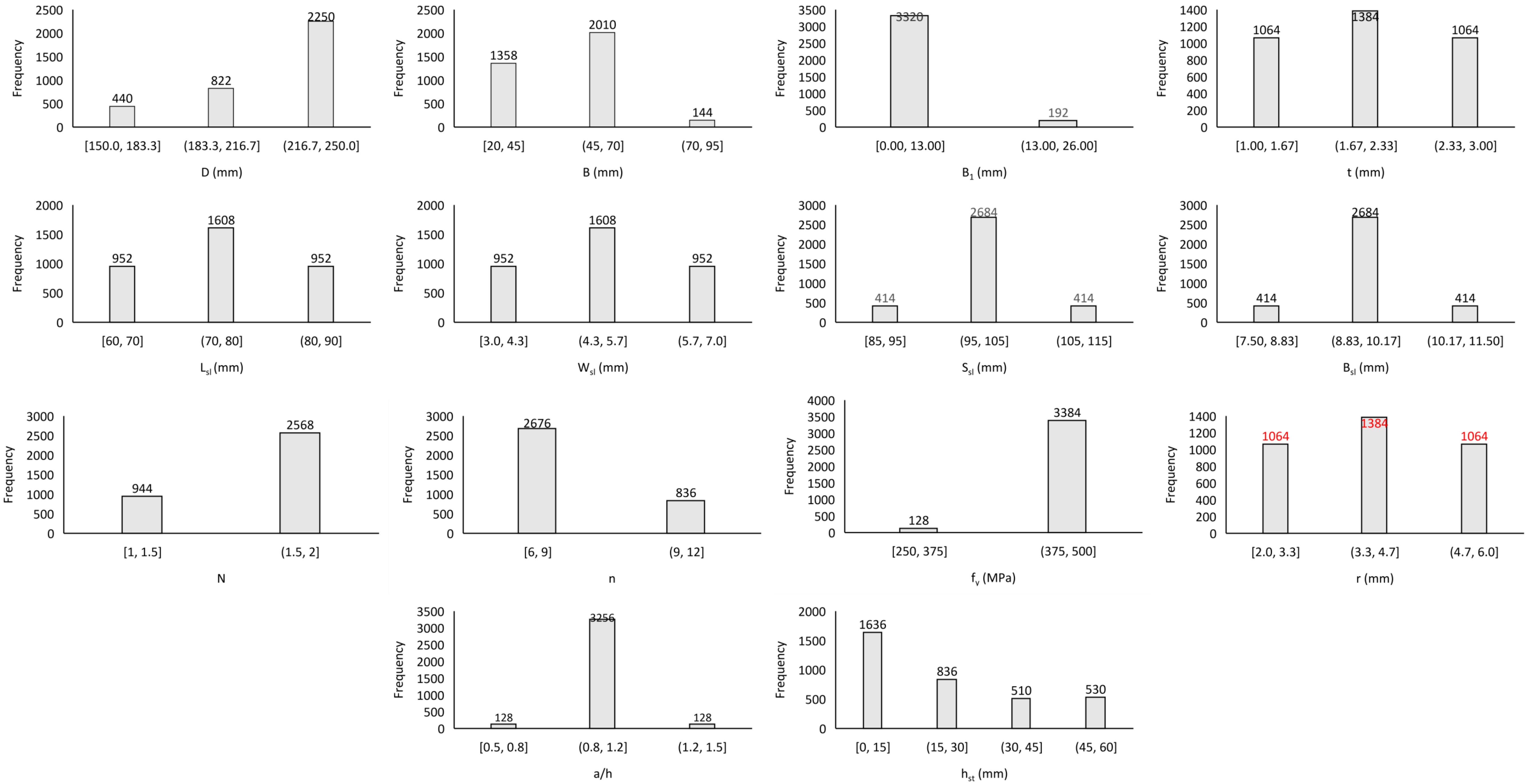
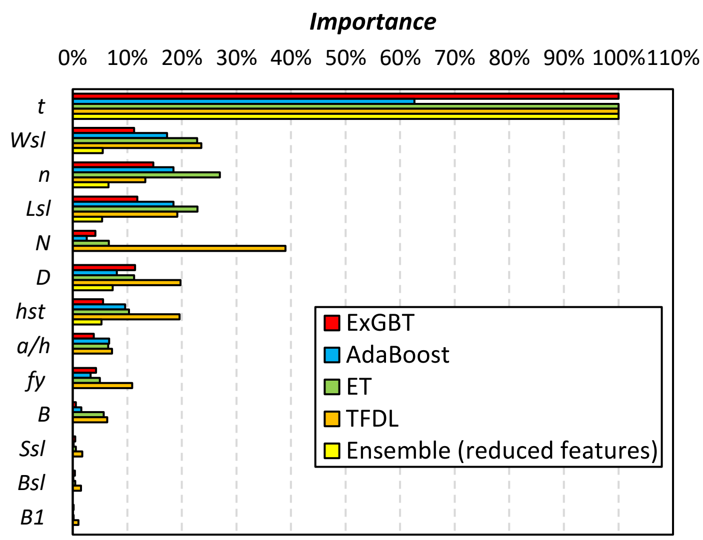



Fig. 5 Frequency of identified features of selected channels in the compiled database

As mentioned earlier, we lack a simple mathematical expression to evaluate the ultimate shear strength of CFS channels with slotted webs. Hence, deriving a *mapping function* will be helpful to engineers and designers to enable using such channels in practical scenarios. In this case study, the identified features that satisfy the set conditions by the four algorithms are $D$, $t$, $L_{sl}$, $W_{sl}$, $n$, and $h_{st}$ (see Fig. 6a), and these features were used to build an ensemble. Similar to the first case study, the ensemble was augmented with a surrogate to yield a *mapping function,* as shown below. The performance of both ensemble and mapping functions is displayed in Table 5 and Fig. 6b. Both cross-examinations show the validity of the ensemble and derived *mapping function,* which attains low error values as compared to the measured shear strength values. In addition, the ensemble and mapping function also score well with respect to $R^2$ (>84%). Finally, one can see that the *mapping function* derived herein also satisfies Smith [57] ($R$= 0.91 > 0.8) and Roy and Roy's [58] ($R_m$ = 0.52 > 0.5) recommendations – thereby ensuring a new layer of validation.

$$V_n = \frac{6174.2Dt}{\max(0.675 \text{ or } L_{sl}+h_{st})} + 1.87W_{sl}n + \frac{55669.19Dt^2}{(L_{sl}\times\max(0.675 \text{ or } 0.988+L_{sl}+1.867W_{sl}n))} - 1644 \quad (11)$$

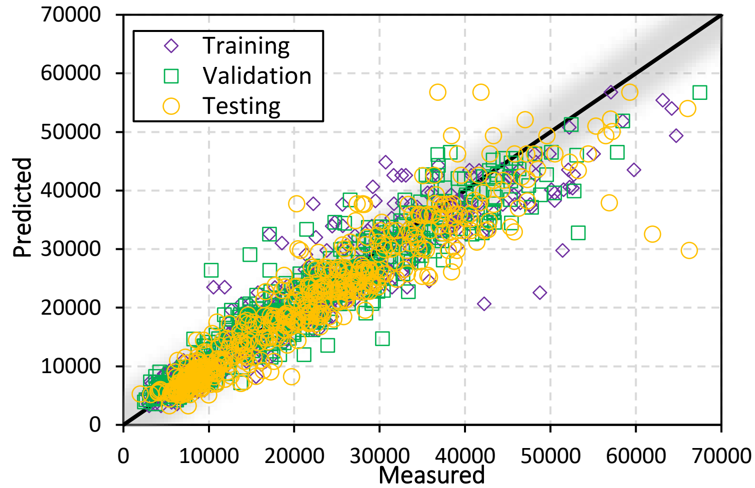



(a) Feature importance

This is a preprint draft. The published article can be found at: https://doi.org/10.1016/j.measurement.2021.110098

Please cite this paper as:

**Naser M.Z.** (2021). "Mapping Functions: A Physics-guided, Data-driven and Algorithm-agnostic Machine Learning Approach to Discover Causal and Descriptive Expressions of Engineering Phenomena." *Measurement.* https://doi.org/10.1016/j.measurement.2021.110098




(b) Measured and predicted ultimate shear strength using *mapping function*

Fig. 6 Evaluation of feature importance and *mapping function*

Table 5 List of selected performance metrics for training/validation/testing regimes.

| ***Metric*** | ***Ensemble*** | | | ***Mapping function*** | | |
|---|---|---|---|---|---|---|
| MAE | 3054.71 | 3200.83 | 3195.00 | 3283.18 | 3420.48 | 3195.13 |
| RMSE | 4467.25 | 5217.82 | 5434.79 | 19.32 | 18.60 | 17.56 |
| $R^2$ | 88.27 | 85.05 | 84.06 | 87.20 | 84.32 | 84.75 |

*Case study 3: mapping functions for cyclic response of reinforced concrete beams strengthened with CFRP*

In this case study, eight RC cantilever beams were collected from the work of Tanarslan [88]. These beams were 200 mm wide and 350 mm deep, with a span of 1600 mm (see Fig. 7). All beams were designed to be shear deficient and hence were reinforced with three 20 mm diameter bars in the compression zone and three 20 mm diameter bars in the tension zone covered with a concrete cover of 30 mm. The examined beams were tested under cyclic loading, and the measured data contained load ($P$) – deformation ($\Delta$) points. Seven beams were strengthened with carbon fiber reinforced concrete rebars following the near-surface mounted method. The beams varied CFRP spacing ($S$) and bar size ($D$) while maintaining a shear span ratio of 5 and compressive strength of concrete at 25 MPa. As such, this database presents a comfortable size of controlled (limited in number) features, as can be seen in Table 6. As one can see, there is a strong correlation and mutual information between load and deformations with minor correlation and mutual information within the other features. Given the nature of the measured data in cyclic tests, correlation and mutual information between some features were not calculated.

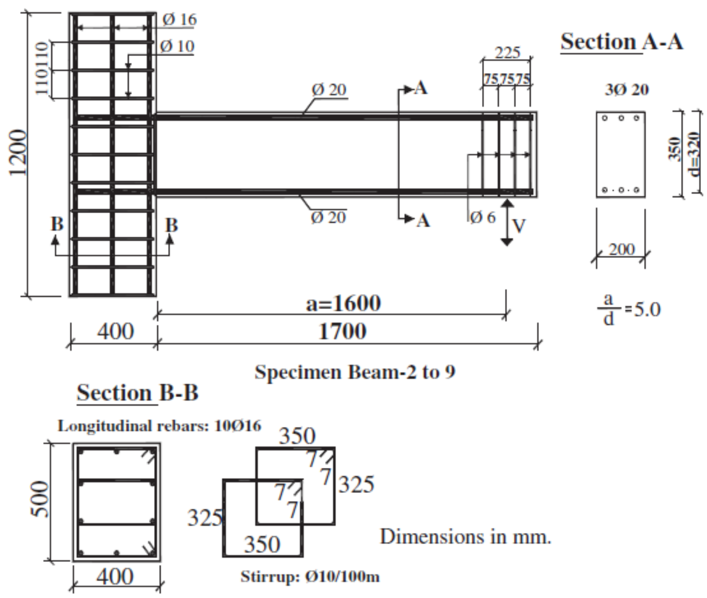



Fig. 7 Reinforcement details of beams (Credit Line: Elsevier, Construction and Building Materials. The effects of NSM CFRP reinforcements for improving the shear capacity of RC beams, H.M. Tanarslan, July 12, 2011, License Number: 5106680425995)

Table 6 Statistical insights from the collected database.

| *Pearson correlation* | | | | | | |
|---|---|---|---|---|---|---|
| *Parameter* | *P* | *a/d* | $f_c$ | *D* | *S* | *Δ* |
| *P* | 1.000 | | | | | |
| *a/d* | - | 1.000 | | | | |
| $f_c$ | - | - | 1.000 | | | |
| *D* | 0.055 | - | - | 1.000 | | |
| *S* | 0.010 | - | - | 0.565 | 1.000 | |
| *Δ* | 0.952 | - | - | 0.077 | -0.019 | 1.000 |
| *Mutual Information* | | | | | | |
| *Parameter* | *P* | *a/d* | $f_c$ | *D* | *S* | *Δ* |
| *P* | 0.732 | | | | | |
| *a/d* | - | - | | | | |
| $f_c$ | - | - | - | | | |
| *D* | 0.046 | - | - | 0.004 | | |
| *S* | 0.039 | - | - | 0.535 | 0.004 | |
| *Δ* | 0.574 | - | - | 0.053 | 0.042 | 1.000 |

This is a preprint draft. The published article can be found at: https://doi.org/10.1016/j.measurement.2021.110098

Please cite this paper as:

**Naser M.Z.** (2021). "Mapping Functions: A Physics-guided, Data-driven and Algorithm-agnostic Machine Learning Approach to Discover Causal and Descriptive Expressions of Engineering Phenomena." *Measurement.* https://doi.org/10.1016/j.measurement.2021.110098

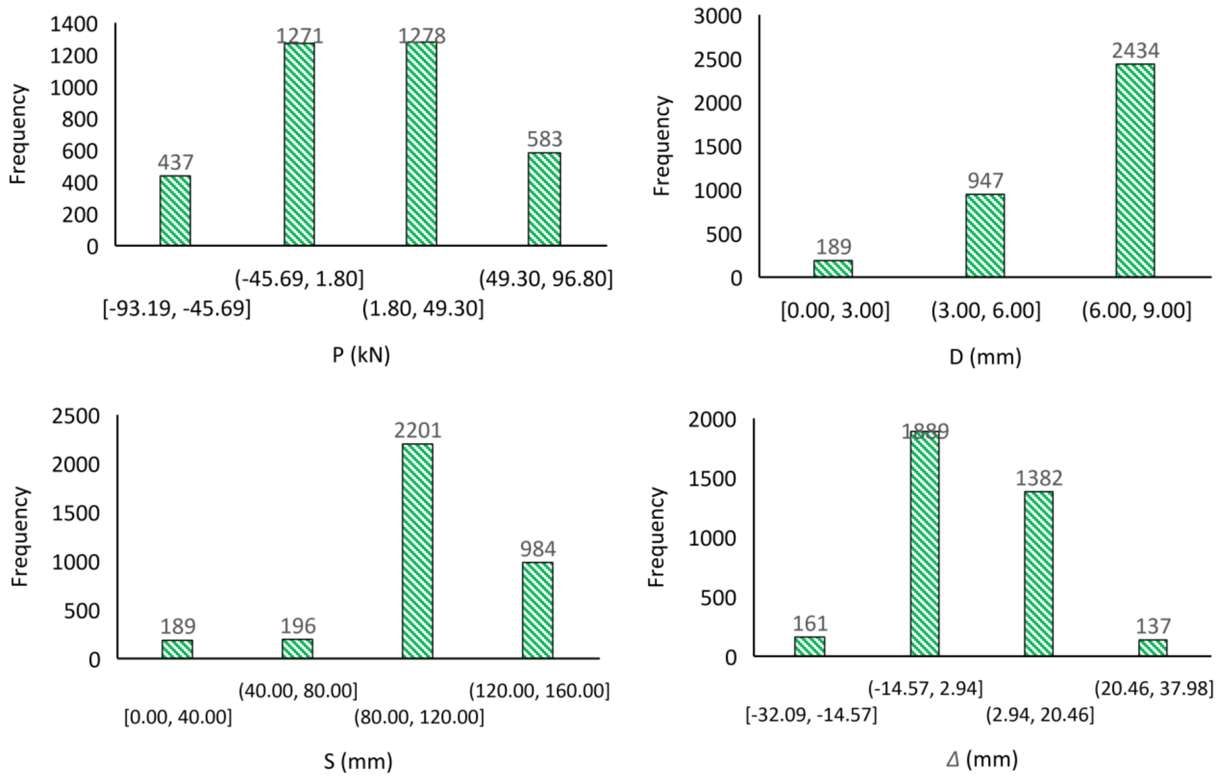



Fig. 8 Frequency of identified features in the compiled database

In this case study, the goal is to derive a *mapping function* that can allow re-construction of load-deformation (cyclic) history response of shear deficient reinforced concrete beams strengthened with CFRP. Figure 9 shows that *P* is the only reoccurring feature that appears to have an importance that satisfies the conditions outlined in this work which from a data examination point of view agrees with the fact that only the *P* and deformation are varied to a much larger extent as opposed to other features (*S* or *D* which are fixed for each particular beam). However, relying on *P* only will not be informative as the response of the eight identical beams examined as part of this work varies due to the different strengthening systems used (in terms of CFRP rebar diameter, *D*, and spacing, *S*, between such rebars). Thus, all three features are used herein to derive a *mapping function*.

The performance of both ensemble and *mapping function* is displayed in Table 7 and Fig. 9. The error observed by performance metrics for ensemble and *mapping function* prediction are low (within 1.5 mm for MAE and with 2.5 mm for RMSE). In addition, the *mapping function* derived herein also satisfies Smith [57] ($R = 0.97 > 0.8$) recommendation, and also passes Roy and Roy's [58] recommendation ($R_m = 0.73 > 0.5$). Also, the cyclic response of three beams was plotted further to showcase the validity of the derived mapping function. Both illustrations show the validity of the ensemble and derived *mapping function*.

$$D = 0.761 + 0.153P + 0.00073S^2D + 0.00017P^2 + 0.0096D^3 + 0.00057PD^2 + 1.357 \times 10^{-6}P^3 + 1.199Step(0.0019P) - 0.0685S - 0.00037PS - 7.35 \times 10^{-5}DS^2 \quad (12)$$

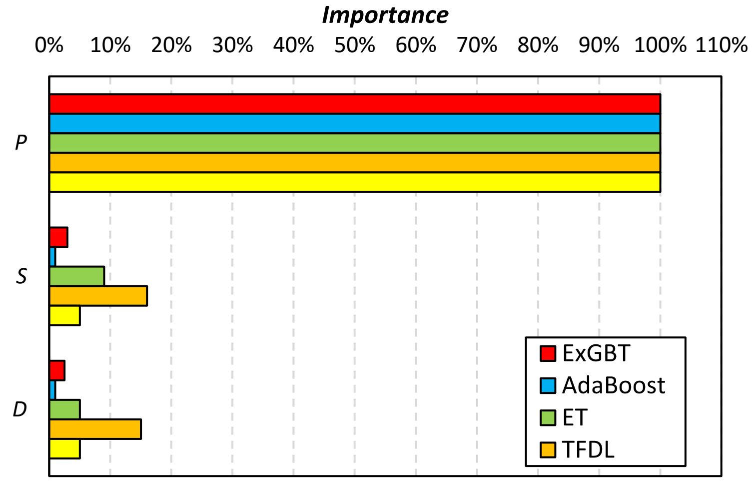
Table 7 List of selected performance metrics for training/validation/testing regimes.

| ***Metric*** | ***Ensemble*** | | | ***Mapping function*** | | |
|---|---|---|---|---|---|---|
| MAE | 1.33 | 1.39 | 1.28 | 1.42 | 1.49 | 1.41 |
| RMSE | 2.25 | 2.40 | 2.14 | 2.32 | 2.48 | 2.21 |
| $R^2$ | 94.88 | 94.44 | 95.25 | 94.54 | 94.03 | 94.95 |

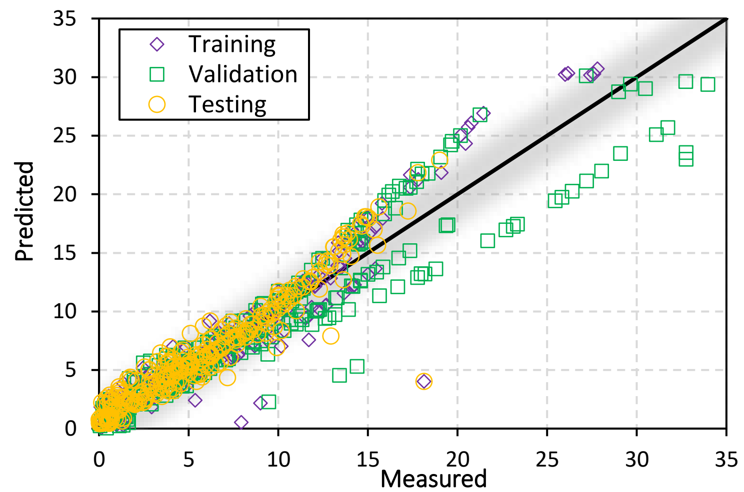
This is a preprint draft. The published article can be found at: https://doi.org/10.1016/j.measurement.2021.110098

Please cite this paper as:

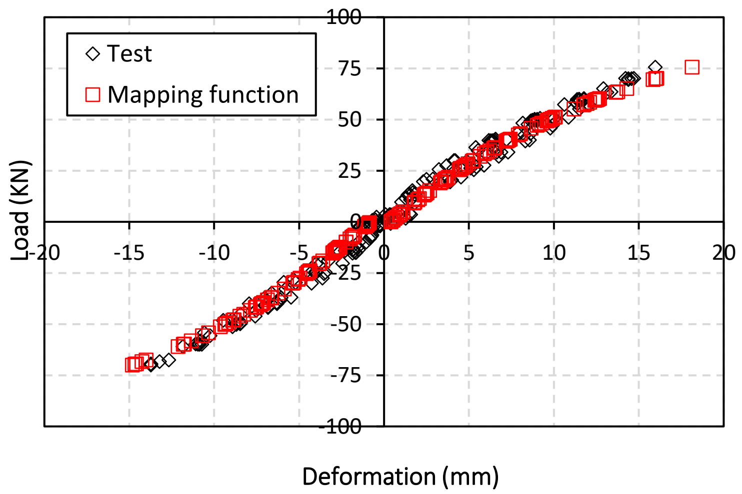
**Naser M.Z.** (2021). "Mapping Functions: A Physics-guided, Data-driven and Algorithm-agnostic Machine Learning Approach to Discover Causal and Descriptive Expressions of Engineering Phenomena." *Measurement.* https://doi.org/10.1016/j.measurement.2021.110098



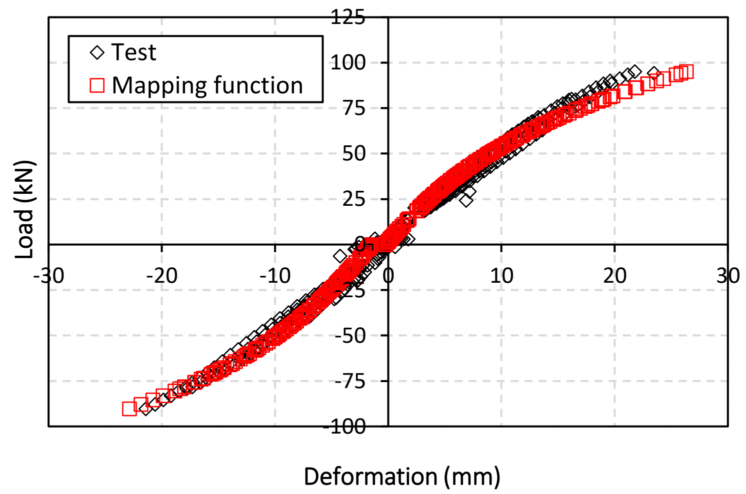

(a) Feature importance



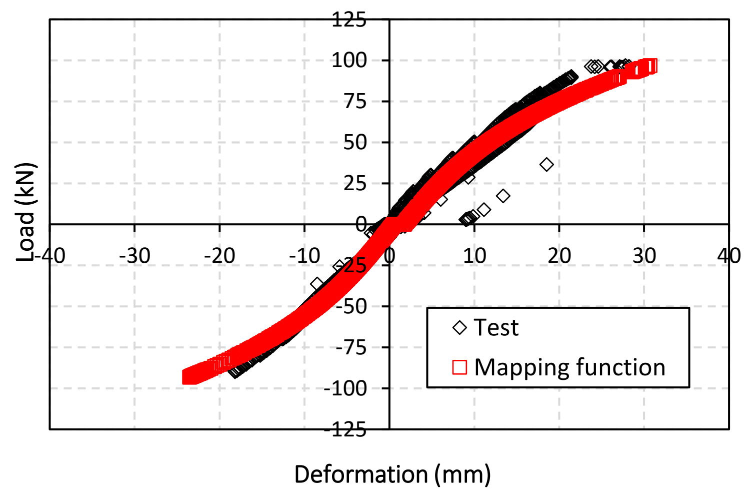

(b) Measured and predicted deformations




(c) Comparison for Beam 5




(d) Comparison for Beam 6




(e) Comparison for Beam 8

Fig. 9 Evaluation of feature importance and *mapping function*

This is a preprint draft. The published article can be found at: https://doi.org/10.1016/j.measurement.2021.110098

Please cite this paper as:

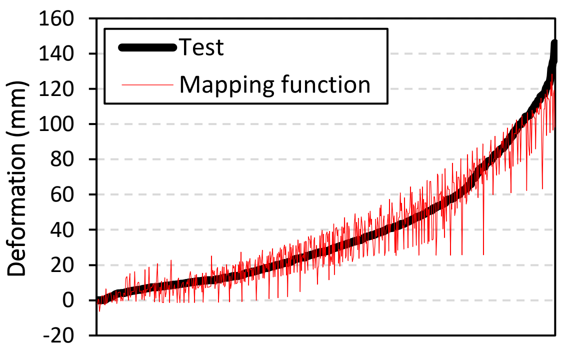
**Naser M.Z.** (2021). "Mapping Functions: A Physics-guided, Data-driven and Algorithm-agnostic Machine Learning Approach to Discover Causal and Descriptive Expressions of Engineering Phenomena." *Measurement.* https://doi.org/10.1016/j.measurement.2021.110098

## Further Insights into *Mapping Functions*

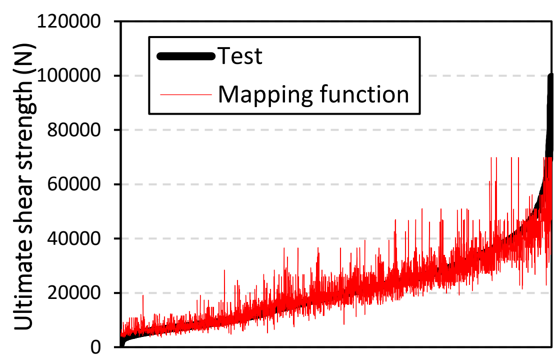
This section presents additional insights and observations that arose during this work to provide the reader with a holistic look into the proposed *mapping function* approach.

### *Global vs. local predictivity*

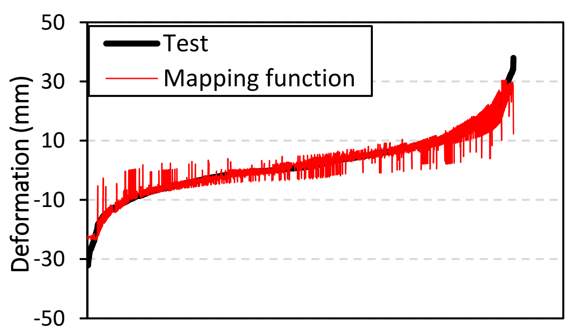
Figure 10 displays a comparison between measured observations from tests to corresponding predicted values obtained from *mapping functions* in the aforenoted three case studies once in an organized manner (e.g., ascending order). As one can see, the mapping functions do seem to adequately capture each examined phenomenon which also meshes with good performance as displayed by performance metrics discussed in each case study. A deep dive into each sub-figure further shows that the derived functions also seem to have lower predictivity than the extreme ranges of each database. This slightly "off" performance is partly due to the limited number of observations belonging to the extreme range and made available for the ensemble. As such, it is also advisable to also cross-check the validity of *mapping functions* across the full range of data and at a local level (i.e., extreme ranges), in addition to that taken by performance metrics which evaluate the global predictivity of the function.




(a) Deformation in RC beams under fire conditions




(b) Ultimate shear strength in CFS channels with slotted webs




(c) Cyclic history of shear deficient CFRP-strengthened RC beams

Fig. 10 Additional insights into the performance of *mapping functions* at local vs. global levels

*Case-by-case predictivity*

It is also advisable to cross-check the predictivity of derived mapping functions on a case-by-case basis (i.e., against each example used in the ML analysis). While the majority of the examples used herein show that the derived mapping functions achieve high predictivity, a few examples were seen to suffer a bit. For example, Fig. 11 showcases such examples as taken from case studies 1 and 3 (which rely on a continuous prediction of performance as opposed to predicting a single value as in case study 2). A close examination of Fig. 11a indicates the worst case seen during this work. It is clear that the mapping function captures the majority of the deformation history on Beam 7 but fails in properly capturing this response beyond the 70-minute mark. A similar observation can also be seen in the case of Beam 2. Figure 11 also shows two examples (Beam 2 and Beam 4) taken from the third case study. This observation can be linked to the need to examine additional generalization techniques to the derived *mapping function*. While predictions from the derived *mapping function* are adequate (especially in Beam 4, instead of Beam 2), these two cases are shown herein to note the small gap apparent near the zero region. This gap is linked to the (Step function) embedded within the derived *mapping function*. Tuning the Step function has been shown to lessen this gap.

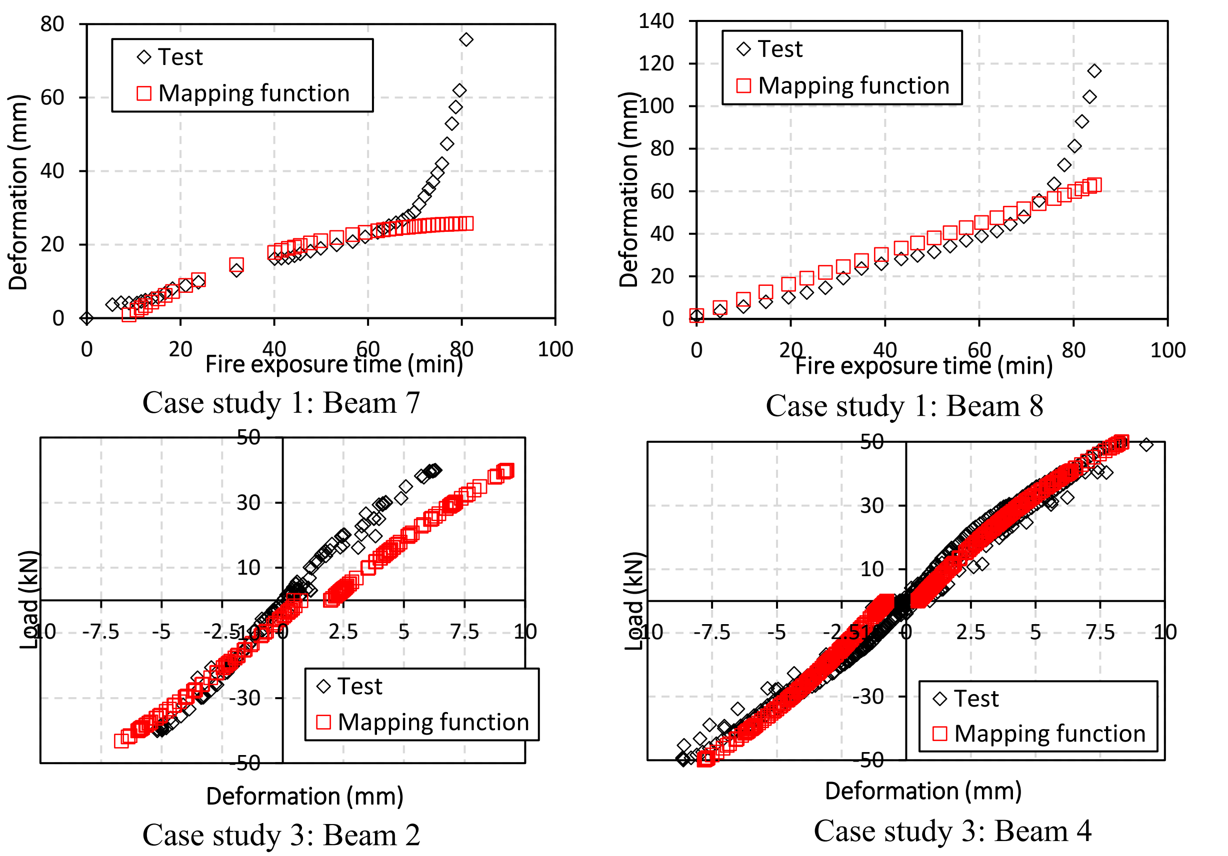



Fig. 11 Additional insights into the performance of mapping functions at a case-by-case basis

*Supplementary thoughts*

Additional items such as those related to specific prioritizing algorithmic families, optimization functions, finetuning hyperparameter, choice of performance metrics are worth investigating, and these will be examined in future works. Similarly, this work concentrated on deriving one-shot *mapping functions* (i.e., those that can be used in a single expression); however, developing multi-*mapping functions* can be of merit, especially to describe phenomena of high dimensionality/ complexity or those of coupled nature. In addition, there could exist a slight trade-off between the convenience of a mapping function and the accuracy of a complex finite element model. This would be further explored in a future work.

## Conclusions

This work presents *mapping functions* as a cognitive ML and simulation-free approach to derive physics-guided expressions to describe engineering phenomena. In this approach, a series of feature selection methods, together with insights from physics principles, are applied to identify critical features that govern the phenomenon on hand. The identified features are then examined via a ML ensemble which is then augmented with a surrogate to derive a *mapping function*. The proposed approach has been examined against three case studies with notable success (as examined across a series of performance metrics); deformation history of beams under fire, ultimate shear strength of cold-formed steel channels, and cyclic response of shear deficient CFRP-strengthened beams. The following list of inferences can also be drawn from the findings of this study:

- Feature selection methods can aid in finetuning the space of search and hence accelerate the development of ML models.
- *Mapping functions* present a modern approach to supplement engineers in evaluating problems via ML. Such functions may reduce the reliance and need for complex and expensive physical tests and numerical models.
- The proposed approach can be further improved with respect to generalizing *mapping functions* in future works. In addition, interested works are invited to explore the space of causality arising from large and small datasets, together with the influence of data quality (as obtained raw from sensors, etc.) and identifying suitable (problem-specific) performance metrics.

## Data Availability

Some or all data, models, or code that support the findings of this study are available from the corresponding author upon reasonable request.

## Acknowledgement

I would like to thank the Editor and Reviewers for their support of this work and constructive comments that enhanced the quality of this manuscript.

## Conflict of Interest

The author declares no conflict of interest.

## References

[1] M.A. Biot, Analytical And Experimental Methods in Engineering Seismology, Trans. Am. Soc. Civ. Eng. (1943). https://doi.org/10.1061/taceat.0005571.

[2] I. Babuska, J.T. Oden, Verification and validation in computational engineering and science: Basic concepts, Comput. Methods Appl. Mech. Eng. (2004). https://doi.org/10.1016/j.cma.2004.03.002.

[3] M.D. Pearl J, The Book of Why_ The New Science of Cause and Effect-Basic Books, 2018.

[4] N.E. Shanmugam, V. Thevendran, Y.H. Tan, Design formula for axially compressed perforated plates, Thin-Walled Struct. (1999). https://doi.org/10.1016/S0263-8231(98)00052-4.

[5] C.A. Ellingwood, B., Galambos, T. V., McGregor, J. G. & Cornell, Development of a Probability Based Load Criterion for American National Standard A58, U.S. Dep. Commer. Natl. Bur. Stand. (1980).

[6] ASCE, Minimum Design Loads for Buildings and Other Structures (ASCE/SEI 7-16), 2016.

[7] K.H. Reineck, D.A. Kuchma, K.S. Kim, S. Marx, Shear database for reinforced concrete members without shear reinforcement, ACI Struct. J. (2003). https://doi.org/10.14359/12488.

[8] E. Nakamura, A.R. Avendaño, O. Bayrak, Shear database for prestressed concrete members, ACI Struct. J. (2013). https://doi.org/10.14359/51686147.

[9] Z. Lai, A.H. Varma, High-Strength Rectangular CFT Members: Database, Modeling, and Design of Short Columns, J. Struct. Eng. (2018). https://doi.org/10.1061/(asce)st.1943-541x.0002026.

[10] S. Thai, H.T. Thai, B. Uy, T. Ngo, Concrete-filled steel tubular columns: Test database, design and calibration, J. Constr. Steel Res. (2019). https://doi.org/10.1016/j.jcsr.2019.02.024.

[11] E. Christodoulou, J. Ma, G.S. Collins, E.W. Steyerberg, J.Y. Verbakel, B. Van Calster, A systematic review shows no performance benefit of machine learning over logistic regression for clinical prediction models, J. Clin. Epidemiol. (2019). https://doi.org/10.1016/j.jclinepi.2019.02.004.

[12] D. Bzdok, N. Altman, M. Krzywinski, Statistics versus machine learning, Nat. Methods. (2018). https://doi.org/10.1038/nmeth.4642.

[13] M.Z. Naser, Machine learning assessment of fiber-reinforced polymer-strengthened and reinforced concrete members, ACI Struct. J. (2020). https://doi.org/10.14359/51728073.

[14] J.L. Hodges, B.Y. Lattimer, K.D. Luxbacher, Compartment fire predictions using transpose convolutional neural networks, Fire Saf. J. (2019). https://doi.org/10.1016/j.firesaf.2019.102854.

[15] D.J. Hand, Probability for Statistics and Machine Learning: Fundamentals and Advanced Topics by Anirban DasGupta, Int. Stat. Rev. (2013). https://doi.org/10.1111/insr.12011_5.

[16] Ž. Ivezic, A.J. Connolly, J.T. VanderPlas, A. Gray, Statistics, Data Mining, and Machine Learning in Astronomy, 2014. https://doi.org/10.23943/princeton/9780691151687.001.0001.

[17] A. Behnood, E.M. Golafshani, Machine learning study of the mechanical properties of concretes containing waste foundry sand, Constr. Build. Mater. (2020). https://doi.org/10.1016/j.conbuildmat.2020.118152.

[18] M.Z. Naser, Fire resistance evaluation through artificial intelligence - A case for timber structures, Fire Saf. J. 105 (2019). https://doi.org/10.1016/j.firesaf.2019.02.002.

[19] S. Feng, H. Zhou, H. Dong, Using deep neural network with small dataset to predict material defects, Mater. Des. (2019). https://doi.org/10.1016/j.matdes.2018.11.060.

[20] M.Z. Naser, H. Zhou, Machine Learning to Derive Unified Material Models for Steel Under Fire Conditions, in: Intell. Data Anal. Decis. Syst. Hazard Mitig., 2021: pp. 213–225. https://doi.org/10.1007/978-981-15-5772-9_11.

[21] K. Siau, W. Wang, Building trust in artificial intelligence, machine learning, and robotics, Cut. Bus. Technol. J. (2018).

[22] A. Giusti, J. Guzzi, D.C. Ciresan, F.L. He, J.P. Rodriguez, F. Fontana, M. Faessler, C. Forster, J. Schmidhuber, G. Di Caro, D. Scaramuzza, L.M. Gambardella, A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots, IEEE Robot. Autom. Lett. (2016). https://doi.org/10.1109/LRA.2015.2509024.

[23] C.D.W. Hildebrand, Ockham, Studies and Selections By Stephen Chak Tornay. LaSalle, Ill.: Open Court Publishing Company, 1938. viii, 207 pages. $1.75., Church Hist. (1938). https://doi.org/10.2307/3160457.

[24] P. Domingos, The role of Occam's Razor in knowledge discovery, Data Min. Knowl. Discov. (1999). https://doi.org/10.1023/A:1009868929893.

[25] Y. Saeys, I. Inza, P. Larrañaga, A review of feature selection techniques in bioinformatics, Bioinformatics. (2007). https://doi.org/10.1093/bioinformatics/btm344.

[26] M. Kuhn, K. Johnson, Applied predictive modeling, 2013. https://doi.org/10.1007/978-1-4614-6849-3.

[27] G. Chandrashekar, F. Sahin, A survey on feature selection methods, Comput. Electr. Eng. (2014). https://doi.org/10.1016/j.compeleceng.2013.11.024.

[28] H. Liu, L. Yu, Toward integrating feature selection algorithms for classification and clustering, IEEE Trans. Knowl. Data Eng. (2005). https://doi.org/10.1109/TKDE.2005.66.

[29] V. Kumar, Feature Selection: A literature Review, Smart Comput. Rev. (2014). https://doi.org/10.6029/smartcr.2014.03.007.

[30] A. Jović, K. Brkić, N. Bogunović, A review of feature selection methods with applications, in: 2015 38th Int. Conv. Inf. Commun. Technol. Electron. Microelectron. MIPRO 2015 - Proc., 2015. https://doi.org/10.1109/MIPRO.2015.7160458.

[31] Engineering Applications of Correlation and Spectral Analysis, Proc. IEEE. (1995). https://doi.org/10.1109/JPROC.1995.1200275.

[32] B.W. Xu, H.S. Shi, Correlations among mechanical properties of steel fiber reinforced concrete, Constr. Build. Mater. (2009). https://doi.org/10.1016/j.conbuildmat.2009.08.017.

[33] J. Hauke, T. Kossowski, Comparison of values of pearson's and spearman's correlation coefficients on the same sets of data, Quaest. Geogr. (2011). https://doi.org/10.2478/v10117-011-0021-1.

[34] P. Embrechts, A. McNeil, D. Straumann, Correlation Pitfalls and Alternatives, Risk Mag. (1999).

[35] M. Beraha, A.M. Metelli, M. Papini, A. Tirinzoni, M. Restelli, Feature Selection via Mutual Information: New Theoretical Insights, in: Proc. Int. Jt. Conf. Neural Networks, 2019. https://doi.org/10.1109/IJCNN.2019.8852410.

[36] J.R. Vergara, P.A. Estévez, A review of feature selection methods based on mutual information, Neural Comput. Appl. (2014). https://doi.org/10.1007/s00521-013-1368-0.

[37] Q. Gu, Z. Li, J. Han, Generalized fisher score for feature selection, in: Proc. 27th Conf. Uncertain. Artif. Intell. UAI 2011, 2011.

[38] M. Naser, A. Alavi, Insights into Performance Fitness and Error Metrics for Machine Learning, Under Rev. (2020).

[39] X.W. Chen, J.C. Jeong, Enhanced recursive feature elimination, in: Proc. - 6th Int. Conf. Mach. Learn. Appl. ICMLA 2007, 2007. https://doi.org/10.1109/ICMLA.2007.44.

[40] P.J.M. van Laarhoven, E.H.L. Aarts, Simulated Annealing: Theory and Applications, 1987. https://doi.org/10.1007/978-94-015-7744-1.

[41] M. Kuhn, K. Johnson, Feature engineering and selection: A practical approach for predictive models, 2019. https://doi.org/10.1201/9781315108230.

[42] N. El Aboudi, L. Benhlima, Review on wrapper feature selection approaches, in: Proc. - 2016 Int. Conf. Eng. MIS, ICEMIS 2016, 2016. https://doi.org/10.1109/ICEMIS.2016.7745366.

[43] M.Z. Naser, Mechanistically Informed Machine Learning and Artificial Intelligence in Fire Engineering and Sciences, Fire Technol. (2021) 1–44. https://doi.org/10.1007/s10694-020-01069-8.

[44] ASCE, Minimum Design Loads for Buildings and Other Structures, 2016.

[45] J.H.H. Fellinger, L. Twilt, Fire Behaviour of Long Span Composite Floor, n.d. http://iafss.org/publications/fss/5/1093/view/fss_5-1093.pdf (accessed February 8, 2019).

[46] M.Z. Naser, AI-based cognitive framework for evaluating response of concrete structures in extreme conditions, Eng. Appl. Artif. Intell. 81 (2019). https://doi.org/10.1016/j.engappai.2019.03.004.

[47] A.H. Buchanan, Fire engineering for a performance based code, Fire Saf. J. 23 (1994) 1–16. https://doi.org/10.1016/0379-7112(94)90058-2.

[48] V. Kodur, M. Naser, Structural Fire Engineering, 1st ed., McGraw Hill Professional, 2020.

[49] F. Wiesner, L. Bisby, The structural capacity of laminated timber compression elements in fire: A meta-analysis, Fire Saf. J. (2018). https://doi.org/10.1016/j.firesaf.2018.04.009.

[50] T. Gernay, Fire resistance and burnout resistance of reinforced concrete columns, Fire Saf. J. (2019). https://doi.org/10.1016/j.firesaf.2019.01.007.

[51] V.K.R. Kodur, M. Garlock, N. Iwankiw, Structures in Fire: State-of-the-Art, Research and Training Needs, Fire Technol. 48 (2012) 825–39. https://doi.org/10.1007/s10694-011-0247-4.

[52] M. Fernández-Delgado, M.S. Sirsat, E. Cernadas, S. Alawadi, S. Barro, M. Febrero-Bande, An extensive experimental survey of regression methods, Neural Networks. (2019). https://doi.org/10.1016/j.neunet.2018.12.010.

[53] A.H. Alavi, A.H. Gandomi, M.G. Sahab, M. Gandomi, Multi expression programming: A new approach to formulation of soil classification, Eng. Comput. 26 (2010) 111–118. https://doi.org/10.1007/s00366-009-0140-7.

[54] M.Z. Naser, A. Seitllari, Concrete under fire: an assessment through intelligent pattern recognition, Eng. Comput. (2019) 1–14. https://doi.org/10.1007/s00366-019-00805-1.

[55] V. V. Degtyarev, Neural networks for predicting shear strength of CFS channels with slotted webs, J. Constr. Steel Res. (2021). https://doi.org/10.1016/j.jcsr.2020.106443.

[56] W.Z. Taffese, E. Sistonen, Machine learning for durability and service-life assessment of reinforced concrete structures: Recent advances and future directions, Autom. Constr. (2017).

https://doi.org/10.1016/j.autcon.2017.01.016.

[57] G. Smith, Probability and statistics in civil engineering., Collins, London, 1986.

[58] P.P. Roy, K. Roy, On some aspects of variable selection for partial least squares regression models, QSAR Comb. Sci. 27 (2008) 302–313. https://doi.org/10.1002/qsar.200710043.

[59] S.K. Babanajad, A.H. Gandomi, A.H. Alavi, New prediction models for concrete ultimate strength under true-triaxial stress states: An evolutionary approach, Adv. Eng. Softw. (2017). https://doi.org/10.1016/j.advengsoft.2017.03.011.

[60] Y. Zhang, H. V. Burton, H. Sun, M. Shokrabadi, A machine learning framework for assessing post-earthquake structural safety, Struct. Saf. (2018). https://doi.org/10.1016/j.strusafe.2017.12.001.

[61] S. Lim, S. Chi, Xgboost application on bridge management systems for proactive damage estimation, Adv. Eng. Informatics. (2019). https://doi.org/10.1016/j.aei.2019.100922.

[62] G. Rätsch, T. Onoda, K.R. Müller, Soft margins for AdaBoost, Mach. Learn. (2001). https://doi.org/10.1023/A:1007618119488.

[63] P. Geurts, D. Ernst, L. Wehenkel, Extremely randomized trees, Mach. Learn. (2006). https://doi.org/10.1007/s10994-006-6226-1.

[64] T. Hope, Y.S. Resheff, I. Lieder, Learning TensorFlow: A Guide to Building Deep Learning Systems, 2017.

[65] TensorFlow, GitHub - tensorflow/tensorflow: An Open Source Machine Learning Framework for Everyone, (2020). https://github.com/tensorflow/tensorflow (accessed February 9, 2021).

[66] Y. Freund, R.E. Schapire, A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting, J. Comput. Syst. Sci. (1997). https://doi.org/10.1006/jcss.1997.1504.

[67] Gradient boosted tree (GBT), (2019). https://software.intel.com/en-us/daal-programming-guide-details-24 (accessed April 9, 2019).

[68] Scikit, sklearn.ensemble.GradientBoostingRegressor — scikit-learn 0.24.1 documentation, (2020). https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html (accessed February 9, 2021).

[69] XGBoost Python Package, Python Package Introduction — xgboost 1.4.0-SNAPSHOT documentation, (2020). https://xgboost.readthedocs.io/en/latest/python/python_intro.html#early-stopping (accessed February 10, 2021).

[70] Scikit, sklearn.ensemble.AdaBoostRegressor — scikit-learn 0.24.1 documentation, (n.d.). https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostRegressor.html (accessed March 10, 2021).

[71] Scikit, sklearn.ensemble.ExtraTreesRegressor — scikit-learn 0.24.1 documentation, (n.d.). https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesRegressor.html?highlight=extratrees (accessed March 10, 2021).

[72] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D.G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, X. Zheng, TensorFlow: A system for large-scale machine learning, in: Proc. 12th USENIX Symp. Oper. Syst. Des. Implementation, OSDI 2016, 2016.

[73] G.L. Albuquerque, A.B. Silva, J.P.C. Rodrigues, V.P. Silva, Behavior of thermally restrained RC beams in case of fire, Eng. Struct. 174 (2018) 407–417. https://doi.org/10.1016/j.engstruct.2018.07.075.

[74] A.H. Shah, U.K. Sharma, Fire resistance and spalling performance of confined concrete columns, Constr. Build. Mater. 156 (2017) 161–74. https://doi.org/10.1016/j.conbuildmat.2017.08.167.

[75] T.B. Carlos, J.P.C. Rodrigues, R.C.A. de Lima, D. Dhima, Experimental analysis on flexural behaviour of RC beams strengthened with CFRP laminates and under fire conditions, Compos. Struct. 189 (2018) 516–28. https://doi.org/10.1016/j.compstruct.2018.01.094.

[76] B. Ellingwood, T.D. Lin, Flexure and Shear Behavior of Concrete Beams during Fires, J. Struct. Eng. 117 (2007) 440–458. https://doi.org/10.1061/(asce)0733-9445(1991)117:2(440).

[77] Y. Jiangtao, W. Yichao, H. Kexu, Y. Kequan, X. Jianzhuang, The performance of near-surface mounted CFRP strengthened RC beam in fire, Fire Saf. J. 90 (2017) 86–94. https://doi.org/10.1016/j.firesaf.2017.04.031.

[78] V.K.R. Kodur, B. Yu, Evaluating the Fire Response of Concrete Beams Strengthened with Near-Surface-Mounted FRP Reinforcement, J. Compos. Constr. 17 (2013) 517–529. https://doi.org/10.1061/(ASCE)CC.1943-5614.0000348.

[79] J.-C. Dotreppe, J.-M. Franssen, A. Bruls, R. Baus, P. Vandevelde, R. Minne, D. van Nieuwenburg, H. Lambotte, Experimental research on the determination of the main parameters affecting the behaviour of reinforced concrete columns under fire conditions, Mag. Concr. Res. (1997). https://doi.org/10.1680/macr.1997.49.179.117.

[80] N. Davey, L. Ashton, Investigations on Building Fires: Part V.: Fire Tests on Structural Elements, 1953.

[81] F. Thomas, C. Webster, Investigations on Building Fires: Part VI.: the Fire Resistance of Reinforced Concrete Columns, 1953.

[82] J.H. Hsu, C.S. Lin, Effect of fire on the residual mechanical properties and structural performance of reinforced concrete beams, J. Fire Prot. Eng. 4 (2008) 245–74. https://doi.org/10.1177/1042391507077171.

[83] L.L. Bai, Z.Q. Wang, Residual Bearing Capacity of Reinforced Concrete Member after Exposure to High Temperature, Adv. Mater. Res. 368 (2011) 577–581. https://doi.org/10.4028/www.scientific.net/amr.368-373.577.

[84] A.H. Gandomi, A.H. Alavi, M.G. Sahab, New formulation for compressive strength of CFRP confined concrete cylinders using linear genetic programming, Mater. Struct. Constr. (2010). https://doi.org/10.1617/s11527-009-9559-y.

[85] V. V. Degtyarev, N. V. Degtyareva, Numerical simulations on cold-formed steel channels with flat slotted webs in shear. Part I: Elastic shear buckling characteristics, Thin-Walled Struct. (2017). https://doi.org/10.1016/j.tws.2017.05.026.

[86] V. V. Degtyarev, N. V. Degtyareva, Numerical simulations on cold-formed steel channels with flat slotted webs in shear. Part II: Ultimate shear strength, Thin-Walled Struct. (2017). https://doi.org/10.1016/j.tws.2017.05.028.

[87] V. V. Degtyarev, N. V. Degtyareva, Numerical simulations on cold-formed steel channels with longitudinally stiffened slotted webs in shear, Thin-Walled Struct. (2018). https://doi.org/10.1016/j.tws.2018.05.001.

[88] H.M. Tanarslan, The effects of NSM CFRP reinforcements for improving the shear capacity of RC beams, Constr. Build. Mater. (2011). https://doi.org/10.1016/j.conbuildmat.2010.12.016.

[89] I. Iguyon, A. Elisseeff, An introduction to variable and feature selection, J. Mach. Learn. Res. (2003).

https://doi.org/10.1162/153244303322753616.

[90] M. Wehenkel, A. Sutera, C. Bastin, P. Geurts, C. Phillips, Random Forests Based Group Importance Scores and Their Statistical Interpretation: Application for Alzheimer's Disease, Front. Neurosci. (2018). https://doi.org/10.3389/fnins.2018.00411.

[91] B.D. Williamson, P.B. Gilbert, N.R. Simon, M. Carone, A unified approach for inference on algorithm-agnostic variable importance, ArXiv. (2020).

[92] D.E. Goldberg, J.H. Holland, Genetic Algorithms and Machine Learning, Mach. Learn. (1988). https://doi.org/10.1023/A:1022602019183.

[93] S.N. Sivanandam, S.N. Deepa, Introduction to genetic algorithms, 2008. https://doi.org/10.1007/978-3-540-73190-0.

[94] L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone, Classification and regression trees, 2017. https://doi.org/10.1201/9781315139470.