European Journal of Mechanics / A Solids 98 (2023) 104869

Contents lists available at ScienceDirect

European Journal of Mechanics / A Solids

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

Optimum design of nonlinear structures via deep neural network-based parameterization framework

Hau T. Mai [a,b], Seunghye Lee [a], Donghyun Kim [a], Jaewook Lee [a], Joowon Kang [c], Jaehong Lee [a,*]

[a] Deep Learning Architectural Research Center, Sejong University, 209 Neungdong-ro, Gwangjin-gu, Seoul 05006, Republic of Korea
[b] Faculty of Mechanical Engineering, Industrial University of Ho Chi Minh City, 12 Nguyen Van Bao Street, Ward 4, Go Vap District, Ho Chi Minh City 70000, Viet Nam
[c] School of Architecture, Yeungnam University, 280, Daehak-Ro, Gyeongsan, Gyeongbuk 38541, Republic of Korea

ARTICLE INFO

Keywords:
Deep neural network
Hyperparameter optimization
Bayesian optimization
Truss optimization
Geometric nonlinear

ABSTRACT

In this paper, a robust deep neural network (DNN)-based parameterization framework is proposed to directly solve the optimum design for geometrically nonlinear trusses subject to displacement constraints. The core idea is to integrate DNN into Bayesian optimization (BO) to find the best optimum structural weight. Herein, the design variables of the structure are parameterized by weights and biases of the network with the spatial coordinates of all joints as the training data. A loss function of the network is built based on the predicted cross-sectional areas and deflection constraints obtained by supporting finite element analysis (FEA) and arc-length method. Accordingly, the optimum weight corresponding to the minimum loss function is indicated as soon as the complete training process. And then it is also serving as an objective of the BO for performing the hyperparameter optimization (HPO) to find the best optimum structural weight. Several illustrative numerical examples for geometrically nonlinear space trusses are examined to determine the efficiency and reliability of the proposed approach. The obtained results demonstrate that our framework can overcome the drawbacks of applications of machine learning in computational mechanics.

1. Introduction

Over the past decade, structural optimization problems have received considerable attention from many researchers in the computational mechanics community. In general, for the linear analysis, the structural displacements have not significantly changed the original geometry in most cases (Thai et al., 2020). However, regarding the design of slender and light structures such as trusses, arches, thin walls, and so on, the geometrically nonlinear effects associated with large deformations must be considered to fully understand the real structural responses (Vo and Lee, 2011; Missoum et al., 2002; Wempner, 1971). Furthermore, it must be remembered that all real structures are nonlinear in some way. Hence, consideration of the nonlinear behaviors were necessary when optimizing the structures for greater safety (Saka and Ulker, 1992).

In general, several different algorithms have been proposed for handling nonlinear structural optimization problems in recent literature and categorized into two main groups. In the first one, gradient-based algorithms have been developed for searching optimal solutions, such as the optimality criterion (OC) (Khot, 1983; Khot and Kamat, 1985), the combining nonlinear and linear goal programming (El-Sayed et al., 1989), the incorporation of the OC and nonlinear analysis technique (Saka and Ulker, 1992), the equivalent loads (Hrinda and Nguyen, 2008), etc. Yet, there are several potential drawbacks of conventional methods such as falling into the local optimum, depending on the initial point selection, concerning the unavailable gradient information. The second one is gradient-free algorithms which rely on evolutionary and population genetics (Kaveh and Rahami, 2006; Kameshki and Saka, 2007, 2001; Pezeshk et al., 2000). Despite these algorithms have been achieved certain success, they require many evaluation functions which is a major drawback of the nonlinear structural optimization problem because of a huge effort of numerical simulations.

In recent years, machine learning (ML) has been widely and successfully applied in computational mechanics problems, including mechanical materials (Truong et al., 2020b; Srinivasan and Saghir, 2013; Lee et al., 2021b), structural analysis (Mai et al., 2022c; Li et al., 2021; Nguyen-Thanh et al., 2020), structural optimization (Mai et al., 2021, 2022a; Trinh et al., 2022), damage detection (Lee et al., 2021a; Truong et al., 2020a, 2022), reliability analysis (Lieu et al., 2022; Afshari et al., 2022; Jia and Wu, 2022), and so on. In specific, physics-informed neural network framework have received much attention in

* Corresponding author.
E-mail addresses: maitienhaunx@gmail.com (H.T. Mai), seunghye@sejong.ac.kr (S. Lee), jhlee@sejong.ac.kr (J. Lee).

https://doi.org/10.1016/j.euromechsol.2022.104869
Received 21 July 2022; Received in revised form 28 October 2022; Accepted 13 November 2022
Available online 21 November 2022

solving partial differential equations due to its mesh-free and other desirable characters (Anitescu et al., 2019). For applications of ML to structural optimization, the data-driven methodology is the most popular approach when the network as a surrogate model is built based on the previously collected data. Therein, the output data is a collection of optimal designs that are found by numerical simulations (e.g., FEA) with respect to the input data containing the information to define the problem. And it has been successfully applied for design optimization of linear (Hajela and Berke, 1991b,a; Ramasamy and Rajasekaran, 1996; Lee et al., 2020; Li et al., 2019) and nonlinear structures (Mai et al., 2021; Abueidda et al., 2020). However, this approach exists limitations that are difficult to overcome such as : (i) the obtained results depend strongly on the amount and quality of the training data as well as the hyperparameters of the network; (ii) estimating the appropriate training data size for each problem is a difficult task; (iii) it requires a large number of costly iterations for collecting data of the nonlinear problem; (iv) and especially, it demands the prior knowledge and expert experience in collecting data and selecting hyperparameters. In recent times, as an alternative approach, the network is used to parameterize the design variable for solving truss (Mai et al., 2022c) and topology optimization (Chandrasekhar and Suresh, 2021; Zehnder et al., 2021; Hoyer et al., 2019). Overall, as compared to the data-driven framework, the advantages of this approach include that the training data only contains the information of the structure without any numerical simulations, its size is simple to determine, and the network directly performs the optimization process. Nevertheless, it faces several challenges due to the tuning hyperparameters as well as the local minimum (Mai et al., 2022c). Additionally, to our knowledge, the ML model directly solving optimization of truss structures with geometrically nonlinear behavior has still not been yet studied thus far. And all these drawbacks motivated us to perform this study where the above-mentioned issues were resolved.

This paper aims to propose the DNN-based parameterization scheme which is established to solve the optimization of geometrically nonlinear truss structure with the self-tuning hyperparameter of the network. Accordingly, the cross-sectional areas of all truss members are parameterized by weights and biases with the spatial coordinates of joints as the training data. Note that this data is a definite size and easily collects without any numerical simulations. Therein, the combination of FEA and arc-length technique plays a supporting role in building the loss function of the network. Once the training is complete, the obtained optimal weight corresponding to the hyperparameters of the network is the objective of the BO to perform the self-adjusting operation. And finally, the best optimum weight of the structure can be indicated with respect to the optimal network. Several numerical examples for design optimization of truss structures under deflection constraints are investigated to evaluate the efficiency and reliability of the proposed framework.

The rest of the paper is organized as follows. Section 2 provides the statement of the size optimization problem of truss structures with deflection constraints. Thereafter, the efficient DNN-based parameterization paradigm and BO algorithm for the HPO problem are presented in Section 3. Several numerical examples are investigated to demonstrate the reliability of the present method in Section 4. And finally, the conclusion is outlined in Section 5.

2. Statement of structural optimization problem

Optimization of structures with nonlinear behavior is known as a complex and time-consuming task related to nonlinear constraints. The sizing optimization of truss structures is one of these problems, which aims to minimize the structural weight that all design constraints are satisfied. Therein, the cross-sectional areas of members are treated as continuous design variables and confined within an acceptable range. Its mathematical model can be expressed as follows

$$\begin{aligned} \text{Minimize} \quad & W(\mathbf{A}) = \sum_{k=1}^{n_g} A_k \sum_{i=1}^{m_k} \rho_i L_i, \\ \text{subjected to} \quad & g_j(\mathbf{A}) = \frac{\Delta_j}{[\Delta]_j} - 1 \le 0, \quad j = 1, 2, \ldots, p, \\ & A_k^{low} \le A_k \le A_k^{up}, \qquad k = 1, 2, \ldots, n_g, \end{aligned} \tag{1}$$

where $W(.)$ is the weight of the truss structure; A_k is the cross-sectional area of the members belonging to the kth group which can range between A_k^{low} and A_k^{up}; n_g denotes the total number of groups in the structure; m_k is the total number of members in the kth group; p denotes the number of constrained displacements; ρ_i and L_i are the material density and length of the ith member; g_j represents the jth constraint function; Δ_j and $[\Delta]_j$ are the jth displacement and allowable displacement, respectively. To obtain the above constraints, Total Lagrangian kinematic description (De Borst et al., 2012; Coda and Paccola, 2014) and arc-length technique (Riks, 1979; Kadapa, 2021) are applied for the resolution of the geometrically nonlinear problem.

In order to solve this problem, a self-adaptive penalty function, as suggested by Sonmez (2011) and Hasançebi (2008), is used in this work to handle the structural displacement constraints. Therefore, Eq. (1) is rewritten as follows

$$\begin{aligned} \text{Minimize} \quad & \mathcal{L}(\mathbf{A}) = \left(1 + \varepsilon_1 c\right)^{\varepsilon_2} W(\mathbf{A}), \\ & c = \sum_{j=1}^{p} \max\left(0, g_j(\mathbf{A})\right), \end{aligned} \tag{2}$$

in which c represents the sum of the violated constraints; the exploration and exploitation rates of the design domain are controlled by two parameters ε_1 and ε_2. According to Refs. Sonmez (2011), Hasançebi (2008), the value of ε_2 is fixed at 1, while the parameter ε_1 is dynamic and self-adaptive based on the feedback from the obtained result in the previous iteration and is described as follows

$$\varepsilon_1^{(t)} = \begin{cases} (1/\kappa)\,\varepsilon_1^{(t-1)} & \text{if } f^{(t-1)} \text{ is feasible,} \\ \kappa\,\varepsilon_1^{(t-1)} & \text{if } f^{(t-1)} \text{ is infeasible,} \end{cases} \tag{3}$$

in which $\varepsilon_1^{(t)}$ and $\varepsilon_1^{(t-1)}$ are the penalty coefficients at iterations t and $(t-1)$, respectively. And ε_1 is set equal to 1 at the beginning of the iteration. κ is the learning parameter of $\varepsilon_1^{(t)}$, and is defined as follows (Sonmez, 2011; Hasançebi, 2008)

$$\kappa = 1 + \frac{1}{p} > 1.01, \tag{4}$$

where p is the total number of constraints.

3. Deep neural network-based parameterization framework

In this section, the DNN-based parameterization paradigm is introduced to directly implement optimization of truss structures with nonlinear behavior. The schematic illustration, as shown in Fig. 1, provides an overall view of the proposed approach. Herein, the cross-sectional areas are parameterized by the parameters $\boldsymbol{\theta}$ of the network. Consequently, the core of our approach is based on optimizing all weights and biases instead of the cross-sectional area of truss members. In addition, BO is adopted to tune the hyperparameters of the network with a view to achieving the best optimum weight design. To reach this goal, a set of initial guess hyperparameters $\mathbf{Y}_{n_0}$ is first generated by Latin Hypercube Sampling (LHS) technique in the search space. For each vector of hyperparameters in the training set $\mathbf{Y}_{n_0}$, DNN is designed with the initial weights and biases are set randomly according to the normal distribution on the interval $[-1, 1]$. The coordinates of members are presented as samples of the input data, while the outputs of the network are the cross-sectional area of members unknown $\hat{\mathbf{A}}$ and their value obtained by the feedforward phase, respectively. Next, the self-adaptive penalty function is treated as the loss function which is derived based on the weight and nonlinear structural responses

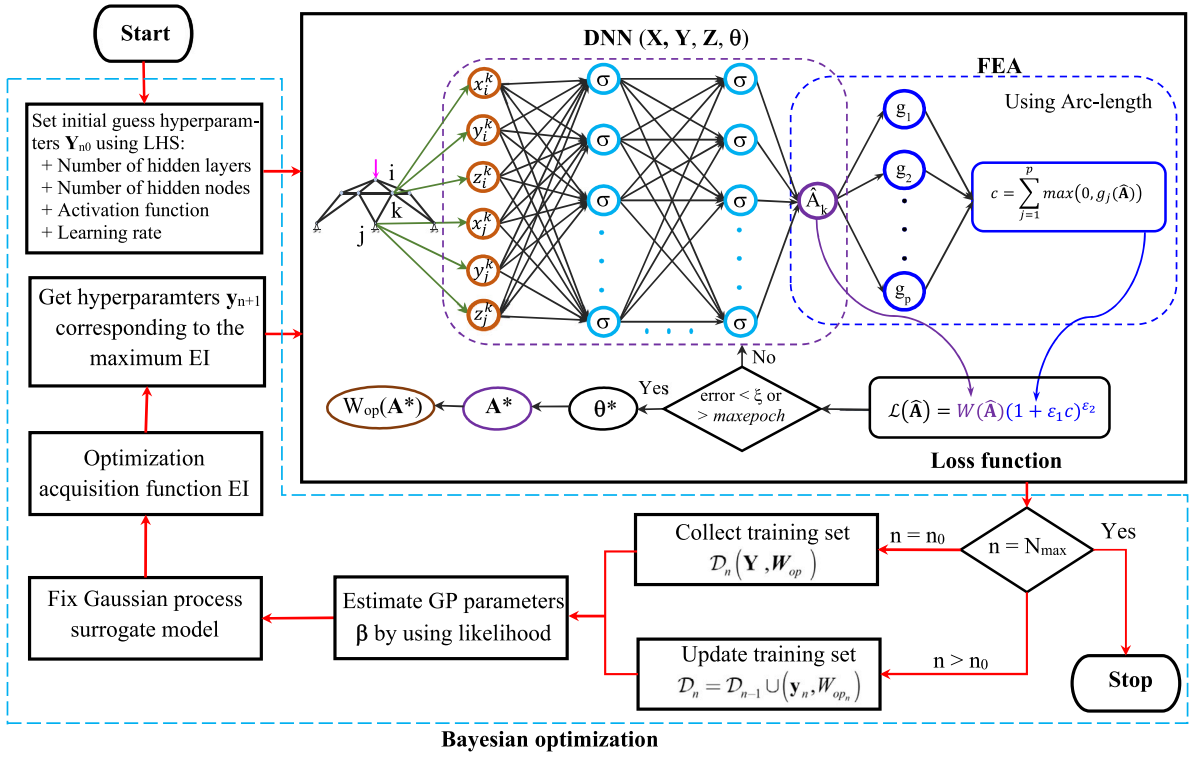

Fig. 1. Schematic of integration of DNN-based parameterization framework into Bayesian optimization for structural optimization.

obtained from numerical simulation based on FEA and arc-length technique with respect to the predicted outputs of the network. After that, all parameters of the network are automatically adjusted by the backpropagation algorithm. And the computation described above is iterated and repeated many times, and it is known as the training. When the network has been properly trained, the optimum weight of structures can be discovered with the minimum loss function. When the training set including the corresponding hyperparameters and the obtained optimum weights is collected or updated. Thereafter, within BO framework, a Gaussian process (GP) surrogate model is used to approximate the training set $\mathcal{D}_n$. When the next candidate point $\mathbf{y}_{n+1}$ is found by maximizing the acquisition equation, and its corresponding optimum weight $W_{op_{n+1}}$ obtained by training the network, respectively. And an iterative infilling process is performed to search the best minimum weight corresponding to the optimal hyperparameters. More generally, our approach is made up of three major components, and the following subsections go into greater detail about them.

3.1. Deep neural network

First, it is worth mentioning that our model is designed based on an unsupervised learning framework. As a result, the training data only contains the input data, while the corresponding output values are not required. In this work, a set of coordinates $(x_i, y_i, z_i, x_j, y_j, z_j)$ of all truss members is chosen as the training data. And the cross-sectional areas $\mathbf{A}$ are the output values of the network which are unknown and must be estimated through the learning process. It is easily be seen that the geometry of the structure makes it simple to obtain training data with small size and independent of sampling techniques. More concretely, its size is $(k \times 6)$ and $(k \times 4)$ for space and planar truss structures, respectively. Here, k indicates the number of truss members.

A fully connected neural network with depth ℓ, as shown in Fig. 2, is employed to obtain the solution of the cross-sectional areas $\mathbf{A}$ within the spatial coordinates of nodes $(\mathbf{x}, \mathbf{y}, \mathbf{z})$. It consists of one input layer, one output layer, and $(\ell - 1)$ hidden layers. Therein, the first layer (0th) has four or six neurons for 2- or 3-dimensional problems, and the last layer (ℓ^{th}) has only one neuron. Meanwhile, the number of hidden layers, as well as the number of neurons in each hidden layer, were hyperparameters optimized by BO. Neurons in the same layer have no connections, but every two neurons in neighboring layers are connected by weight (w). In addition, the hidden neuron is added by bias (b) into the sum of the weighted inputs.

Generally, the training of the network is a cyclic process performed to adjust the weights and biases in order to minimize the loss function. To achieve this goal, the feedforward and backpropagation mechanisms were utilized for each iteration. Firstly, the information flow is passed in one direction from the input layer to the output layer in the feedforward process which is defined by a mapping: $\mathbb{R}^6 \rightarrow \mathbb{R}$. Consequently, the relationship between the input and output values of each layer is defined as follows

$$\begin{aligned}
&\text{input layer} \quad : \quad \mathbf{h}^0 = \left[x_i, y_i, z_i, x_j, y_j, z_j\right] \in \mathbb{R}^6, \\
&\text{hidden layers} : \quad \mathbf{h}^k = f_1\left(\mathbf{W}^{k^T}\mathbf{h}^{(k-1)} + \mathbf{b}^k\right) \in \mathbb{R}^{m_k}, \\
&\qquad for \;\; 1 \le k \le (\ell - 1), \\
&\text{output layer} \quad : \quad \mathbf{h}^\ell = f_2\left(\mathbf{W}^{\ell^T}\mathbf{h}^{(\ell-1)} + \mathbf{b}^\ell\right) = \hat{A} \in \mathbb{R},
\end{aligned} \tag{5}$$

where $\mathbf{W}^{(.)}$ is the weight matrix; $\mathbf{b}^{(.)}$ is the bias vector; m_k is the number of units in the kth hidden layer; $f(.)$ denotes the activation function, which allows the network to learn about the complete relationship between the input and output. There are several popular activation functions, for example, Tanh, Sigmoid, ReLU, LeakyReLU, Softmax, Linear, and so on. In this study, the softmax function is applied for the output layer, while the activation function of the hidden layer is estimated by BO.

And then, the loss function $\mathcal{L}$, which will be explained in detail in Section 3.2, is defined based on the outputs and their corresponding responses. Next, to minimize this function, its gradient information

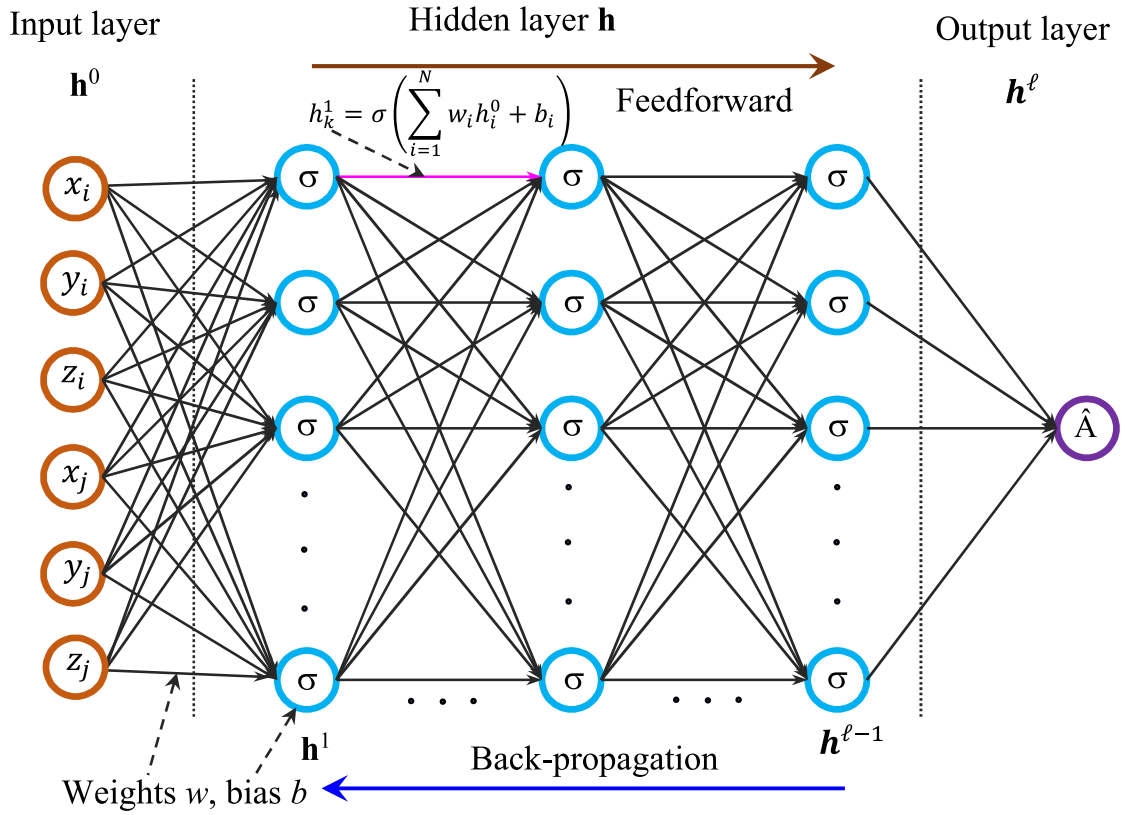

Fig. 2. A fully-connected deep neural network architecture.

concerning the parameters of the network is determined by the back-propagation algorithm. And the weights and biases will be updated in combination with the optimizer. Several optimizers are available for training the DNN model such as SGD, RMSprop, Adagrad, etc. One of the most widely used optimizers is Adam (Kingma and Ba, 2014) and demonstrated through recent literature in the field of analysis and optimization of the structure (Mai et al., 2021, 2022c,a). Hence, it is used in this work to perform the training process. As a result, the network parameters at iteration t are expressed as

$$\boldsymbol{\theta_t} = \boldsymbol{\theta}_{t-1} - \eta \frac{\mathbf{m}_t \sqrt{1-\beta_2^t}}{\left(1-\beta_1^t\right)\left(\sqrt{\mathbf{v}_t} + \varepsilon\sqrt{1-\beta_2^t}\right)}, \tag{6}$$

where $\mathbf{m}_t$ and $\mathbf{v}_t$ are obtained by

$$\begin{aligned} \mathbf{m}_t &= \beta_1 \mathbf{m}_{t-1} + \left(1-\beta_1\right).\nabla\mathcal{L}_t\left(\boldsymbol{\theta}_{t-1}\right), \\ \mathbf{v}_t &= \beta_2 \mathbf{v}_{t-1} + \left(1-\beta_2\right).\nabla\mathcal{L}_t\left(\boldsymbol{\theta}_{t-1}\right), \end{aligned} \tag{7}$$

in which $\boldsymbol{\theta}$ is the trainable parameter vector that consists of weights and biases; $\mathbf{m}_t$ and $\mathbf{v}_t$ are the first and second raw moment vectors which are controlled by two exponential decay rates β_1, $\beta_2 \in [0, 1)$; ε denotes a constant added to ensure numerical stability, and η is the learning rate. In this study, BO was introduced to indicate the optimal learning rate, while other parameters with their default values as suggested by Kingma and Ba (2014) were utilized to train the model. Interested readers can refer to Kingma and Ba (2014) for details. $\nabla\mathcal{L}$ is the sensitivity of the loss function which is focused in the next section.

3.2. Loss function

Clearly, in our approach, the cross-sectional areas are previously unknown, and their predicted values $\hat{\mathrm{A}}_i(\mathbf{x}, \mathbf{y}, \mathbf{z}, \boldsymbol{\theta})$ are expressed by the coordinates of truss members and the parameters of the network. According to the above Fig. 1, the loss function is strictly derived from the predicted cross-sectional areas and structural responses obtained using FEA with respect to these predicted values, and it is also the penalty function of structural optimization in Eq. (2). It is rewritten as follows

$$\mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{z}, \boldsymbol{\theta}) = \left(1+\varepsilon_1 c\right)^{\varepsilon_2} W\left(\hat{\mathbf{A}}(\mathbf{x}, \mathbf{y}, \mathbf{z}, \boldsymbol{\theta})\right). \tag{8}$$

As shown in Eq. (8), it is easily seen that the design variables are now the parameters of the network instead of the cross-sectional areas as the conventional approach. Hence, the training process, which is also the structural optimization task, is to minimize the loss function to find the optimal parameters of the network.

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \left(\mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{z}, \boldsymbol{\theta})\right). \tag{9}$$

In order to train the model, the sensitivity of the loss function to the parameter change is given by

$$\frac{\partial \mathcal{L}}{\partial \theta_i} = \sum_{k=1}^{n_g} \left[\varepsilon_2\left(1+\varepsilon_1 c\right)^{\varepsilon_2 - 1} W \frac{\partial c}{\partial \hat{A}_k} + \left(1+\varepsilon_1 c\right)^{\varepsilon_2} \frac{\partial W}{\partial \hat{A}_k}\right] \frac{\partial \hat{A}_k}{\partial \theta_i}. \tag{10}$$

From Eq. (10), the derivative of the output $\frac{\partial \hat{A}_k}{\partial \theta_i}$ is computed automatically with the backpropagation algorithm. And, an automatic differentiation tool JAX (Bradbury et al., 2020) is applied for computing the gradient of constraints (Mai et al., 2022a; Chandrasekhar et al., 2021). While the other term $\frac{\partial W}{\partial \hat{A}_k}$ can be easily and exactly determined by the following formulation

$$\frac{\partial W}{\partial \hat{A}_k} = \sum_{i=1}^{m_k} \rho_i L_i. \tag{11}$$

And clearly, when the sensitivity of the loss function is entirely defined by the above terms and hence it facilitates training to adjust the parameters of the network as Eq. (6).

3.3. Hyperparameter tuning

Hyperparameters are a set of parameters that cannot be directly obtained from the training. It can be categorized into two types: one relates to model training, e.g. learning rate, batch size, etc., and the other deals with the structure of networks, such as the number of hidden layers, hidden neurons, and so on. They are often based on the trial-and-error process, and hence it depends on the experience and intuition of expert users (Hertel et al., 2020). In addition, HPO plays a crucial role in the success of applying DNN to practical problems, and it is known as a black-box optimization problem (Asali et al., 2021). Hence, traditional algorithms are poorly suited for such tuning tasks.

In this article, BO, which is a sequential surrogate model-based algorithm, is utilized to estimate the best hyperparameters of the network. Accordingly, a probabilistic surrogate model is developed to fit all currently observed samples into the target function. Then, the next candidate point is estimated by maximizing the acquisition function to detect promising hyperparameter regions, and the above steps are repeated until the stop criterion is satisfied. The mathematical formulation of the unconstrained black-box optimization can be expressed as follows

$$\mathbf{y}^* = \underset{\mathbf{y}\in\mathcal{X}\subset\mathbb{R}^d}{\arg\min}\, W_{op}(\mathbf{y}), \tag{12}$$

where $\mathbf{y}$ is the hyperparameter vector including real-, integer-, and categorical-valued attributes; $\mathcal{X}$ denotes a compact set of $\mathbb{R}^d$ with d dimensions; W_{op} is the obtained minimum mass of the truss structure by the network corresponding to hyperparameters, and it is selected as the objective function of the tuning process.

To approximate the above objective function, GP is utilized to provide a distribution over functions. Let us examine a set of initial observations $D_n = \left\{\mathbf{y}_i, W_{op_i}\right\}$ in which the obtained minimum weights by the network $\mathbf{W}_{op} = \left[W_{op_1}, W_{op_2}, \ldots, W_{op_n}\right]^T$ are noise-free at $\mathbf{Y} = \left[\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n\right]^T$. When the posterior probability distribution of $W_{op}(\mathbf{y})$ at a new point $\mathbf{y}$ can be obtained via the Bayes' rule as follows

$$W_{op}(\mathbf{y})\Big|\,\mathbf{y}, D_n = \mathcal{N}\left(\mu_n(\mathbf{y}),\ \sigma_n^2(\mathbf{y})\right), \tag{13}$$

with

$$\begin{aligned}\mu_n(\mathbf{y}) &= \mu(\mathbf{y}) + \mathbf{k}^T\mathbf{K}^{-1}\left(\mathbf{W}_{op} - \mu(\mathbf{Y})\right),\\ \sigma_n^2(\mathbf{y}) &= k(\mathbf{y},\mathbf{y}) - \mathbf{k}^T\mathbf{K}^{-1}\mathbf{k},\end{aligned} \tag{14}$$

where $\mu_n(.)$ and $\sigma_n^2(.)$ are the posterior mean and covariance function, respectively; $\mathbf{k}$ and $\mathbf{K}$ are the covariance vector and matrix; The *Matérn* kernel function is chosen to calculate the covariance values (Jones et al., 1998).

Next, an acquisition function is established to indicate the next hyperparameter set by its maximizing value based on the previous observations D_n. It must be carefully selected to trade off exploitation in current areas and exploration over the search space. And there are several popular choices for this function, such as the lower confidence bound (LCB), probability of improvement (PI), entropy search, and so on. In this work, the expected improvement (EI), which is the most common acquisition function, is selected because of its good performance (Jones et al., 1998). Its mathematical expression is rewritten as follows

$$EI(\mathbf{y}) = \left(W_{op}(\mathbf{y}^*) - \mu_n(\mathbf{y})\right)\Phi\left(\frac{W_{op}(\mathbf{y}^*)-\mu_n(\mathbf{y})}{\sigma_n(\mathbf{y})}\right) + \sigma_n(\mathbf{y})\,\phi\left(\frac{W_{op}(\mathbf{y}^*)-\mu_n(\mathbf{y})}{\sigma_n(\mathbf{y})}\right), \tag{15}$$

where $\Phi(.)$ and $\phi(.)$ are the standard normal cumulative distribution function and probability density function, respectively. From Eq. (15), the first term is devoted to the local search phase to evaluate points with low mean, while the other term is dedicated to the global search phase to estimate points with high uncertainty. Interested readers can refer to Ref. Jones et al. (1998) for more details.

Algorithm 1 Generic pseudo-code for BO without constraints

Input: n_0: number of the initial hyperparameters combinations
N_{max}: maximum iterations
β: GP parameter space
Output: $\mathbf{y}^*, W_{op}^*$: the best solution

Using LHS to obtain n_0 combinations of hyperparameters $\mathbf{Y}_{n_0}$ from the design space
Training the network with the above hyperparameters to obtain the weight values $\mathbf{W}_{op_{n_0}}$
Collect a set of initial observations $D_{n_0} = \left\{\mathbf{y}_i, W_{op_i}\right\}\ \forall i = 1, 2, ..., n_0$
Target $W_{op}(\mathbf{y}^*) = \min \mathbf{W}_{op}$
Set $n = n_0$
while $n \le N_{max}$ **do**
  Find GP hyperparameter β by the maximum likelihood estimation method
  Build the GP model on D_n
  Find y_{n+1} by maximizing Eq. (15)
  Training the network with the hyperparameters $\mathbf{y}_{n+1}$ to evaluate $W_{op_{n+1}}$
  Append $D_{n+1} = D_n \cup \left\{\left(\mathbf{y}_{n+1}, W_{op_{n+1}}\right)\right\}$
  Update $W_{op}(\mathbf{y}^*)$
  $n = n + 1$
end while

A pseudo-code of the general BO framework for the hyperparameter tuning is summarized in Algorithm 1. In order to evaluate its efficiency, a single variable test function with five initial samples is considered to estimate the minimum value. And the performance process is depicted in Fig. 3. In which, the solid black line presents the true function, the blue dashed line denotes the GP posterior mean. The respective infill function is shown by the solid red line, and the cyan area represents the 95% confidence interval. It is easily seen that the EI value tends to increase near the minimum posterior mean, and the maximum posterior uncertainty. The next samples are determined through the combination of measured values and uncertainties. And it only needs nine evaluation functions to find the near-global optimal solution.

4. Numerical examples

In this section, four well-known benchmark problems are investigated to justify the efficiency of our approach for optimization of structures with geometrically nonlinear behavior. The obtained results will be compared with the DE and previous works. In order to enhance the reliability and performance of the network, BO algorithm is employed to automatically tune the hyperparameters, including the number of hidden layers, the number of neurons in each hidden layer, the activation function, and the learning rate of the used Adam optimizer, and their value range are summarized in Table 1. In addition, Softmax is adopted as an activation function for the output layer during the training process. In all cases, the initial and maximum number of iterations for BO are set to $n_0 = 10$ and $N_{max} = 25$, respectively. While the maximum number of epochs equal to 300 for the training network. Note that the learning process is terminated when either the maximum number of epochs reaches or the norm of the gradient value is less than 0.01. In addition, HPO is implemented 10 independent runs to evaluate the influence of uncertain quantities in BO.

For DE algorithm, its parameters are set similar to Hau et al. (Mai et al., 2021). Due to the stochastic nature of the algorithm, the number of independent runs is only limited to 10 times. To ensure a fair comparison of the various method, all numerical examples are implemented based on Tensorflow library with Python language and run on a desktop computer with Core i5-8500 CPU @3.0 GHz, 16 GB RAM of memory.

Table 1
Configuration space for the hyperparameters of the network.

Hyperparameter	Search space	Type
No. of hidden layers	[1, 4]	Integer
No. of hidden neurons	[20, 60]	Integer
Activation function	[ReLU, Sigmoid, Softmax, Tanh, LeakyReLU]	Categorical
Learning rate	[0.001, 0.1]	Real

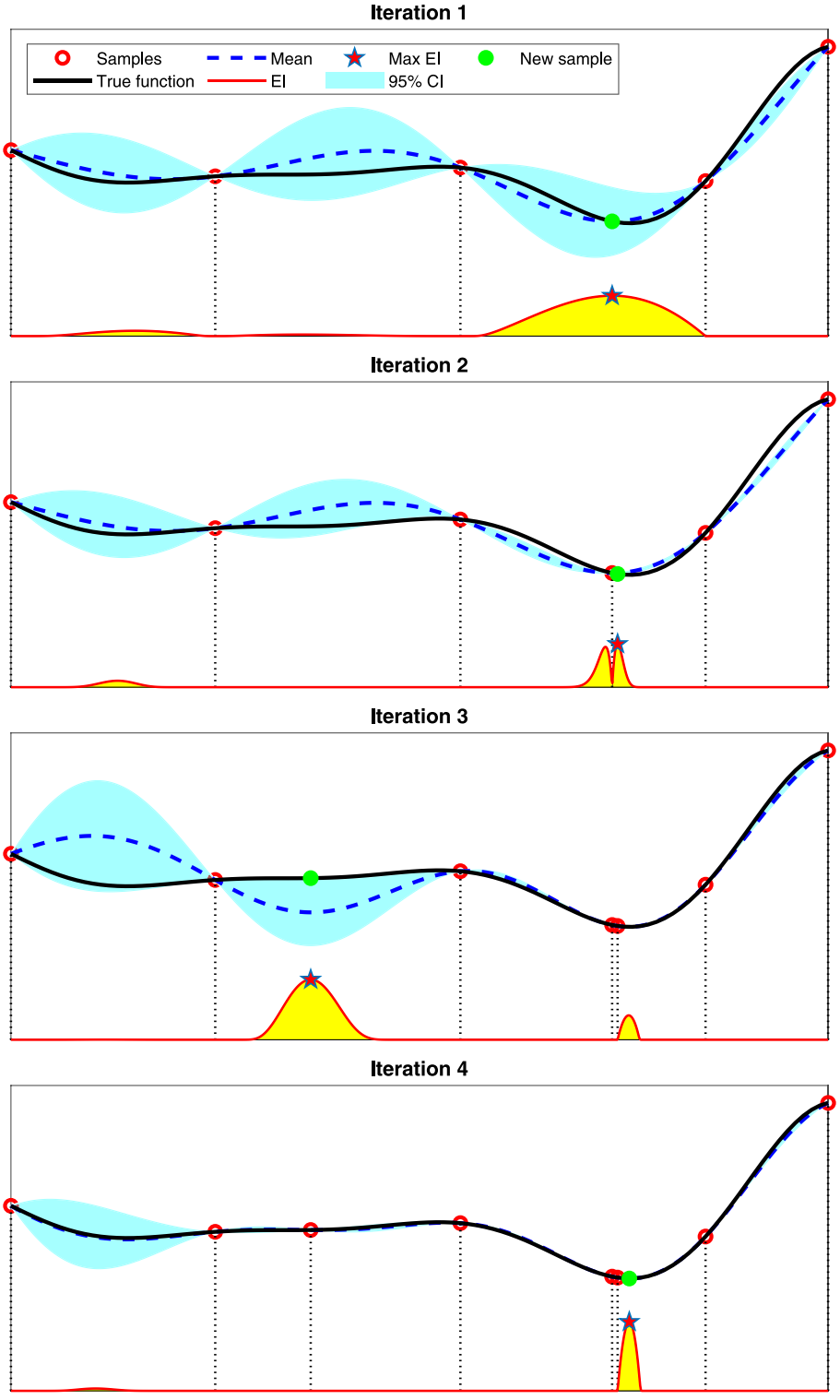

Fig. 3. An example of EI based Bayesian optimization for solving the one-dimensional minimization problem.

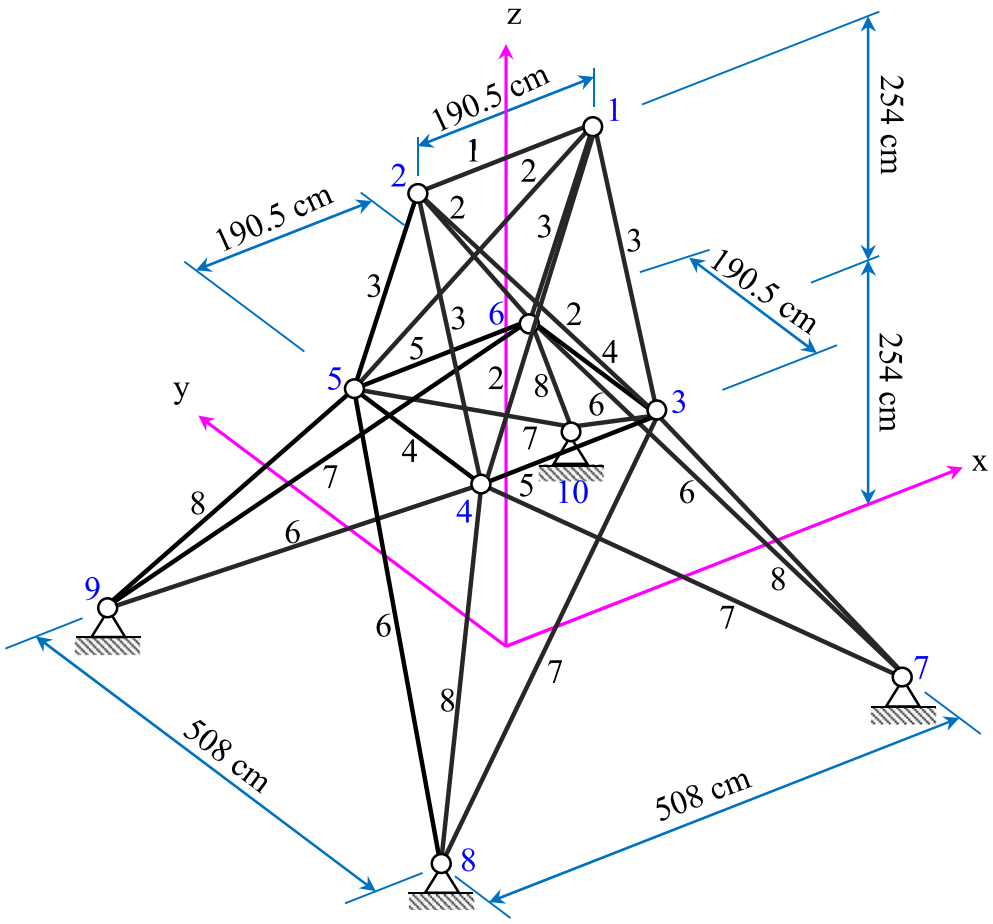

Fig. 4. A 25-bar space truss structure.

Table 2
Material properties, upper and lower bounds on design variables.

Test problems	Young's modulus E (kN/mm^2)	Material density ρ (kg/cm^3)	Cross-sectional area (mm^2)
25-bar space truss	207	0.00785	$2 \le A_i \le 40$
52-bar dome truss	210	0.00785	$2 \le A_i \le 100$
56-bar space truss	210	0.00785	$5 \le A_i \le 150$
120-bar dome truss	210	0.00785	$2 \le A_i \le 50$

Table 3
Loading condition for the 25-bar space truss.

Nodes	Loading (kN)		
	x	y	z
1	−80	−120	30
2	−60	−100	30
3	−30	0	0
6	−30	0	0

4.1. 25-bar space truss

A 25-bar space truss structure, as shown in Fig. 4, is investigated as the first numerical example for the optimum design of geometrically nonlinear truss structures. Young's modulus, material properties, and design variable bounds are summarized in Table 2. Cross-sectional areas of members are categorized into 8 groups with respect to 8 design variables as depicted in Fig. 4. It is subjected to the loading condition defined in Table 3. The displacement at nodes 1 and 2 are limited in the interval [−10, 10] mm in the x and y directions. In order to get a fair comparison, a set of 10 initial hyperparameter combinations is used for all infill strategies of the BO.

The optimal hyperparameters of the network found by various infill strategies are reported in Table 4. Besides, Table 5 summarizes best, worst, mean, standard deviation (Std), and 95% confidence interval (95% CI) of the optimal weight. Firstly, it is easily seen that there is a good agreement between the optimal weights obtained by different procedures. Clearly, the differences here were not significant between the best, worst, and mean weights. Therein, the EI achieves the lightest design overall (W_{best} = 504.231 kg; Std = 0.069 kg; 95% CI [504.556, 504.643] kg), and then the PI (W_{best} = 504.577 kg; Std = 0.189 kg; 95% CI [504.863, 505.097] kg), and LCB (W_{best} = 504.675 kg; Std = 0312 kg; 95% CI [505.223, 505.609] kg). In addition, the optimal architecture of the network obtained by the EI (6-39-39-1) is smaller than PI (6-37-37-37-1), and LCB (6-60-60-60-60-1). Hence, it requires the least number of weights and biases with only 1873 parameters, while the other strategies require a larger number of parameters, e.g., PI with 3109 parameters, and LCB with 11461 parameters. Furthermore, the obtained activation function is ReLU which is simple for computing, non-parameters, avoids gradient saturation problems, and converges much faster than others (Korenciak, 2018). The convergence histories of the best tuning hyperparameters process with various infill strategies are depicted in Fig. 5. Note that they coincide at the first 10 iterations

Table 4
Optimum hyperparameters of the network obtained using the BO with different acquisition functions for the 25-bar space truss.

Infill strategy	Hyperparameters			
	No. of hidden layers	No. of hidden neurons	Activation function	Learning rate
PI	3	37	ReLU	0.01489
LCB	4	60	LeakyReLU	0.00425
EI	2	39	ReLU	0.03796

Table 5
Statistics of the optimal weight with different acquisition functions for the 25-bar space truss.

Weight (kg)	Acquisition functions		
	PI	LCB	EI
Best	504.577	504.675	**504.231**
Worst	506.489	507.362	**504.988**
Mean	504.980	505.416	**504.600**
Std	0.189	0.312	**0.069**
95% CI	504.863–505.097	505.223–505.609	**504.556–504.643**

because of the common use of initial hyperparameters generated by the LHS technique. But after that, their convergence rates are different at all. Clearly, the EI shows its efficiency in detecting the optimal hyperparameters with the minimum weight. Consequently, it is chosen as an infill strategy for the BO in this study.

The optimal results gained by our approach with the optimal network architecture (6-39-39-1) and other algorithms are tabulated as Tables 6–7. It is easily seen that the optimum weight (504.231 kg) obtained by the present approach is smaller than other studies (Saka (Saka and Ulker, 1992): 507 kg; FEA-DE (Mai et al., 2021): 504.315 kg; DNN-DE (Mai et al., 2021): 505.493 kg; SQP: 652.255 kg). More formally, all constraints are satisfied as shown in Table 7. Additionally, Fig. 7 shows the convergence histories of various methods. Clearly, the convergence rate of the proposed approach with the optimal network is much faster than of the FEA-DE, DNN-DE, and sequential quadratic programming (SQP). Furthermore, it tends to be stable around 100 analyses and reaches the optimal solution after only 175 analyses. On the contrary, the FEA-DE and DNN-DE require 1,490,799 and 18,120 times of nonlinear analyses (Mai et al., 2021), respectively. Otherwise, both SQP and our model are the gradient-based methods, so their convergence speed accelerate in the early iteration. However, the optimal weight found by SQP (652.255 kg) is a local optimum instead of a global optimum. This can easily be explained by the fact that it depends on the choice of initial point and hard to choose a suitable starting point. Fig. 6 represents the iteration history of the SQP algorithm for two different initial points in the feasible and infeasible domains, respectively. As can be seen on the plot, it cannot escape from the local minima. Meanwhile, the proposed method has completely overcome this drawback by using BO. It allowed the automatic tuning of hyperparameters of the network. And this process is the automatic selection of the starting point. Because when the hyperparameters were changed, the initial weights and biases of the network leaded to change the position of starting point. Its utility is demonstrated through the obtained results shown in Table 5 and Fig. 5. Therefore, our approach is capable of effectively handling design problems containing the local minima.

4.2. 52-bar dome truss

Next, a 52-bar dome truss illustrated in Fig. 8 is examined. This benchmark has been previously studied by Saka and Ulker (1992). The data of the optimization problem is tabulated in Table 2. All members of the structure are classified into 8 groups with respect to the design variables as labeled in the same figure. An external load of 150 kN is applied in the z-axis' positive direction at joints 6–13. Since this is a symmetric structure, the vertical displacements of nodes 1, 2, 6, and 7 are restricted to 10 mm.

As the previously investigated example, the optimal hyperparameters of the network, including hidden layers (3), neurons in each hidden layer (37), activation function (ReLU), and learning rate (0.01489) as shown in Table 8, are found by the BO with respect to the minimum weight of 2,141.01 kg. From Table 9, it is easily be seen that the best (2,141.01 kg), mean (2,142.715 kg) and 95% CI (2,142.280 kg - 2,143.149 kg) values of the optimum weight are close with the Std less than 0.7 kg. Fig. 9 shows the graph of the convergence curve which gives a more detailed performance view for tuning hyperparameter. Clearly, the optimal hyperparameters achieved after only 20 times of training. On the other hand, the optimal results corresponding to the optimal network, including the cross-sectional areas, weight as well as deflection constraints, are summarized in comparison with other studies in Tables 10–11. It is obvious that our approach outperforms others in terms of the minimum weight, while all displacements are free from any violations of constraints. More specifically, the best optimum weight obtained by the proposed method (2,141.01 kg) is smaller than DNN-DE (2,142.41 kg), FEA-DE (2,141.9 kg), and Saka (5,161 kg). Clearly, the optimal weight found by Saka is still far behind comparing to other studies. This is due to the fact that it probably traps in the local optimum. Meanwhile, our model entirely overcomes this challenge. In addition, a comparison of the convergence rates is shown in Fig. 10. As can be seen on the plot, the mass of structures rapidly decreases in the first ten epochs and find the solution only through 237 epochs. Again, our procedure converges very fast to the optimal solution, while the others are still a long way from the target value. It can easily be interpreted by the fact that the training network works based on the gradient-based optimization. Hence, the number of evaluations will go down much more quickly. Furthermore, the obtained results demonstrate the efficiency of the proposed framework.

4.3. 56-bar space truss

The next optimization problem deals with the 56-bar space truss shown in Fig. 11. As shown in this figure, the structure has 4 stories and the cross-sectional areas of members as design variables are collected in 4 groups. It is subjected to concentrated loads of 45.5 kN in the x-axis' positive direction at nodes 1, 2, 5, 7, 9, 11, 13, and 15 and 91 kN in the negative z-direction at nodes 1, 2, 3, and 4. The data dealing with the design, including Young's modulus, density, and design variable bounds, are listed in Table 2. All displacements of joints in the x-direction are limited in the interval [−30, 30] mm.

The optimal results, including the hyperparameters, design variables, total weight, and deflection constraints, are reported in Tables 8, 9, 12, and 13. Accordingly, the best combination of hyperparameters (4, 59, ReLU, 0.09405) is found by the BO with respect to the minimum weight (14,549.691 kg) that it only needs 25 times for the training network. It can be observed that the best optimum weight is close to mean (14,550.723 kg) and worst weights (14,554.903 kg) with the error less than 0.008% and 0.036%, respectively. Furthermore, the best optimal masses is found with the small Std values (0.508 kg). In addition, the lower (14,550.422 kg) and upper (14,551.023 kg) bound values of the 95% CI are not significantly different and close to the best weight. The convergence history of the BO is depicted in Fig. 12. As observed, the best-fitted network model achieves after only 5 samples which are found by the infill strategy EI with 10 initial samples. Additionally, the optimum weight attained by the network with the optimal hyperparameters is the best design without violation constraints. Note that although Saka (Saka and Ulker, 1992) was given the smallest weight (13,577.160 kg), the displacement constraints are violated at nodes 1 (31.1542 mm) and 2 (32.2244 mm) in the x-direction. As indicated in Ref. Mai et al. (2021), the sensitivity of the nonlinear response and control parameters of the numerical method used by Saka

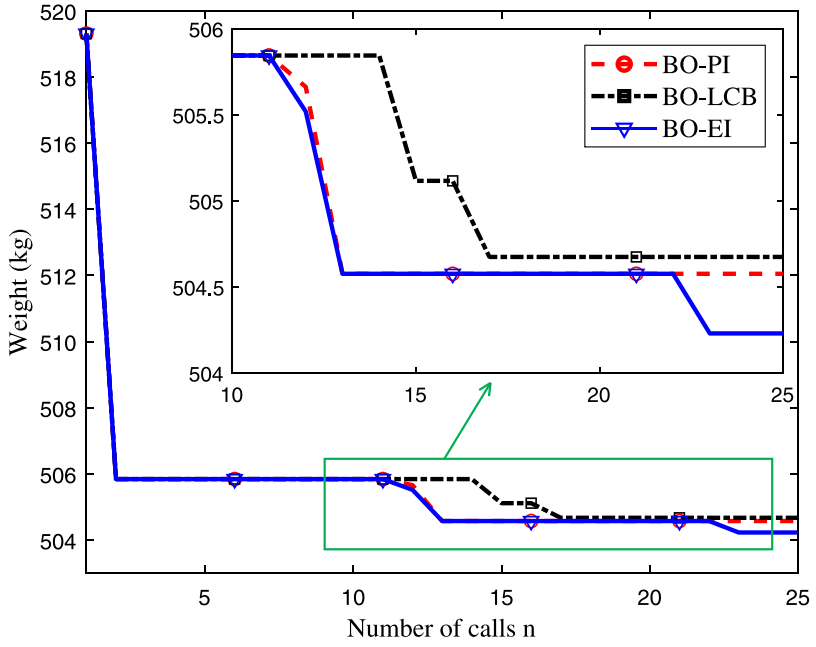

Fig. 5. The convergence histories of the HPO using BO for the 25-bar space truss structure.

Table 6
Comparison of optimal results for the 25-bar space truss.

Design variable A_i (cm^2)	Saka (Saka and Ulker, 1992)	FEA-DE (Mai et al., 2021)	DNN-DE (Mai et al., 2021)	SQP	Present
1	2.000	2.000	2.000	2.330	2.003
2	7.500	3.098	4.099	11.498	2.934
3	13.120	16.910	15.113	8.940	17.025
4	2.000	2.000	2.000	2.352	2.035
5	4.270	6.566	3.906	2.271	6.736
6	3.800	3.891	4.851	10.238	3.876
7	4.220	3.330	3.744	9.544	3.265
8	17.150	18.320	17.779	14.404	18.439
Best weight (kg)	507	504.315	505.493	652.255	**504.231**

Table 7
The displacement constraints of the 25-bar space truss.

Displacements (mm)	Saka (Saka and Ulker, 1992)	FEA-DE (Mai et al., 2021)	DNN-DE (Mai et al., 2021)	SQP	Present
u_1	−6.704	−9.764	−8.313	−6.170	−9.027
v_1	−10.000	−10.000	−10.000	−9.535	−9.997
u_2	−6.289	−9.346	−7.905	−5.760	−8.639
v_2	−8.850	−8.323	−8.601	−8.610	−8.427

Table 8
Optimum hyperparameters obtained by using the BO for different problems.

Test problems	Hyperparameters			
	No. of hidden layers	No. of hidden neurons	Activation function	Learning rate
52-bar dome truss	3	37	ReLU	0.01489
56-bar space truss	4	59	ReLU	0.09405
120-bar dome truss	2	45	ReLU	0.04433

can affect the optimal result. Fig. 13 displays the convergence curves of the present method and FEA-DE for the structural weight. Once again shows that the convergence speed accelerates within 50 first epochs and achieves the optimal solution after only 198 times nonlinear analyses.

4.4. 120-bar dome truss

The last problem done herein is to optimize a 120-bar dome truss structure as shown in Fig. 14. All cross-sectional areas of members are categorized into 7 groups corresponding to design variables. The data concerning the design of this benchmark is indicated in Table 2. The system is subjected to vertical loads in the negative direction of the z-axis which are 60 kN at node 1, 30 kN at nodes 2–13, and 10 kN at nodes 14–37. The vertical displacement of free nodes under this loading is limited to 10 mm.

The optimal hyperparameters were obtained after 25 training times, as shown in Table 8 and Fig. 15. With respect to the optimal network, the optimum results, including the weight, statistics, design variables, and constraint values, are reported in Tables 9, 14 and 15. Firstly,

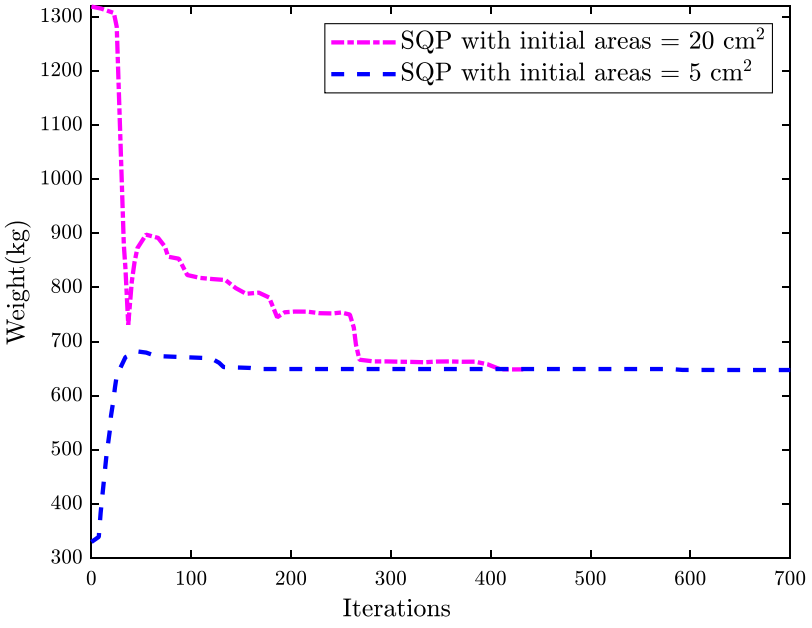

Fig. 6. Iteration history of the SQP algorithm for different initial areas for the 25-bar space truss..

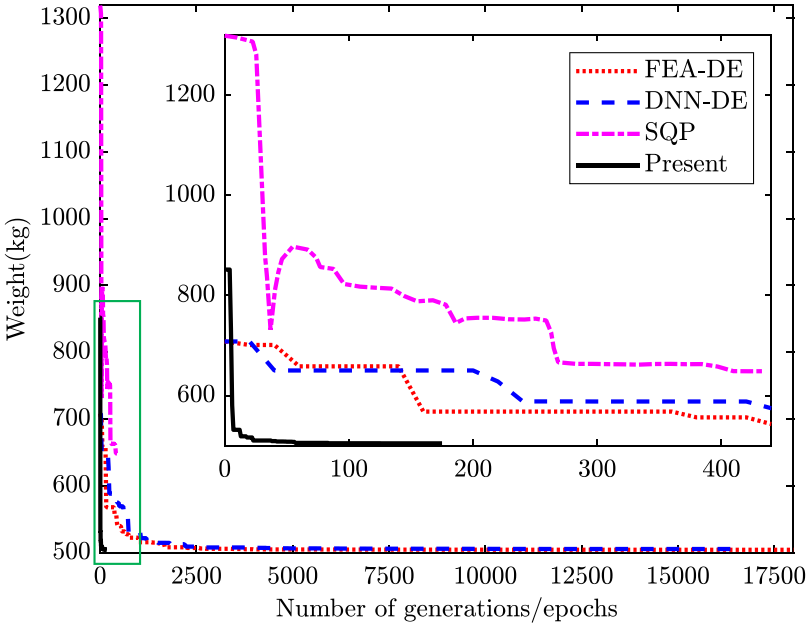

Fig. 7. The weight convergence histories of the optimal network and other studies for the 25-bar space truss.

Table 9
Statistics of the optimal weight with different problems.

Test problems	Weight (kg)				
	Best	Worst	Mean	Std	95% CI
52-bar dome truss	2,141.010	2,146.903	2,142.715	0.665	2,142.280–2,143.149
56-bar space truss	14,549.691	14,554.903	14,550.723	0.508	14,550.422–14,551.023
120-bar dome truss	5,836.717	5,841.903	5,838.019	0.605	5,837.661–5,838.376

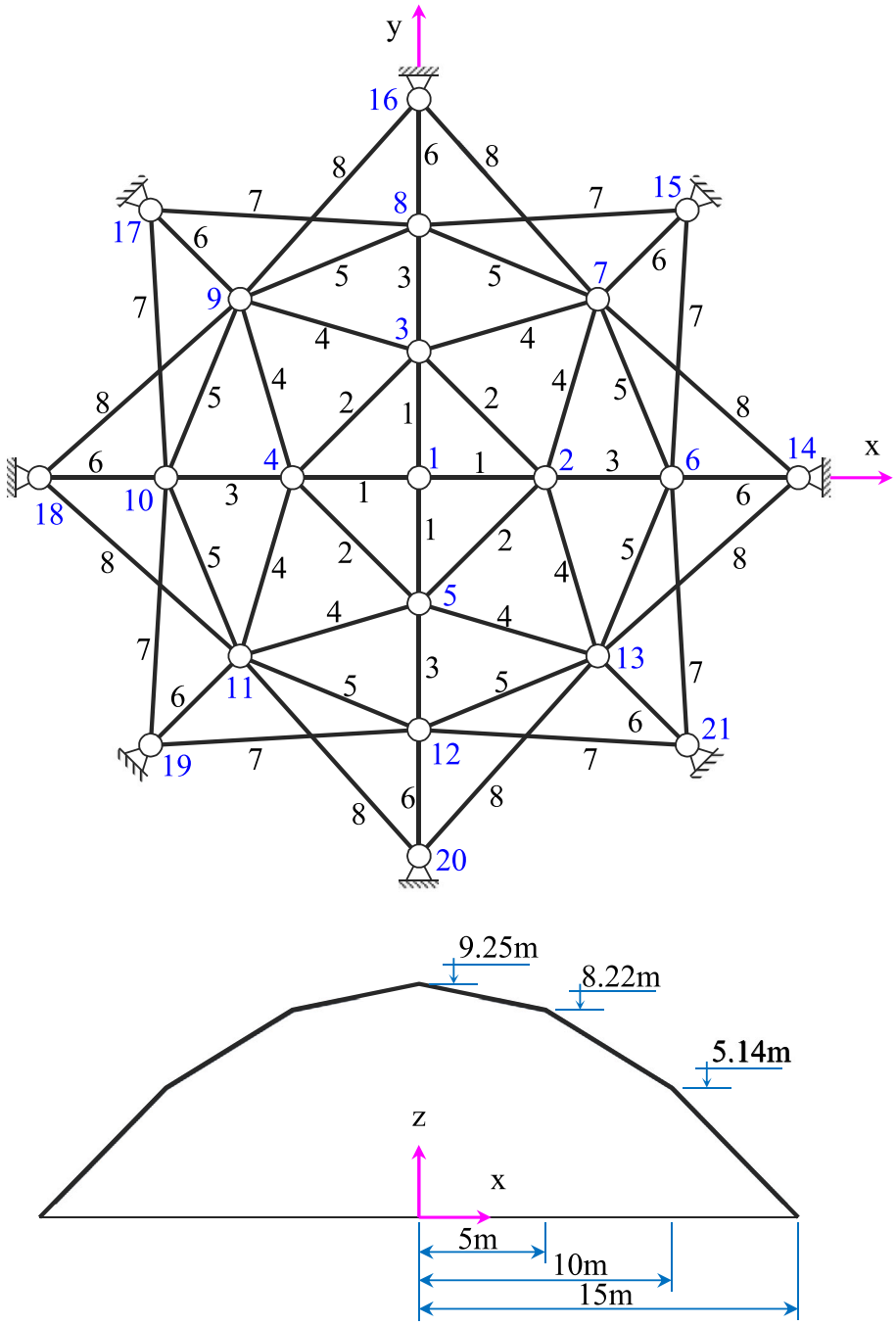

Fig. 8. Schematic of a 52-bar dome truss structure.

from the data in Table 9, the range of confidence interval (95% CI = 5,837.661 kg to 5,838.376 kg) changes for narrow and close to the best (5,836.717 kg), mean (5,838.019 kg) and worst (5,841.903 kg) weight with the small Std (0.605 kg). Next, it can be seen that the optimum weight found by Saka (Saka and Ulker, 1992) (7,587 kg) was the heaviest and violated the design constraints (−10.015mm). In addition, it is interesting that in this example, the optimum weight gained by the FEA-DE (6,504.674 kg) is also much larger than the proposed paradigm (5,836.717 kg) without violating constraints. This can be explained that the DE algorithm may trap in the local optimum. And clearly, our approach gives the best result in terms of both the optimum weight and constraints. Hence, tuning hyperparameters by the BO is an important role in improving accuracy and searching the global solution, as shown in Fig. 15. A comparison of the convergence history between the optimal network and FEA-DE is illustrated in Fig. 16. As the above examples, the proposed framework always converges much more rapidly than the FEA-DE. It only requires 200 times of nonlinear analysis, while the DE requires a large number of nonlinear analyses (13,740). Again, this demonstrates the efficiency of the self-tuning DNN for solving the optimum design problem with geometrically nonlinear behavior.

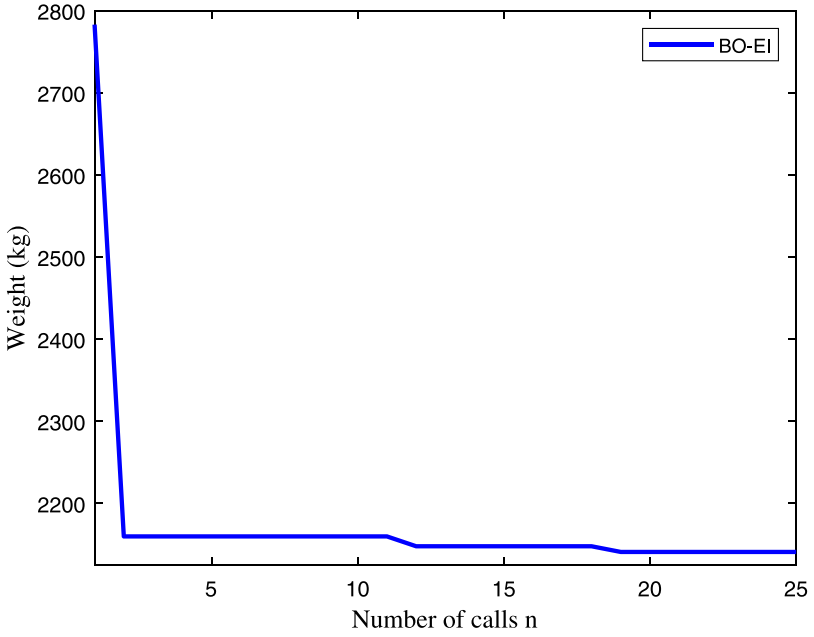

Fig. 9. The convergence history of the HPO using BO for the 52-bar dome truss structure.

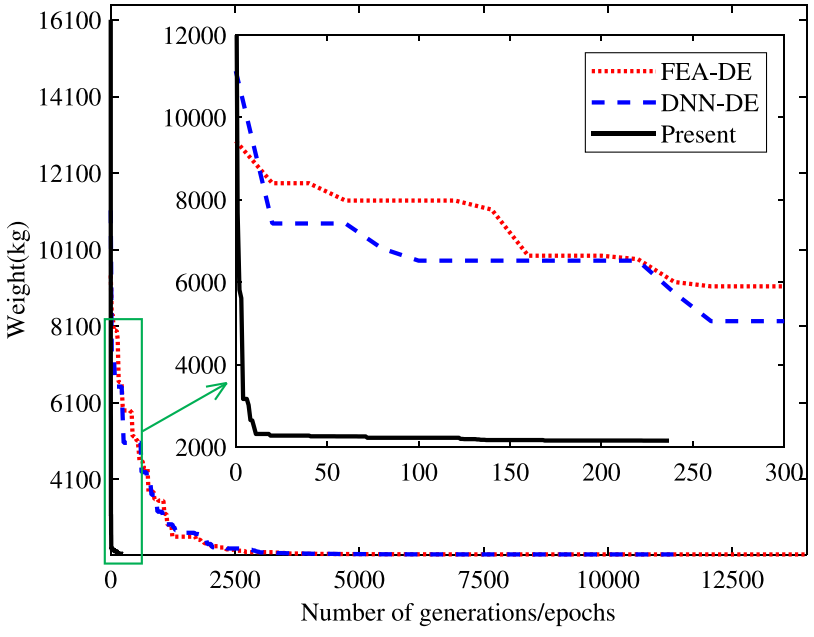

Fig. 10. The weight convergence histories of the optimal network and other works for the 52-bar dome truss.

Table 10
Comparison of optimal results for the 52-bar dome truss.

Design variables A_i (cm^2)	Saka (Saka and Ulker, 1992)	FEA-DE (Mai et al., 2021)	DNN-DE (Mai et al., 2021)	Present
1	81.820	2.000	2.000	2.004
2	22.410	2.000	2.000	2.000
3	33.580	2.000	2.000	2.004
4	14.450	2.000	2.000	2.001
5	10.640	16.672	16.753	16.692
6	25.160	17.585	17.869	17.544
7	2.000	2.519	2.301	2.509
8	2.000	2.000	2.000	2.008
Best weight (kg)	5,161	2,141.9	2,142.41	**2,141.010**

Table 11
The displacement constraints of the 52-bar dome truss.

Displacements (mm)	Saka (Saka and Ulker, 1992)	FEA-DE (Mai et al., 2021)	DNN-DE (Mai et al., 2021)	Present
w_1	−2.772	1.328	1.206	1.339
w_2	−2.826	0.720	0.742	0.743
w_6	13.045	10.000	10.000	10.000
w_7	9.491	10.000	9.648	9.987

Table 12
Comparison of optimal results for the 56-bar space truss.

Design variables A_i (cm^2)	Saka (Saka and Ulker, 1992)	FEA-DE	Present
1	7.440	8.002	7.951
2	111.020	115.427	115.609
3	5.000	5.001	5.005
4	46.460	52.761	52.642
Best weight (kg)	13,577.160	14,550.004	**14,549.691**

Table 13
The displacement constraints of the 56-bar space truss.

Displacements (mm)	Saka (Saka and Ulker, 1992)	FEA-DE	Present
u_1	31.1542	29.0021	28.9957
u_2	32.2244	30.0002	30.0000
u_3	27.0624	25.3639	25.3520
u_4	26.7943	25.1181	25.1044
u_5	24.3559	22.5559	22.5625
u_6	23.2623	21.5386	21.5389
u_7	19.9771	18.5977	18.5931
u_8	20.2158	18.8190	18.8160
u_9	14.6074	13.4029	13.4083
u_{10}	14.8078	13.5845	13.5914
u_{11}	13.9259	12.8816	12.8889
u_{12}	12.7964	11.8268	11.8277
u_{13}	7.7910	7.1061	7.1175
u_{14}	6.6797	6.0706	6.0757
u_{15}	5.4646	4.9662	4.9702
u_{16}	5.6804	5.1646	5.1701

Table 14
Comparison of optimal results of the 120 dome truss.

Design variables A_i (cm^2)	Saka (Saka and Ulker, 1992)	FEA-DE	Present
1	17.500	9.693	5.496
2	45.560	45.096	42.184
3	25.450	25.785	25.077
4	8.440	4.829	2.077
5	22.300	24.116	24.805
6	15.960	14.997	15.454
7	3.900	2.000	2.048
Best weight (kg)	7,587.000	6,504.674	**5,836.717**

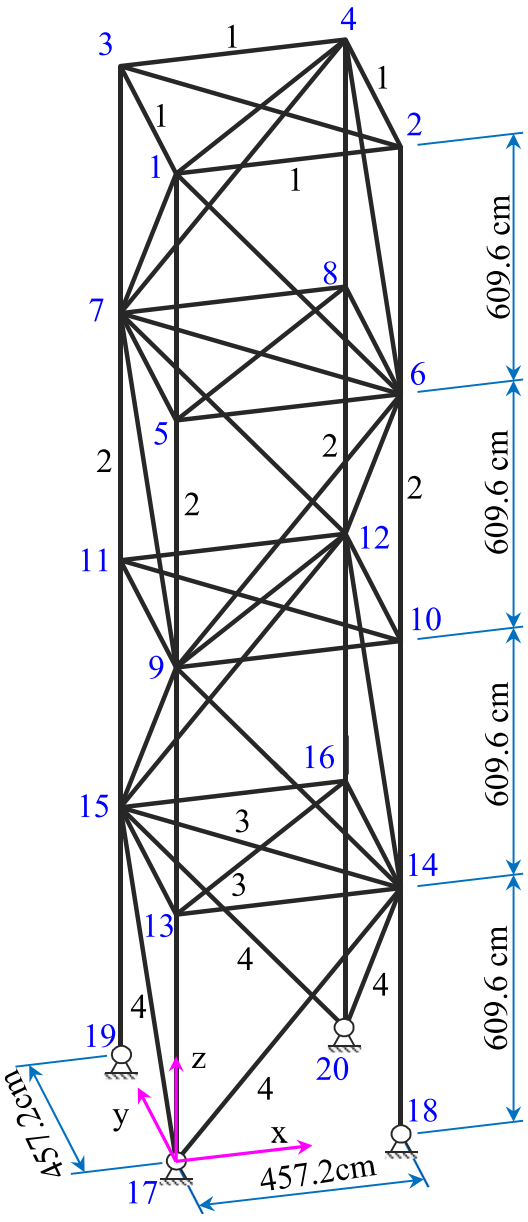

Fig. 11. A 56-bar space truss structure.

Table 15
The displacement constraints of the 120-bar dome truss.

Displacements (mm)	Saka (Saka and Ulker, 1992)	FEA-DE	Present
w_1	−7.518	−8.336	−10.000
w_5	−10.015	−9.838	−10.000
w_{19}	0.907	0.641	0.606
w_{20}	0.125	−4.401	−5.766

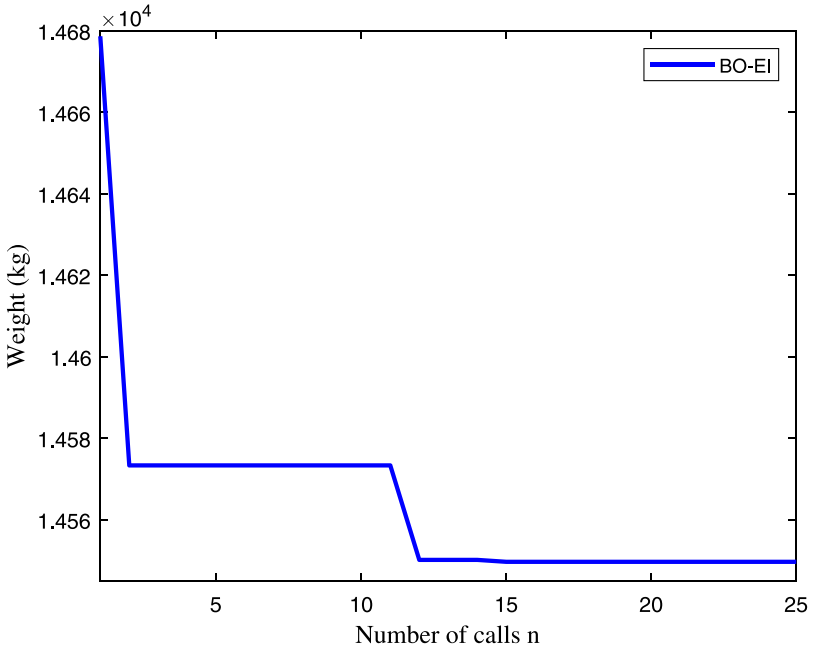

Fig. 12. The convergence history of the HPO using BO for the 56-bar space truss structure.

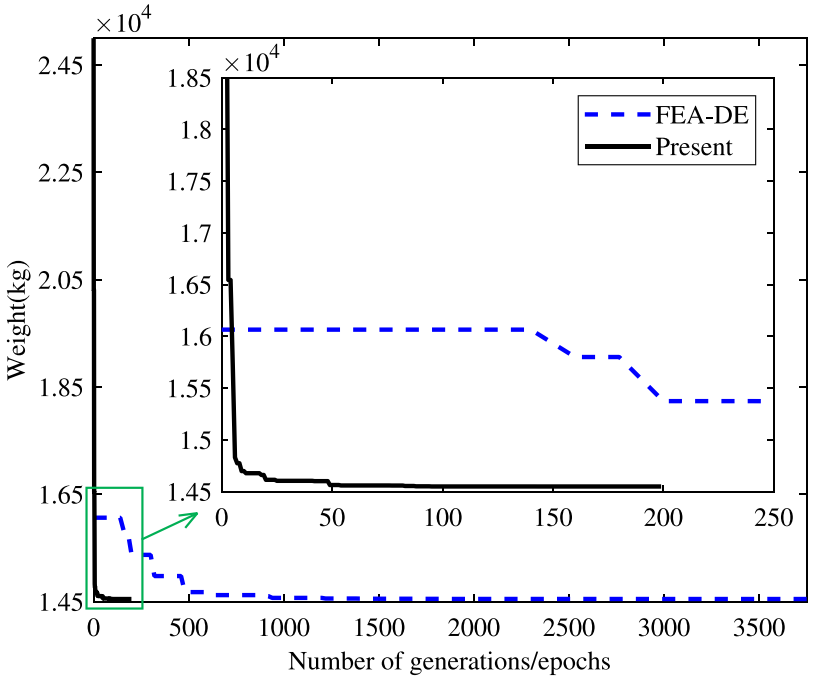

Fig. 13. The weight convergence histories of the optimal network and FEA-DE for the 56-bar space truss.

5. Conclusions

In this study, an efficient self-tuning hyperparameter DNN-based framework is developed to solve the design optimization of truss structures with geometrically nonlinear behavior. The DNN is built to parameterize the cross-sectional area of members. In addition, the BO framework is integrated with the network's training process to self-adjusting search hyperparameters. And instead of looking for the cross-sectional areas, the parameters of the network as the new design variables for structural optimization are estimated by training to minimize the loss function. Therein, the FEA is utilized to support the construction of the loss function and the network directly performs the structural optimization problem. When the training phase ends, the minimum weight of the structure is pointed out immediately. And then, BO is utilized to determine the best optimum weight corresponding the optimal hyperparameters of the network. Several numerical examples for the optimum design of geometrically nonlinear truss structure are examined to prove the reliability and efficiency of our approach. And

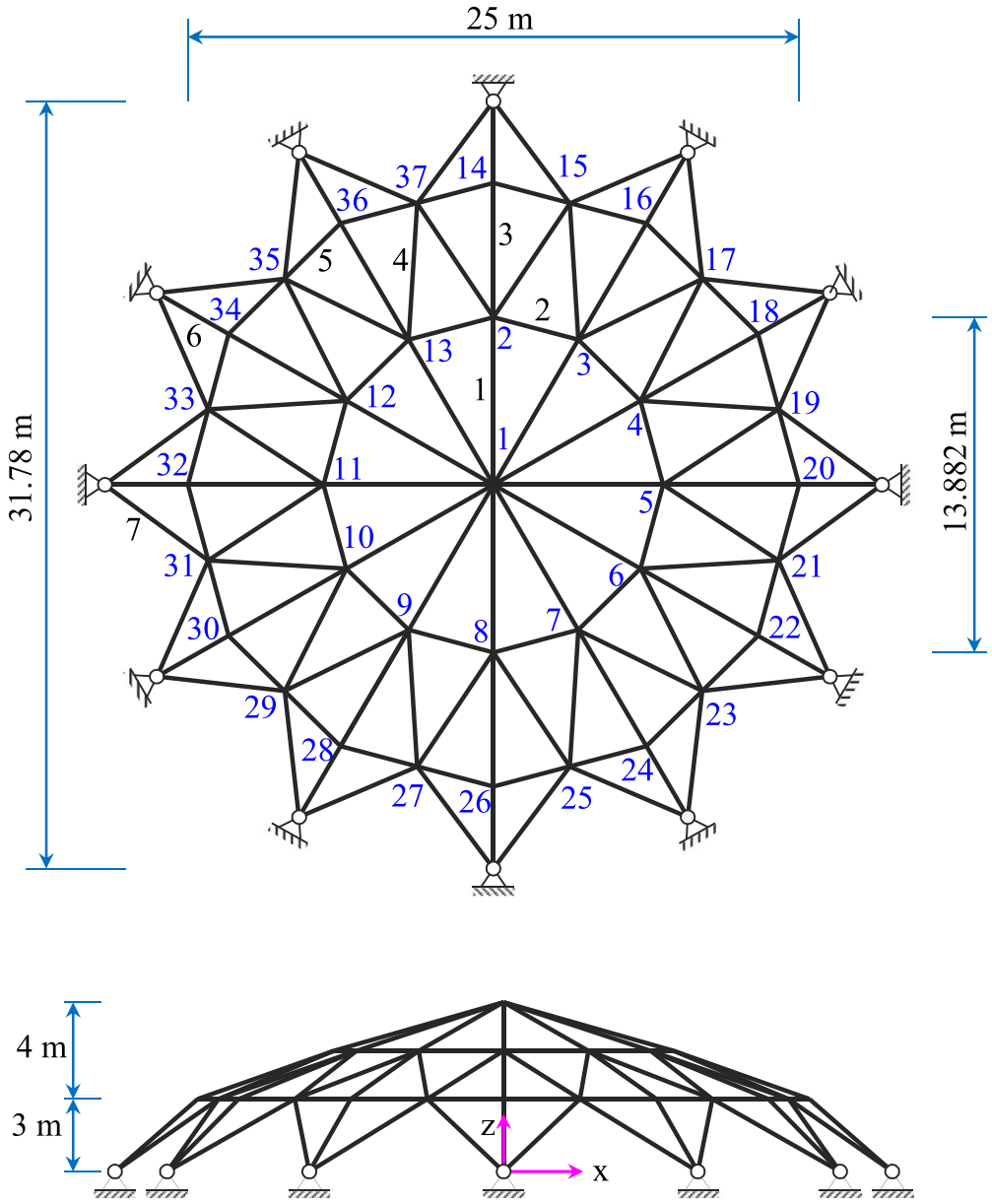

Fig. 14. 120-bar dome space truss structure.

the numerical results indicated that our model always outperforms others in terms of the quality solution and convergence rate. In specific, it is interesting that the network can automatically tune hyperparameters in the learning process to avoid being trapped in a local optimum, and one of the major strengths of this approach. In light of the above outstanding features, it is a promising alternative to solving complex problems with nonlinear behavior.

CRediT authorship contribution statement

Hau T. Mai: Conceptualization, Methodology, Software, Formal analysis, Investigation, Writing – original draft, Writing – review & editing, Visualization. **Seunghye Lee:** Writing – review, Validation. **Donghyun Kim:** Data curation, Validation. **Jaewook Lee:** Data curation, Validation. **Joowon Kang:** Data curation, Validation. **Jaehong Lee:** Conceptualization, Methodology, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The authors do not have permission to share data.

Acknowledgment

This research was supported by a grant (NRF- 2020R1A4A2002855) from NRF (National Research Foundation of Korea) funded by MEST (Ministry of Education and Science Technology) of Korean government.

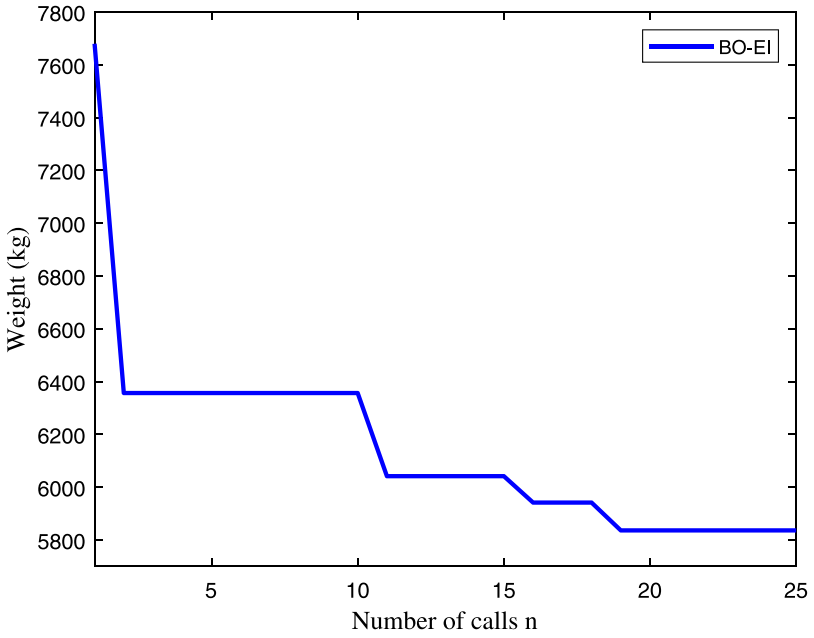

Fig. 15. The convergence histories of the HPO using BO for the 120-bar dome truss structure.

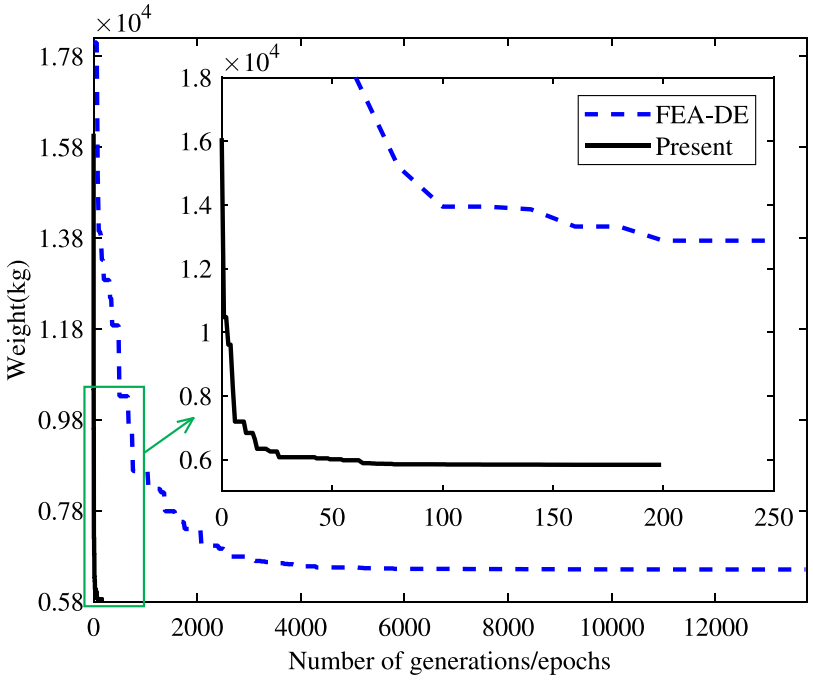

Fig. 16. The weight convergence histories of the optimal network and FEA-DE for the 120-bar dome truss.

References

Abueidda, D.W., Koric, S., Sobh, N.A., 2020. Topology optimization of 2D structures with nonlinearities using deep learning. Comput. Struct. 237, 106283.

Afshari, S.S., Enayatollahi, F., Xu, X., Liang, X., 2022. Machine learning-based methods in structural reliability analysis: A review. Reliab. Eng. Syst. Saf. 219, 108223.

Anitescu, C., Atroshchenko, E., Alajlan, N., Rabczuk, T., 2019. Artificial neural network methods for the solution of second order boundary value problems. Comput. Mater. Contin. 59 (1), 345–359.

Asali, A., Ravid, D., Shalev, H., David, L., Yogev, E., Yogev, S.S., Schonman, R., Biron-Shental, T., Miller, N., 2021. Intrahepatic cholestasis of pregnancy: Machine-learning algorithm to predict elevated bile acid based on clinical and laboratory data. Arch. Gynecol. Obstet. 304 (3), 641–647.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M.J., Leary, C., Maclaurin, D., Wanderman-Milne, S., 2020. JAX: composable transformations of Python+ NumPy programs, 2018, 4. p. 16, URL http://github.com/google/jax.

Chandrasekhar, A., Sridhara, S., Suresh, K., 2021. AuTO: A framework for automatic differentiation in Topology Optimization. Struct. Multidiscip. Optim. 64 (6), 4355–4365.

Chandrasekhar, A., Suresh, K., 2021. TOuNN: Topology Optimization using Neural Networks. Struct. Multidiscip. Optim. 63 (3), 1135–1149.

Coda, H.B., Paccola, R.R., 2014. A total-Lagrangian position-based FEM applied to physical and geometrical nonlinear dynamics of plane frames including semi-rigid connections and progressive collapse. Finite Elem. Anal. Des. 91, 1–15.

De Borst, R., Crisfield, M.A., Remmers, J.J., Verhoosel, C.V., 2012. Nonlinear Finite Element Analysis of Solids and Structures. John Wiley & Sons.

El-Sayed, M.E., Ridgely, B.J., Sandgren, E., 1989. Nonlinear structural optimization using goal programming. Comput. Struct. 32 (1), 69–73.

Hajela, P., Berke, L., 1991a. Neural network based decomposition in optimal structural synthesis. Comput. Syst. Eng. 2 (5–6), 473–481.
Hajela, P., Berke, L., 1991b. Neurobiological computational models in structural analysis and design. Comput. Struct. 41 (4), 657–667.
Hasançebi, O., 2008. Adaptive evolution strategies in structural optimization: Enhancing their computational performance with applications to large-scale structures. Comput. Struct. 86 (1–2), 119–132.
Hertel, L., Collado, J., Sadowski, P., Ott, J., Baldi, P., 2020. Sherpa: Robust hyperparameter optimization for machine learning. SoftwareX 12, 100591.
Hoyer, S., Sohl-Dickstein, J., Greydanus, S., 2019. Neural reparameterization improves structural optimization. arXiv preprint arXiv:1909.04240.
Hrinda, G.A., Nguyen, D.T., 2008. Optimization of stability-constrained geometrically nonlinear shallow trusses using an arc length sparse method with a strain energy density approach. Finite Elem. Anal. Des. 44 (15), 933–950.
Jia, D.-W., Wu, Z.-Y., 2022. A Laplace asymptotic integral-based reliability analysis method combined with artificial neural network. Appl. Math. Model..
Jones, D.R., Schonlau, M., Welch, W.J., 1998. Efficient global optimization of expensive black-box functions. J. Global Optim. 13 (4), 455–492.
Kadapa, C., 2021. A simple extrapolated predictor for overcoming the starting and tracking issues in the arc-length method for nonlinear structural mechanics. Eng. Struct. 234, 111755.
Kameshki, E., Saka, M., 2001. Optimum design of nonlinear steel frames with semi-rigid connections using a genetic algorithm. Comput. Struct. 79 (17), 1593–1604.
Kameshki, E., Saka, M., 2007. Optimum geometry design of nonlinear braced domes using genetic algorithm. Comput. Struct. 85 (1–2), 71–79.
Kaveh, A., Rahami, H., 2006. Nonlinear analysis and optimal design of structures via force method and genetic algorithm. Comput. Struct. 84 (12), 770–778.
Khot, N., 1983. Nonlinear analysis of optimized structure with constraints on systemstability. AIAA J. 21 (8), 1181–1186.
Khot, N., Kamat, M., 1985. Minimum weight design of truss structures with geometric nonlinear behavior. AIAA J. 23 (1), 139–144.
Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
Korenciak, M., 2018. Go Game Move Prediction Using Convolutional Neural Network. Jyväskylän ammattikorkeakoulu.
Lee, S., Kim, H., Lieu, Q.X., Lee, J., 2020. CNN-based image recognition for topology optimization. Knowl.-Based Syst. 198, 105887.
Lee, S., Park, S., Kim, T., Lieu, Q.X., Lee, J., 2021a. Damage quantification in truss structures by limited sensor-based surrogate model. Appl. Acoust. 172, 107547.
Lee, S., Vo, T.P., Thai, H.-T., Lee, J., Patel, V., 2021b. Strength prediction of concrete-filled steel tubular columns using Categorical Gradient Boosting algorithm. Eng. Struct. 238, 112109.
Li, W., Bazant, M.Z., Zhu, J., 2021. A physics-guided neural network framework for elastic plates: Comparison of governing equations-based and energy-based approaches. Comput. Methods Appl. Mech. Engrg. 383, 113933.
Li, B., Huang, C., Li, X., Zheng, S., Hong, J., 2019. Non-iterative structural topology optimization using deep learning. Comput. Aided Des. 115, 172–180.
Lieu, Q.X., Nguyen, K.T., Dang, K.D., Lee, S., Kang, J., Lee, J., 2022. An adaptive surrogate model to structural reliability analysis using deep neural network. Expert Syst. Appl. 189, 116104.
Mai, H.T., Kang, J., Lee, J., 2021. A machine learning-based surrogate model for optimization of truss structures with geometrically nonlinear behavior. Finite Elem. Anal. Des. 196, 103572.
Mai, H.T., Lieu, Q.X., Kang, J., Lee, J., 2022a. A novel deep unsupervised learning-based framework for optimization of truss structures. Eng. Comput. 1–24.
Mai, H.T., Lieu, Q.X., Kang, J., Lee, J., 2022c. A robust unsupervised neural network framework for geometrically nonlinear analysis of inelastic truss structures. Appl. Math. Model..
Missoum, S., Gürdal, Z., Gu, W., 2002. Optimization of nonlinear trusses using a displacement-based approach. Struct. Multidiscip. Optim. 23 (3), 214–221.
Nguyen-Thanh, V.M., Zhuang, X., Rabczuk, T., 2020. A deep energy method for finite deformation hyperelasticity. Eur. J. Mech. A Solids 80, 103874.
Pezeshk, S., Camp, C., Chen, D., 2000. Design of nonlinear framed structures using genetic optimization. J. Struct. Eng. 126 (3), 382–388.
Ramasamy, J., Rajasekaran, S., 1996. Artificial neural network and genetic algorithm for the design optimizaton of industrial roofs—A comparison. Comput. Struct. 58 (4), 747–755.
Riks, E., 1979. An incremental approach to the solution of snapping and buckling problems. Int. J. Solids Struct. 15 (7), 529–551.
Saka, M., Ulker, M., 1992. Optimum design of geometrically nonlinear space trusses. Comput. Struct. 42 (3), 289–299.
Sonmez, M., 2011. Artificial Bee Colony algorithm for optimization of truss structures. Appl. Soft Comput. 11 (2), 2406–2418.
Srinivasan, S., Saghir, M.Z., 2013. Modeling of thermotransport phenomenon in metal alloys using artificial neural networks. Appl. Math. Model. 37 (5), 2850–2869.
Thai, H.-T., Nguyen, T.-K., Lee, S., Patel, V.I., Vo, T.P., 2020. Review of nonlinear analysis and modeling of steel and composite structures. Int. J. Struct. Stab. Dyn. 20 (04), 2030003.
Trinh, D.T., Lee, S., Kang, J., Lee, J., 2022. Force density-informed neural network for prestress design of tensegrity structures with multiple self-stress modes. Eur. J. Mech. A Solids 104584.
Truong, T.T., Dinh-Cong, D., Lee, J., Nguyen-Thoi, T., 2020a. An effective Deep Feedforward Neural Networks (DFNN) method for damage identification of truss structures using noisy incomplete modal data. J. Build. Eng. 30, 101244.
Truong, T.T., Lee, S., Lee, J., 2020b. An artificial neural network-differential evolution approach for optimization of bidirectional functionally graded beams. Compos. Struct. 233, 111517.
Truong, T.T., Lee, J., Nguyen-Thoi, T., 2022. Joint damage detection of structures with noisy data by an effective deep learning framework using autoencoder-convolutional gated recurrent unit. Ocean Eng. 243, 110142.
Vo, T.P., Lee, J., 2011. Geometrical nonlinear analysis of thin-walled composite beams using finite element method based on first order shear deformation theory. Arch. Appl. Mech. 81 (4), 419–435.
Wempner, G.A., 1971. Discrete approximations related to nonlinear theories of solids. Int. J. Solids Struct. 7 (11), 1581–1599.
Zehnder, J., Li, Y., Coros, S., Thomaszewski, B., 2021. NTopo: Mesh-free topology optimization using implicit neural representations. Adv. Neural Inf. Process. Syst. 34.