Focussed on Steels and Composites for Engineering Structures

Visual Abstract

Weight optimization of steel lattice transmission towers based on Differential Evolution and machine learning classification technique

Tran-Hieu Nguyen, Anh-Tuan Vu
Hanoi University of Civil Engineering, Viet Nam
hieunt2@nuce.edu.vn, https://orcid.org/0000-0002-1446-5859
tuanva@nuce.edu.vn

ABSTRACT. Transmission towers are tall structures used to support overhead power lines. They play an important role in the electrical grids. There are several types of transmission towers in which lattice towers are the most common type. Designing steel lattice transmission towers is a challenging task for structural engineers due to a large number of members. Therefore, discovering effective ways to design lattice towers has attracted the interest of researchers. This paper presents a novel method that integrates Differential Evolution (DE), a powerful optimization algorithm, and a machine learning classification model to minimize the weight of steel lattice towers. A classification model based on the Adaptive Boosting algorithm is developed in order to eliminate unpromising candidates during the optimization process. A feature handling technique is also introduced to improve the model quality. An illustrated example of a 160-bar tower is conducted to demonstrate the efficiency of the proposed method. The results show that the application of the Adaptive Boosting model saves about 40% of the structural analyses. As a result, the proposed method is 1.5 times faster than the original DE algorithm. In comparison with other algorithms, the proposed method obtains the same optimal weight with the least number of structural analyses.

KEYWORDS. Structural Optimization; Machine Learning Classification; Differential Evolution; Transmission Tower.

OPEN ACCESS

Citation: Nguyen, T.-H., Vu, A.-T., Weight optimization of steel lattice transmission towers based on Differential Evolution and machine learning classification technique, Frattura ed Integrità Strutturale, 59 (2022) 172-187.

Received: 19.08.2021
Accepted: 12.10.2021
Published: 01.01.2022

INTRODUCTION

Electricity is an essential part of modern life. Electricity is not only a big part of daily life in homes but also plays an important role in industrial production. The electrical grid consists of two components which are the electric power transmission and the electric power distribution. The transmission network carries electricity from power plants to substations, while the distribution network delivers electricity from substations to customers. Nowadays, due to the increasing demand for energy as well as the exploitation of new energy sources such as wind power, solar power, the transmission network is constantly being upgraded and expanded.

In the transmission network, the power is usually transported at high voltages through overhead power lines due to the lower installation cost comparing with the underground systems. Overhead power lines are supported by transmission towers which are often steel lattice truss structures. Designing steel lattice transmission towers is always a challenging task for structural engineers due to a large number of design variables. In addition, a typical design of a tower is applied many times in the transmission lines. Therefore, minimizing the weight of typical tower results in a significant economic effect. Because of the above reasons, the weight optimization of steel lattice transmission towers is an interesting topic for a long time. Numerous outstanding studies in this field can be listed as follows.

In 1995, Rao [1] applied the Hookes-Jeeves method to optimize the shape of a 400 kV transmission tower. The weight of the optimized tower is 12% less than the initial design. In 2004, Taniwaki and Ohkubo [2] considered the node coordinates, the cross-sectional areas, and the materials of members as design variables when optimizing the weight of transmission towers subject to simultaneously static and seismic loads. The optimization process was separated into two stages, in which the first stage aims to optimize the shape and the sections of the tower when the materials are fixed. At the second stage, the best pair of the cross-section and material for each member is identified while maintaining the shape of the tower. Shea and Smith [3] developed a new method that combines structural grammars and the Simulated Annealing (SA) algorithm for optimizing the topology and the shape of transmission towers. The weight of an existing tower in Switzerland was reduced by 16.7% after optimizing by the proposed method. In [4], four types of optimization including sizing optimization, shape optimization, topology optimization, and configuration optimization are carried out based on the adaptive Genetic Algorithm (GA). Kaveh et al. [5] presented the Multi Metaheuristic-based Search method in which several metaheuristic algorithms are simultaneously employed on subsets of the initial population in order to increase the diversity over the design space. The proposed method was applied to optimize four different steel towers. Souza et al. [6] proposed a new method to optimize the size, shape, and topology of steel towers based on the Firefly Algorithm and the Backtracking Search Algorithm. In this proposed method, the considered tower is divided into modules and the configuration of each module is selected from pre-established templates. Tort et al. [7] developed a tool that integrates the finite element analysis commercial software PLS-Tower and the SA algorithm for optimizing transmission towers. Recently, Khodzhaiev and Reuter [8] used a modified version of the GA to optimize the topology, shape, and sizes of towers.

Based on the literature, it can be seen that the meta-heuristic algorithms have been commonly used in studies related to the optimization of steel towers due to their simplicity and ease of implementation. However, one of the disadvantages of meta-heuristic algorithms is that the computing time is prohibitively long due to a huge number of structural analyses. As an illustration, in Ref. [6], 180,000 structural analyses were performed when optimizing the 115 kV transmission tower. Several studies have been carried out with the aim of reducing the number of structural analyses. For example, Couceiro et al. [9] integrated the sensitivity analysis into the SA algorithm in order to accelerate the optimization process. This technique reduced the computing time from 109,095 s to 8850 s for a real 220 kV transmission tower containing 684 elements.

In recent years, Machine Learning (ML) has been increasingly used in structural engineering. Several common types of ML tasks include regression, classification, dimensionality reduction, etc. ML regression models are frequently used to simulate complex data which are difficult to derive explicit formulas such as characteristics of a material [10,11], load-bearing capacities of a structural member [12], or a whole structure [13]. Unlike regression, the classification aims to predict a discrete class label like the failure mode of concrete columns [14] or the safety state of structures [15]. Another ML task is to reduce the complexity of the data. Some dimensionality reduction techniques can be listed: Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), and Proper Orthogonal Decomposition (POD), in which the POD technique has been widely employed in the crack identification [16] as well as the damage detection [17].

It is noted that damage detection is formulated as an optimization problem with the aim of minimizing errors between measured values and calculated values. Therefore, meta-heuristic algorithms are often applied to increase the precision of the damage detection as Slime Mould algorithm by Tiachacht et al. [18], Atom Search Optimization (ASO) by Khatir et al. [19]. Additionally, the combination of a ML model with a meta-heuristic algorithm has gained increasing attention in recent times. Some recognized works in this field are carried out by Khatir et al. [20-22], by Zenzen et al. [23].

Regarding the structural optimization of steel towers, ML has been used to construct surrogate models in order to reduce the computational cost. For instance, Kaveh et al [24] used Neural Networks (NNs) as a structural analysis tool when optimizing the shape and size of transmission towers. In 2019, Taheri, Ghasemi, and Dizangian [25] used radial basis function (RBF) neural networks to approximate the response of towers. The developed RBF networks were then combined with the Artificial Bee Colony (ABC) algorithm to shorten the computing time. Similarly, Hosseini, Ghasemi, and Dizangian [26] introduced the BBO-ANFIS optimization method in which the Adaptive Neuro-Fuzzy Inference System (ANFIS) models were embedded to the Biogeography Based Optimization (BBO) algorithm to save the optimization time. Although the application of surrogate modeling is very time-efficient, the designs found by the surrogate-assisted methods are not the optimal solutions due to the approximation of the structural behavior [25,26].

This paper aims to present a hybrid method for the sizing optimization of transmission towers. Different from previous studies, this study uses a ML classification technique called Adaptive Boosting to evaluate the safety state of transmission towers. The developed model is then integrated into the Differential Evolution algorithm to discard the unnecessary structural analyses. The proposed method is used to minimize the weight of a 160-bar tower. The obtained designs are then compared with previous designs in the literature to illustrate the effectiveness of the proposed method.

FORMULATION OF THE WEIGHT OPTIMIZATION PROBLEM OF TRANSMISSION TOWERS

In general, the overall cost of a transmission tower consists of several components such as the material cost, the manufacturing cost, the transportation cost, the erection cost, as well as the maintenance cost. However, there exists a linear relationship between the overall cost and the weight of the structure. Therefore, the weight of structural members is frequently used as the objective function of optimization problems. Because steel transmission towers are not constrained by architectural and aesthetic requirements, optimizing their shape and topology has been received great attention from researchers. However, there is a fact that the complex configurations obtained from the shape and topology optimization are difficult to apply in practice. Hence, only cross-sectional areas of tower members are considered as design variables in the present work.
The weight optimization of transmission towers is formulated as follows:

$$\begin{aligned} &\text{find} && \mathbf{A} = \{A_i \mid i = 1,2,...,ng\} \\ &\text{to minimize} && W(\mathbf{A}) = \rho \sum_{i=1}^{ng} A_i \sum_{j=1}^{nm(i)} L_j \\ &\text{subject to} && g_k(\mathbf{A}) \le 0 \mid k = 1,2,...,nc \\ & && A_i \in \mathbf{S} = \{S_1, S_2, ..., S_d\} \end{aligned} \tag{1}$$

where: **A** represents the vector containing ng design variables; A_i is a design variable which denotes the cross-sectional area of members of the i-th group; ng is the number of member groups; W(**A**) is the weight of the tower; ρ is the unit weight of steel; $nm(i)$ is the number of members belong to the i-th group; L_j represents the length of j-th member; g_k(**A**) is the k-th constraint; nc is the number of constraints; A_i is selected from the set **S** containing d profiles.
Design constraints of transmission towers include stress, slenderness, buckling, displacement, as well as natural frequency limitations. The formula of each constraint condition depends on the design specification used in the project. The most widely used specification for steel lattice towers is ASCE 10-97 [27]. In addition, there are also geometric requirements in which the lower leg members must be larger than the upper ones.
To apply meta-heuristic algorithms for solving this problem, some techniques should be employed. Firstly, the penalty function method is often used for dealing with constraints. In more detail, the objective function is modified by adding a term to penalize when there is any constraint violation:

$$Fit(\mathbf{A}) = W(\mathbf{A}) + PF \times \left\{1 + \max\left[\max_j\left(0, g_j(\mathbf{A})\right)\right]\right\} \tag{2}$$

in which: Fit(**A**) is called the fitness function; W(**A**) is the objective function that is the weight of the tower in this case; PF is the penalty factor.
In this work, PF is a very large value. By this setting, the selection between two candidates complies with the following rules:

- Between two feasible candidates, the one having the lower value of the objective function is selected,
- Between two infeasible candidates, the one having the lower degree of constraint violation is selected,
- Between a feasible candidate and an infeasible candidate, the feasible one is selected.

Besides, for handling discrete optimization problems, the design variable A_i is replaced by the sequence number I_i representing the position of A_i in the list **S**.

THE PROPOSED METHOD

The proposed method combines Differential Evolution (DE), a powerful optimization algorithm, and the Adaptive Boosting classification technique. In the following sections, the DE algorithm and the AdaBoost technique are briefly presented. Next, the proposed method is introduced.

Differential Evolution algorithm

The DE algorithm was developed by Price and Storn [28]. Basically, the DE algorithm consists of four basic operators: initialization, mutation, crossover, and selection. There exist many mutation strategies such as "rand/1", "rand/2", "best/1", "best/2", etc. In the present work, the variant "target-to-best/1" of the DE algorithm is employed. First of all, the DE generates an initial population of NP individuals $\{\mathbf{x}_k^{(0)} \mid k=1,2,...,NP\}$ using the following operator:

$$x_{k,i}^{(0)} = l_i + \text{rand}[0,1](u_i - l_i) \tag{3}$$

where: $\mathbf{x}_k^{(0)}$ is a D-dimensional vector representing a solution of the optimization problem; $x_{k,i}^{(0)}$ is the i-th component of $\mathbf{x}_k^{(0)}$; rand[0,1] is a uniformly distributed random real value in the range [0,1]; l_i and u_i are the min. and max. bounds of $x_{k,i}$, respectively.

At the generation t, a trial vector $\mathbf{u}_k^{(t)}$ which is different from the target vector $\mathbf{x}_k^{(t)}$ is produced using two operators of mutation:

$$\mathbf{v}_k^{(t)} = \mathbf{x}_k^{(t)} + F \times \left(\mathbf{x}_{best}^{(t)} - \mathbf{x}_k^{(t)}\right) + F \times \left(\mathbf{x}_{r1}^{(t)} - \mathbf{x}_{r2}^{(t)}\right) \tag{4}$$

and crossover:

$$u_{k,i}^{(t)} = \begin{cases} v_{k,i}^{(t)} & \text{if } i = K \text{ or } \text{rand}[0,1] \le Cr \\ x_{k,i}^{(t)} & \text{otherwise} \end{cases} \tag{5}$$

where: $r1 \neq r2 \neq k$ are randomly selected between 1 and NP; F is the scaling factor; $\mathbf{x}_{best}^{(t)}$ is the best individual of the t-th population; $u_{k,i}^{(t)}$, $v_{k,i}^{(t)}$, and $x_{k,i}^{(t)}$ are the i-th component of $\mathbf{u}_k^{(t)}$, $\mathbf{v}_k^{(t)}$, $\mathbf{x}_k^{(t)}$, respectively; K is a randomly chosen integer in the interval [1,D]; and Cr is the crossover rate.

The k-th individual of the (t+1) generation $\mathbf{x}_k^{(t+1)}$ is selected using the operator selection as follows:

$$\mathbf{x}_k^{(t+1)} = \begin{cases} \mathbf{v}_k^{(t)} \text{ if } Fit\left(\mathbf{u}_k^{(t)}\right) \le Fit\left(\mathbf{x}_k^{(t)}\right) \\ \mathbf{x}_k^{(t)} \text{ otherwise} \end{cases} \tag{6}$$

in which: $Fit(.)$ is the fitness function.

The final optimal solution is obtained after repeatedly performing three operators of mutation, crossover, and selection for (max_iter-1) times.

Adaptive Boosting

The ensemble method Adaptive Boosting (a.k.a AdaBoost) was developed by Freund and Schapire in 1997 [29]. Despite this method can handle both regression tasks as well as multi-label classification tasks, this study only employs the AdaBoost binary classifier. The idea behind this method is to create a strong classifier based on multiple weak classifiers that are sequentially trained on the weighted dataset as clearly illustrated in Fig. 1. At the first step, every sample of the training data is weighted a uniform value and a classifier is trained by the whole data. Then, the samples that are wrongly predicted by the first classifier are re-weighted with a higher value. It helps these samples get the attention of the next classifier. This process is repeatedly carried out to obtain a number of classifiers. Each classifier is also assigned a weight based on its accuracy. The final prediction is made by the weighted majority voting technique. The implementation process of the AdaBoost binary classifier is shortly presented as follows.

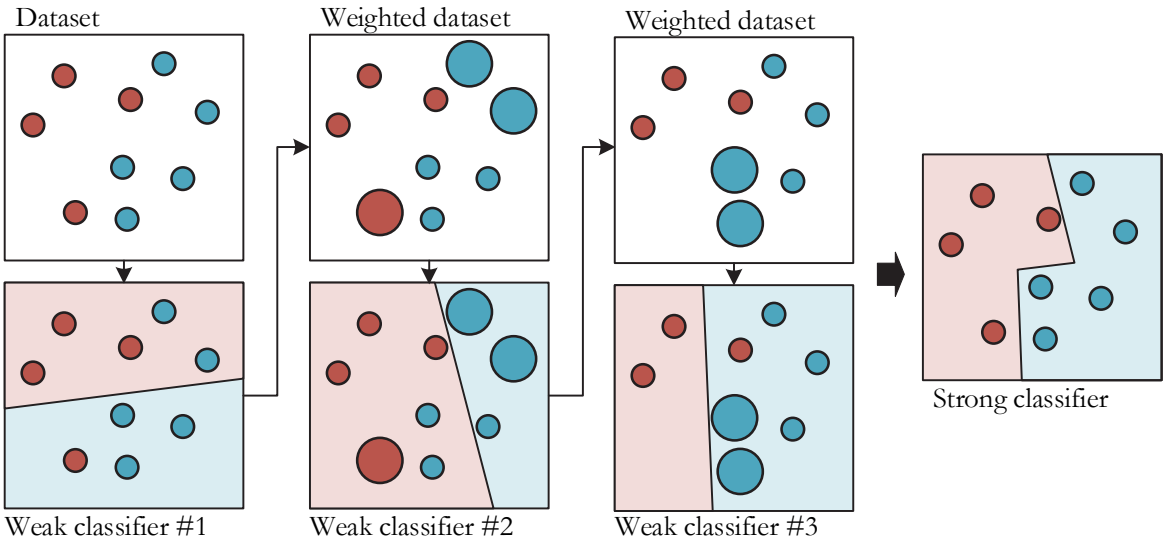

Figure 1: Illustration of the AdaBoost.

Assume that the training dataset contains N samples $\{(\mathbf{x}_i, y_i) \mid i=1,\ldots,N\}$ where $\mathbf{x}_i$ is a vector that comprises input features, $y_i \in \{+1,-1\}$ is the output feature. Firstly, a set of the same weight $\mathbf{D}_1=\{w_{1,i}=1/N \mid i=1,\ldots,N\}$ is assigned to all samples. A weak classifier C_1 is then trained by the weighted dataset. Secondly, the error rate of the trained classifier C_1 is determined using the following formula:

$$err_1 = \sum_{i=1}^{N} w_{1,i} I\left(C_1\left(\mathbf{x}_i\right) \neq y_i\right) \tag{7}$$

where: $I(.)$ is a function that returns 0 if the prediction is correct while 1 if the prediction is wrong.
Third, the weight α_1 of the weak classifier C_1 is computed from its error rate:

$$\alpha_1 = \frac{1}{2}\ln\left(\frac{1-err_1}{err_1}\right) \tag{8}$$

Next, the set of weights is adjusted to $\mathbf{D}_2=\{w_{2,i} \mid i=1,\ldots,N\}$ for the second iteration, in which the weight $w_{2,i}$ is calculated as follows:

$$w_{2,i} = \frac{w_{1,i}\exp\left(-\alpha_1 y_i C_1\left(\mathbf{x}_i\right)\right)}{\sum_{i=1}^{N} w_{2,i}} \tag{9}$$

The weak classifier C_2 is then trained by the new weighted dataset. After T iterations, there are totally T weak classifiers that are sequentially trained. For new sample $\mathbf{x}_{new}$, the label y_{new} is estimated by taking the weighted sum of the predictions of these weak classifiers:

$$y_{new} = \text{sign}\left(\sum_{t=1}^{T} \alpha_t C_t\left(\mathbf{x}_{new}\right)\right) \tag{10}$$

where: sign(.) is a function that extracts the sign of a real number.
The weak classifier used in the AdaBoost can be any machine learning classification algorithm, but the Decision Tree is commonly used. The background theory of the Decision Tree is not presented here since it is out of the scope of this article.

AdaBoost classification-assisted Differential Evolution

The idea to use the machine learning classification technique to distinguish and discard unpromising candidates has been presented in Ref. [30]. In the previous study, a deep NN model is developed to predict the safety state of structures. The optimization process is separated into two stages. The first stage employs the original DE algorithm with the aim of exploring the design space. In the second stage, each trial vector produced by the mutation and crossover operators is preliminarily assessed by the neural network model. If it is predicted to violate any constraint and its objective function is larger than that of the target vector, the trial vector is eliminated without conducting the exact fitness evaluation. Using the NN model during the optimization process can reduce unnecessary fitness evaluations. The numerical example of a planar 47-bar tower conducting in Ref. [30] shows that the reduction rate reaches 20%.

However, it can be observed that the fitness evaluations performed in the first stage can be used as the training data of the classification model. Secondly, the neural network model used in Ref. [30] has low accuracy. Therefore, in the present work, some modifications are proposed to enhance the optimization process:

- During the first stage, every trial vector is exactly evaluated and saved into the database. At the end of stage I, the collected data is used for training the machine learning model.
- A previous study [31] demonstrates the superiority of the AdaBoost model over other ML models like NN or Support Vector Machine (SVM) in evaluating the safety state of trusses. Therefore, the AdaBoost model is employed instead of the NN model in order to improve the classification accuracy in this study.

The proposed method is called the AdaBoost-DE (short for AdaBoost classification-assisted Differential Evolution) and its flowchart is presented in Fig. 2.

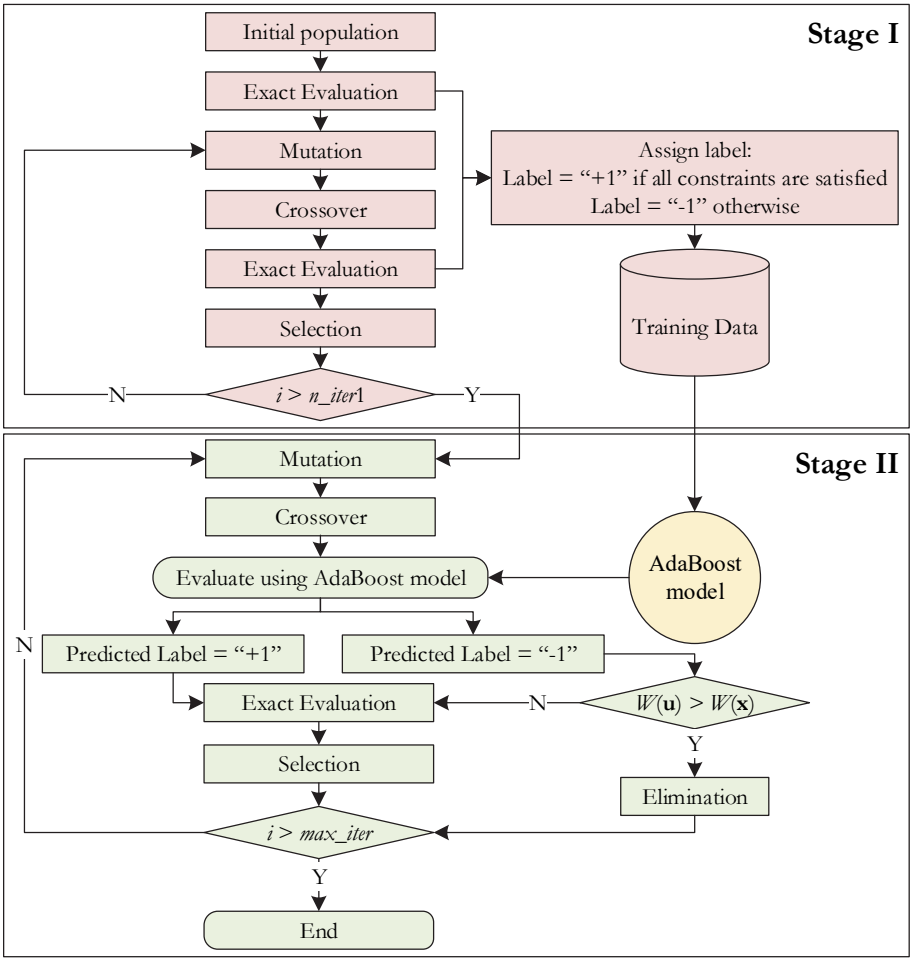

Figure 2: Flowchart of the AdaBoost-DE.

NUMERICAL EXAMPLE

Design data

A 160-bar transmission tower is carried out to demonstrate the efficiency of the proposed method. This structure has been optimized using numerous algorithms, such as the Multi-Metaheuristic based Search Method (MMSM) [5], the rank-based ant system (RBAS) [32], the adaptive elitist DE (aeDE) [33], the modified symbiotic organisms search (mSOS) algorithm [34], and the particle swarm optimizer cultural (PSOC) [35]. The configuration of the tower is presented in Fig. 3. The tower consists of 52 nodes and 160 members in which the information of nodes and members are listed in Tab. 1 and Tab. 2. The density of the steel material equals ρ=0.00785 kg/cm^3, and the modulus of elasticity is equal to E=2.047E+06 kgf/cm^2. This tower subjects to eight independent load cases which are introduced in Tab. 3. This problem has 38 design variables corresponding to cross-sectional areas of 38 member groups as shown in Tab. 2. The list of 42 available profiles used in this tower is given in Tab. 4. Members in the tower are designed according to the tensile stress and buckling stress conditions. The limitation of the tensile stresses equals 1500 kgf/cm^2 for all members while the critical buckling stresses of the members can be determined using the following equation:

$$\sigma_b = \begin{cases} 1300 - (kL/r)^2/24 & \text{if } kL/r \le 120 \\ 10^7/(kL/r)^2 & \text{otherwise} \end{cases} \tag{11}$$

in which: k is the effective length factor which is assumed to be 1.0 for all members; L is the length of the member; r is the radius of gyration of the member's section.

It must be noted that constraints on displacement, natural frequency, as well as geometric requirements need to be considered in practice. However, these constraints are neglected as in previous studies [32-35] for a fair comparison.

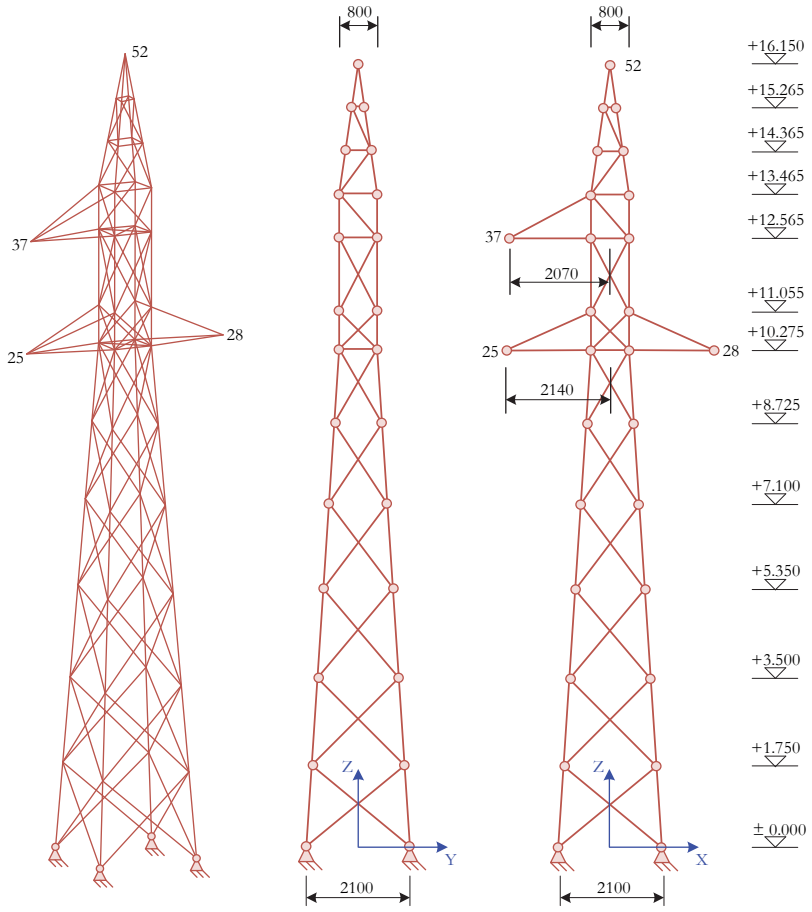

Figure 3: Layout of the 160-bar tower.

Node	*X* (cm)	*Y* (cm)	*Z* (cm)	Node	*X* (cm)	*Y* (cm)	*Z* (cm)	Node	*X* (cm)	*Y* (cm)	*Z* (cm)
1	-105.000	-105.000	0.000	19	60.085	60.085	710.000	37	-207.000	0.000	1256.500
2	105.000	-105.000	0.000	20	-60.085	60.085	710.000	38	40.000	40.000	1256.500
3	105.000	105.000	0.000	21	-49.805	-49.805	872.500	39	-40.000	40.000	1256.500
4	-105.000	105.000	0.000	22	49.805	-49.805	872.500	40	-40.000	-40.000	1346.500
5	-93.929	-93.929	175.000	23	49.805	49.805	872.500	41	40.000	-40.000	1346.500
6	93.929	-93.929	175.000	24	-49.805	49.805	872.500	42	40.000	40.000	1346.500
7	93.929	93.929	175.000	25	-214.000	0.000	1027.500	43	-40.000	40.000	1346.500
8	-93.929	93.929	175.000	26	-40.000	40.000	1027.500	44	-26.592	-26.592	1436.500
9	-82.859	-82.859	350.000	27	40.000	-40.000	1027.500	45	26.592	-26.592	1436.500
10	82.859	-82.859	350.000	28	214.000	0.000	1027.500	46	26.592	26.592	1436.500
11	82.859	82.859	350.000	29	40.000	40.000	1027.500	47	-26.592	26.592	1436.500
12	-82.859	82.859	350.000	30	-40.000	40.000	1027.500	48	-12.737	-12.737	1526.500
13	-71.156	-71.156	535.000	31	-40.000	-40.000	1105.500	49	12.737	-12.737	1526.500
14	71.156	-71.156	535.000	32	40.000	-40.000	1105.500	50	12.737	12.737	1526.500
15	71.156	71.156	535.000	33	40.000	40.000	1105.500	51	-12.737	12.737	1526.500
16	-71.156	71.156	535.000	34	-40.000	40.000	1105.500	52	0.000	0.000	1615.000
17	-60.085	-60.085	710.000	35	-40.000	-40.000	1256.500				
18	60.085	-60.085	710.000	36	40.000	-40.000	1256.500				

Table 1: Nodal coordinates.

Element no.	Group	Node		Element no.	Group	Node		Element no.	Group	Node		Element no.	Group	Node	
		i	*j*			*i*	*j*			*i*	*j*			*i*	*j*
1	1	1	5	11	2	4	5	21	4	7	12	31	6	10	15
2	1	2	6	12	2	1	8	22	4	8	11	32	6	11	14
3	1	3	7	13	3	5	9	23	4	8	9	33	6	11	16
4	1	4	8	14	3	6	10	24	4	5	12	34	6	12	15
5	2	1	6	15	3	7	11	25	5	9	13	35	6	12	13
6	2	2	5	16	3	8	12	26	5	10	14	36	6	9	16
7	2	2	7	17	4	5	10	27	5	11	15	37	7	13	17
8	2	3	6	18	4	6	9	28	5	12	16	38	7	14	18
9	2	3	8	19	4	6	11	29	6	9	14	39	7	15	19
10	2	4	7	20	4	7	10	30	6	10	13	40	7	16	20

Table 2: Element connectivity.

Element no.	Group	Node		Element no.	Group	Node		Element no.	Group	Node		Element no.	Group	Node	
		i	*j*			*i*	*j*			*i*	*j*			*i*	*j*
41	8	13	18	71	14	24	26	101	22	33	38	131	31	38	39
42	8	14	17	72	14	21	30	102	22	34	39	132	31	39	35
43	8	14	19	73	15	26	27	103	23	33	39	133	32	40	44
44	8	15	18	74	15	27	29	104	23	32	35	134	32	41	45
45	8	15	20	75	15	29	30	105	23	31	36	135	32	42	46
46	8	16	19	76	15	30	26	106	23	34	38	136	32	43	47
47	8	16	17	77	16	25	26	107	24	32	38	137	33	40	45
48	8	13	20	78	16	27	28	108	24	33	36	138	33	41	46
49	9	17	21	79	16	25	30	109	24	34	35	139	33	42	47
50	9	18	22	80	16	29	28	110	24	31	39	140	33	43	44
51	9	19	23	81	17	25	31	111	25	37	35	141	34	44	45
52	9	20	24	82	17	28	32	112	25	37	39	142	34	45	46
53	10	17	22	83	17	28	33	113	26	37	40	143	34	46	47
54	10	18	21	84	17	25	34	114	26	37	43	144	34	44	47
55	10	19	24	85	18	26	31	115	27	35	40	145	35	44	48
56	10	20	23	86	18	27	32	116	27	36	41	146	35	45	49
57	11	18	23	87	18	29	33	117	27	38	42	147	35	46	50
58	11	19	22	88	18	30	34	118	27	39	43	148	35	47	51
59	11	20	21	89	19	26	32	119	28	35	38	149	36	45	48
60	11	17	24	90	19	27	31	120	28	36	39	150	36	46	49
61	12	21	26	91	19	29	34	121	29	36	40	151	36	47	50
62	12	22	27	92	19	30	33	122	29	38	41	152	36	44	51
63	12	23	29	93	20	27	33	123	29	39	42	153	37	48	49
64	12	24	30	94	20	29	32	124	29	35	43	154	37	49	50
65	13	21	27	95	20	30	31	125	30	40	41	155	37	50	51
66	13	22	26	96	20	26	34	126	30	41	42	156	37	48	51
67	13	23	30	97	21	26	29	127	30	42	43	157	38	48	52
68	13	24	29	98	21	27	30	128	30	43	40	158	38	49	52
69	14	22	29	99	22	31	35	129	31	35	36	159	38	50	52
70	14	23	27	100	22	32	36	130	31	36	38	160	38	51	52

Table 2: Element connectivity (continued).

Load case	Node	P_X	P_Y	P_Z	Load case	Node	P_X	P_Y	P_Z
LC1	52	-868	0	-491	LC5	52	-917	0	-491
	37	-996	0	-546		37	-951	0	-546
	25	-1091	0	-546		25	-1015	0	-546
	28	-1091	0	-546		28	-636	1259	-428
LC2	52	-493	1245	-363	LC6	52	-917	0	-491
	37	-996	0	-546		37	-572	1303	-428
	25	-1091	0	-546		25	-1015	0	-546
	28	-1091	0	-546		28	-1015	0	-546
LC3	52	-917	0	-491	LC7	52	-917	0	-491
	37	-951	0	-546		37	-951	0	-546
	25	-1015	0	-546		25	-1015	0	-546
	28	-1015	0	-546		28	-636	1303	-428
LC4	52	-917	0	-546	LC8	52	-498	1460	-363
	37	-572	1259	-428		37	-951	0	-546
	25	-1015	0	-546		25	-1015	0	-546
	28	-1015	0	-546		28	-1015	0	-546

Table 3: Load cases (unit: kgf).

No.	A (cm²)	r (cm)	No.	A (cm²)	r (cm)	No.	A (cm²)	r (cm)	No.	A (cm²)	r (cm)
1	1.84	0.47	12	7.44	1.26	23	21.12	2.16	34	46.94	2.94
2	2.26	0.57	13	8.06	1.36	24	23.2	2.36	35	51	2.92
3	2.66	0.67	14	8.66	1.46	25	25.12	2.57	36	52.1	3.54
4	3.07	0.77	15	9.4	1.35	26	27.5	2.35	37	61.82	3.96
5	3.47	0.87	16	10.47	1.36	27	29.88	2.56	38	61.9	3.52
6	3.88	0.97	17	11.38	1.45	28	32.76	2.14	39	68.3	3.51
7	4.79	0.97	18	12.21	1.55	29	33.9	2.33	40	76.38	3.93
8	5.27	1.06	19	13.79	1.76	30	34.77	2.97	41	90.6	3.92
9	5.75	1.16	20	15.39	1.95	31	39.16	2.54	42	94.13	3.92
10	6.25	1.26	21	17.03	1.74	32	43	2.93			
11	6.84	1.15	22	19.03	1.94	33	45.65	2.94			

Table 4: List of available profiles.

Feature handling for machine learning models

When developing a machine learning model, the input features should be identified. Numerous studies used the cross-sectional areas of truss members as inputs [24-26,30]. However, as can be seen in Eq. (11), the critical buckling stress of a member depends on its effective length kL and the radius of gyration r. Among the two parameters, the effective length does not change during the sizing optimization process, but the radius of gyration r varies according to the profile of the member. Therefore, the radius of gyration r is also an important characteristic that should be included in the machine learning model.

In the present work, a feature handling technique is proposed to improve the accuracy of the machine learning model. In particular, there are 38 input features $\mathbf{x}=\{x_i | i=1,2,\ldots,38\}$ that are fed into the model, in which each feature x_i represents the ratio of the area and the radius of gyration of a profile used in this structure:

$$x_i = \frac{A_i}{r_i} \tag{12}$$

where: A_i, r_i are the cross-sectional area and the radius of gyration of the profile, respectively.

In this way, the model can learn both important characteristics including the cross-sectional area and the radius of gyration while keeping the number of input features.

Setups

To exhibit the advantages of the proposed method, the 160-bar tower is optimized according to both the original DE algorithm and the AdaBoost-DE algorithm. The parameters of the two algorithms are the same as follows: the scaling factor F=0.8, the crossover rate Cr=0.9, the population size NP=50, the maximum number of generations max_iter=750. Particularly for the AdaBoost-DE algorithm, the number of generations of the first stage is set to be n_iter1=50 which means that the amount of samples of the training data equals 1000. The AdaBoost classification model consists of T=100 weak classifiers in which each weak classifier is a decision tree with the maximum depth max_depth=1.

Both the DE and the AdaBoost-DE algorithms are written in Python programming language and performed by the personal computer having the system configuration as follows: processor Intel Core i5 2.5 GHz, RAM 4.00 Gb. The AdaBoost classification model is developed using the machine learning library scikit-learn [36]. The in-house Python code for finite element analysis (FEA) is based on the direct stiffness method.

Results

The statistical results of 30 independent runs are summarized in Tab. 5. The results of the previous studies are also presented in Tab. 5 for comparison. The value in bold indicates this is the best result among the compared algorithms. Fig. 4 plots the convergence histories of the DE and the AdaBoost-DE, in which Fig. 4(a) shows the relationship of the weight and the number of structural analyses that must be carried out during the optimization process while Fig. 4(b) displays the reduction of the weight over computing time.

		RBAS [32]	aeDE [33]	mSOS [34]	Present work	
					DE	AdaBoost-DE
Weight (kg)	Best	1336.7493	**1336.634**	**1336.634**	**1336.634**	**1336.634**
	Mean	1342.6083	1355.875	**1338.245**	1350.648	1353.370
	Worst	1353.4686	1410.611	**1343.821**	1464.828	1475.435
	SD	-	18.805	**2.080**	22.728	26.491
Average number of structural analyses		90,000	23,925	24,000	37,500	**22,488**
Average computing time (s)		-	-	-	3036.6	**2024.5**

Table 5: Statistical results for the optimization of the 160-bar transmission tower.

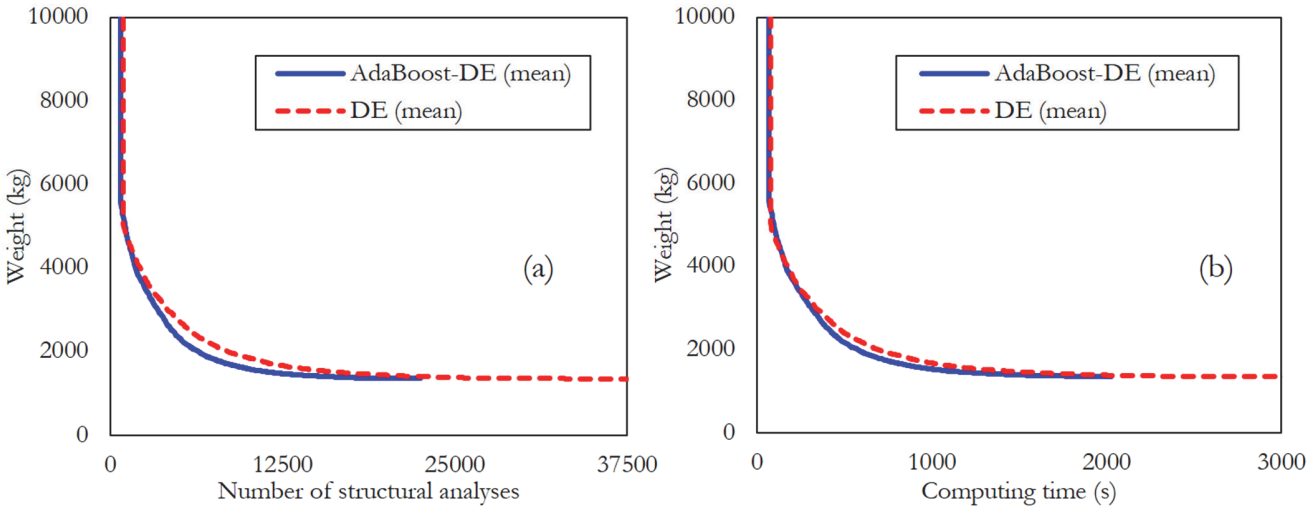

Figure 4: Convergence curves of the AdaBoost-DE and the DE.

Tab. 5 shows that the optimal results of the AdaBoost-DE, the DE, the aeDE, and the mSOS are the same and better than the RBAS (1336.634 kg for the AdaBoost-DE, the DE, the aeDE, the mSOS, and 1336.7493 kg for the RBAS). The optimal weights found by the MMSM (1329.715 kg) [5] and by the PSOC (1327.048 kg) [35] are smaller than the optimal weight found in the present work. However, according to our verification, these designs slightly violate the buckling constraint at elements 52 and 80. Thus, they are not reported in Tab. 5. Despite having the same optimal results, the advantage of the proposed method is the speed. The AdaBoost-DE conducts 22488 structural analyses while the required number of structural analyses of the RBAS, the DE, the aeDE, the mSOS are 90000 times, 37500 times, 23925 times, and 24000 times, respectively.

Fig. 4 clearly exhibits the benefit of integrating the AdaBoost classification technique into the DE algorithm. During 750 iterations, the DE requires 37500 analyses while the AdaBoost-DE performs only 22488 analyses. In other words, the application of the AdaBoost technique reduces 40% of structural analyses. Compared with the results in Ref. [30], the modifications proposed in the present work increase the reduction rate from 20% to 40%.

Regarding the computing time, although the proposed method requires additional time for training and employing the machine learning model, it is still more time-efficient than the original DE algorithm. According to Tab. 5, the average time consumed by the DE and the AdaBoost-DE are 3036.6 s and 2024.5 s, respectively. It means the AdaBoost-DE is approximately 1.5 times faster than the DE in terms of computing time.

Validation of the optimal design using the commercial software

In this section, the best solution found by the AdaBoost-DE is re-checked using the commercial software CSI SAP2000. This is a well-known software that has been widely utilized by structural engineers over the world. The section properties including sectional areas as well as the radius of gyration of 38 member groups of the best solution found by the AdaBoost-DE are reported in Tab. 6.

A 3D model containing 52 nodes and 160 elements is simulated by using SAP2000 as schematized in Fig. 5(a). The material steel used in this model is specified with the following properties: the modulus of elasticity E=2.047E+06 kgf/cm^2 and the weight per unit volume ρ=7.85E-03 kg/cm^3. Elements of the model are assigned to 38 different sections with the section properties as presented in Tab. 6. Eight static load cases are defined where the force magnitudes of these load cases are taken from Tab. 3.

The model is linearly analyzed and the stress contour of the envelope load combination is plotted in Fig. 5(b). Next, the tensile stress and buckling stress constraints are checked based on the obtained results. The maximum tensile stress found at Element 86 reaches 912.1 kgf/cm^2, which equals 61% of the limitation. The critical element according to the buckling stress condition is Element 105. The compression stress in this element is equal to -386.3 kgf/cm^2, achieving 100% of the allowable buckling stress. Thus, the optimal design does not violate any design constraints. Furthermore, the stresses of 160 elements obtaining by SAP2000 and by in-house FEA code are compared in Fig. 6. The coefficient of determination R^2=0.9996 is very close to 1.0 indicating the high similarity of SAP2000 and the developed FEA code.

Group	A (cm²)	r (cm)	Group	A (cm²)	r (cm)	Group	A (cm²)	r (cm)	Group	A (cm²)	r (cm)
1	19.03	1.94	11	5.75	1.16	21	2.66	0.67	31	2.26	0.57
2	5.27	1.06	12	12.21	1.55	22	8.06	1.36	32	3.88	0.97
3	19.03	1.94	13	6.25	1.26	23	5.75	1.16	33	1.84	0.47
4	5.27	1.06	14	5.75	1.16	24	6.25	1.26	34	1.84	0.47
5	19.03	1.94	15	3.88	0.97	25	5.75	1.16	35	3.88	0.97
6	5.75	1.16	16	7.44	1.26	26	2.26	0.47	36	1.84	0.47
7	15.39	1.95	17	1.84	0.47	27	4.79	0.97	37	1.84	0.47
8	5.75	1.16	18	8.66	1.46	28	2.66	0.67	38	3.88	0.97
9	13.79	1.76	19	2.66	0.67	29	3.47	0.87			
10	5.75	1.16	20	3.07	0.77	30	1.84	0.47			

Table 6: Section properties of the best solution.

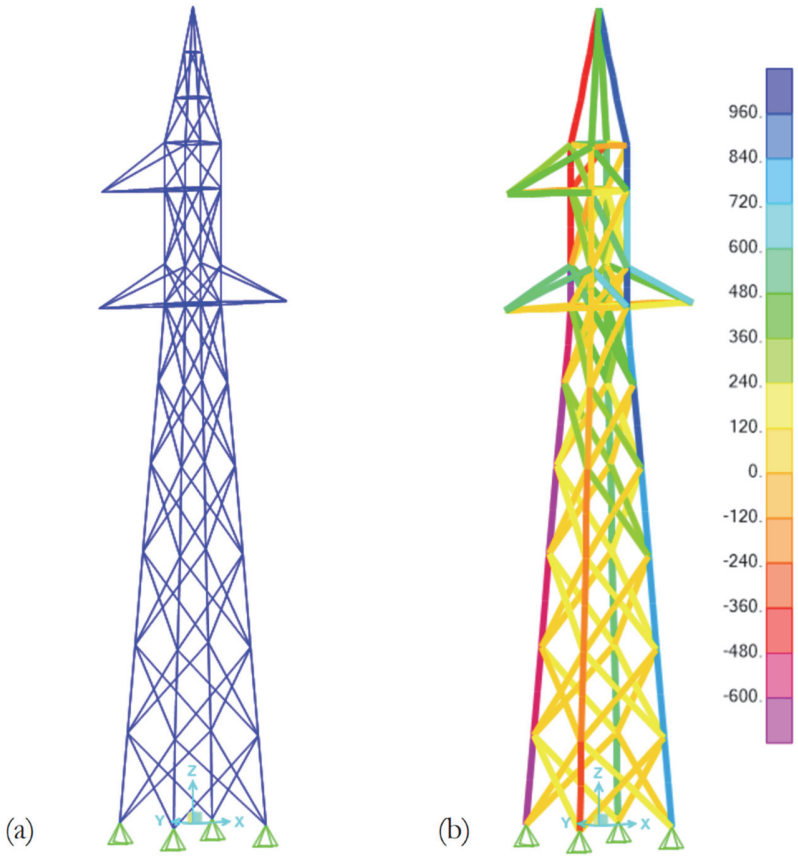

Figure 5: Model in SAP2000.

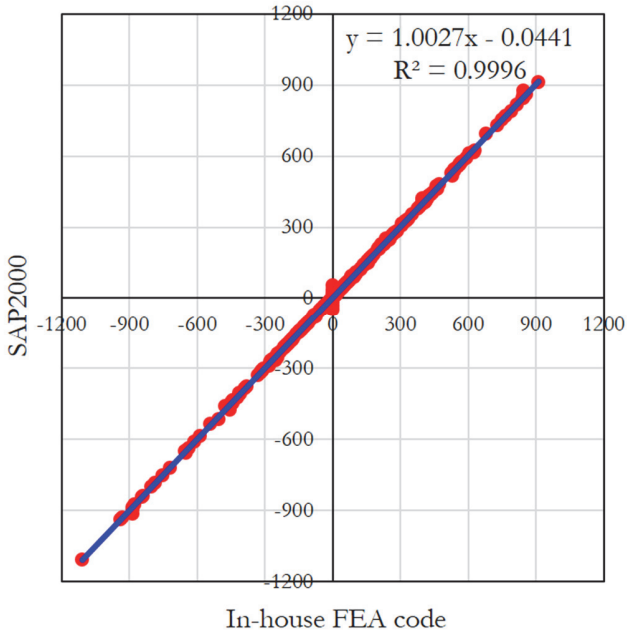

Figure 6: Comparison between the stresses obtained from SAP2000 and in-house FEA code.

Conclusions

In this paper, a method that combines the Differential Evolution algorithm and the Adaptive Boosting classification technique to minimize the weight of steel lattice towers is proposed. In the sooner generations, the original Differential Evolution with four basic operators is employed with the aim of exploring the design space and collecting training data. An Adaptive Boosting model is trained by the obtained data and then utilized to discard worse candidates in the later generations. Additionally, a feature handling technique is introduced for the purpose of improving the quality of the machine learning model.

The proposed method is applied to a 160-bar tower to illustrate its effectiveness. The optimal design found by the AdaBoost-DE is as good as the results obtained by other meta-heuristic algorithms. However, the advantage of the AdaBoost-DE is the fast convergence speed. This advantage is achieved because the AdaBoost-DE reduces the number of structural analyses by approximately 40% compared to the original Differential Evolution algorithm. Consequently, the optimization time of the AdaBoost-DE is significantly shortened. The numerical example conducted in this paper indicates that the AdaBoost-DE is an efficient method to optimize steel lattice towers.

Despite the advantages mentioned above, the proposed method also has some disadvantages. First, the integration of the AdaBoost model into the DE increases the complexity of the algorithm. This makes it difficult for structural engineers who are not familiar with advanced programming techniques. Next, the proposed method needs to set seven parameters instead of four as in the original DE. Consequently, the hyperparameter tuning process consumes more computational cost. Finally, the AdaBoost-DE requires additional time for training ML models. In general, the proposed method is suitable for problems where the design constraint verification is very time-consuming. The potential of the AdaBoost-DE for the optimization of large-scale towers should be further investigated.

Acknowledgment

The author T.-H. Nguyen was funded by Vingroup Joint Stock Company and supported by the Domestic Ph.D. Scholarship Programme of Vingroup Innovation Foundation (VINIF), Vingroup Big Data Institute (VINBIGDATA) code VINIF.2020.TS.134.

REFERENCES

[1] Rao, G.V. (1995). Optimum designs for transmission line towers. Computers & structures, 57(1), pp.81-92. DOI: 10.1016/0045-7949(94)00597-V.

[2] Taniwaki, K. and Ohkubo, S. (2004). Optimal synthesis method for transmission tower truss structures subjected to static and seismic loads. Structural and Multidisciplinary Optimization, 26(6), pp.441-454. DOI: 10.1007/s00158-003-0367-7.

[3] Shea, K. and Smith, I.F. (2006). Improving full-scale transmission tower design through topology and shape optimization. Journal of structural engineering, 132(5), pp.781-790. DOI: 10.1061/(ASCE)0733-9445(2006)132:5(781).

[4] Guo, H.Y. and Li, Z.L. (2011). Structural topology optimization of high-voltage transmission tower with discrete variables. Structural and Multidisciplinary Optimization, 43(6), pp.851-861. DOI: 10.1007/s00158-010-0561-3.

[5] Kaveh, A., Kalatjari, V.R. and Talebpour, M.H. (2016). Optimal design of steel towers using a multi-metaheuristic based search method. Periodica Polytechnica Civil Engineering, 60(2), pp.229-246. DOI: 10.3311/PPci.8222.

[6] de Souza, R.R., Miguel, L.F.F., Lopez, R.H., Miguel, L.F.F. and Torii, A.J. (2016). A procedure for the size, shape and topology optimization of transmission line tower structures. Engineering Structures, 111, pp.162-184. DOI: 10.1016/j.engstruct.2015.12.005.

[7] Tort, C., Şahin, S. and Hasançebi, O. (2017). Optimum design of steel lattice transmission line towers using simulated annealing and PLS-TOWER. Computers & Structures, 179, pp.75-94. DOI: 10.1016/j.compstruc.2016.10.017.

[8] Khodzhaiev, M. and Reuter, U. (2021). Structural optimization of transmission towers using a novel Genetic Algorithm approach with a variable length genome. Engineering Structures, 240, p.112306. DOI: 10.1016/j.engstruct.2021.112306.

[9] Couceiro, I., París, J., Martínez, S., Colominas, I., Navarrina, F. and Casteleiro, M. (2016). Structural optimization of lattice steel transmission towers. Engineering Structures, 117, pp.274-286. DOI: 10.1016/j.engstruct.2016.03.005.

[10] Nguyen, H., Vu, T., Vo, T. P. and Thai, H. T. (2021). Efficient machine learning models for prediction of concrete strengths. Construction and Building Materials, 266, 120950. DOI: 10.1016/j.conbuildmat.2020.120950.

[11] Ouladbrahim, A., Belaidi, I., Khatir, S., Magagnini, E., Capozucca, R., and Wahab, M. A. (2021). Prediction of Gurson Damage Model Parameters Coupled with Hardening Law Identification of Steel X70 Pipeline Using Neural Network. Metals and Materials International, 2021, pp.1-15. DOI: 10.1007/s12540-021-01024-4.

[12] Degtyarev, V. V. (2021). Neural networks for predicting shear strength of CFS channels with slotted webs. Journal of Constructional Steel Research, 177, 106443. DOI: 10.1016/j.jcsr.2020.106443.

[13] Truong, V. H., Vu, Q. V., Thai, H. T., and Ha, M. H. (2020). A robust method for safety evaluation of steel trusses using Gradient Tree Boosting algorithm. Advances in Engineering Software, 147, 102825. DOI: 10.1016/j.advengsoft.2020.102825.

[14] Mangalathu, S., and Jeon, J. S. (2019). Machine learning–based failure mode recognition of circular reinforced concrete bridge columns: Comparative study. Journal of Structural Engineering, 145(10), 04019104. DOI: 10.1061/(ASCE)ST.1943-541X.0002402.

[15] Shen, Z., Pan, P., Zhang, D., and Huang, S. (2020). Rapid structural safety assessment using a deep neural network. Journal of Earthquake Engineering, 2020, pp.1-17. DOI: 10.1080/13632469.2020.1785586.

[16] Benaissa, B., Hocine, N. A., Belaidi, I., Hamrani, A., and Pettarin, V. (2016). Crack identification using model reduction based on proper orthogonal decomposition coupled with radial basis functions. Structural and Multidisciplinary Optimization, 54(2), pp.265-274. DOI: 10.1007/s00158-016-1400-y.

[17] Samir, K., Brahim, B., Capozucca, R., and Wahab, M. A. (2018). Damage detection in CFRP composite beams based on vibration analysis using proper orthogonal decomposition method with radial basis functions and cuckoo search algorithm. Composite Structures, 187, pp.344-353. DOI: 10.1016/j.compstruct.2017.12.058.

[18] Tiachacht, S., Khatir, S., Le Thanh, C., Rao, R. V., Mirjalili, S., and Wahab, M. A. (2021). Inverse problem for dynamic structural health monitoring based on slime mould algorithm. Engineering with Computers, 2021, pp.1-24. DOI: 10.1007/s00366-021-01378-8.

[19] Khatir, S., Tiachacht, S., Le Thanh, C., Tran-Ngoc, H., Mirjalili, S., and Wahab, M. A. (2021). A new robust flexibility index for structural damage identification and quantification. Engineering Failure Analysis, 2021, 105714. DOI: 10.1016/j.engfailanal.2021.105714.

[20] Khatir, S., Tiachacht, S., Thanh, C.L., Bui, T.Q. and Wahab, M.A., (2019). Damage assessment in composite laminates using ANN-PSO-IGA and Cornwell indicator. Composite Structures, 230, p.111509. DOI: 10.1016/j.compstruct.2019.111509.

[21] Khatir, S., Boutchicha, D., Le Thanh, C., Tran-Ngoc, H., Nguyen, T.N. and Abdel-Wahab, M., (2020). Improved ANN technique combined with Jaya algorithm for crack identification in plates using XIGA and experimental analysis. Theoretical and Applied Fracture Mechanics, 107, p.102554. DOI: 10.1016/j.tafmec.2020.102554.

[22] Khatir, S., Tiachacht, S., Le Thanh, C., Ghandourah, E., Mirjalili, S. and Wahab, M.A., (2021). An improved Artificial Neural Network using Arithmetic Optimization Algorithm for damage assessment in FGM composite plates. Composite Structures, p.114287. DOI: 10.1016/j.compstruct.2021.114287.

[23] Zenzen, R., Khatir, S., Belaidi, I., Le Thanh, C., & Wahab, M. A. (2020). A modified transmissibility indicator and Artificial Neural Network for damage identification and quantification in laminated composite structures. Composite Structures, 248, 112497.

[24] Kaveh, A., Gholipour, Y. and Rahami, H. (2008). Optimal design of transmission towers using genetic algorithm and neural networks. International Journal of Space Structures, 23(1), pp.1-19. DOI: 10.1260/026635108785342073.

[25] Taheri, F., Ghasemi, M.R. and Dizangian, B., 2020. Practical optimization of power transmission towers using the RBF-based ABC algorithm. Structural Engineering and Mechanics, 73(4), pp.463-479. DOI: 10.12989/sem.2020.73.4.463.

[26] Hosseini, N., Ghasemi, M.R. and Dizangian, B., 2021. ANFIS-based Optimum Design of Real Power Transmission Towers with Size, Shape and Panel Variables using BBO Algorithm. IEEE Transactions on Power Delivery. DOI: 10.1109/TPWRD.2021.3052595.

[27] ASCE, 2000, March. Design of latticed steel transmission structures. American Society of Civil Engineers.

[28] Storn, R. and Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization, 11(4), pp.341-359. DOI: 10.1023/A:1008202821328.

[29] Freund, Y. and Schapire, R.E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences, 55(1), pp.119-139. DOI: 10.1006/jcss.1997.1504.

[30] Nguyen, T.-H., and Vu, A.-T. (2022). Application of Artificial Intelligence for Structural Optimization. In: Modern Mechanics and Applications, Singapore, Springer, pp. 1052-1064. DOI: 10.1007/978-981-16-3239-6_82.

[31] Nguyen, T.-H. and Vu, A.-T. (2021). Evaluating structural safety of trusses using Machine Learning. Frattura ed Integrità Strutturale, 15(58), pp. 308–318. DOI: 10.3221/IGF-ESIS.58.23.

[32] Capriles, P.V., Fonseca, L.G., Barbosa, H.J. and Lemonge, A.C. (2007). Rank-based ant colony algorithms for truss weight minimization with discrete variables. Communications in Numerical Methods in Engineering, 23(6), pp. 553-575. DOI: 10.1002/cnm.912.

[33] Ho-Huu, V., Nguyen-Thoi, T., Vo-Duy, T. and Nguyen-Trang, T. (2016). An adaptive elitist differential evolution for optimization of truss structures with discrete design variables. Computers & Structures, 165, pp.59-75. DOI: 10.1016/j.compstruc.2015.11.014.

[34] Do, D.T. and Lee, J. (2017). A modified symbiotic organisms search (mSOS) algorithm for optimization of pin-jointed structures. Applied soft computing, 61, pp.683-699. DOI: 10.1016/j.asoc.2017.08.002.

[35] Jafari, M., Salajegheh, E. and Salajegheh, J., (2021). Optimal design of truss structures using a hybrid method based on particle swarm optimizer and cultural algorithm. Structures, 32, pp. 391-405. DOI: 10.1016/j.istruc.2021.03.017.

[36] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. The Journal of Machine Learning Research, 12, pp.2825-2830.