Applied Soft Computing 70 (2018) 737–750

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Stacking ensemble with parsimonious base models to improve generalization capability in the characterization of steel bolted components

A. Pernía-Espinoza, J. Fernandez-Ceniceros, J. Antonanzas, R. Urraca, F.J. Martinez-de-Pison*

EDMANS Group, Department of Mechanical Engineering, University of La Rioja, Logroño 26004, Spain

ARTICLE INFO

Article history:
Received 31 July 2017
Received in revised form 8 May 2018
Accepted 2 June 2018
Available online 22 June 2018

Keywords:
Stacking ensemble model
Parsimonious models
Bolted connection
GA-PARSIMONY
T-stub

ABSTRACT

This study presents a new soft computing method to create an accurate and reliable model capable of determining three key points of the comprehensive force–displacement curve of bolted components in steel structures. To this end, a database with the results of a set of finite element (FE) simulations, which represent real responses of bolted components, is utilized to create a stacking ensemble model that combines the predictions of different parsimonious base models. The innovative proposal of this study is using GA-PARSIMONY, a previously published GA-method which searches parsimonious models by optimizing feature selection and hyperparameter optimization processes. Therefore, parsimonious solutions created with a variety of machine learning methods are combined by means of a nested cross-validation scheme in a unique meta-learner in order to increase diversity and minimize the generalization error rate. The results reveal that efficiently combining parsimonious models provides more accurate and reliable predictions as compared to other methods. Thus, the informational model is able to replace costly FE simulations without significantly comprising accuracy and could be implemented in structural analysis software.

1. Introduction

Structural steel connections play an essential role in the stability of frames and buildings. Since the introduction of the semi-rigid concept [1] numerous methods have been developed to accurately predict the moment-rotation response of steel connections [2]. The semi-rigid approach represented an important breakthrough in the design of these elements that led to significant reductions in weight and costs. Nevertheless, complex and advanced analyses were required to tackle the nonlinearities involved in the calculation process.

Nowadays, the use of component-based models to assess semi-rigid steel connections is widely accepted among practitioners. The basic principle of these models consists of dividing the connection into individual components represented as springs, each one characterized by their force–displacement response in terms of initial stiffness and maximum strength. The moment-rotation curve of the connection is then determined by assembling the individual responses of all components in a mechanical system. For example, in beam-to-column bolted joints (Fig. 1a), tensile components represent the main source of deformability; hence, they are responsible for the rotation capacity of the entire joint. These components can be effectively characterized by means of the equivalent T-stub model (Fig. 1b).

Regulatory codes, such as Eurocode-3 [3], have adopted this approach which is suitable for hand calculation and reasonably accurate in determining rotational stiffness and moment resistance. However, further research is still necessary to adequately characterize components and obtain the comprehensive non-linear response of connections, including rotation capacity (Fig. 1c). Alternatives to the existing analytical models [4] should be able to handle material plasticity, contacts, progressive damage and failure. At the same time, practitioners need cost-effective methods in terms of computational effort that are also suitable to be incorporated into steel structure software.

To date, numerical simulations based on the finite element (FE) method represent one of the most accurate ways to predict the behavior of steel connections (Fig. 2). Nevertheless, the FE method continues to be a time-consuming process in spite of growing computer power. The complexity of analysis codes seems to keep pace

* Corresponding author.
E-mail address: fjmartin@unirioja.es (F.J. Martinez-de-Pison).

https://doi.org/10.1016/j.asoc.2018.06.005
1568-4946/

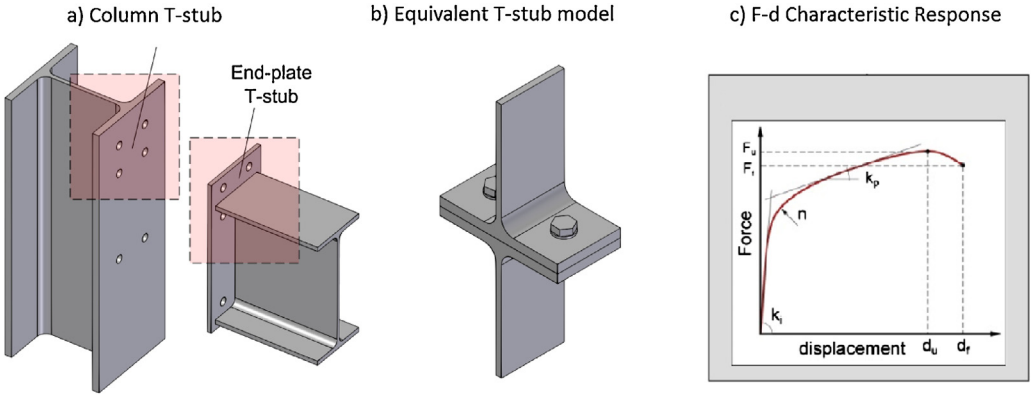


Fig. 1. Tension zone in steel connections. (a) End-plate beam-to-column connection, (b) equivalent T-stub model and (c) force–displacement curve of the T-sub model to be obtained.

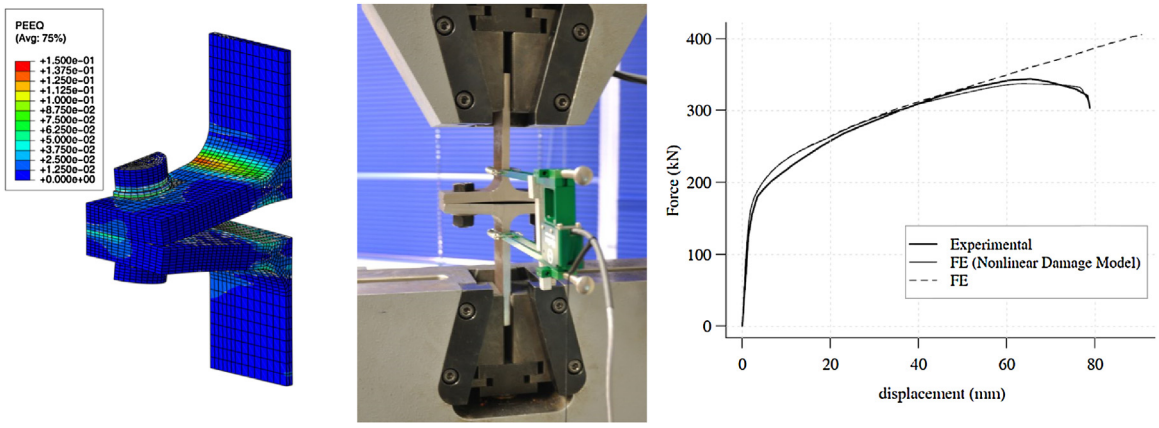


Fig. 2. Characterization of the force–displacement curve of a bolted component with an FE model validated with experimental tests.

with continuous advances in software and hardware [5]. Thus, the FE method remains inefficient in terms of current requirements when design and optimization are required for hundred of joints in a steel structure. Hence, a second level of abstraction is needed that would alleviate the computational burden required by numerical simulations.

Machine learning models created from an FE simulation database represent a low-cost approximation of computationally expensive simulations [6]. These models capture the underlying relationship between input variables and FE simulation results that can be expressed mathematically as:

$$y = f(x, \varphi) + \epsilon \tag{1}$$

where y is the actual value of the output, f contains the modeling function, $x = [x_1, \ldots, x_n]$ represents the array of n input variables, $\Phi = [\varphi_1, \ldots, \varphi_m]$ denotes the array of m unknown parameters to adjust the function f, and ϵ includes both the error of fitting the machine learning model to the simulation results and the intrinsic error corresponding to the simulation.

Concerning steel structures, artificial neural network (ANN) were utilized by Shahin et al. [7] to predict the ultimate pure bending of steel circular tubes. A specific example of the design of semi-rigid joints can be found in the work reported by Diaz et al. [8] who employed kriging and genetic algorithms (GA) for cost-optimization of bolted end-plate connections. Recently, Fernandez-Ceniceros et al. [9,10] have also offered insight into the use of machine learning models to assess steel bolted connections.

Model generalization capability involves the ability to predict correct responses for new designs. This capability depends on the machine learning algorithm, model complexity and the training process. In the context of modeling steel bolted connection behavior, the challenge is to obtain models capable of predicting highly accurate force–displacement curves for new designs trained with a relatively small database created with FE simulations. Selecting the most appropriate method for each case study remains a challenge for practical design. Model accuracy is generally influenced by several factors, such as the nature of the output variables, the number of computational FE experiments necessary to approximate the model, and the dimensionality of the problem in terms of number of input features. Regarding the former, highly nonlinear problems may require more flexible modeling techniques to capture their complex behavior. However, flexible techniques provide excellent fit for computer experiments at the expense of lower generalization capacity. Therefore, a methodology to obtain an efficient trade-off between flexibility and generalization capability is essential in order to overcome the phenomenon of over-fitting.

This article presents a new hybrid methodology which couples numerical and informational models to predict key parameters of steel connections such as initial stiffness, maximum resistance and displacement at failure. To this end, numerical models based on the finite element method (FEM) are developed first to reproduce the real response of bolted components selected with a design of experiments. These models incorporate progressive damage mechanisms and failure criteria to accurately estimate the displacement at fracture. In order to minimize the computational burden of the FEM, the results of a set of simulations are utilized to create a database and then, a stacking ensemble model is constructed. This

model, also called a meta-ensemble model, is capable of accurately determining the curve parameters for unseen data.

The primary innovation of this proposal is that it uses a soft computing (SC) method previously published by the authors, GA-PARSIMONY [11], to obtain parsimonious and accurate base solutions with different algorithms, such as ANN, support vector machines (SVM), CUBIST, gradient boosting machines (GBM) and ridge regression, among others. GA-PARSIMONY obtains accurate parsimonious models for the first level by selecting the best features and parameters with a special parsimonious model selection (PMS) process based on a dual criteria which considers accuracy and complexity separately. Finally, the predictions of the best parsimonious models obtained with each algorithm are combined with a second level stacking ensemble learner, also optimized with GA-PARSIMONY, in order to reduce variance and improve the generalization capability.

The rest of the article is organized as follows: Section 2 begins by introducing methods for improving model generalization capability and explaining its relationship with new SC techniques. Section 3 presents the proposed methodology which combines the FE method to construct a database and, by combining it with GA-PARSIMONY, builds an accurate stacking ensemble with parsimonious first level models created with different machine learning algorithms. Section 4 describes the experiments performed in the case study of a T-stub component. Then, Section 5 deals with the results of the proposed method and compares them with previous approaches. And lastly, Section 6 presents the conclusions and contributions of this study.

2. Related research on using soft computing for improving generalization capability

In machine learning, generalization capability refers to the ability of a model to handle unseen data. A model may perform well with the training data, but it may still fail to predict the response of a new dataset. This capability depends on an adequate training process and model complexity. Thus, poor generalization is obtained when the model is over-trained or when its degree of freedom is higher than that of the training data [12]. This second aspect is related to the complexity of a model, which can be defined by the model's internal structure [13], such as the number of leafs or levels in model trees, or the sum of squared coefficients in ridge regression, among others methods. For instance, the sum of squared weights is used in neural networks because overly large weights can magnify the noise from inputs and propagate it to the output [14]. Therefore, this metric defines the "flatness" of a model, which is directly related to the variance of the prediction. Other complexity metrics are the *Vapnik-Chervonenkis* (VC) dimension [15], the *degrees of freedom* (GDF) [16] or the number of input features selected (N_{SF}), which is included in the penalty terms of the *Akaike information criterion* [17] and the *Bayesian information criterion* (BIC) [18].

In this context, it is widely accepted that, among several accurate models, the least complex model (more parsimonious) should be selected. This model will probably be more robust and reliable against new data, perturbations and noise.

2.1. Trade-off between bias and variance

Models should be tested with unseen patterns in order to determine their generalization capability. Suppose that a real value y is explained as $y = f(\mathbf{x}) + \epsilon$, where $f(\mathbf{x})$ is the real function of the problem with n independent attributes x_i, and ϵ is the intrinsic error with zero mean and variance σ^2. Minimizing a metric such as the Mean Squared Error (MSE) allows us to search for models $\hat{f}(x)$ as close as possible to $f(x)$. Then, the expected MSE with an unseen sample $\mathbf{x}$, also known as generalization error (GE), can be decomposed as follows [19]:

$$E[(y - \hat{f}(\mathbf{x}))^2] = Bias[\hat{f}(\mathbf{x})]^2 + Var[\hat{f}(\mathbf{x})] + \sigma^2 \quad (2)$$

with:

$$Bias[\hat{f}(\mathbf{x})] = E[\hat{f}(\mathbf{x}) - f(x)] \quad (3)$$

$$Var[\hat{f}(\mathbf{x})] = E[\hat{f}(\mathbf{x})^2] - E[\hat{f}(\mathbf{x})]^2 \quad (4)$$

where the $Bias^2$ is related to the accuracy of the model, Var is the variance and represents the model repeatability or how much $\hat{f}(x)$ will move around the mean, and σ^2 is the irreducible error present in the original data.

The complexity of the model has opposite effects in the reduction of bias and variance. Fig. 3 shows the training error and U-shaped MSE curves for two test samples of unseen data in regard

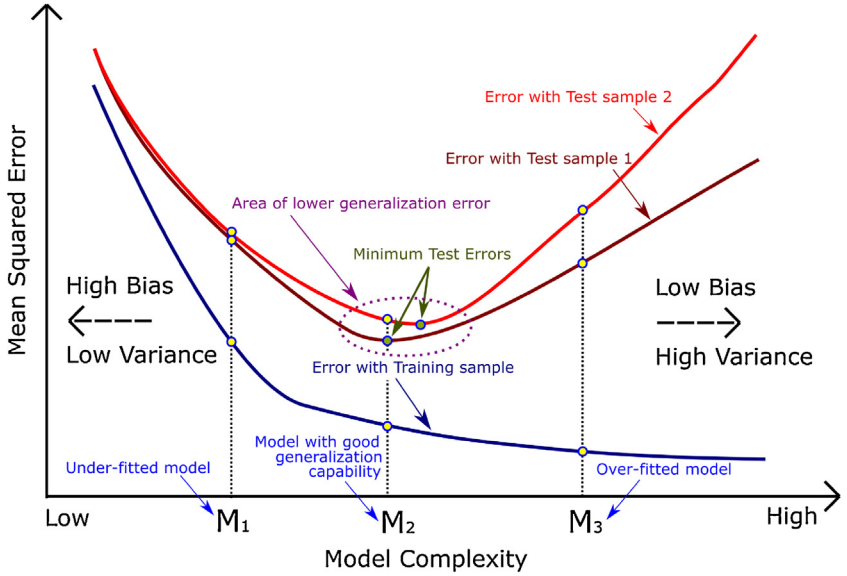


Fig. 3. Bias-variance and its relationship with model complexity. M_1 is an under-fitted simple model with high bias but low variance. M_3 is an over-fitted highly flexible model with low bias but high variance in unseen data. M_2 obtains a trade-off between bias and variance to achieve a satisfactory expected error with the test samples.

to model complexity (horizontal axis). A highly flexible model like M_3 obtains a low training MSE but the variance increases in the test samples. On the other hand, a low complex model, such as M_1, shows a high bias because it cannot adapt correctly to the data. Therefore, controlling the bias-variance trade-off is necessary to obtain strong overall models within the area of lower generalization errors, as occurs with model M_2. This control is called *regularization* and takes into account both the complexity and accuracy of prediction in the training process.

2.2. Regularization of wrapper models using soft computing

Regularization is included in the training process by minimizing a regularized "Loss+Penalty" function [19]:

$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{minimize}}\{L(\mathbf{X}, \mathbf{y}, \beta) + \lambda P(\beta)\} \tag{5}$$

where $\mathbf{X}$ is the input data, $\mathbf{y}$ the response variable, and β the model weights. The objective is to reduce $L(\mathbf{X}, \mathbf{y}, \beta)$, which is the loss function that quantifies how the model fits the training data ($\mathbf{X}$, $\mathbf{y}$), and $P(\beta)$, which is the penalty term for the complexity of the model. λ is a non-negative parameter that balances both terms and must be adjusted along with other tuning parameters.

An increasing number of studies have reported SC strategies where the optimization process considers the whole KDD-wrapper scheme. These schemes generally include not only hyperparameter optimization (HO) but also other tasks such as feature selection (FS), feature engineering (FE), data transformation (DT), or the use of several learners. Ma and Xia [20] present a tribe competition-based genetic algorithm for FS in pattern classification to obtain an optimal feature subset and produce more accurate classifiers. Wei et al. [21] develop a binary particle swarm optimization (BPSO) with SVM based on memory renewal and enhanced mutation mechanisms for FS. Perez-Rodriguez et al. [22] demonstrate that using evolutionary computation for simultaneous instance and feature selection plus feature weighting can improve model accuracy significantly. Huang and Chang [23] propose GA for FS and HO for SVM in microarray classification. Ding [24] use PSO in hyperspectral classification of remote sensing images. Vieira et al. [25] employ a binary PSO with a wrapper SVM approach to predict surviving or deceased patients with septic shock. Huang and Dun [26] design a distributed PSO for FS and HO. Wan et al. [27] present a modified binary coded ant colony optimization algorithm combined with GA for FS. Ahila et al. [28] use PSO for classification of power system disturbances. Dhiman et al. [29] propose a hybrid approach with a GA-SVM scheme for FS and HO in detecting epileptic seizures from background electroencephalogram signals. Winkler et al. [30] report a variety of evolutionary strategies to optimize different machine learning models in the identification of tumor markers. Wang et al. [31] perform HO and FS in medical diagnoses with a chaotic moth-flame optimization strategy. Medjahed et al. [32] propose the use of a gray wolf optimizer in hyperspectral image classification.

In general, these works use a more or less complex fitness function, but they only include a regularization mechanism in the training algorithm and not in the whole wrapper model selection process. Few methodologies search for the best parsimonious solution based on minimizing the cost and complexity of the whole wrapper. Chen et al. [33] use an evolutionary approach to simultaneously reduce complexity and optimize weights of learning vector quantification networks for bankruptcy prediction. Avalos et al. [34] present a methodology based on two penalization schemes, PAM1 and PAM2, for carrying out both function estimation and variable selection to obtain parsimonious additive models (PAM). Escobar and Morales-Menendez [35] develop a penalized maximum probability of correct decision (PMPCD) model selection criterion to efficiently solve the trade-off between N_{SF} and prediction in an ultrasonic metal welding quality control application. With this methodology, the selected parsimonious model had only 4 inputs from a highly unbalanced database formed by 54 features.

2.3. Searching parsimonious solutions with GA-PARSIMONY

Small differences in cost can be superfluous in many real applications. For example, researchers looking for a model that predicts the temperature set points (within 700–1000 °C) of a steel furnace prefer robust solutions with few inputs rather than marginal reductions in the fitness function. A difference between two models of less than ±1 °C can be insignificant. However, obtaining a parsimonious solution with high accuracy can be very useful to mitigate the uncertainty caused by perturbations in the production lines, such as noise and the tolerance of sensors. Simpler models are also easier to implement, update and understand.

For this purpose, Sanz-Garcia et al. [11] presented GA-PARSIMONY, a SC methodology based on GA to search accurate parsimonious models. This SC method optimizes a KDD-wrapper scheme with GA by using a parsimonious model selection (PMS). In each GA generation, the best solutions are first sorted by their cost (J), and then individuals with less complexity are moved to the top positions when the absolute difference between their Js is lower than a threshold value (α). Therefore, the selection of less complex solutions among those with similar accuracy promotes the evolution of robust models with better generalization capabilities. Another advantage of this methodology is that it eliminates the use of the regularization penalty weight λ because complexity and cost are evaluated separately.

Fig. 4 shows an example of how the selection process of GA-PARSIMONY works with four individuals and $\alpha = 1$. The process begins by defining an initial population of chromosomes Λ_0 : $\{\lambda_0^1, \lambda_0^2, \ldots, \lambda_0^p\}$ using a Latin Hypercube Sampling (LHS()):

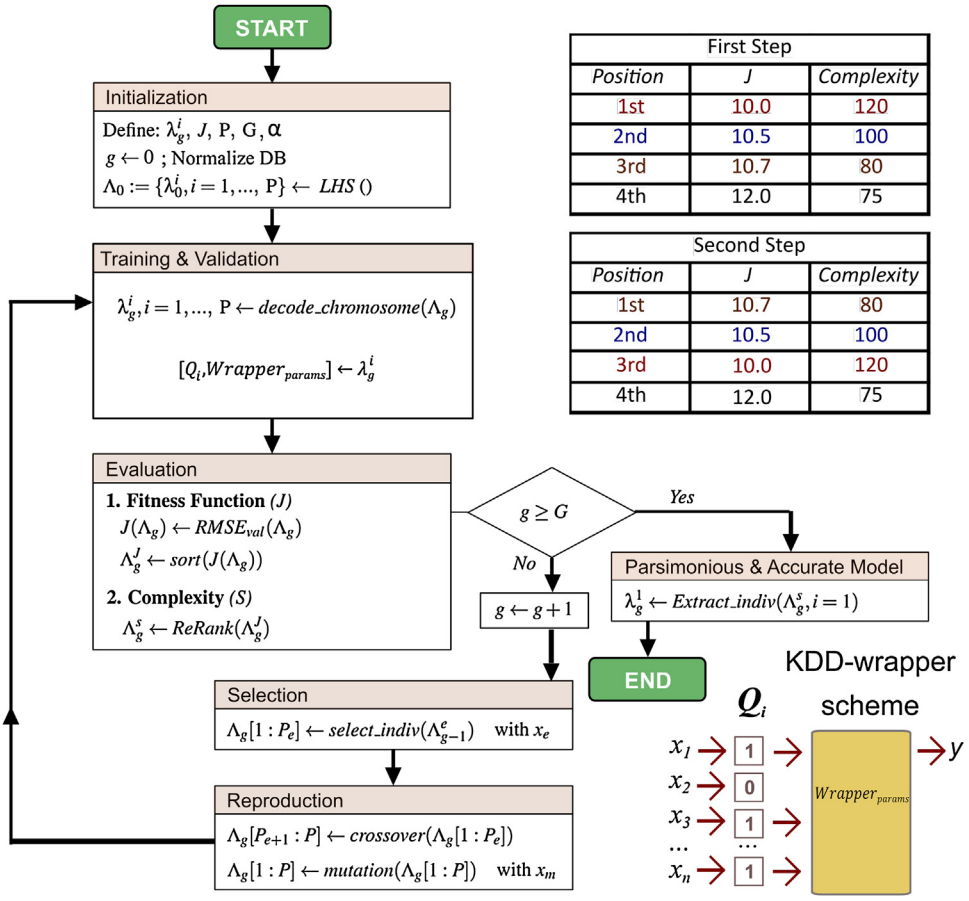
$$\Lambda_0 : \{\lambda_0^i, \quad i = 1, \ldots, P\} \longleftarrow LHS() \tag{6}$$

where λ_0^i is the i chromosome of the first generation ($g = 0$) and P is the number of individuals. The maximum number of generations (G), the cost function (J) and α have to be defined as well. Each chromosome i composed by two parts: (i) a binary coded array (Q_i) that represents the inputs used, and (ii) the parameters of the wrapper model being optimized ($Wrapper_{params}$) (Eq. (7)).

$$\lambda_g^i = [Q_i, Wrapper_{params}] \tag{7}$$

The evaluation and selection processes consist of a separated evaluation of cost and complexity. Continuing with the example of temperature set points, in the first step, individuals are sorted by their J, which is obtained with the Root Mean Squared Error (RMSE) using repeated cross-validation. Differences smaller than 1°C are not considered significant. Therefore, in the second step, individuals are rearranged by their *Complexity* if the difference between their Js is lower than $\alpha = 1.0$. Consequently, the parsimonious individual with *Complexity* = 80 and a difference with the lowest J of 0.7 is moved to the first position.

GA-PARSIMONY has been successfully applied with support vector regression (SVR), random forest (RF), ANN and GBM in a wide range of fields such as solar radiation forecasting [36], industrial processes [37] or hotel room demand estimation [38]. Furthermore, GA-PARSIMONY has already been used to predict the force–displacement of bolted components [10]. In that study, GA-PARSIMONY is used to optimize the search of parsimonious SVR models for seven key parameters of the curve. The SC-based models reported a high degree of accuracy with new predictions.



First Step		
Position	*J*	*Complexity*
1st	10.0	120
2nd	10.5	100
3rd	10.7	80
4th	12.0	75

Second Step		
Position	*J*	*Complexity*
1st	10.7	80
2nd	10.5	100
3rd	10.0	120
4th	12.0	75

Fig. 4. Flowchart of GA-PARSIMONY with a small example how the selection process works for four individuals with $\alpha = 1$.

2.4. Ensembled methods for reducing bias and variance

Ensemble methods (EM) combine multiple models into one model that is more accurate than the best of these models. EM are useful when the predictive accuracy of the problem is more important than model interpretability. *Bagging* [39] is one of the most popular methods among some classic EM, such as Random Forest (RF), AdaBoost, Stacking or Gradient Boosting Machines (GBM). It consists of creating n base models with different random samples obtained with replacement from the original dataset. The response is calculated by averaging the fitted learners, for regression problems, or with the plurality vote or the mean of probabilities, for classification. The advantages of bagging [13] are: it reduces the test error by smoothing out variance leaving the bias unchanged, it does not over-fit the data when the number of base models increase significantly, and it is easily parallelizable.

Stacking, also called *meta ensembling* or *stacked generalization*, is another well-known method used to minimize the generalization error rate by deducing biases of the generalizers [40]. It combines the outputs of multiple predictive models (level-0 base models) as training data for another model (level-1 stacked model) to approximate the same target function. The second level model learns where each base learner performs better, thereby providing improved prediction accuracy [41]. This meta-model can also be extended to 3 or more levels, with level-n models learning from level $(n-1)$. The main improvement occurs when there is a high diversity between the base model predictions [42]. Therefore, the second level model can improve the learning process when several algorithms are used to construct level-0 models because they provide different points of view of the problem.

3. Stacking ensemble model with GA-PARSIMONY

Despite the reasonable prediction capability of SVR models in the characterization of steel bolted components [10], more accurate tools are required to deal with the increasingly demanding quality requirements of structural design. Therefore, a stacking ensemble model constructed with parsimonious solutions and different algorithms has been considered in this study.

The stacking structure is depicted in Fig. 5. The basic idea is to combine parsimonious models of different types in a unique meta-learner to increase diversity and minimize the generalization error rate. First level is formed by n parsimonious models created with different algorithms: linear ridge regression, multilayer perceptron, regression tree, support vector regression, etc. In order to reduce variance and therefore smooth the generalization error rate of the base learners (level-0 models), a bagging model is created with each algorithm. Then, the level-1 model uses the predictions of base models plus the original database to reduce biases and improve accuracy. This stacking approximation is called *Restacking* because it considers not only the outputs of previous models, but also the original database as inputs. Furthermore, the search for reliably accurate parsimonious bagging models is performed with GA-PARSIMONY by optimizing feature selection (FS), hyperparameter optimization (HO) and model parsimonious selection (MPO). Therefore, a robust and reliable model is obtained for each algorithm and level.

In order to avoid leaking in the training process, the stacked generalization method with GA-PARSIMONY is based on a nested cross-validation scheme (see Fig. 6). A training dataset is divided in n outer folds. For each i outer fold, the best parsimonious model (OF-

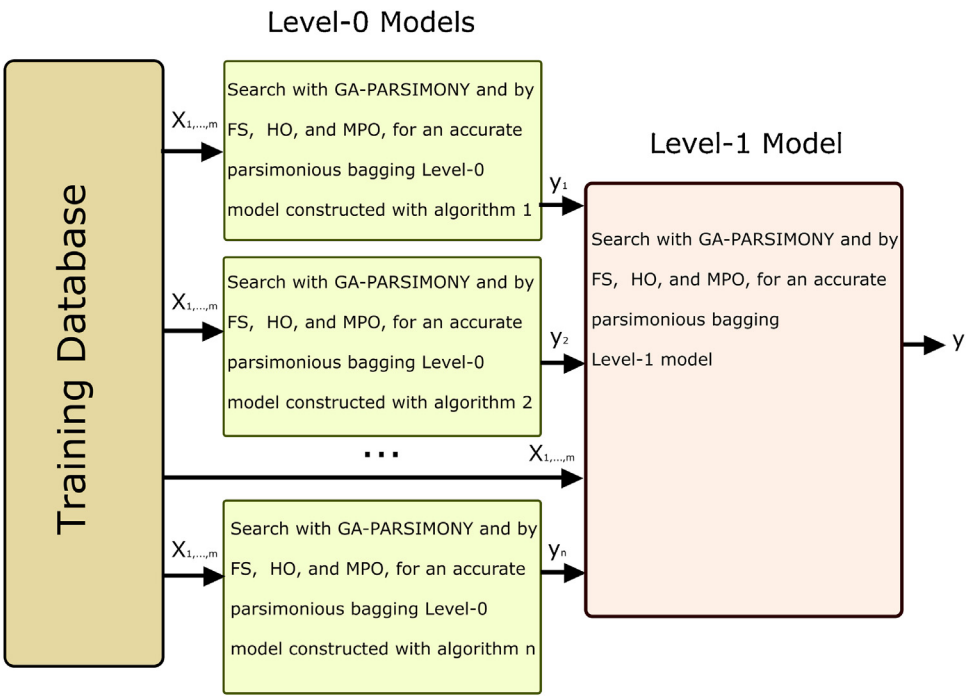


Fig. 5. Meta ensemble model constructed with n parsimonious bagging models in the first level, and a second level parsimonious model which learns from the original database and predictions of level-0 base learners. Each parsimonious model is performed with GA-PARSIMONY by optimizing FS, HO and MPO.

i model) is constructed with the other $n-1$ outer folds by using GA-PARSIMONY. The evaluation of each individual in the GA selection is performed with an inner k-fold cross-validation (CV) process. Finally, we use the best OF-i model to predict the response for the i outer fold. The response with a new dataset can also be obtained with the average of the n OF-models' predictions.

Fig. 6 shows an example of how a nested cross-validation scheme works for $n=4$ and $k=5$. In this case, the training database is divided in $n=4$ outer folds. For the $i=4$ outer fold, the GA-PARSIMONY method is used to search for the best parsimonious OF-4 model by FS, HO and PMS, and with a dataset created with the 1, 2, and 3 outer folds. Evaluation of the best individuals in the GA process is performed with this dataset and a $k=5$ inner CV. When the GA optimization process ends, the best parsimonious OF-4 model is used to predict the response for the outer 4-fold and also, for the new database. At the end of the process, the predictions for all outer folds are obtained and four predictions for the new database that can be averaged into a single response.

The process can be carried out with each algorithm (ANN, CUBIST, SVR, GBM, Ridge Linear Regression, etc.) but the following settings must be defined before using GA-PARSIMONY:

- α, the maximum absolute difference between two model costs (J) considered similar.
- A complexity function which measures the model parsimony.
- The ranges of the parameters that need to be tuned.
- GA settings: number of individuals per population, maximum of generations, percentage of elitism and mutation, etc.

Once the best parsimonious models have been created, the second step is to build the stacking ensemble model (Fig. 7). Our proposal creates a new training database by combining the original one and the response predictions with each OF-i model and for each i outer fold. Furthermore, the new database is constructed in the same way. Finally, a stacking ensemble model is created with this training database and using GA-PARSIMONY with a robust machine learning algorithm.

4. Experiments

4.1. T-stub component and FE model

The T-stub component comprises two t-shape profiles tied by their flanges with one or more bolt rows (Fig. 1b) [43]. The tensile load applied to the web is transferred by the flange in bending and the bolts in tension. During this process, the contact between flanges produces a prying action that increases the forces developed in the bolts. The contact area, as well as the pressure magnitude, evolves during the loading process, thereby rendering an adequate evaluation of the force–displacement response difficult. Non-linear material laws, large deformations and the existence of different failure patterns also further complicate the calculation of the T-stub component.

Numerical approaches such as the FE method constitute a reliable tool for assessing steel connections. An advanced FE model of the T-stub component, contrasted with laboratory experiments, is described in detail by the authors in [44]. The FE model includes complete stress-strain nonlinear material relationships and a refined characterization of the bolt, including threaded length, nut and washers (Fig. 8a). Additionally, the main novelty of the numerical model is the implementation of a continuum damage mechanics model to simulate the failure of the bolted connection. Thus, the force–displacement response of the T-stub can be fully characterized, from the initial stiffness up to the fracture point (Fig. 8b).

4.2. Generation of the training and test dataset with design of computer experiments

Design of Computer Experiments (DoCE) accounts for the deterministic nature of computer experiments assuming that numerical noise is negligible. For these cases, space-filling sampling techniques are appropriate because they uniformly distribute the points over the entire design space.

One of the most widely used space-filling designs is the Latin Hypercube Sampling (LHS) [45]. LHS divides each input into n

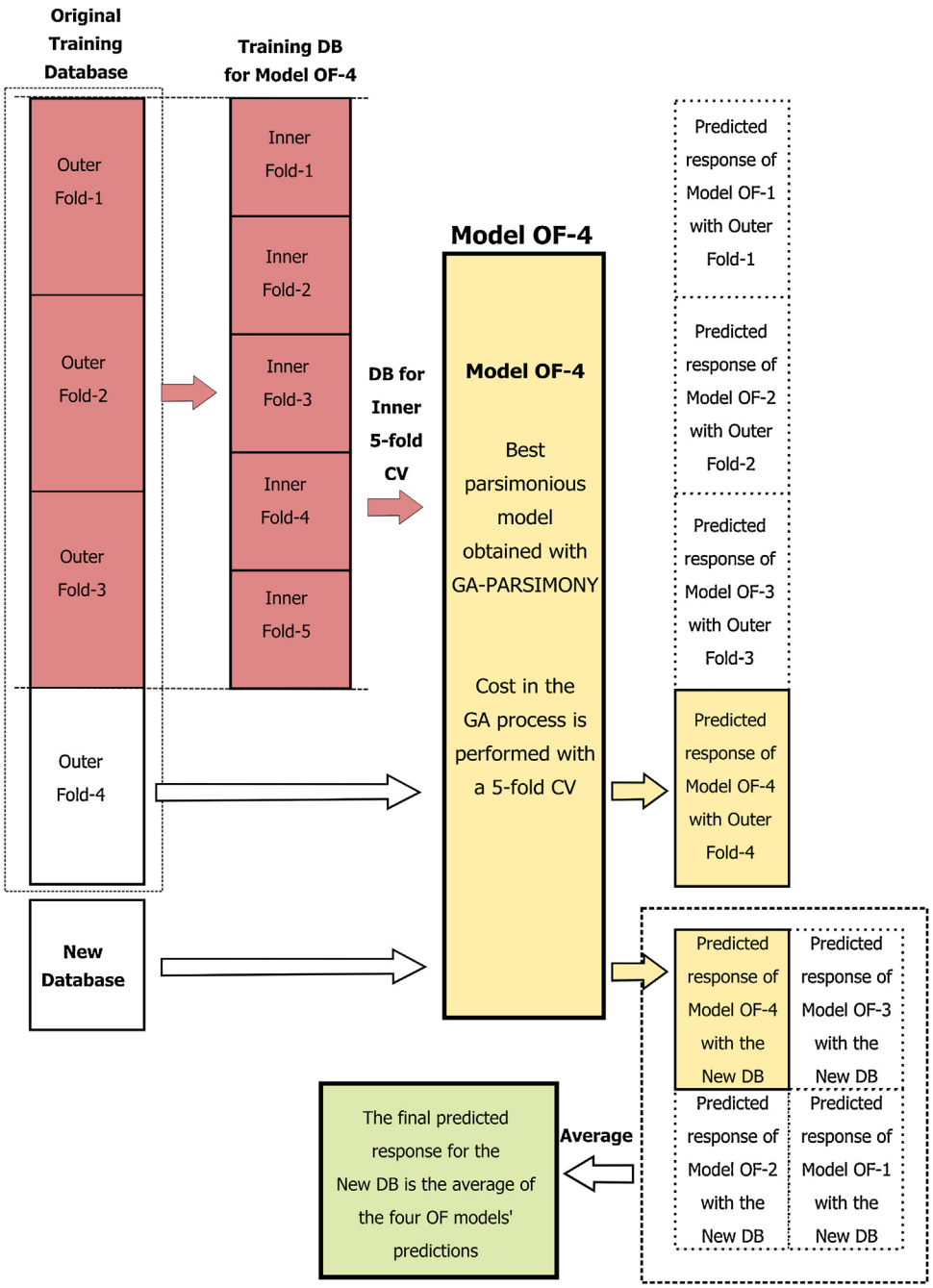


Fig. 6. Example of the search for the best parsimonious model for outer fold-4 and using GA-PARSIMONY with a nested cross-validation process with 4 outer folds and 5 inner folds.

equally probable intervals and selects a random value in each interval. The principal advantage of this method is that each input variable is represented in every division of its range [46]. These space-filling designs are especially useful in conjunction with non-parametric metamodeling techniques, as employed in this study.

The primary goal of the DoCE is to gain as much information as possible using the minimum number of sample points. However, determining sample size still remains a challenge for practitioners. This parameter relies mainly on the complexity and nonlinearity of the function to be approximated, the dimensionality of the problem, and the modeling technique selected.

To create the training dataset, the LHS method was used to define the input values of 820 FE simulations experiments. This number was experimentally determined by prior research [44]. Additionally, two test datasets composed of 76 samples were generated separately so as to check the accuracy and generalization capacity of meta-models in the prediction of unseen data; but one of them was also used to select the best α parameter. Therefore, for each combination of input values, an FE simulation was conducted to characterize the response of the T-stub component. Regarding the outputs, the performance of meta-models was evaluated according to their prediction of three key parameters of the force–displacement curve: initial stiffness (k_i), maximum strength (F_u) and displacement at failure (d_f) (Fig. 1c).

Table 1 describes the feasible ranges of geometrical parameters and mechanical properties of hot-rolled profiles and bolts used herein. A graphical description of the T-stub geometry is also included in Fig. 9.

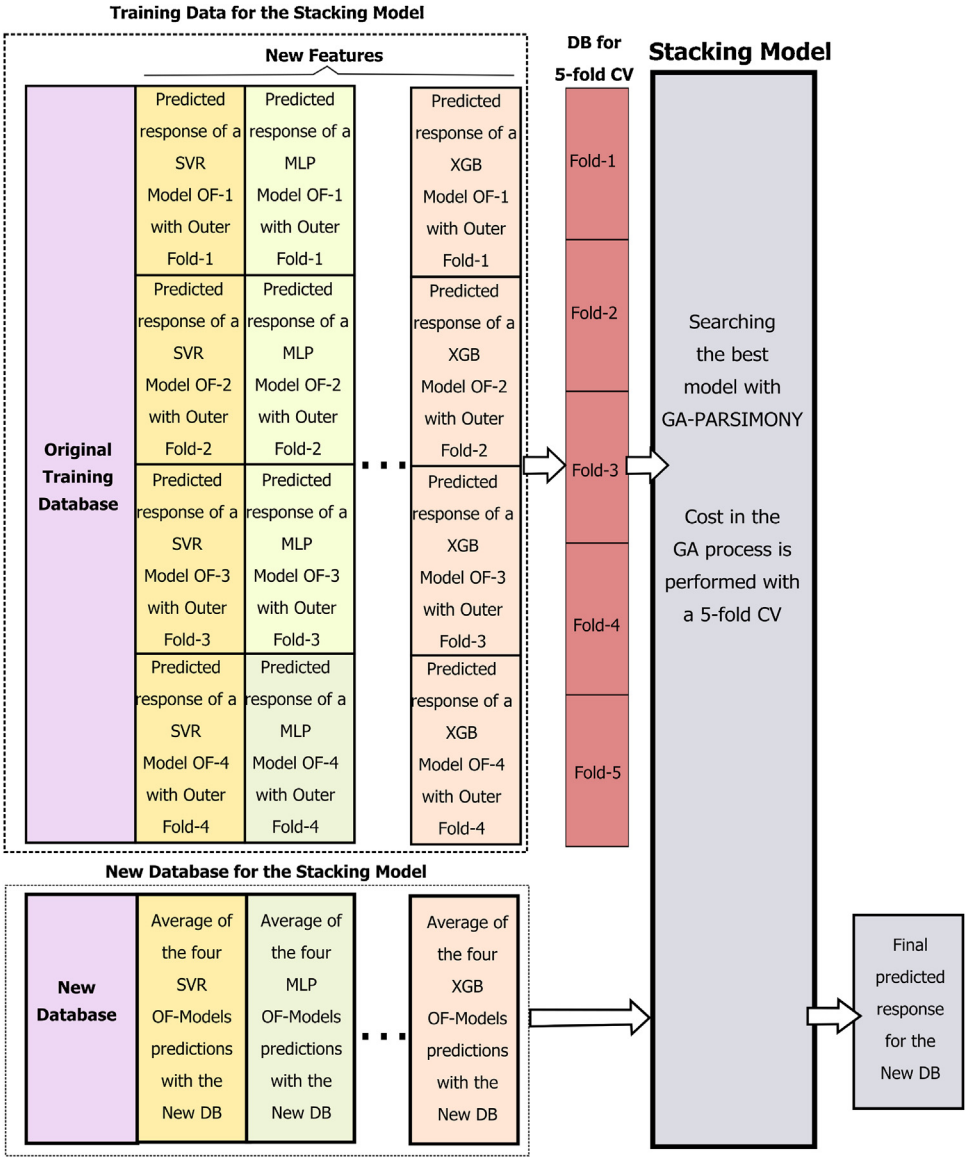


Fig. 7. Stacking ensemble model with a training dataset built with the original datasets plus the predictions of best parsimonious models created with different algorithms and using GA-PARSIMONY.

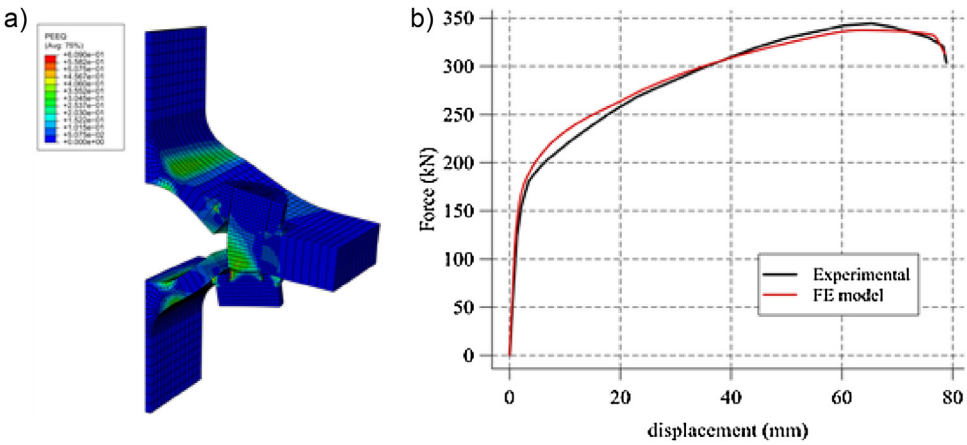


Fig. 8. Advanced FE model of the T-stub bolted component [44]. (a) FE simulation: equivalent plastic strain (PEEQ); (b) force–displacement response: FE model vs. experimental test.

Table 1
Ranges of the input features included in the DoCE.

Variable	Description [units]	Range
d_{bolt}	Nominal bolt diameter [–]	[M12–M27]
clearance	Difference between bolt hole and bolt diameter [mm]	[0.50–3.50]
t_{flange}	Flange thickness of the T-shape profile [mm]	[8.00–30.00]
t_{web}	Web thickness of the T-shape profile [mm]	[5.00–20.00]
L_{flange}	Flange length of the T-shape profile [mm]	[52.00–180.00]
r	Flange-to-web connection radius [mm]	[9.75–43.00]
n	Dist. from center of bolt hole to flange's free edge [mm]	[15.75–106.00]
b	Width of the T-shape profile [mm]	[42.00–187.00]
L_{thread}	Thread length of the bolt [mm]	[2.50–60.25]
σ_y	Yield strength of the structural steel [MPa]	[200–400]
σ_u	Stress at the maximum tensile load of the structural steel [MPa]	[300–800]
E_h	Strain-hardening coefficient of the structural steel [MPa]	[1000–3000]
σ_{yb}	Yield strength of the bolt steel [MPa]	[640–1098]
σ_{ub}	Stress at the maximum tensile load of the bolt steel [MPa]	[800–1200]
ϵ_{ub}	Strain at the maximum tensile load of the bolt steel [–]	[0.07–0.14]

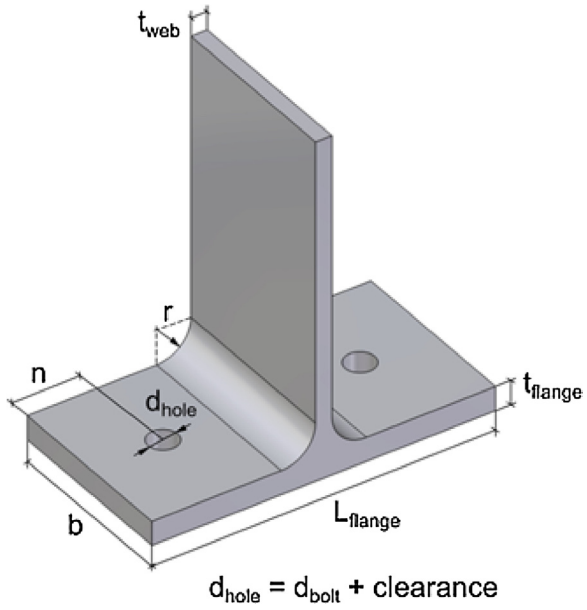


Fig. 9. T-stub geometry.

4.3. Regression techniques

Once the training and test datasets were generated, the next step was to create the most parsimonious solutions to be included in the first level of the stacking ensemble model. In this study, six popular regression algorithms were selected:

- Linear ridge regression (LIN) [47].
- Model tree algorithm (CUBIST) which is based on Quinlan's M5 model tree [48] and uses a separate-and-conquer strategy to create a tree with linear models in each leaf.
- Instance-based learning (KNN) method [49] which is a popular *k*-nearest neighbors regressor.
- Single-hidden-layer artificial neural network (ANN) [50] with *Broyden-Fletcher-Goldfarb-Shanno* (BFGS) training algorithm and weight decay.
- Support vector regression (SVR) technique [51] with radial-basis functions (RBF).
- EXtreme Gradient Boosting (XGB) machines [52] which is based on gradient boosting machines (GBM) [53].

Table 2 shows the ranges of the setting parameters included in the GA-PARSIMONY optimization process.

In addition, model complexity was defined as follows:

$$Complexity = 10^6 N_{SF} + C_{\text{mod}} \tag{8}$$

where N_{SF} is the number of input selected features and C_{mod} is the internal model complexity [54]. This expression gives priority to the N_{SF} term to penalize individuals with a higher number of input features. Internal complexity plays an important role when two models have the same number of inputs. Due to the fact that N_{SF} is weighted with a high value (10^6), C_{mod} is trunked to a value

Table 2
Range of parameters for regression techniques.

Algorithm	Parameter description	Range
LIN	*Ridge* – Regularization parameter	[0.00000001–0.99999999]
CUBIST	*committe* – Number of boosting committees	[1–20]
	neighbors – Number of neighbors in prediction	[1–9]
IBK	*k* – Number of nearest neighbours	[1–40]
	distW – Type of distance weighting	[0 = No, 1 = 1/dist 2 = 1-dist]
ANN	*n* – Number of hidden neurons	[1–25]
	decay – Weight decay	[0.0001–0.9999]
	maxit – Maximum number of iterations	1000
SVR	*C* – Error penalty coefficient	$10^{[(-3.9)-2.5]}$
	γ – Parameter of the RBF kernel	[0.000001–0.999999]
	ϵ – Insensitive loss parameter	[0.000001–0.999999]
XGB	*max_depth* – Max. depth of a tree	[1–3]
	min_child – Min. sum of instance weight in a child	[0.10–50.00]
	subsample – Subsample ratio of training instance	[0.50–1.00]
	alpha – L1 regularization term on weights	[0.00–1.00]
	num_trees – Number of trees	[10–2000]
	col_sample – Subsample ratio of columns for split	1.0
	eta – Step size shrinkage in update (learning rate)	0.05

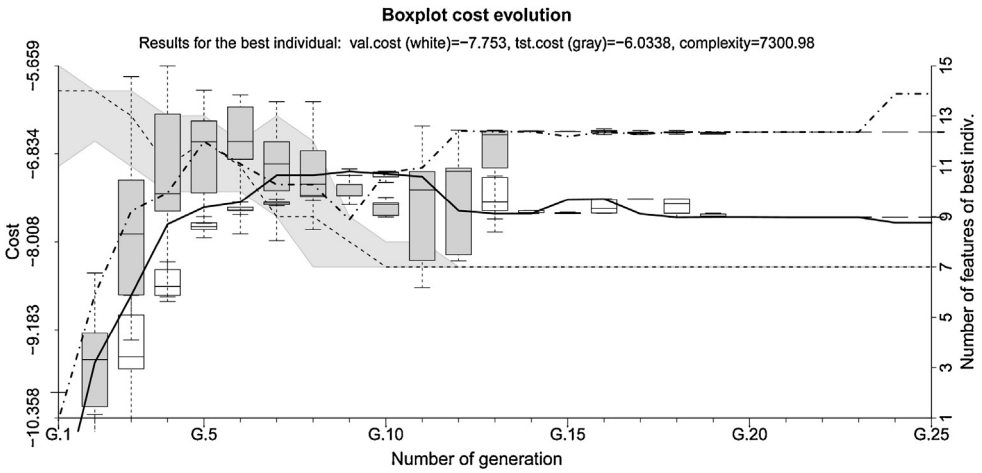


Fig. 10. Evolution of the OF-1 BaggSVR models for d_f and using GA-PARSIMONY with $\alpha = 1.0$. White and grey box-plots represent $RMSE_{val}$ and $RMSE_{tst}$ elitist evolution, respectively. Continuous and dashed-dotted lines show the $RMSE_{val}$ and $RMSE_{tst}$ of the best individual, respectively. The shaded area delimits the maximum and minimum number of N_{FS}, and the dashed line, the N_{FS} of the best individual.

of 999999 if it exceeds this upper limit. This hybrid complexity metric has demonstrated adequate behavior in PMS in previous experiments developed by the authors [11,36,55,37,38].

The following C_{mod} metrics were defined for each regression technique:

- LIN: $C_{\text{mod}} = \sum \beta_i^2$ where β_i are the coefficients of the equation.
- CUBIST: C_{mod} corresponds to the mean of the number of trees rules.
- KNN: $C_{mod} = (10^6/K) - 1$ where K is the number of nearest neighbors.
- ANN: $C_{\text{mod}} = \sum w_i^2$ where w_i are the weights of the network.
- SVR: C_{mod} corresponds to the number of support vectors.
- XGB: $C_{mod} = 10^4 max\_depth$ which depends on the tree depth.

Finally, a bagging ensemble with 10 models was considered to reduce variance as base model for each technique: BaggLIN, BaggCUBIST, BaggKNN, BaggANN, BagSVR, and BaggXGB.

4.4. GA-PARSIMONY settings

For each i outer-fold and regression technique, GA-PARSIMONY searched for the best parsimonious OF-i model by validating the individuals with an inner 5-CV Root Mean Squared Error ($J = RMSE_{val}$). In addition, several experiments were performed with different α values for each method and response variable (k_i, F_u and d_f), to promote parsimonious solutions into the PMS process.

The GA optimization process was performed with the following settings. A population size of 80 and a maximum number of generations of 100 but with an early stopping strategy when J of the best individual does not decrease in 10 generations. The PMS selection method started at the 10th generation. The mutation percentage was 10% for all individuals except the three best elite of each generation. The elitism percentage was 20%. The selection method was based on *linear-rank*. *heuristic blending* was used for parameter crossing [56] and *binary random swapping* for feature crossing.

All experiments were conducted with the statistical software `R` [57], `GAparsimony` package [58] and nine Intel 24-core servers of `Beronia` HPC cluster (Intel ® Xeon ® E5-2670 @ 2.30GHz).

Table 3
J, $RMSE_{tst}$ and *Complexity* of the best BaggSVR models for d_f response. Parsimonious models were obtained with GA-PARSIMONY and different α values for controlling the trade-off between J and *Complexity*.

α	$J = RMSE_{val}$	Complexity	$RMSE_{tst}$
0	7.699	8 750390.7	6.332
0.001	7.717	8 750353.6	6.315
0.01	7.849	8 500317.0	6.788
0.08	7.707	7 750259.6	6.884
0.1	7.736	7 750209.2	6.170
0.2	7.787	7 500183.8	5.906
0.4	7.880	7 000146.3	5.890
0.7	8.121	7 000108.7	5.783
0.8	8.160	7 000103.0	5.850
0.9	8.413	6 750128.2	6.096
1.0	**8.270**	**7 000099.5**	**5.765**
1.2	9.122	6 750047.5	6.175
1.5	10.403	5 750066.0	7.553
1.7	10.645	5 500047.2	8.396
2.0	10.774	4 750076.4	9.965
2.2	10.657	4 750086.4	10.367
2.5	11.571	4 500056.8	11.610

5. Results

Table 3 shows J, *Complexity*, and the RMSE with a new test dataset ($RMSE_{tst}$) of the best BaggSVR models for d_f and different values of α. In this table, a clear trend of increasing $RMSE_{val}$ can be observed when α grows but, inversely, model complexity decreases when α is higher. The best trade-off between J and *Complexity* is achieved with $\alpha = 1.0$, reducing the testing error to 9.83% compared to the best model with Parsimony Model Selection (PMS) disabled ($\alpha = 0$). Therefore, the model with the best generalization capability is achieved with GA-PARSIMONY and $\alpha = 1.0$. For this BaggSVR, 7 of 15 initial features are selected, with a mean of $C_{mod} = 99.5$ support vectors of all SVR models. Similar behavior can be observed with other algorithms and response variables.

Fig. 10 displays, for d_f response, the search of the best parsimonious BaggSVR model for the outer-fold 1 (OF-1 BaggSVR) with GA-PARSIMONY and $\alpha = 1.0$. This figure shows the evolution of negative values of $J = RMSE_{val}$ and $RMSE_{tst}$ (in order to represent the best solutions on the top), and the number of model features selected (N_{FS}) of elitists. In this example, with $\alpha = 1.0$, GA-PARSIMONY starts the PMS method at the 10th generation to search parsimonious solutions, in spite of J getting worse. Therefore, GA-PARSIMONY improves $RMSE_{tst}$ by reducing model complexity.

Table 4
Value of α to obtain the best GA-PARSIMONY model for each response variable and algorithm. The seventh column shows the percentage of $RMSE_{tst}$ reduction versus the error of the best model with PMS disabled ($\alpha = 0$). The last column presents the total elapsed time in a 24-core server to search the best model with GA-PARSIMONY and with the 17 α values of Table 3.

Resp.	Algorithm	α	$RMSE_{val}$	*Complexity*	$RMSE_{tst}$	$RMSE_{tst}$ reduction	Total minutes
d_f	BaggCUBIST	0.20	7.842	7 000008.2	5.799	3.59%	2605.4
d_f	BaggKNN	0.00	12.193	6 258332.3	11.458	0.00%	13.0
d_f	BaggLIN	0.20	10.438	7 500001.3	8.397	5.30%	26.8
d_f	**BaggANN**	0.20	6.494	9 500079.9	**4.971**	2.10%	1750.8
d_f	BaggSVR	1.00	8.270	7 000100.0	5.765	9.83%	543.7
d_f	BaggXGB	0.20	8.330	8 020000.1	5.989	3.45%	376.5
F_u	BaggCUBIST	0.40	17.034	9 500006.1	13.785	4.00%	2393.4
F_u	BaggKNN	0.00	48.704	6 454165.8	36.301	0.00%	16.2
F_u	BaggLIN	1.70	30.546	7 000000.2	26.362	4.82%	33.9
F_u	**BaggANN**	0.20	11.779	11 750039.1	**9.469**	8.16%	3215.3
F_u	BaggSVR	0.08	16.876	10 250314.9	13.180	1.73%	1174.7
F_u	BaggXGB	1.00	21.205	8 767500.0	16.566	2.26%	692.9
k_i	BaggCUBIST	0.08	11.279	9 500011.2	10.968	1.80%	3396.6
k_i	BaggKNN	0.70	39.641	4 927380.0	34.533	0.32%	14.4
k_i	BaggLIN	0.10	25.805	9 250001.0	22.582	0.22%	33.9
k_i	**BaggANN**	0.10	8.243	11 000039.4	**7.822**	1.46%	3034.1
k_i	BaggSVR	0.01	9.603	9 750348.1	8.154	1.21%	1814.3
k_i	BaggXGB	0.00	17.146	7 770000.0	16.016	0.00%	929.3

Table 5
Results of the stacking model with PMS (StackANN) versus a stacking model constructed with the same methodology but with PMS disabled ($\alpha = 0$) (Stack_NOPMS). Results of the best model of the first level are also displayed (BaggANN from Table 4) and of a single SVR developed in previous research [10]. Table presents α and $RMSE_{tst}$. Additionally, *RMSE*, and the mean and standard deviation of the squared error with a new second testing dataset (*tst*2) are shown.

Resp.	Algorithm	α	$RMSE_{tst}$	$RMSE_{tst2}$	SE_{tst2}^{mean}	SE_{tst2}^{sd}
d_f	**StackANN**	0.10	**4.709**	**3.681**	**13.547**	**29.378**
d_f	$BaggANN_{L0}$	0.20	4.971	4.534	20.560	39.934
d_f	StackNO_PMS	0.00	5.237	4.166	17.358	36.555
d_f	Fernandez-Ceniceros et al. [10]		7.260			
F_u	**StackANN**	0.10	**9.031**	**8.640**	**74.649**	**125.840**
F_u	$BaggANN_{L0}$	0.20	9.469	9.192	84.493	169.375
F_u	StackNO_PMS	0.00	10.486	10.052	101.038	169.788
F_u	Fernandez-Ceniceros et al. [10]		15.970			
k_i	**StackANN**	0.08	**7.637**	**5.111**	**26.118**	**57.892**
k_i	$BaggANN_{L0}$	0.10	7.822	5.204	27.087	69.772
k_i	StackNO_PMS	0.00	7.823	5.993	35.922	81.273
k_i	Fernandez-Ceniceros et al. [10]		11.750			

Table 4 shows the α value of the bagging model with the best generalization capability (best $RMSE_{tst}$) for each algorithm and response variable. $J = RMSE_{val}$, $RMSE_{tst}$, and *Complexity* are also presented. The 7th column corresponds to the relative percentage reduction of $RMSE_{tst}$ versus the model selected with PMS of GA-PARSIMONY disabled ($\alpha = 0$). Finally, the last column presents the total minutes, in a 24-core server, used to search for the best model with GA-PARSIMONY and with the 17 α values of Table 3.

For the three response variables, BaggANN obtains the best solutions as compared to the other algorithms. In addition, BaggSVR, BaggCUBIST and BaggXGB present accurate solutions for d_f. However, the computational effort necessary is much greater with these algorithms than compared to the other methods. Moreover, an important improvement in the generalization capability is achieved with PMS (more than 5% RMSEtst reduction) in BaggLIN and BaggSVR d_f models, and in BaggLIN and BaggANN F_u models. For k_i, BaggANN and BaggSVR obtain the best accuracy, but PMS improves these results slightly.

Table 5 presents the final results of the stacking model (StackANN), which was trained as explained in Section 3. The training database is formed by combining the outer-folds' responses from the models of Table 4 and the original training dataset. The last three columns correspond to the errors with a new testing database (*tst*2). Additionally, the $RMSE_{tst2}$, mean (SE_{tst2}^{mean}), and standard deviation (SE_{tst2}^{sd}) of the squared error with this new dataset are included. The objective is to check the models' generalization capability with a new database that has not been used to select the best α.

StackANN corresponds to a stacking parsimonious Bagging ANN model which was built with the original database and the predictions of the best base models from level-0, as explained in Section 3. The second line corresponds to the results of the best bagging ANN level-0 model ($BaggANN_{L0}$) of Table 4). The third line shows errors with the stacking bagging ANN model (StackNO_PMS) constructed with the same methodology, but where all base and second level models were obtained with PMS of GA-PARSIMONY disabled ($\alpha = 0$). Finally, results of a previous study [10] are presented.

The three StackANN models significantly improve errors in both testing databases versus StackNO_PMS models. This proves that incorporating PMS into the GA-PARSIMONY process helps to obtain parsimonious models with better generalization capabilities. Moreover, the stacking method of StackANN improves results for the first level model ($BaggANN_{L0}$) by increasing diversity and reducing the generalization error rate. Finally, it can be observed that SE_{tst2}^{sd} is reduced with StackANN for the three response variables. Therefore, the proposed methodology creates more accurate, robust, and reliable models thanks to the combination of parsimonious base models in the first level.

Fig. 11 shows error of StackANN model vs the three measured responses from test database. *df* and *Ki* have a normal distribution

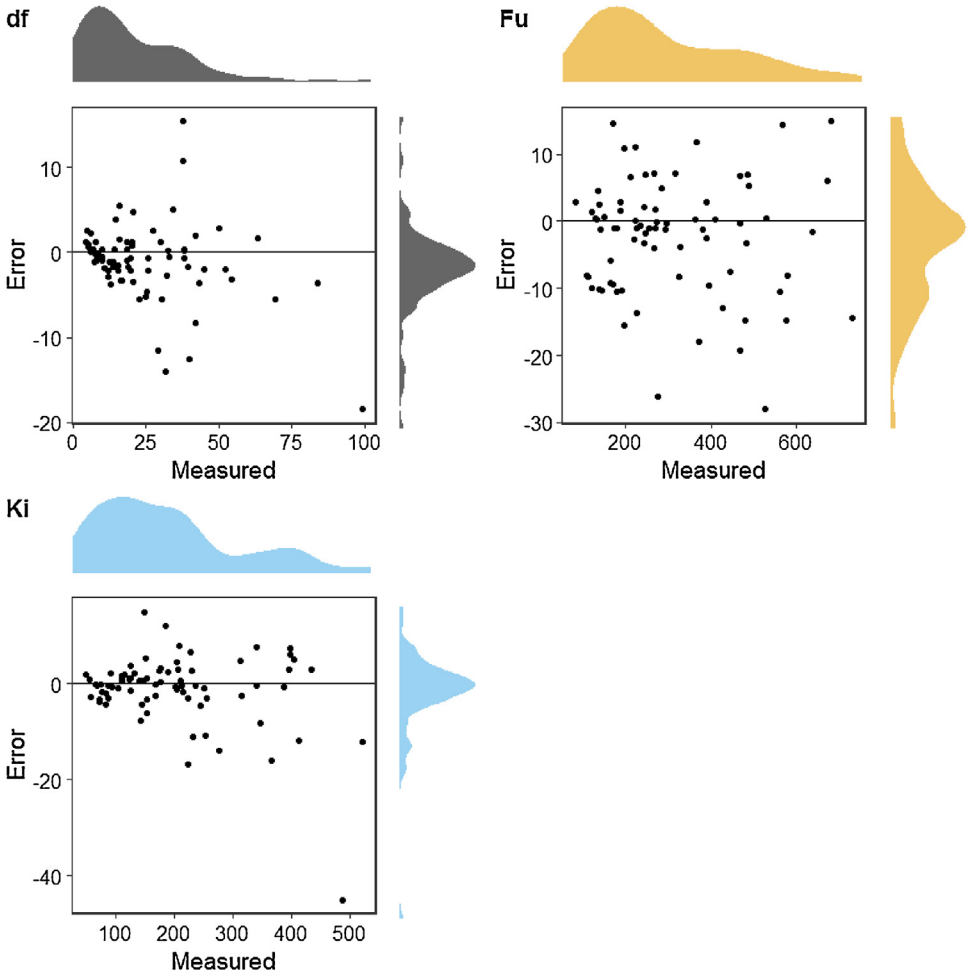


Fig. 11. Scatterplot of the errors obtained with StackANN in the testing database.

with the mean close to 0. Finally, Fig. 12 represents relative error in percentage that it is defined by:

$$Re_i = 100\frac{\hat{y}_i - y_i}{y_i} \tag{9}$$

This figure highlights the prediction capability for the three output variables. These relative errors represent useful information for structure designers about the applicability of the proposed model. Regarding the prediction of initial stiffness k_i and maximum strength F_u, most of the cases were predicted within a scatter band of ±5%. These results are very accurate compared with previous models [10] and represent a great improvement over analytical models included in current regulatory codes, such as Eurocode-3.

As for the prediction of the displacement at failure d_f, it presents higher relative errors, most of them within a scatter band of ±25%. This lower prediction capability compared to the other two variables is explained by the intrinsic randomness of the damage process. The variable d_f implicitly accounts for the degradation mechanisms which lead to the failure of the bolted connection. These mechanisms include very non-linear effects such as large deformations, necking, nucleation and coalescence of voids, and can only be predicted by means of advanced FE simulations. Taking all of this into consideration, the obtained relative errors for this variable are reasonably accurate. In fact, none of the existing standards and regulatory codes are able to predict displacement at fracture, despite the importance of this variable from the point of view of design and safety.

Overall, the proposed model represents a powerful tool capable of making real-time predictions with higher accuracy than current regulatory codes. This is only possible through the combination of advanced numerical methods (FEM) of modeling complex phenomena with SC techniques to alleviate the computation burden, as used in this work.

6. Conclusions

Current software applications based on machine learning models demand robust and accurate predictions from unseen new data. Predictions of such quality are mandatory in steel structure design from the point of view of safety. In particular, in the assessment of steel bolted connections, the challenge is to create models capable of estimating the force–displacement responses with great precision and reliability.

Today, FEM is one of the most accurate methods to predict behavior of steel bolted components but, it is a extremely time-consuming process and hence inefficient for designing the hundred of joints that comprise a steel structure. Thus, the main objective of this research was to create a robust and accurate meta-model that could be used in real-time in structural analysis and software design. The model learned from a database created with FEM simulations that was developed with a Design of Experiments (DoCE) in order to obtain a homogeneous distribution of actual design solutions. Therefore, this article presents a new methodology for building robust and reliable models of three key parameters of the force–displacement curve: maximum resistance, initial stiffness

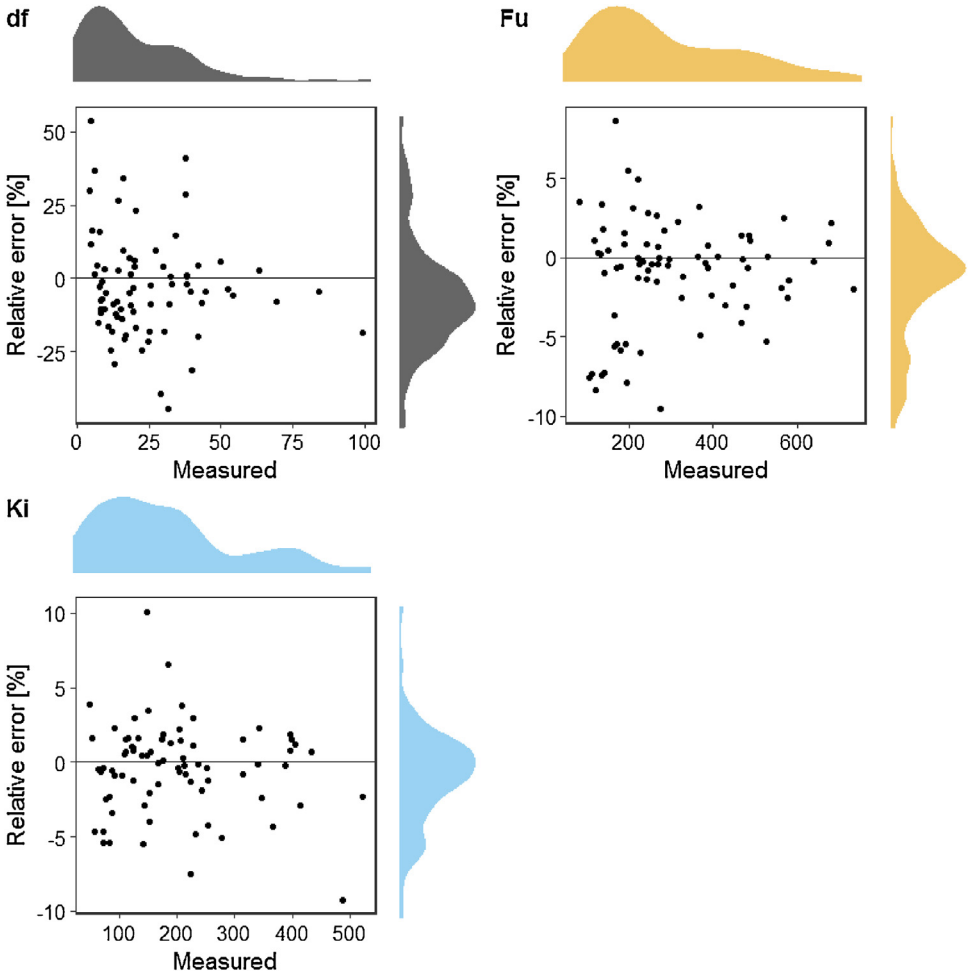


Fig. 12. Scatterplot of the relative errors obtained with StackANN in the testing database.

and displacement at failure. The main objective of this study was to develop a robust stacking ensemble model with parsimonious base models created with different machine learning methods.

For this purpose, GA-PARSIMONY, a GA soft computing methodology based on feature selection, parameter tuning and parsimonious model selection was employed. GA-PARSIMONY searched for robust and reliable bagging models with only one tuning parameter, α, which controls the trade-off between accuracy and model flexibility. Finally, in order to achieve greater generalization capability, the best parsimonious models obtained with different algorithms were combined in a stacking ensemble model by means of a nested cross validation process.

Experiments performed with a small database created with FE simulations demonstrated that the combination of parsimonious models in a stacking ensemble model improved accuracy with unseen data, against classical searching methods based on feature selection and parameter tuning.

Although, it seems that Stacking with PMS method included in GA-PARSIMONY could improve generalization capability in applications where models must be built from small databases, other experiments are necessary to corroborate this conclusion.

Finally, this accurate tool could assist designers in optimizing steel structures. This would translate into smaller quantities of steel necessary, lighter structures, economic savings, and less embodied CO2.

Acknowledgements

We are greatly indebted to *Banco Santander* for the APPI16/05, APPI17/04 and REGI2018/43 fellowships, and to the University of La Rioja for the EGI16/19 fellowship. This work used the Beronia cluster (Universidad de La Rioja), which is supported by FEDER-MINECO grant number UNLR-094E-2C-225.

References

[1] K.M. Abdalla, Review and Classification of Semi-Rigid Connections, Springer Netherlands, Dordrecht, 2000, pp. 169–178.
[2] C. Díaz, P. Martí, M. Victoria, O.M. Querin, Review on the modelling of joint behaviour in steel frames, J. Construct. Steel Res. 67 (5) (2011) 741–758.
[3] 3 Eurocode, Design of Steel Structures, Part 1–8: Design of Joints. EN 1993-1-8:2005, European Committee for Standardisation (CEN), 2005.
[4] C. Faella, V. Piluso, G. Rizzano, Structural Steel Semirigid Connections: Theory, Design, and Software, New Directions in Civil Engineering, Taylor & Francis, 1999.
[5] G.G. Wang, S. Shan, Review of metamodeling techniques in support of engineering design optimization, J. Mech. Des. 129 (4) (2007) 370–380.
[6] M. Meckesheimer, A.J. Booker, R.R. Barton, T.W. Simpson, Computationally inexpensive metamodel assessment strategies, AIAA J. 40 (2002) 2053–2060.
[7] M. Shahin, M. Elchalakani, Neural networks for modelling ultimate pure bending of steel circular tubes, J. Construct. Steel Res. 64 (6) (2008) 624–633.
[8] C. Díaz, M. Victoria, O.M. Querin, P. Martí, Optimum design of semi-rigid connections using metamodels, J. Construct. Steel Res. 78 (2012) 97–106.
[9] J. Fernandez, A. Pernia, F.J. Martinez-de-Pison, R. Lostado, Prediction models for calculating bolted connections using data mining techniques and the finite element method, Eng. Struct. 32 (10) (2010) 3018–3027.
[10] J. Fernandez-Ceniceros, A. Sanz-Garcia, F. Antonanzas-Torres, F.J. Martinez-de-Pison, A numerical-informational approach for characterising

the ductile behaviour of the t-stub component. Part 2: Parsimonious soft-computing-based metamodel, Eng. Struct. 82 (2015) 249–260.
[11] A. Sanz-Garcia, J. Fernandez-Ceniceros, F. Antonanzas-Torres, A. Pernia-Espinoza, F.J. Martinez-de Pison, GA-PARSIMONY: a GA-SVR approach with feature selection and parameter optimization to obtain parsimonious solutions for predicting temperature settings in a continuous annealing furnace, Appl. Soft Comput. 35 (2015) 13–28.
[12] S. Urolagin, P.K.V, N.V.S. Reddy, Generalization capability of artificial neural network incorporated with pruning method, in: P.S. Thilagam, A.R. Pais, K. Chandrasekaran, N. Balakrishnan (Eds.), Advanced Computing, Networking and Security, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 171–178.
[13] G. Seni, J. Elder, Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions, Morgan and Claypool Publishers, 2010.
[14] Y. Lee, J.-S. Lee, S.-Y. Lee, C.H. Park, Improving generalization capability of neural networks based on simulated annealing, 2007 IEEE Congress on Evolutionary Computation (2007) 3447–3453, http://dx.doi.org/10.1109/CEC.2007.4424918.
[15] V.N. Vapnik, A.Y. Chervonenkis, On the uniform convergence of relative frequencies of events to their probabilities, Theory Probab. Appl. 16 (2) (1971) 264–280.
[16] J. Ye, On measuring and correcting the effects of data mining and model selection, J. Am. Stat. Assoc. 93 (441) (1998) 120–131.
[17] H. Akaike, A new look at the statistical model identification, IEEE Trans. Autom. Control 19 (6) (1974) 716–723, http://dx.doi.org/10.1109/TAC.1974.1100705.
[18] G. Schwarz, Estimating the dimension of a model, Ann. Stat. 6 (2) (1978) 461–464, http://dx.doi.org/10.1214/aos/1176344136.
[19] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning: Data Mining, Inference and Prediction, 2nd ed., Springer, 2009.
[20] B. Ma, Y. Xia, A tribe competition-based genetic algorithm for feature selection in pattern classification, Appl. Soft Comput. 58 (2017) 328–338.
[21] J. Wei, R. Zhang, Z. Yu, R. Hu, J. Tang, C. Gui, Y. Yuan, A BPSO-SVM algorithm based on memory renewal and enhanced mutation mechanisms for feature selection, Appl. Soft Comput. 58 (2017) 176–192.
[22] J. Pérez-Rodríguez, A.G. Arroyo-Peña, N. García-Pedrajas, Simultaneous instance and feature selection and weighting using evolutionary computation: proposal and study, Appl. Soft Comput. 37 (2015) 416–443.
[23] H.-L. Huang, F.-L. Chang, ESVM: evolutionary support vector machine for automatic feature selection and classification of microarray data, Biosystems 90 (2) (2007) 516–528.
[24] S. Ding, Spectral and wavelet-based feature selection with particle swarm optimization for hyperspectral classification, J. Softw. 6 (7) (2011) 1248–1256.
[25] S.M. Vieira, L.F. Mendonza, G.J. Farinha, J.M. Sousa, Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients, Appl. Soft Comput. 13 (8) (2013) 3494–3504.
[26] C.-L. Huang, J.-F. Dun, A distributed PSO-SVM hybrid system with feature selection and parameter optimization, Appl. Soft Comput. 8 (4) (2008) 1381–1391.
[27] Y. Wan, M. Wang, Z. Ye, X. Lai, A feature selection method based on modified binary coded ant colony optimization algorithm, Appl. Soft Comput. 49 (2016) 248–258.
[28] R. Ahila, V. Sadasivam, K. Manimala, An integrated {PSO} for parameter determination and feature selection of {ELM} and its application in classification of power system disturbances, Appl. Soft Comput. 32 (2015) 23–37.
[29] R. Dhiman, J. Saini, Priyanka, Genetic algorithms tuned expert model for detection of epileptic seizures from EEG signatures, Appl. Soft Comput. 19 (2014) 8–17.
[30] S.M. Winkler, M. Affenzeller, G. Kronberger, M. Kommenda, S. Wagner, W. Jacak, H. Stekel, Analysis of selected evolutionary algorithms in feature selection and parameter optimization for data based tumor marker modeling, in: R.R.Z. Moreno-Diaz, F. Pichler, A. Quesada-Arencibia (Eds.), EUROCAST (1), Vol. 6927 of Lecture Notes in Computer Science, Springer, 2011, pp. 335–342.
[31] M. Wang, H. Chen, B. Yang, X. Zhao, L. Hu, Z. Cai, H. Huang, C. Tong, Toward an optimal kernel extreme learning machine using a chaotic moth-flame optimization strategy with applications in medical diagnoses, Neurocomputing 267 (2017) 69–84, http://dx.doi.org/10.1016/j.neucom.2017.04.060 http://www.sciencedirect.com/science/article/pii/S0925231217308305.
[32] S. Medjahed, T.A. Saadi, A. Benyettou, M. Ouali, Gray wolf optimizer for hyperspectral band selection, Appl. Soft Comput. 40 (2016) 178–186.
[33] N. Chen, B. Ribeiro, A. Vieira, J. Duarte, J.C. Neves, A genetic algorithm-based approach to cost-sensitive bankruptcy prediction, Expert Syst. Appl. 38 (10) (2011) 12939–12945.
[34] M. Avalos, Y. Grandvalet, C. Ambroise, Parsimonious additive models, Comput. Stat. Data Anal. 51 (6) (2007) 2851–2870, http://dx.doi.org/10.1016/j.csda.2006.10.007 http://www.sciencedirect.com/science/article/pii/S0167947306003732.
[35] C.A. Escobar, R. Morales-Menendez, Process-monitoring-for-quality – a model selection criterion, Manuf. Lett. 15 (2018) 55–58, http://dx.doi.org/10.1016/j.mfglet.2018.01.001 http://www.sciencedirect.com/science/article/pii/S221384631830004X.
[36] F. Antonanzas-Torres, R. Urraca, J. Antonanzas, J. Fernandez-Ceniceros, F.J. Martinez-de-Pison, Generation of daily global solar irradiation with support vector machines for regression, Energy Convers. Manage. 96 (2015) 277–286.
[37] A. Sanz-García, J. Fernández-Ceniceros, F. Anto nanzas-Torres, F.J. Martínez-de Pisón, Parsimonious support vector machines modelling for set points in industrial processes based on genetic algorithm optimization, in: International Joint Conference SOCO13-CISIS13-ICEUTE13, Vol. 239 of Advances in Intelligent Systems and Computing, Springer International Publishing, 2014, pp. 1–10.
[38] R. Urraca-Valle, A. Sanz-Garcia, J. Fernandez-Ceniceros, E. Sodupe, F.J. Martinez-de-Pison, Improving hotel room demand forecasting with a hybrid GA-SVR methodology based on skewed data transformation, feature selection and parsimony tuning, in: E. Onieva, I. Santos, E. Osaba, H. Quintián, E. Corchado (Eds.), Hybrid Artificial Intelligent Systems – 10th International Conference, HAIS 2015, Bilbao, Spain, June 22–24, 2015, Proceedings, Vol. 9121 of Lecture Notes in Computer Science, Springer, 2015, pp. 632–643, http://dx.doi.org/10.1007/978-3-319-19644-2.
[39] L. Breiman, Bagging predictors, Mach. Learn. 24 (2) (1996) 123–140, http://dx.doi.org/10.1023/A:1018054314350.
[40] D.H. Wolpert, Stacked generalization, Neural Netw. 5 (1992) 241–259.
[41] K.M. Ting, I.H. Witten, Issues in stacked generalization, CoRR abs/1105.5466. arXiv:1105.5466. http://arxiv.org/abs/1105.5466.
[42] Q. Dai, R. Ye, Z. Liu, Considering diversity and accuracy simultaneously for ensemble pruning, Appl. Soft Comput. 58 (2017) 75–91, http://dx.doi.org/10.1016/j.asoc.2017.04.058 http://www.sciencedirect.com/science/article/pii/S1568494617302405.
[43] A.M.G. Coelho, L.S. da Silva, F.S. Bijlaard, Characterization of the Nonlinear Behaviour of Single Bolted T-stub Connections, Connections in Steel Structures V, Amsterdam, The Netherlands, 2004.
[44] J. Fernandez-Ceniceros, A. Sanz-Garcia, F. Antoñanzas-Torres, F. Martinez-de Pison, A numerical-informational approach for characterising the ductile behaviour of the t-stub component. Part 1: Refined finite element model and test validation, Eng. Struct. 82 (2015) 236–248.
[45] M.D. McKay, R.J. Beckman, W.J. Conover, Comparison of three methods for selecting values of input variables in the analysis of output from a computer code, Technometrics 21 (2) (1979) 239–245.
[46] J. Sacks, W.J. Welch, T.J. Mitchell, H.P. Wynn, Design and analysis of computer experiments, Stat. Sci. (1989) 409–423.
[47] I.H. Witten, E. Frank, M.A. Hall, Data Mining: Practical Machine Learning Tools and Techniques, 3rd ed., Morgan Kaufmann, Amsterdam, 2011.
[48] J.R. Quinlan, Learning with continuous classes, 5th Australian Joint Conference on Artificial Intelligence (1992) 343–348.
[49] D.W. Aha, D. Kibler, Instance-based learning algorithms, Mach. Learn. (1991) 37–66.
[50] R. Menéndez de Llano, J.L. Bosque, Study of neural net training methods in parallel and distributed architectures, Future Gener. Comput. Syst. 26 (2) (2010) 267–275.
[51] H. Drucker, Chris, B.L. Kaufman, A. Smola, V. Vapnik, Support vector regression machines, Advances in Neural Information Processing Systems 9, Vol. 9 (1997) 155–161.
[52] T. Chen, T. He, M. Benesty, XGBoost: Extreme Gradient Boosting, R package version 0.4-3, 2015 https://github.com/dmlc/xgboost.
[53] J.H. Friedman, Greedy function approximation: a gradient boosting machine, Ann. Stat. 29 (5) (2001) 1189–1232.
[54] R. Urraca, E. Sodupe-Ortega, J. Antonanzas, F. Antonanzas-Torres, F.M. de Pison, Evaluation of a novel GA-based methodology for model structure selection: the GA-parsimony, Neurocomputing 271 (2018) 9–17, http://dx.doi.org/10.1016/j.neucom.2016.08.154 http://www.sciencedirect.com/science/article/pii/S0925231217312171.
[55] F.J. Martinez-de Pison, E. Fraile-Garcia, J. Ferreiro-Cabello, R. Gonzalez, A. Pernia, Searching Parsimonious Solutions with GA-PARSIMONY and XGBoost in High-Dimensional Databases, Springer International Publishing, Cham, 2017, pp. 201–210.
[56] Z. Michalewicz, C.Z. Janikow, Handling constraints in genetic algorithms, ICGA (1991) 151–157.
[57] R Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, 2013.
[58] F. Martinez-de-Pison, GAparsimony package for R, 2017 https://github.com/jpison/GAparsimony.