Finite Elements in Analysis and Design 196 (2021) 103572

Contents lists available at ScienceDirect

Finite Elements in Analysis and Design

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

A machine learning-based surrogate model for optimization of truss structures with geometrically nonlinear behavior

Hau T. Mai [a], Joowon Kang [b], Jaehong Lee [a,*]

[a] Deep Learning Architectural Research Center, Sejong University, 209 Neungdong-ro, Gwangjin-gu, Seoul 05006, Republic of Korea
[b] School of Architecture, Yeungnam University, Gyeongsan, Gyeongbuk, Republic of Korea

ARTICLE INFO

Keywords:
Surrogate model
Deep neural network
Neural network
Machine learning
Geometric nonlinear
Truss optimization

ABSTRACT

Design optimization of geometrically nonlinear structures is well known as a computationally expensive problem by using incremental-iterative solution techniques. To handle the problem effectively the optimization algorithm needs to ensure that the trade-off between the computational time and the quality of the solution is found. In this study, a deep neural network (DNN)-based surrogate model which integrates with differential evolution (DE) algorithm is developed and applied for solving the optimum design problem of geometrically nonlinear space truss under displacement constraints and refer to the approach as DNN-DE. Accordingly, this surrogate model, also is known as a deep neural network, is established to replace conventional finite element analyses (FEAs). Each dataset is created based on FEA which employs the total Lagrangian formulation and the arc-length procedure. Several numerical examples are given to demonstrate the efficiency and validity of the proposed paradigm. These results indicate that the proposed approach not only reduces the computational cost dramatically but also guarantees convergence.

1. Introduction

Design optimization of structures with linear response has attracted the interest of many researchers during the last decade. However, when most structures exhibit significant nonlinear responses in one way or another, nonlinearities have to be considered in the analysis to evaluate their influence [1,2]. In particular, the design of light and slender structures such as arches, bars, thin-walled constructions, etc., are complicated due to the nonlinear geometric effects associated with large deformations [3–8]. Therefore, these structural optimization methods require geometrically nonlinear analysis to obtain their behaviour under the applied loads [7].

Some researchers have reported a variety of algorithms for solving nonlinear structural optimization problems and categorized them into three distinct approaches. The first one is the gradient-based approach which relies on gradient information to determine the optimum solution. For instance, Khot et al. [9,10] developed an algorithm based on an optimality criterion to minimize the weight of the space truss with geometrically nonlinear behavior. Haririan et al. [11] has tried to address this type of problem by using the computer program ADINA associated with design sensitivity analysis. Besides, a combining model based on the nonlinear and linear goal programming was delivered by El-Sayed et al. [12]. To save the computational cost, Saka and Ulker [7] developed a coupling methodology between the optimality criteria approach and the nonlinear analysis technique. An enlargement of the displacement-based optimization procedure to speed up the analysis and to enhance accuracy was released by S. Missoum et al. [8]. More recently, Shin et al. [13] has proposed an approach that uses the equivalent loads to reduce the number of nonlinear analyses. Hrinda et al. [14] has successfully developed a new algorithm by integrating the arc-length method and a design-variable update scheme. Nonetheless, all gradient-based approaches still have some disadvantages such as depending on the initial value selection, converging to local minima [13]. Next, the non-gradient-based algorithm is characterized as the second approach [15–18]. Although algorithms for this kind of processing have the ability to find the near-global optimum solution, they require a huge number of evaluation function for reaching a good solution [19,20]. The final approach, referred to as surrogate-assisted optimization, has received much attention due to the computationally expensive problems. These methods have proved the effectiveness

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

https://doi.org/10.1016/j.finel.2021.103572
Received 8 September 2020; Received in revised form 19 March 2021; Accepted 25 March 2021
Available online 18 June 2021
0168-874X/

for solving small-dimensional, expensive, black-box global optimization problems [21–25]. However, two major existing drawbacks of this approach are the curse of dimensionality and the small feasible region [21].

Machine learning (ML) has been successfully applied to a wide range of fields to assist in decision-making [26], such as image recognition, computer vision, healthcare, self-driving systems, et. Among ML models, deep neural network (DNN) has attracted remarkable attention in computational mechanics [27–29]. It contains multiple hidden layers between the input and output layers and is able to model complicated nonlinear relationship between inputs and outputs. In recent years, a number of studies have proved the effectiveness in applying DNN to structural analysis [30–32] and optimization [33–36]. Hajela and Berke [34,37,38] were among the first people who used neural networks (NNs) for replacing structural analysis steps in the optimization process. A surrogate of FEA based on DNN model is developed to estimate the stress distributions of the human thoracic aorta by Liang et al. [39]. Besides, Lee et al. [40] implemented a survey to assess the efficiency and accuracy of DNN in structural analysis. Also, Tam et al. [41] applied deep feedforward neural networks to detect the location of damaged elements in truss structures. Recently, Chandrasekhar and Suresh [42] have proposed a conventional neural network to directly execute topology optimization. Additionally, Kollmann et al. [36] utilized a convolutional neural network to obtain optimal metamaterial designs. In another work, Zhou et al. [43] developed a data-driven model to accelerate the truss topology optimization process. A combination of the generative adversarial network and the super-resolution generative adversarial network to near-optimal structural topology of heat transfer problems were examined by Li et al. [44]. Also, Yildiz et al. [45] proposed an integrated procedure, which consists of neural network and feature-based approaches, predicts optimized designs. In general, the applications of ML models have yielded satisfactory results in structural analyses and optimization designs. Although most of the studies in the literature apply ML algorithms to accelerate the design optimization process, these applications are limited to linear elastic materials and small deformations, with linear optimization constraints. In most of practical problems, we have to due with a lot of structures that have nonlinear characteristics. Indeed, these issues are much more complicated than linear problems because of a huge requirement of numerical iterations. Furthermore, the optimization of nonlinear structures demands a lot of computing effort. However, the applications of ML models in solving the optimization problem with geometrically nonlinear behavior have not been actually studied much.

In this paper, an efficient integration of DNN model with DE algorithm is developed to resolve the optimization of truss structures with geometrically nonlinear behavior. In this approach, the constructed DNN model serves an surrogate model to directly estimate the displacement at nodes without utilizing the performance of FEA. For this to be realized, the DNN model must be trained first to learn a mapping between the input and output quantities. In which, the set of input variables is generated relies on Latin Hypercube Sampling (LHS) [46], the set of output responses is obtained from the geometrically nonlinear analysis by using arc-length method. Several optimizers and activation functions of the DNN model are surveyed to achieve a better performance of the network. Meanwhile, K-fold cross-validation and dropout techniques are considered to train the models and avoid the over-fitting problem. Additionally, mini-batch is applied to speed up the training process. Finally, DE as an optimizer is used to find the global solution for a new optimization problem involving the original objective and constraints obtained from DNN. The optimum design problem of geometrically nonlinear space truss under displacement constraints with the linear elastic material will be discussed without considering the stability. The effectiveness and reliability of the present method are also investigated through three numerical examples.

The rest of this article is organized as follows: In Section 2, a brief description of the governing equation and arc-length technique is given.

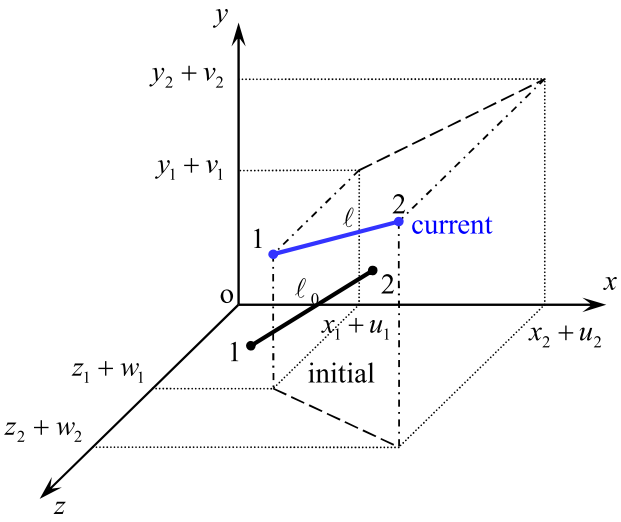

Fig. 1. Kinematics of Total Lagrangian space truss elements.

The statement of truss layout and size optimization problem with the integration model is presented in Section 3. Section 4 provides the extension of DNN-DE in optimization of truss structures with geometrically nonlinear behavior. Numerical examples are illustrated in Section 5. Conclusions, challenges, and opportunities are outlined in Section 6.

2. Geometrically nonlinear analysis of space truss

The primary objective of nonlinear analysis is to search for the equilibrium configuration of structures, which is under the action of external loads. In this section, the resolution of the geometrically nonlinear problem is described by using the Total Lagrangian (TL) kinematic description [47–49] and linearized arc-length technique (Riks-Wempner algorithm) [6,50,51].

The first one provides briefly the derivation of internal force vector and tangent stiffness matrix. The interested readers may refer to Ref. [47] for a complete treatment of the problem. Let us consider a space truss element in its initial and current configurations shown in Fig. 1. The coordinates (x_1, y_1, z_1) and (x_2, y_2, z_2) represent the initial configuration at the ends of the element, respectively. So, the initial length of member can be written as

$$\ell_0 = \sqrt{\ell_{ox}^2 + \ell_{oy}^2 + \ell_{oz}^2}, \tag{1}$$

in which $\ell_0^x = x_2 - x_1$, $\ell_0^y = y_2 - y_1$, $\ell_0^z = z_2 - z_1$

Let (u_1, v_1, w_1) and (u_2, v_2, w_2) correspond to displacements at the two ends of the current configuration. As a consequence, the current length of member is expressed as

$$\ell = \sqrt{\ell_x^2 + \ell_y^2 + \ell_z^2}, \tag{2}$$

where $\ell_x = \ell_{0x} + u_2 - u_1, \ell_y = \ell_{0y} + v_2 - v_1, \ell_z = \ell_{0z} + w_2 - w_1$

According to Crisfield et al. [47], Green's strain is defined as

$$\varepsilon = \frac{\ell^2 - \ell_0^2}{2\ell_0^2}. \tag{3}$$

The internal force vector is given by

$$\mathbf{F}_{el} = EA\varepsilon\ell_0\mathbf{B}^T, \tag{4}$$

where E, and A are the elastic modulus and the cross-section area, respectively; the matrix $\mathbf{B}$ is defined as follows

$$\mathbf{B} = \frac{1}{\ell_0^2}\begin{bmatrix} -\ell_x & -\ell_y & -\ell_z & \ell_x & \ell_y & \ell_z \end{bmatrix}. \tag{5}$$

The tangent stiffness matrix is obtained by differentiating the internal force vector with respect to the nodal displacements.

$$\mathbf{K}_{el} = \mathbf{K}_L + \mathbf{K}_G, \tag{6}$$

where $\mathbf{K}_L$ and $\mathbf{K}_G$ are the material stiffness matrix and geometric stiffness matrix, respectively, as follows

$$\mathbf{K}_L = EA\ell_0\mathbf{B}^T\mathbf{B}, \tag{7}$$

$$\mathbf{K}_G = \frac{AE\varepsilon}{\ell_0}\begin{bmatrix} \mathbf{I}_3 & -\mathbf{I}_3 \\ -\mathbf{I}_3 & \mathbf{I}_3 \end{bmatrix}, \tag{8}$$

in which $\mathbf{I}_3$ is the identity matrix of order 3.

According to Crisfield [47], the nonlinear system is given by

$$\mathbf{g}(\mathbf{u}, \lambda) = \mathbf{f}(\mathbf{u}) - \lambda\mathbf{q} = \mathbf{0}, \tag{9a}$$

$$c(\mathbf{u}, \lambda) = \left(\Delta\mathbf{u}^T\Delta\mathbf{u} + \Delta\lambda^2\psi^2\mathbf{q}^T\mathbf{q}\right) - \Delta l^2 = 0, \tag{9b}$$

where $\mathbf{u}$ is the displacement vector; $\mathbf{f}$ denotes the global internal force vector; $\mathbf{q}$ is the fixed total load vector; $\mathbf{g}$ indicates the residual load vector; λ refers to the load parameter; ψ is the scaling parameter, and Δl refers to the arc-length increment.

The second part presents a combination of incremental and iterative procedures, which is used to solve Eq. (9). Assume the existence solution $({}^t\mathbf{u}, {}^t\lambda)$ in the tth load step, the total parameters of load $\lambda^{(k)}$ and nodal displacments $\mathbf{u}^{(k)}$ at the kth iteration are calculated by [47].

$$\lambda^{(k)} = {}^t\lambda + \Delta\lambda^{(k)}, \tag{10}$$

$$\mathbf{u}^{(k)} = {}^t\mathbf{u} + \Delta\mathbf{u}^{(k)}, \tag{11}$$

where $\Delta\lambda^{(k)}$ and $\Delta\mathbf{u}^{(k)}$ are the load increment and the displacement increment, respectively. They are determined by the following equation

$$\begin{bmatrix} \Delta\mathbf{u}^{(k)} \\ \Delta\lambda^{(k)} \end{bmatrix} = \begin{bmatrix} \Delta\mathbf{u}^{(k-1)} \\ \Delta\lambda^{(k-1)} \end{bmatrix} + \begin{bmatrix} \delta\mathbf{u}^{(k)} \\ \delta\lambda^{(k)} \end{bmatrix}, \tag{12}$$

where $\delta\mathbf{u}^{(k)}$ and $\delta\lambda^{(k)}$ are called the sub-incremental displacements vector and the sub-incremental load, respectively. Following Crisfield, Riks and Wempner [6,47,50,51], these values can be obtained by combining Newton-Raphson's method with Eq. (9) and the linearized version of the arc-length method is shown in Fig. 2, as follows

$$\delta\lambda^{(k)} = -\frac{\Delta\mathbf{u}^{(0)T}\delta\mathbf{u}_g^{(k)}}{\Delta\mathbf{u}^{(0)T}\delta\mathbf{u}_q^{(k)}}, \tag{13}$$

$$\delta\mathbf{u}^{(k)} = \delta\mathbf{u}_g^{(k)} + \delta\lambda^{(k)}\delta\mathbf{u}_q^{(k)}, \tag{14}$$

where $\delta\mathbf{u}_g^{(k)}$, $\delta\mathbf{u}_q^{(k)}$ and $\Delta\mathbf{u}^{(0)}$ are obtained by

$$\delta\mathbf{u}_g^{(k)} = -\left[\mathbf{K}\left(\mathbf{u}^{(k-1)}\right)\right]^{-1}\mathbf{g}^{(k-1)}, \tag{15}$$

$$\delta\mathbf{u}_q^{(k)} = \left[\mathbf{K}\left(\mathbf{u}^{(\mathbf{k-1})}\right)\right]^{-1}\mathbf{q}, \tag{16}$$

$$\Delta\mathbf{u}^{(0)} = \Delta\lambda^{(0)t}\delta\mathbf{u}_q, \tag{17}$$

in which $\mathbf{K}$ is the global stiffness matrix which is constructed by assembling individual element stiffness matrics $\mathbf{K}_{el}$

Riks and Wempner [6,51] first proposed the arc-length method, which is illustrated in Fig. 3. It is easily be seen that the iterative path at each step is always orthogonal to the initial tangent. The initial load increment $\Delta\lambda^{(0)}$ is indicated by

$$\Delta\lambda^{(0)} = \frac{\Delta l}{\|{}^t\delta\mathbf{u}_q\|}. \tag{18}$$

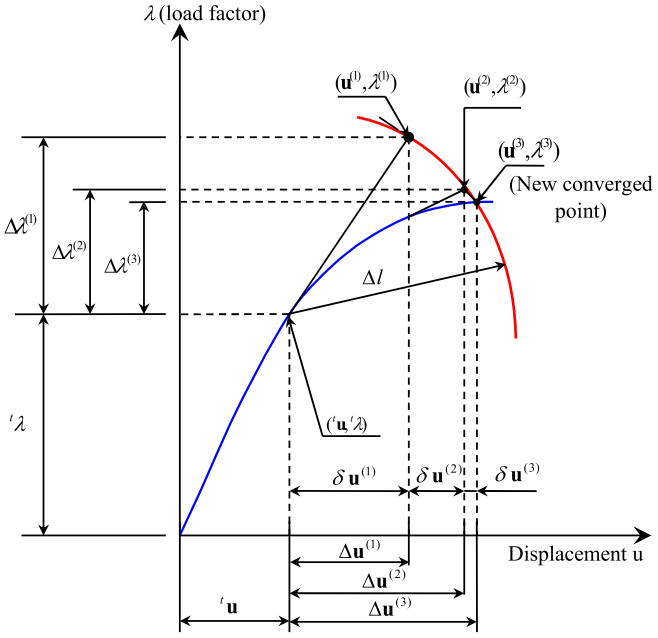

Fig. 2. The linearized arc-length method for single degree of freedom.

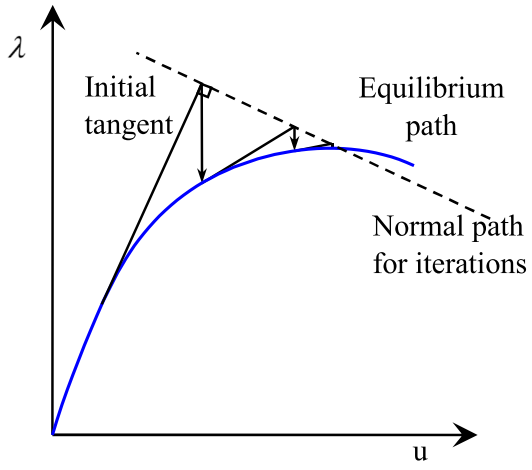

Fig. 3. The Arc-length (Riks and Wempner) using constant line search.

Note that the arc-length increment Δl in the current time step can be used as the control parameter as follows

$$\Delta l = {}^t\Delta l\sqrt{\frac{2}{{}^tk}}, \tag{19}$$

where ${}^t\Delta l$ is the increment of arc-length in the previous time step, and ${}^tk$ is the number of iterations needed to converge in the previous time step.

In dealing with the nonlinear problem, the convergence criterion is given by

$$\left\|\mathbf{g}^{(k)}\right\| \le tol.\left\|\mathbf{q}\right\|, \tag{20}$$

where the tolerance *tol* is provided by the user. The algorithm for geometrically nonlinear analysis using the linearized arc-length technique is summarized in several steps in Algorithm. 1.

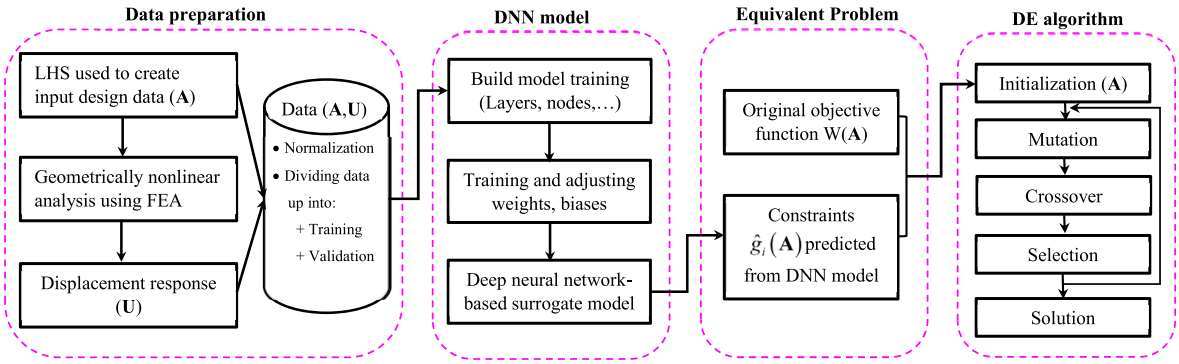

Fig. 4. Flowchart depicting the deep neural network-based surrogate model for optimization.

3. Optimum design problem

For the optimization of geometrically nonlinear truss structures under displacement constraints, the goal is to determine the member cross-sectional areas that will minimize the weight of the structure. In which, the area of members is considered the continuous design variables, as well as a predefined feasible region [7,8]. The general formulation can be represented as follows

$$\begin{aligned} &\text{Minimize} \ : \quad W(\mathbf{A}) = \sum_{i=1}^{m} A_i \rho_i \ell_i, \\ &\text{Subjectto} : \quad g_j(\mathbf{A}) = \frac{u_j}{[u]_j} - 1 \le 0, \quad j = 1, 2, \ldots, p, \\ &\qquad\qquad A_{i,\min} \le A_i \le A_{i,\max} \qquad i = 1, 2, \ldots, m, \end{aligned} \tag{21}$$

where $W(.), g(.)$ are the objective and constraint functions; A_i, ρ_i, ℓ_i are the cross-sectional area, the material density, and the length of the ith member, respectively; m is the total number of members of the structure; u_j and $[u]_j$ denote the displacement and the allowable deflection of the jth joint; p is the number of constrained displacements; $A_{i,\min}$ and $A_{i,\max}$ are the lower and upper bounds on the design variable A_i. We note that the displacement constraint value achieved at the maximum load factor of 1.0, and the stress constraints are not considered in this study.

In order to solve the constrained optimization problem, two of the most popular approaches are evolutionary and gradient-based algorithms. According to Eq. (21), this problem is referred to as the grey-box model [52], which contains the computationally cheap objective function and all computationally expensive constraints [52–54]. So the above algorithms encountered some difficulties in identifying the near global optimal solution. Consequently, we draw a deep neural network-based surrogate model that is used to learn and replace the constraint functions. Once we obtain the result after the training, DE algorithm will apply to handle an equivalent optimization problem that contains an original objective and the approximate constraint functions.

4. DNN-based surrogate model for optimization

In this section, the schematic illustration of the integration model, as shown in Fig. 4, gives an overall view of the presented study, which consists of four main blocks as below:

(i) LHS technique is employed to generate a number of samples (**A**) and then the displacement responses (**U**) are collected through the analysis phase. The input values of the dataset are normalized before training.
(ii) Build a DNN model and training, then take the well trained network as a surrogate model of the constraints.
(iii) An equivalent optimization problem is constructed based on the original objective function and the DNN model.
(iv) The DE algorithm is introduced to resolve the optimization problem.

4.1. Data preparation

First of all, data-collection plays a vital role in the performance of the DNN. In this study, cross-sectional areas of truss members are used as the input data and the output data are defined as displacements of nodes. The LHS technique [46] is employed to generate samples that allow covering the full design space. Only the cross-sectional areas are adopted as design variables and the other conditions of the geometric nonlinear problem are fixed. The geometric nonlinear problem, which has a specific condition, is implemented to collect the output values corresponding to samples by using the arc-length technique.

The input and output data have a different range of variations. As mentioned by Berke et al. [38] and Arpat et al. [55], the input values of the dataset are normalized to the interval of [0.1, 0.9] due to the nature of the activation function. On the other hand, the normalization process makes the statistical distribution of each input data roughly uniform. It has a significant influence on the data-collection process to be appropriate for training. There are many different types of normalization methods usually used to rescale data. In this work, the min-max scaling method is applied to normalize the original data X into the interval range [a, b].

$$\overline{X} = a + \frac{(X - X_{\min})(b - a)}{(X_{\max} - X_{\min})}. \tag{22}$$

The DNN model can get sufficiently good approximations when the amount of data are reasonable. The experiments were also conducted on the 30-bar dome truss to investigate the effect of the size of dataset. 10-fold cross-validation was applied to determine the errors for training and validation sets in [4-335-335-335-335-2] architecture. ReLu is applied to all the hidden layers, while in the output layer linear function is used. The network is trained by Adam optimizer, with the dropout ratio 0.2, learning rate 0.001, batch-size of 32 and 1000 epochs. Mean square error (MSE) and root mean square error (RMSE) are reported in Table 1. It is easily seen that the errors of the model with 1320 samples is smallest among all datasets. Fig. 5 show the best loss obtained from four different sizes of the dataset. The first two models are obvious difference the loss of training and validation. It is shown that the size of training dataset is not enough. When the number of samples increase to 1320, the loss values of both training and validation shares many similar characteristics and the network stable, converge after 400th epoch.

4.2. Deep neural network

The multi-layer neural network is a set of mathematical relationships between the inputs and outputs through a training process. The network learns through analyzing the training data by adjusting weights

Algorithm 1 Geometrically nonlinear analysis using the linearized arc-length technique.

Input: Geometric parameters, material properties, boundary conditions, parameters control

Output: Displacements, stress

Set $u = 0,\ \Delta u = 0,\ \lambda = 0$ for the initial configuration $(t = 0)$
for $t = 1$ *to* $n_{\max}$ **do**
  Calculate $\mathbf{g},\ \Delta\mathbf{u},\ \Delta\lambda\ (k = 0)$ by Eq. (9a), Eq. (17), and Eq. (18)
  for $k = 1$ *to* $i_{\max}$ **do**
    Compute the sub-incremental displacement, load $\left(\delta\mathbf{u}^{(k)},\ \delta\lambda^{(k)}\right)$ at the k^{th} iteration by Eq. (13) and Eq. (14)
    Update $\left(\Delta\mathbf{u}^{(k)},\ \Delta\lambda^{(k)}\right)$ displacement and load increment at the k^{th} iteration by Eq. (12)
    Update the total parameters of load and displacements $\left(\lambda^{(k)},\ \mathbf{u}^{(k)}\right)$ at the k^{th} iteration by Eq. (10) and Eq. (11)
    Evaluate the residual force vector $\mathbf{g}\left(\lambda^{(k)},\ \mathbf{u}^{(k)}\right)$ at the k^{th} iteration by Eq. (9a)
    if $\|\mathbf{g}\| \le tol.\ \|\mathbf{q}\|$ **then**
      break and go on for the next step else then return to step 5
  $t = t + 1$, update the displacement vector ${}^{t}\mathbf{u}$, load level parameter ${}^{t}\lambda$, and the arc-length increment Δl at the current time step
  Return to step 3

Table 1
MSE and RMSE values with different size of dataset.

Size Data	MSE(10^{-2})		RMSE(%)	
	Training	Validation	Training	Validation
700	3.726	4.101	19.302	10.270
900	0.237	0.255	4.857	5.045
1100	0.189	0.200	4.332	4.469
1320	0.059	0.051	2.429	2.273

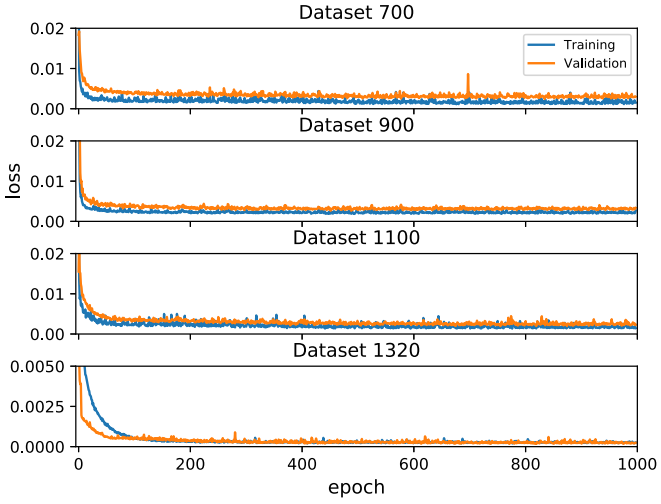

Fig. 5. The best convergence history of loss function with different sizes of data.

and biases [56]. A neural network with one hidden layer is called the shallow neural network, whereas a neural net with two or more hidden layers is known as a deep neural network. In this study, a DNN model is established to predict the deflection of the truss structure with geometrically nonlinear behavior. DNN architecture with full connected layer is depicted in Fig. 6. There are an input layer, an output layer, and $(N - 1)$ hidden layers. Each layer consists of a number of neurons (or units). The units of the present layer are connected to all units in the previous layer via weights and bias. The output values of the ith hidden

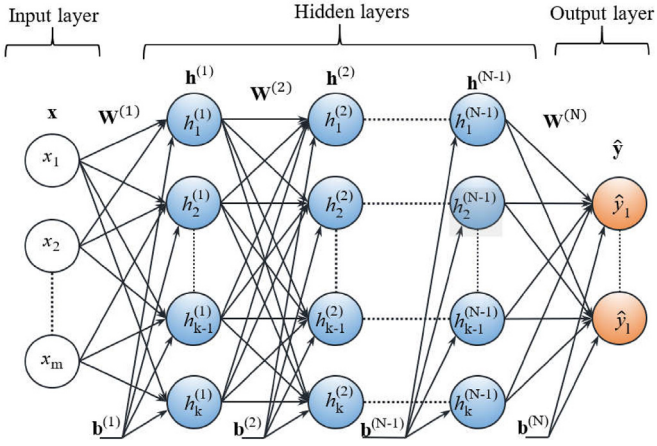

Fig. 6. A generic architecture of a deep feedforward neural network.

layer and output layer are

$$\mathbf{h}^{(i)} = f\left(\mathbf{W}^{(i)}\mathbf{h}^{(i-1)} + \mathbf{b}^{(i)}\right) \qquad \forall i = 1,\ 2, \ldots, (N-1), \tag{23}$$

$$\hat{\mathbf{y}} = O\left(\mathbf{W}^{(N)}\mathbf{h}^{(N-1)} + \mathbf{b}^{(N)}\right), \tag{24}$$

where $\mathbf{h}^{(0)} = \mathbf{x}$ is the input vector, $\mathbf{h}^{(i)}$ is the output vector of ith hidden layer for $i \neq 0$; $\mathbf{W}^{(i)}$ denotes the weight matrix connecting the $(i - 1)^{th}$ layer to ith layer; $\mathbf{b}^{(i)}$ indicates the bias vector; f and O are the activation functions; $\hat{\mathbf{y}}$ refers to the output vector.

Choosing a suitable activation function (AF) for the hidden layer would be capable of learning more complex patterns and enhance the accuracy of the regression problem [40]. According to Nwankpa's survey [57], the rectified linear unit (ReLU) and sigmoid are most widely used.

In supervised learning, all datasets are labeled in supervised learning. The best DNN model is determined by training to identify the optimal parameters (weights and bias). To do this, an optimization algorithm to minimize the loss function between the predicted results and the simulated results is utilized. MSE [40,58] is the most commonly used regression model as a loss function which is defined as follows

$$E_{MSE} = \frac{1}{n}\sum_{j=1}^{n}\left(y_j - \hat{y}_j\right)^2, \tag{25}$$

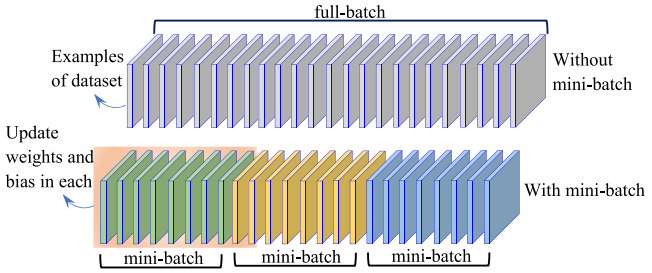

Fig. 7. Splitting the dataset of the mini-batch approach.

in which y_j and $\hat{y}_j$ are the true and predicted output values; n is the number of units output multiplied by the number of training samples.

Back-propagation is one of the most popular training algorithm, which relies on the information gradient of the loss function to modify weights and bias, respectively. Up to now, various algorithms have been presented and investigated by researchers, such as SGD [59], Adagrad [60], Adadelta [61], RMSprop [62], and so on. Adam [63] which combines the advantages of Adagrad with RMSprop, is a robust and well-suited method to a wide range of non-convex optimization problems. More recent, several studies [40,44,64] have been demonstrated the effectiveness of deep learning method for solving the analysis and optimal structure problem. Therefore, it is chosen to fit the model on the training datasets.

Mini-batch gradient descent is commonly used to speed-up in the training phase. The training data is divided into subsets which are called mini-batchs as shown in Fig. 7 [65]. Instead of using all data at once or the samples at a time to update parameters, the incremental update is executed on each subsets with respect to several instances at a time.

There is a very brief description of the Adam method to update parameters for the network using mini-batch technique. The first gradient of MSE with respect to the parameters is given by

$$\mathbf{g}_t = \nabla E_{MSE_t}(\boldsymbol{\theta}), \tag{26}$$

where

$$E_{MSE_t} = \frac{1}{n_b}\sum_{j=1}^{n_b}\left(y_j - \hat{y}_j\right)^2, \tag{27}$$

in which $\boldsymbol{\theta}$ is the parameter vector, which contains elements including weights and bias; $t \in \mathbb{N}$ is the timestep; n_b is the number of samples in the batch; E_{MSE_t} is MSE of the subset at the timestep t.

Then, two exponential decay rates $\mathbf{m}_t$ and $\mathbf{v}_t$ at timestep t

$$\mathbf{m}_t = \beta_1 \mathbf{m}_{t-1} + (1-\beta_1)\,\mathbf{g}_t, \tag{28}$$

$$\mathbf{v}_t = \beta_2 \mathbf{v}_{t-1} + (1-\beta_2)\,\mathbf{g}_t^2, \tag{29}$$

where $\beta_1,\ \beta_2 \in [0,1)$ are the hyper-parameters, which illustrative to control the exponential decay rates of these moving averages. $\mathbf{m}_0$ and $\mathbf{v}_0$ are initialized as a vector of $\mathbf{0}'s$. Hence, the decay rates are small at the initial time steps. To eliminate this effect, the bias corrections for variables $\mathbf{m}_t$ and $\mathbf{v}_t$ are calculated as follows

$$\hat{\mathbf{m}}_t = \frac{\mathbf{m}_t}{1-\beta_1^t}, \tag{30}$$

$$\hat{\mathbf{v}}_t = \frac{\mathbf{v}_t}{1-\beta_2^t}. \tag{31}$$

Finally, the network parameters are updated by the Adam algorithm as

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \eta\frac{\hat{\mathbf{m}}_t}{\sqrt{\hat{\mathbf{v}}_t}+\varepsilon}, \tag{32}$$

Here, η is the learning rate; ε is a constant added to maintain numerical stability; $\hat{\mathbf{m}}_t$ and $\hat{\mathbf{v}}_t$ are the bias-corrected first and second raw moment, respectively. As suggested Kingma and Ba [63] our investigation utilizes the default setting of the algorithm with $\eta = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\varepsilon = 10^{-8}$. The main steps of the Adam method are summarized in Algorithm. 2.

It is easily be seen that the existence of noise in data by the error analysis of the numerical method. On the other hand, the network contains multiple non-linear hidden layers, which can learn a complex mapping between the network's inputs and outputs. Therefore, overfitting can occur in the training process. To avoid this phenomenon, many methods have been investigated to reduce its influence such as validation, early stopping, train with more data, and so on. One of the most common explicit techniques to prevent overfitting is dropout, which is developed by Srivastava et al. [66], and applied successfully for structural optimization problem [40,64]. According to this method, some arbitrary units and all its incoming and outgoing connections will be removed while training the network. In this study, the dropout technique and k-fold cross-validation are used to improve the performance of the DNN model. In addition, RandomizedSearchCV hyperparameter optimization technique available in the Sklearn package in python is used to select the number of units and hidden layers with the search space [10, 500] and [1,5], respectively.

4.3. Differential evolution algorithm

DE, introduced by Storn and Price [16], has been demonstrated effective and robust for solving global optimization problems [18,67–69]. In this article, it is employed as an optimizer for solving the equivalent optimization problem where the value of deflections are predicted from DNN model. The basic steps of the DE algorithm are summarized in the following.

- *Initialization*

Firstly, the initial population including np individuals is randomly created from the search space. The ith individual is a vector with d design variables $\mathbf{x}_i = \{\mathrm{x}_{i,1}, \mathrm{x}_{i,2}, \ldots, \mathrm{x}_{i,d}\}$ and is defined by

$$\mathrm{x}_{i,j} = \mathrm{x}_{\min,j} + \mathrm{rand}_{i,j}[0,1]\left(\mathrm{x}_{\max,j} - \mathrm{x}_{\min,j}\right) \quad i = 1,2,\ldots,np;\quad j = 1,2,\ldots,d, \tag{33}$$

where $\mathrm{x}_{\max,j}$ and $\mathrm{x}_{\min,j}$ are the upper and lower bounds of x_j, respectively; and $\mathrm{rand}_{i,j}[0,1]$ is a uniformly distributed random number on the interval [0, 1].

- *Mutation*

Secondly, a mutant vector $\mathbf{v}_i$ is generated by the mutation strategy corresponding to a target vector $\mathbf{x}_i$. Four popular mutation operations are usually employed in the DE as follows

$$\text{rand/1} \quad \mathbf{v}_i = \mathbf{x}_{r_1} + F\left(\mathbf{x}_{r_2} - \mathbf{x}_{r_3}\right), \tag{34a}$$

$$\text{rand/2} \quad \mathbf{v}_i = \mathbf{x}_{r_1} + F\left(\mathbf{x}_{r_2} - \mathbf{x}_{r_3}\right) + F\left(\mathbf{x}_{r_4} - \mathbf{x}_{r_5}\right), \tag{34b}$$

$$\text{best/1} \quad \mathbf{v}_i = \mathbf{x}_{best} + F\left(\mathbf{x}_{r_1} - \mathbf{x}_{r_2}\right), \tag{34c}$$

$$\text{best/2} \quad \mathbf{v}_i = \mathbf{x}_{best} + F\left(\mathbf{x}_{r_1} - \mathbf{x}_{r_2}\right) + F\left(\mathbf{x}_{r_3} - \mathbf{x}_{r_4}\right), \tag{34d}$$

where r_1, r_2, r_3, r_4 and r_5 are five different integer numbers that are chosen from $\{1,2,\ldots,i-1,i+1,\ldots,np\}$; $\mathbf{x}_{best}$ is the best individual in the population; and the scale factor F is randomly picked out from the interval (0, 1].

According to Eq. (34), the values of the mutant vector $\mathbf{v}_i$ may be violated its boundary constraint. Hence it is adjusted to the allowable

Algorithm 2 Adam (Kingma and Ba [63]).

Input: parameters $\boldsymbol{\theta}_0$ (the initial values of weights and biases)
Output: parameters $\boldsymbol{\theta}_0$ (the optimal values of weights and biases)
Given hyper-parameters $\eta = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\varepsilon = 10^{-8}$
Set $\mathbf{m}_0 = \mathbf{0}, \mathbf{v}_0 = \mathbf{0}, t = 0$
while $\boldsymbol{\theta}$ *not convergence* **do**
  $t = t + 1$
  Compute g_t using current mini-batch at timestep t by Eq. (26)
  Set $\mathbf{m}_t = \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1) .\mathbf{g}_t$ [see Eq. (28)]
  Set $\mathbf{v}_t = \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2) .\mathbf{g}_t^2$ [see Eq. (29)]
  Set $\hat{\mathbf{m}}_t = \mathbf{m}_t / (1 - \beta_1^t)$ [see Eq. (30)]
  Set $\hat{\mathbf{v}}_t = \mathbf{v}_t / (1 - \beta_2^t)$ [see Eq. (31)]
  Update parameters $\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \eta \hat{\boldsymbol{m}}_t / \left(\sqrt{\hat{\boldsymbol{v}}_t} + \varepsilon\right)$ [see Eq. (32)]

limits of the variable designs by the following equation

$$\mathrm{v}_{i,j} = \begin{cases} 2\mathrm{x}_{\min,j} - \mathrm{v}_{ij} & \text{if } \mathrm{v}_{ij} < \mathrm{x}_{\min,j}, \\ 2\mathrm{x}_{\max,j} - \mathrm{v}_{ij} & \text{if } \mathrm{v}_{ij} > \mathrm{x}_{\max,j}, \\ \mathrm{v}_{ij} & \text{otherwise.} \end{cases} \tag{35}$$

- *Crossover*

Next, a trial vector $\mathbf{u}_i$ is generated from the mutant vector $\mathbf{v}_i$ and its target vector $\mathbf{x}_i$ based on the crossover operator. It can be expressed as

$$\mathrm{u}_{i,j} = \begin{cases} \mathrm{v}_{ij} & \text{if } j = \mathrm{K} \text{ or } \mathrm{rand}[0,1] \le \mathrm{Cr}, \\ \mathrm{x}_{i,j} & \text{otherwise,} \end{cases} \tag{36}$$

where Cr is the crossover control parameter which is randomly selected in [0, 1]; K is an integer number randomly in [1, *np*].

- *Selection*

Finally, a selection operation is implemented to chose better individuals by comparing the objective function value of each trial vector $f(\mathbf{u}_i)$ and target vector $f(\mathbf{x}_i)$ in the current population. The operation is expressed as follows

$$\mathbf{x}_i = \begin{cases} \mathbf{u}_i & \text{if } f(\mathbf{u}_i) \le f(\mathbf{x}_i), \\ \mathbf{x}_i & \text{otherwise.} \end{cases} \tag{37}$$

5. Numerical examples

5.1. Nonlinear analysis

In this section, two numerical examples are investigated to demonstrate the effectiveness and reliability of the analysis results before implementing the optimization step. The own weight of the truss is neglected in the analyzes.

5.1.1. 24-bar dome truss

A star space dome consisting of 24 bars is shown in Fig. 8. All members have the same axial stiffness $EA = 8.0 \times 10^7$ N. The central node of the structure is subjected the concentrated force *P*. The parameters of the arc-length procedure used in the simulations were: $i_{max} = 100; \Delta l = 0.15$ and $tol = 1.0 \times 10^{-6}$. This example has been studied by Bonet [70], Greco [48] and Krishnamoorthy [71].

Fig. 9 shows the equilibrium path at the apex of the truss. The computational result indicates that the arc-length method is able to capture the full nonlinear behavior and overcome the limit points. The result of this problem shows a good agreement with the equilibrium path obtained by Bonet.

5.1.2. 60-bar dome truss

The second structure is considered to evaluate the geometric nonlinear behavior of a more complex dome truss. The geometry and loading shown in Fig. 10 has been solved by Choong [72]. It comprises 25

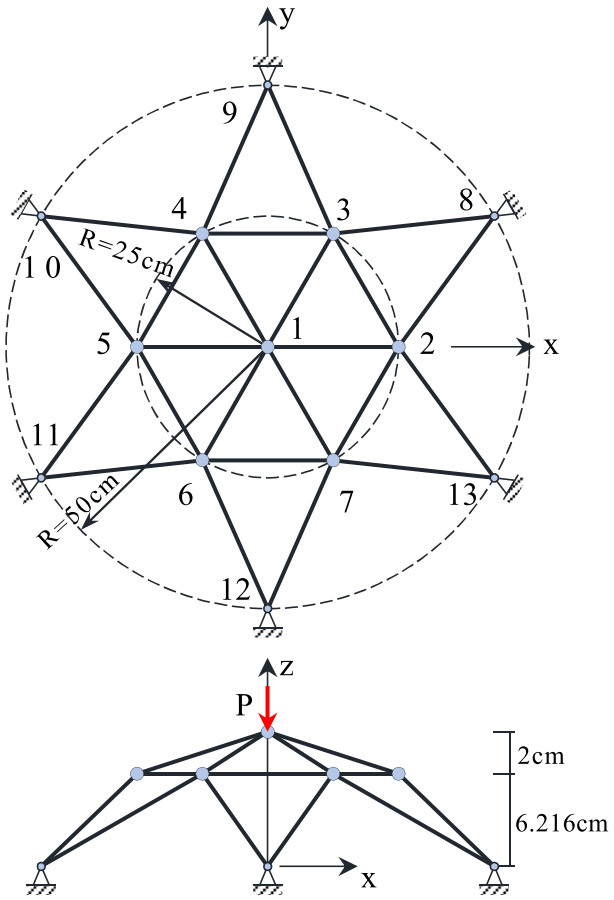

Fig. 8. Star dome space truss with 24 bars.

nodes and 60 elements with the same axial stiffness $EA = 1.0 \times 10^4$. The vertical loads *P* are applied at joints 13–18. The parameters used for the path-following technique were $i_{max} = 100; \Delta l = 2.0$ and $tol = 1.0 \times 10^{-6}$.

The vertical displacement at node 1 versus load P curve obtained with the proposed approach was a good agreement with that of Choong et al. [72], are shown in Fig. 11. It can be seen that the methodology has the ability to trace the complete equilibrium path with several bifurcation points and limit points. These results proved that the developed computational codes are enough reliability to solve nonlinear problems.

5.2. Optimization

In this section, three well-known benchmark problems are explored to accurately evaluate the integration model. The parameters of DE are

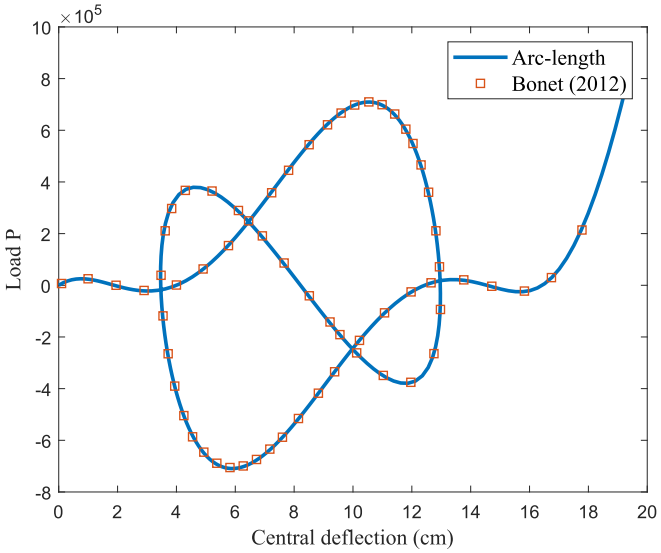

Fig. 9. Load-deflection path for the star dome truss with 24 bars.

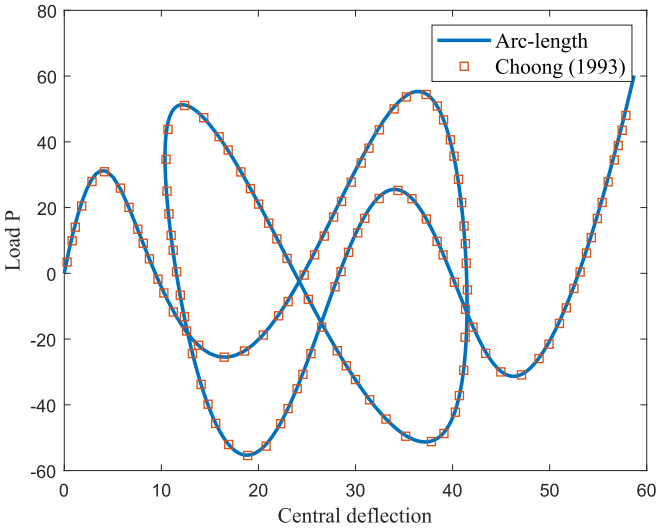

Fig. 11. Load-deflection curve for the star dome truss with 60 bars.

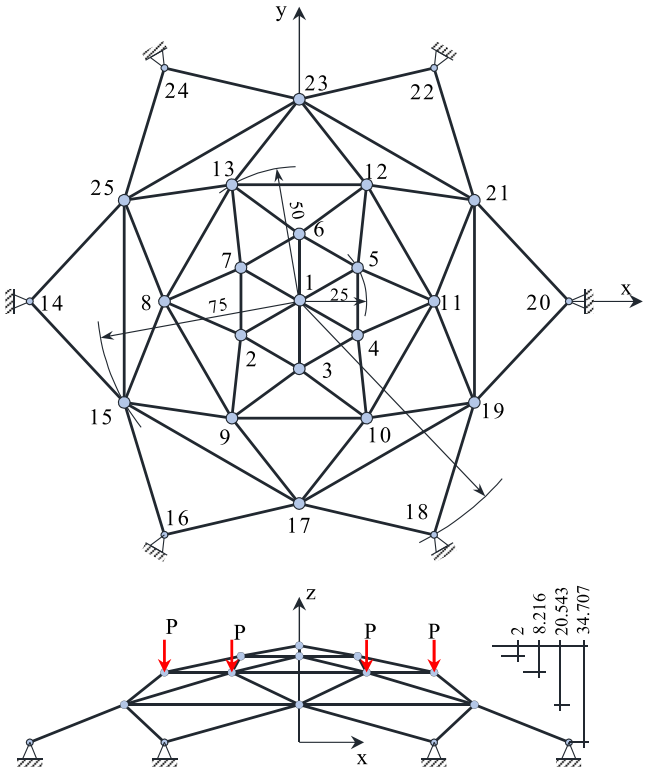

Fig. 10. Star dome space truss with 60 bars.

set similar for all investigated examples as follows [16–18,73]: population size *npop* = 20, maximum number of generation *MaxGen* = 2000, scaling factor $F = 0.8$, crossover control parameter $Cr = 0.9$, and value tolerance $tol = 10^{-6}$. The DE is a stochastic nature of metaheuristic algorithms. Therefore, 30 independent runs are implemented for each problem. In all applications, FEA is utilized to obtain the dataset with 1320 data pairs. It is assumed that the material is a linear elastic material. The input data are cross-section areas and the displacements at free nodes are defined as the output data of the DNN model. The network uses a batch size of 32, 1000 epochs, and learning rate 0.001. In order to avoid overfitting, 10-fold cross-validation and dropout with ratio of 0.2 were applied. The network architectures are found by using randomized search, and they are summarized in Table 2. The results obtained from the integration model are presented in terms of the best weight, and constraints evaluation. Accordingly, the result of the best run is reported for comparison with the DE algorithm, and various algorithms. To get a fair comparison between the different methods, all the tests are implemented in PyCharm anaconda Python environment using Sklearn, Keras, Tensorflow libraries, and a desktop computer Core(TM) i5-8500 CPU 3.0 GHz with 16 GB RAM.

5.2.1. 30-bar dome truss

The first example is the 30-bar dome space truss, which was first introduced by Khot and Kamat [10]. Cross-section areas of the structure are categorized into four groups with pin supports, geometry, dimensions, boundary conditions as shown in Fig. 12. The upper and lower bounds of the design variable are respectively 0.1in^2 and 2in^2. The density of the material is $\rho = 0.1\text{lbs/in}^3$ and Young's modulus is equal to $E = 10^7\text{lbs/in}^2$. The vertical z-direction force F = 2000lb is applied at node 1 of the truss. Since this is a symmetric structure, the displacement of nodes 1–2 in the z-direction are not allowed exceed ±10in. The network architecture that is given in Table 2, is used to build the model. The combinations of optimizers and activation functions are adopted to investigate of the DNN model, and the results are reported in Table 3. It can be seen that Adam and SGD are, respectively, the best optimizer and the worst optimizer. And meanwhile, Softmax performs worse than

Table 2
The architecture of DNN model.

Problem	Input layer	Hidden layers			Output layer	
	No. Units	No. layers	No. Units/layer	Activation function	No. Units	Activation function
30-bar dome truss	4	4	335	ReLU	2	Linear
52-bar dome truss	8	5	71	Sigmoid	4	Linear
25-bar space truss	8	3	126	ReLU	4	Linear

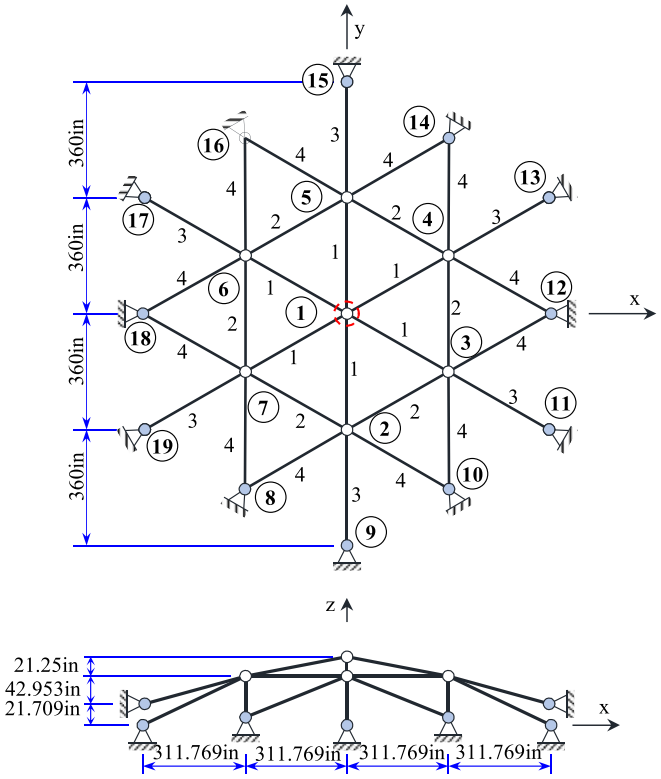

Fig. 12. 30-bar dome space truss structure.

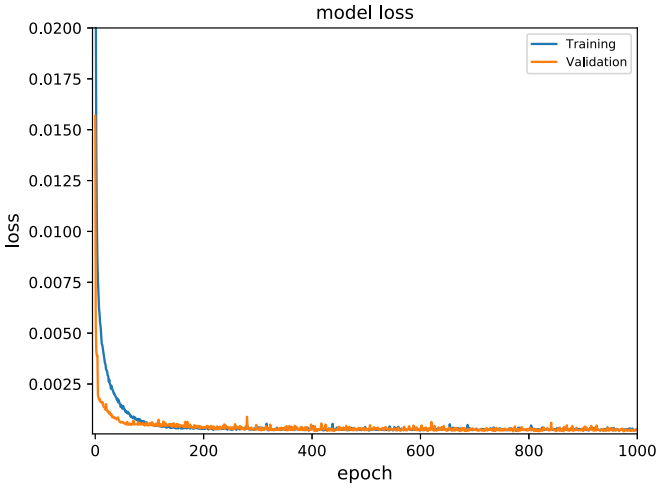

Fig. 13. The convergence history of loss function of 30-bar dome space truss structure.

the other activation functions on most of the optimizers, whereas ReLU is either the best or the second best on 5 of 6 combining. Two of the most remarkable results to emerge from combining ReLU, Sigmoid into Adam are slightly lower than the other combinations. Hence, they are suggested for implementing in this study. Fig. 13 illustrates the convergence history of the loss function. From displayed results, MSE of both training and validation sets approach zero after 500 epochs. Table 4 summarizes mean, standard deviation (Std) and 95% confidence intervals (CI) of the errors with respect to training and validation sets. It can be seen that MSE are smaller than 0.2×10^{-3} and the range where the 95%CI RMSE values are lower than 4%. It has been proved that the network tackled the issue of overfitting and underfitting.

The results of the first benchmark gained by this study are presented in Table 7 and comparison between the present method and others. Firstly, it can be seen that there is a large difference between the optimization considering linear and nonlinear analysis. The optimum weights were 333.778lb and 730.106lb with respect to linear and nonlinear behaviour and the difference was 118.74%. This show that consideration of the geometric nonlinearity effect is essential for the design of the structure. Therefore, the structure can be dangerous when the geometric nonlinearity effects are considered. On the other hand, the optimum weight obtained from DNN-DE is better than the results given by Khot [10], Missoum [8], and Hrinda [14]. Although there is a negligible discrepancy (relative errors is small 0.48%) in the best weight, it is a large computation time difference between the DNN-DE and FEA-DE. Specifically, the average times of the FEA-DE take 8,959s to find the optimal solution with 2,821,111 FE analyses while DNN-DE only requires 3746.936s with 4120 predictions, including 2549s (68.03%) of the data-collection, 1197.376s (31.95%) of the training session and 0.56s (0.02%) of the optimization session. It is easily explained that FEA-DE takes a lot of time to evaluate due to the incremental iterative algorithm. In general, DE takes approximately three days to complete

Table 3
MSE of the 30-bar dome truss (10^{-4}) with respect to various optimizers and activation functions.

Optimizer	Adam		RMSprop		Adagrad		SGD		Adadelta		Adamax	
Activation function	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
Softmax	220.4	203.7	246.8	226.7	45.74	44.67	223.80	206.3	231.63	219.8	223.4	206.2
Softplus	1.355	3.620	3.006	5.680	92.27	84.46	94.267	85.96	24.369	19.88	1.187	8.121
Tanh	0.304	4.578	0.451	13.17	14.70	15.58	94.529	87.04	13.720	25.92	2.435	5.898
Sigmoid	0.680	2.293	2.284	2.838	19.45	14.70	219.64	205.7	32.899	24.61	1.890	1.412
ReLU	0.174	3.803	0.789	6.144	0.321	4.571	17.078	17.89	0.4706	5.327	0.290	5.734

Table 4
Results of MSE and RMSE for the dome structure with 30 members.

Data	Statistics	MSE(10^{-3})			RMSE(%)		
		w_1	w_2	Total	w_1	w_2	Total
Training	Mean	0.161	1.361	0.761	1.248	3.678	2.749
	Std	0.066	0.227	0.129	0.235	0.303	0.232
	95% CI	0.11–0.21	1.2–1.52	0.67–0.85	1.08–1.42	3.46–3.89	2.58–2.91
Validation	Mean	0.177	1.485	0.831	1.308	3.796	2.842
	Std	0.079	0.574	0.314	0.261	0.699	0.508
	95% CI	0.12–0.23	1.07–1.9	0.61–1.06	1.12–1.49	3.3–4.3	2.48–3.21

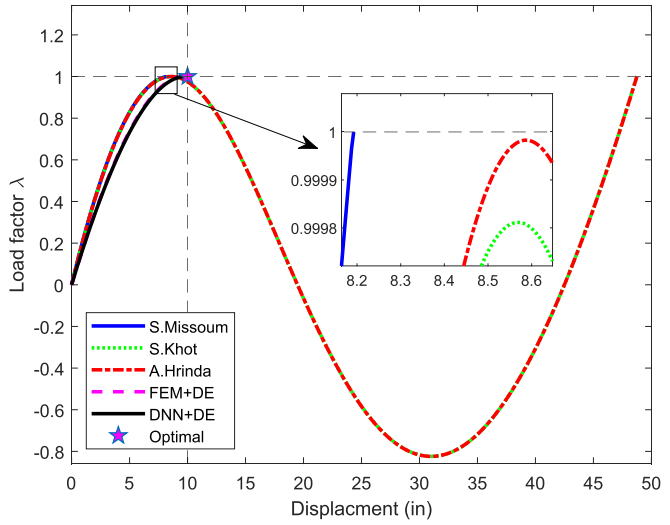

Fig. 14. Load–displacement curve of the first constraint for the optimum solution at node 1 in 30-bar dome truss.

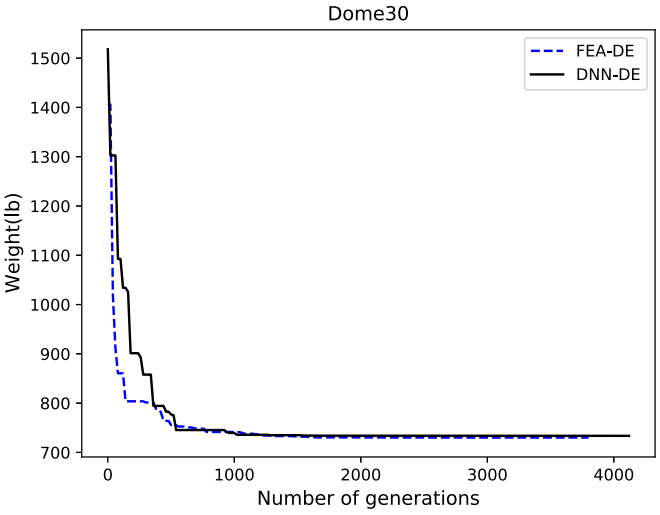

Fig. 15. The weight convergence histories of the 30-bar dome truss.

30 independent optimization runs based on the FEA model, while it only spends 3763.176s on the DNN model. Therefore, the integration model helps to save a huge amount of computational cost.

The areas obtained from Table 7 were used for performing the nonlinear analysis to achieve the displacement constraint values, which are shown in Table 8. The equilibrium paths of the center node are displayed in Fig. 14. It should be noted that the results obtained from Khot [10], Missoum [8], and Hrinda [14] are studied on the structural design optimization problems without displacement constraints. Specifically, Khot [10] defined the optimization problem with the total energy constraint not exceeding total potential energy. At the same time, Khot and Hindra indicated that the total potential energy associated with the optimum design at the nonlinear critical point and then the load increment equals 1. As shown in Fig. 14, the first critical points achieved as the displacement between 8in and 12in, where the critical point of FEA-DE obtained at 9.6927in. For the integration model, the output values of data points were collected in the training dataset when the load increment equals 1. Consequently, if optimization algorithms are good enough and the load increment constraint value is set to 1, the critical point position will not change when the displacement constraint value is an arbitrary choice between 10in and 12in. Therefore, the comparison between our approach and the others has equivalent significance. As shown in Fig. 14, the load-deflection plots obtained from Khot and Hrinda are two critical-limit points for each curve. The first critical-limit points are achieved when the load factors corresponding to 0.9998 and 0.99999 are less than 1. These differences are not much significant, and they can be deal with control parameters and the tolerance for the stopping criteria of the nonlinear solution technique. These reasons, along with the sensitivity of the nonlinear response, can make the large displacements obtained from Khot and Hrinda as shown in 8. For the DNN-DE approach, the constraint results agree well with the FEA-DE, and close to the constraint limit. The weight convergence histories obtained using the algorithms for this example are given in Fig. 15. It is found that the proposed approach saves lots of computational efforts to reach the near-global optimal solution.

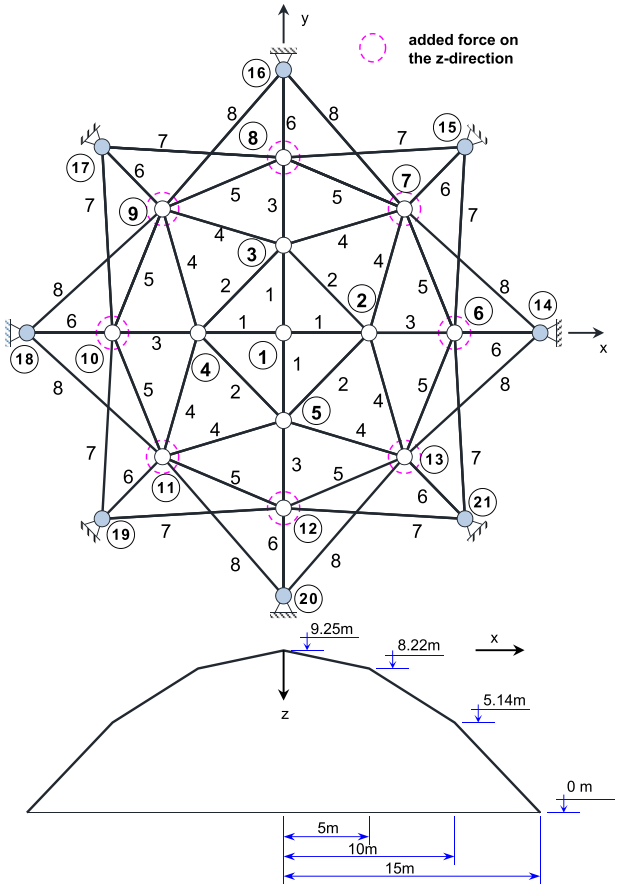

Fig. 16. 52-bar dome truss.

5.2.2. 52-bar dome truss

In the second example, the optimization problem for the 52-bar dome truss is investigated. This problem has been previously examined by Saka [7]. The geometry and finite element representation are shown in Fig. 16. All members of the structure are categorized into 8 groups, which are labeled in the same figure. The Young's modulus is E = 21000 kN/cm^2 and material density is ρ = 7.85 N/cm^3 for all elements. The cross-section areas of members are permitted to vary between 2 cm^2 and 100 cm^2. The system is subjected to an external loading in the direction of the z-axis which was 150 kN at joints 6–13. The vertical displacements of all free nodes are restricted to 10 mm. The network architecture and activation function mentioned in Table 2 is used for the training of process operators. The total time requires 3194s to generate the training dataset, 753.326s to do the training process. The loss model depicting convergence of MSE of training and valida-

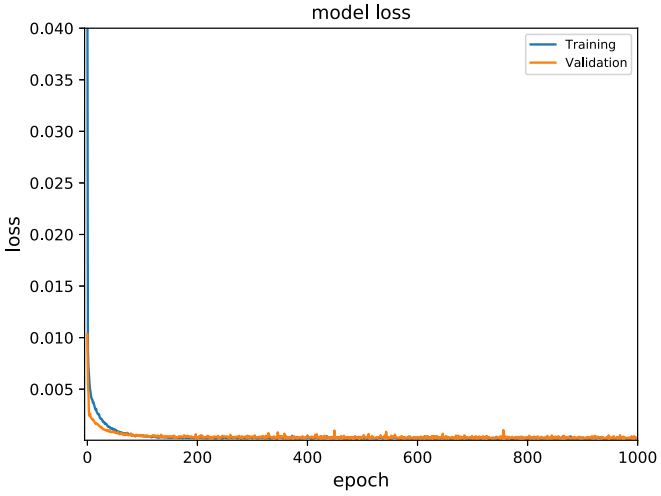

Fig. 17. The convergence history of loss function of 52-bar dome truss.

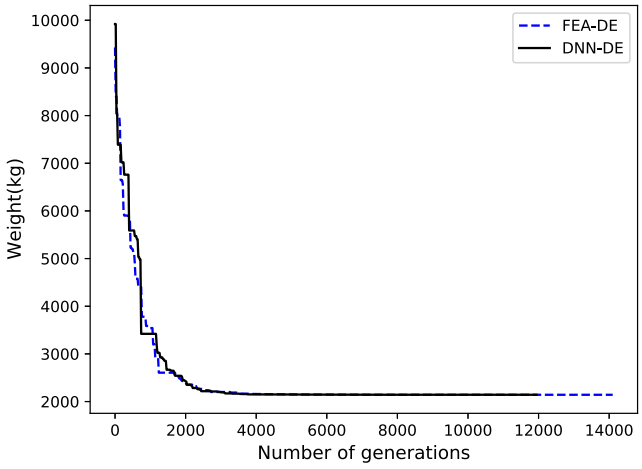

Fig. 18. The weight convergence histories of the 52-bar dome truss.

tion sets are shown as Fig. 17. It can be seen that the loss of training and validation set tends to decrease at first and then converge to zeros and stable after 400 epochs. In Table 5, the mean and standard deviation are listed along with the 95% confidence interval for each and all outputs of the network. The results indicate that MSE close to zeros and the 95% confidence interval of RMSE values are smaller than 2%.

As the previously indicated example, Table 9 summarizes the optimal results obtained by this work, and comparing with those available from the existing literature. Table 10 shows that the results of constraints obtained from the geometrically nonlinear analysis by using the arc-length method. The third constraint obtained from Saka [7] is larger than the others. It can be explained by the sensitivity of the nonlinear response and control parameters. On the other hand, the results of DNN-DE attain a quite good agreement with the FEA-DE. More specifically, the optimal solution obtained from this study is very close to FEA-DE (the relative error of the optimum weight is very small 0.0024%). Furthermore, it takes 3948.976s with only 14100 predictions to achieve the optimal solution while FEA-DE takes 68,408s with 12,762,048 FE analyses to get the goal.

Consequently, the integration model can reduce the computational time more than 17 times with FEA-DE. As observed, all displacements of the DE based methods are free from any violations of constraints. The weight convergence histories obtained using the DNN-DE and FEA-DE for this example are shown on Fig. 18. The convergence rate of DNN-DE is faster than FEA-DE. Hence, it has been demonstrated that our method is effective, reduces the computational time, and guarantees solution accuracy.

5.2.3. 25-bar space truss

A space truss containing 25 bars shown on Fig. 19 is considered as the last problem. This example has been studied by Saka [7] for the optimization problem with geometrically nonlinear behavior. Cross-sectional areas of all members are divided into 8 groups corresponding to 8 design variables in the same figure. The bounds of the sizing design variables are $2\ \text{cm}^2 \le A_i \le 20\ \text{cm}^2$. The density of the material is assumed constant in each element $\rho = 7.85\ \text{N/cm}^3$ and Young's modulus is $E = 20700\ \text{kN/cm}^2$. This structure is subjected to the loading condition listed in Table 11. Allowable displacements at joints 1 and 2 are restricted to 10 mm in the x and y directions. The optimal mapping is defined by the network architecture in Table 2. Fig. 20 depicted the

Table 5
Results of MSE and RMSE for the dome structure with 52 members.

Data	Statistics	MSE(10^{-4})					RMSE(%)				
		w_1	w_2	w_6	w_7	Total	w_1	w_2	w_6	w_7	Total
Training	Mean	1.656	1.334	1.445	1.703	1.535	1.275	1.145	1.195	1.251	1.228
	Std	0.506	0.394	0.313	0.987	0.425	0.185	0.159	0.138	0.392	0.171
	95% CI	1.29–2.02	1.05–1.62	1.22–1.67	1–2.41	1.23–1.84	1.14–1.41	1.03–1.26	1.1–1.29	0.97–1.53	1.11–1.35
Validation	Mean	3.028	2.431	2.327	3.041	2.707	1.658	1.486	1.473	1.631	1.590
	Std	2.143	1.597	1.445	2.198	1.585	0.557	0.498	0.417	0.652	0.444
	95% CI	1.49–4.56	1.29–3.57	1.29–3.36	1.47–4.61	1.57–3.84	1.26–2.06	1.13–1.84	1.17–1.77	1.16–2.1	1.27–1.91

Table 6
Results of MSE and RMSE for the 25-bar space truss.

Data	Statistics	MSE(10^{-4})					RMSE(%)				
		u_1	v_1	u_2	v_2	Total	u_1	v_1	u_2	v_2	Total
Training	Mean	0.515	0.434	0.491	0.431	0.468	0.705	0.650	0.694	0.646	0.677
	Std	0.205	0.165	0.145	0.167	0.142	0.139	0.117	0.103	0.125	0.104
	95% CI	0.37–0.66	0.32–0.55	0.39–0.6	0.31–0.55	0.37–0.57	0.61–0.8	0.57–0.73	0.62–0.77	0.56–0.74	0.6–0.75
Validation	Mean	0.758	0.941	0.737	0.954	0.847	0.840	0.913	0.833	0.906	0.878
	Std	0.441	0.745	0.404	0.861	0.582	0.243	0.346	0.219	0.386	0.293
	95% CI	0.44–1.07	0.41–1.47	0.45–1.03	0.34–1.57	0.43–1.26	0.67–1.01	0.67–1.16	0.68–0.99	0.63–1.18	0.67–1.09

Table 7
Comparison of optimal results of the dome structure with 30 members.

Design variables A_i (in^2)	Linear	Geometric nonlinear				
	FEA-DE	Khot [10]	Missoum [8]	Hrinda [14]	FEA-DE	DNN-DE
1	0.6947	1.6926	1.6904	1.69407	1.6113	1.51
2	0.4825	1.3754	1.3833	1.37499	1.4629	1.581
3	0.1	0.1	0.1	0.10037	0.1	0.1
4	0.1627	0.2693	0.2674	0.26346	0.1	0.1
Best weight (lb)	333.778	766.19	766.72	765	730.106	733.629
No. function evaluations	4540	–	–	–	2821111	4120
Time average (sec)	6.07	–	–	–	8959	3746.936

Table 8
Two displacement constraints of the 30-bar dome truss.

Displacements (in)	Khot [10]	Missoum [8]	Hrinda [14]	FEA-DE	DNN-DE
w_1	48.733	8.1923	48.7505	9.6972	9.6927
w_2	2.05010	0.0899	2.06800	1.1664	1.2489

Table 9
Comparison of optimal results of the dome structure with 52 members.

Design variables A_i (cm^2)	Geometric nonlinear		
	Saka [7]	FEA-DE	DNN-DE
1	81.82	2	2
2	22.41	2	2
3	33.58	2	2
4	14.45	2	2
5	10.64	16.672	16.753
6	25.16	17.585	17.869
7	2	2.519	2.3013
8	2	2	2
Best weight (kg)	5161	2141.87	2142.41
No. function evaluations	–	12762048	14100
Time average (sec)	–	68408	3948.976

Table 10
The second displacement constraints of the 52-bar dome truss.

Displacements (mm)	Saka [7]	FEA-DE	DNN-DE
w_1	−2.772	1.328	1.206
w_2	−2.826	0.72	0.742
w_6	13.045	10	10
w_7	9.491	10	9.648

convergence history of the loss consisting of MSE of training and validation sets. The mean, standard deviation, and 95% confidence interval of MSE and RMSE are presented in Table 6 in which the values of MSE and RMSE are smaller than 10^{-4} and 1%, respectively.

The optimal results obtained by the present approach and the others are tabulated as Table 12. Again, we can see that the optimum weight obtained by the integration model and FEA-DE are similar and

Table 11
Loading condition for 25-bar spatial truss.

Node	Loading (kN)		
	X	Y	Z
1	−80	−120	30
2	−60	−100	30
3	−30	0	0
6	−30	0	0

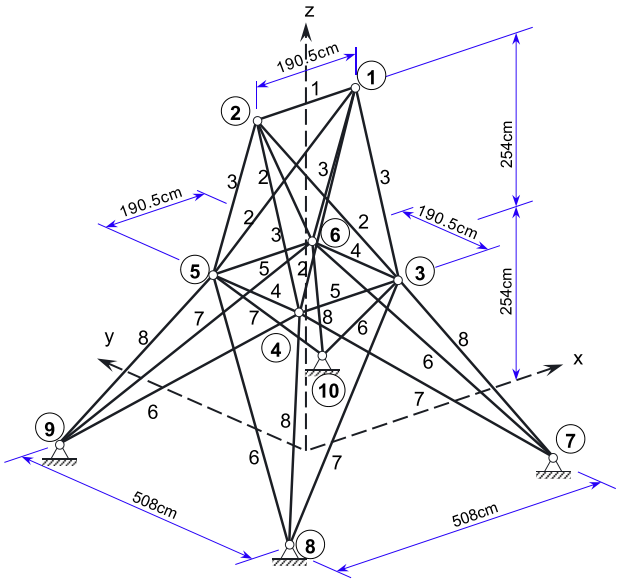

Fig. 19. 25-bar space truss.

smaller than the result given by Saka [7]. Meanwhile, DNN-DE saves a lot of time and the small relative error (0.23%) to FEA-DE. More specifically, it only spends 1853s for the data-collection, 895.793s for the training and optimization process while FEA-DE takes very long time (28,179s). Our approach is only requires 18,120 predictions while FEA-DE needs 1,490,799 FE analyses to obtain the optimal global solution. More importantly, none of the deflection constraints do not exceed the allowable displacements as shown in Table 13. Fig. 21 provides the weight convergence histories obtained by two different approaches. Although the convergence rate of the DNN-DE is lower than FEA-DE, DNN-DE shows its accuracy and effectiveness in significantly reducing the computational cost.

6. Conclusions

In this paper, a novel deep neural network-based surrogate model is proposed for the optimization of truss structures with geometrically nonlinear behavior. DNN is employed to learn the model complex relationship between the inputs and the outputs as a surrogate model and then displacements of the truss are quickly predicted without solving nonlinear programming problems. DE algorithm is carried out to solve

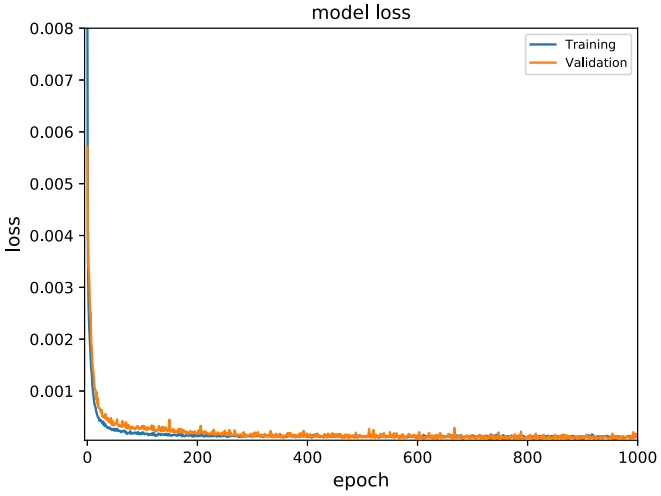

Fig. 20. The convergence history of loss function of 25-bar space truss.

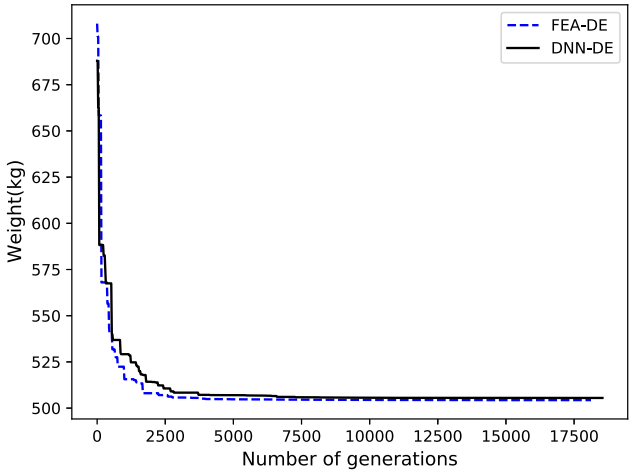

Fig. 21. The weight convergence histories of the 25-bar space truss.

Table 12
Comparison of optimal results of the 25-bar space truss.

Design variables A_i (cm^2)	Geometric nonlinear		
	Saka [7]	FEA-DE	DNN-DE
1	2	2	2
2	7.5	3.098	4.099
3	13.12	16.91	15.113
4	2	2	2
5	4.27	6.566	3.906
6	3.8	3.891	4.851
7	4.22	3.33	3.744
8	17.15	18.32	17.779
Best weight (kg)	507	504.315	505.493
No. function Evaluations	–	1490799	18120
Time average (sec)	–	28179	2748.793

Table 13
The second displacement constraints of the 25-bar spatial truss.

Displacements (mm)	Saka [7]	FEA-DE	DNN-DE
u_1	−6.704	−9.764	−8.313
v_1	−10	−10	−10
u_2	−6.289	−9.346	−7.905
v_2	−8.85	−8.323	−8.601

a new structural optimization, which combines the original objective with constraints obtained from DNN model. Three numerical examples are tested to verify the accuracy, effectiveness, and robustness of the proposed approach. The attained results indicated that the integration of DNN and DE saves computational cost dramatically in almost all problems. Meanwhile, DNN model guarantees capable of good predicting displacements with average errors smaller than 0.5% in the optimum weight. Therefore, the integration model is an important contribution of this study and promising to extend its applications to more complex engineering problems, such as plate structures, shell structures, etc.

In supervised learning, one of the challenges facing many researchers today indicates the optimal DNN architecture, which will affect the computation time and accuracy of the training model, and this study is no exception. Whereas the number of nodes in the input and output layers is indicated by the physical characteristics of the problem, the type of activation function, the number of hidden layers and the nodes in each of these hidden layers often require considerable user experience blue [34,38]. To circumvent this limitation, one promising direction will be improved in the next our work by using Bayesian optimization techniques to determine the optimal architecture of DNN.

Author contributions

Hau T. Mai: Conceptualization, Methodology, Software, Formal analysis, Investigation, Writing – original draft, Visualization. Joowon Kang: Formal analysis, review & editing. Jaehong Lee: Supervision, Writing review & editing, 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.

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.

References

[1] H.T. Thai, T. Nguyen, S. Lee, V. Patel, T. Vo, Review of Nonlinear Analysis and Modeling of Steel and Composite Structures, 2020.
[2] H.X. Nguyen, E. Atroshchenko, H. Nguyen-Xuan, T.P. Vo, Geometrically nonlinear isogeometric analysis of functionally graded microplates with the modified couple stress theory, Comput. Struct. 193 (2017) 110–127.
[3] S. Hou, Q. Li, S. Long, X. Yang, W. Li, Design optimization of regular hexagonal thin-walled columns with crashworthiness criteria, Finite Elem. Anal. Des. 43 (2007) 555–565.
[4] T.P. Vo, J. Lee, Geometrical nonlinear analysis of thin-walled composite beams using finite element method based on first order shear deformation theory, Arch. Appl. Mech. 81 (2011) 419–435.
[5] T.P. Vo, J. Lee, Geometrically nonlinear theory of thin-walled composite box beams using shear-deformable beam theory, Int. J. Mech. Sci. 52 (2010) 65–74.
[6] G.A. Wempner, Discrete approximations related to nonlinear theories of solids, Int. J. Solid Struct. 7 (1971) 1581–1599.
[7] M. Saka, M. Ulker, Optimum design of geometrically nonlinear space trusses, Comput. Struct. 41 (1991) 1387–1396.
[8] S. Missoum, Z. Gürdal, W. Gu, Optimization of nonlinear trusses using a displacement-based approach, Struct. Multidiscip. Optim. 23 (2002) 214–221.
[9] N. Khot, Nonlinear analysis of optimized structure with constraints on systemstability, AIAA J. 21 (1983) 1181–1186.
[10] N. Khot, M. Kamat, Minimum weight design of truss structures with geometric nonlinear behavior, AIAA J. 23 (1985) 139–144.
[11] M. Haririan, J. Cardoso, J. Arora, Use of adina for design optimization of nonlinear structures, Comput. Struct. 26 (1987) 123–133.
[12] M.E. El-Sayed, B.J. Ridgely, E. Sandgren, Nonlinear structural optimization using goal programming, Comput. Struct. 32 (1989) 69–73.

[13] M.-K. Shin, K.-J. Park, G.-J. Park, Optimization of structures with nonlinear behavior using equivalent loads, Comput. Methods Appl. Mech. Eng. 196 (2007) 1154–1167.
[14] G.A. Hrinda, D.T. Nguyen, 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 (2008) 933–950.
[15] K. Deb, S. Gulati, Design of truss-structures for minimum weight using genetic algorithms, Finite Elem. Anal. Des. 37 (2001) 447–465.
[16] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (1997) 341–359.
[17] V. Ho-Huu, T. Nguyen-Thoi, T. Truong-Khac, L. Le-Anh, T. Vo-Duy, An improved differential evolution based on roulette wheel selection for shape and size optimization of truss structures with frequency constraints, Neural Comput. Appl. 29 (2018) 167–185.
[18] V. Ho-Huu, T. Vo-Duy, T. Luu-Van, L. Le-Anh, T. Nguyen-Thoi, Optimal design of truss structures with frequency constraints using improved differential evolution algorithm based on an adaptive mutation scheme, Autom. ConStruct. 68 (2016) 81–94.
[19] A. Kaveh, H. Rahami, Nonlinear analysis and optimal design of structures via force method and genetic algorithm, Comput. Struct. 84 (2006) 770–778.
[20] E. Kameshki, M. Saka, Optimum geometry design of nonlinear braced domes using genetic algorithm, Comput. Struct. 85 (2007) 71–79.
[21] S. Shan, G.G. Wang, Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions, Struct. Multidiscip. Optim. 41 (2010) 219–241.
[22] L. Chen, H. Qiu, L. Gao, C. Jiang, Z. Yang, Optimization of expensive black-box problems via gradient-enhanced kriging, Comput. Methods Appl. Mech. Eng. 362 (2020) 112861.
[23] R. Shi, L. Liu, T. Long, Y. Wu, Y. Tang, Filter-based adaptive kriging method for black-box optimization problems with expensive objective and constraints, Comput. Methods Appl. Mech. Eng. 347 (2019) 782–805.
[24] N. Bartoli, T. Lefebvre, S. Dubreuil, R. Olivanti, R. Priem, N. Bons, J.R.R.A. Martins, J. Morlier, Adaptive modeling strategy for constrained global optimization with application to aerodynamic wing design, Aero. Sci. Technol. 90 (2019) 85–102.
[25] A.I.J. Forrester, A.J. Keane, Recent advances in surrogate-based optimization, Prog. Aero. Sci. 45 (2009) 50–79.
[26] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, F.E. Alsaadi, A survey of deep neural network architectures and their applications, Neurocomputing 234 (2017) 11–26.
[27] D.W. Abueidda, S. Koric, N.A. Sobh, H. Sehitoglu, Deep learning for plasticity and thermo-viscoplasticity, Int. J. Plast. 136 (2020) 102852.
[28] E. Samaniego, C. Anitescu, S. Goswami, V.M. Nguyen-Thanh, H. Guo, K. Hamdia, X. Zhuang, T. Rabczuk, An energy approach to the solution of partial differential equations in computational mechanics via machine learning: concepts, implementation and applications, Comput. Methods Appl. Mech. Eng. 362 (2020) 112790.
[29] S. Lee, H. Kim, Q.X. Lieu, J. Lee, Cnn-based Image Recognition for Topology Optimization, Knowledge-Based Systems, 2020, p. 105887.
[30] G.X. Gu, C.-T. Chen, M.J. Buehler, De novo composite design based on machine learning algorithm, Extreme Mechanics Letters 18 (2018) 19–28.
[31] M. Mozaffar, R. Bostanabad, W. Chen, K. Ehmann, J. Cao, M. Bessa, Deep learning predicts path-dependent plasticity, Proc. Natl. Acad. Sci. Unit. States Am. 116 (2019) 26414–26420.
[32] H. Nguyen, T. Vu, T. P. Vo, H.-T. Thai, Efficient machine learning models for prediction of concrete strengths, Construct. Build. Mater. 266 (????) 120950.
[33] Y. Yu, T. Hur, J. Jung, I.G. Jang, Deep learning for determining a near-optimal topological design without any iteration, Struct. Multidiscip. Optim. 59 (2019) 787–799.
[34] P. Hajela, L. Berke, Neurobiological computational models in structural analysis and design, Comput. Struct. 41 (1991) 657–667.
[35] D.W. Abueidda, S. Koric, N.A. Sobh, Topology optimization of 2d structures with nonlinearities using deep learning, Comput. Struct. 237 (2020) 106283.
[36] H.T. Kollmann, D.W. Abueidda, S. Koric, E. Guleryuz, N.A. Sobh, Deep learning for topology optimization of 2d metamaterials, Mater. Des. 196 (2020) 109098.
[37] P. Hajela, L. Berke, Neural network based decomposition in optimal structural synthesis, Comput. Syst. Eng. 2 (1991) 473–481.
[38] L. Berke, P. Hajela, Applications of artificial neural nets in structural mechanics, Struct. Optim. 4 (1992) 90–98.
[39] L. Liang, M. Liu, C. Martin, W. Sun, A deep learning approach to estimate stress distribution: a fast and accurate surrogate of finite-element analysis, J. R. Soc. Interface 15 (2018) 20170844.
[40] S. Lee, J. Ha, M. Zokhirova, H. Moon, J. Lee, Background information of deep learning for structural engineering, Arch. Comput. Methods Eng. 25 (2018) 121–129.
[41] T.T. Truong, D. Dinh-Cong, J. Lee, T. Nguyen-Thoi, An effective deep feedforward neural networks (dfnn) method for damage identification of truss structures using noisy incomplete modal data, Journal of Building Engineering 30 (2020) 101244.
[42] A. Chandrasekhar, K. Suresh, TOuNN: Direct Topology Optimization Using Neural Networks, Submitted to Struct Multidisc Optim, 2020, p. 15.
[43] Y. Zhou, H. Zhan, W. Zhang, J. Zhu, J. Bai, Q. Wang, Y. Gu, A new data-driven topology optimization framework for structural optimization, Comput. Struct. 239 (2020) 106310.
[44] B. Li, C. Huang, X. Li, S. Zheng, J. Hong, Non-iterative structural topology optimization using deep learning, Comput. Aided Des. 115 (2019) 172–180.
[45] A. Yildiz, N. Öztürk, N. Kaya, F. Öztürk, Integrated optimal topology design and shape optimization using neural networks, Struct. Multidiscip. Optim. 25 (2003) 251–260.
[46] C.-B. Yun, E.Y. Bahng, Substructural identification using neural networks, Comput. Struct. 77 (2000) 41–52.
[47] M. Crisfield, Non-linear Finite Element Analysis of Solids and Structures: Advanced Topics, Non-linear Finite Element Analysis of Solids and Structures, Wiley, ????
[48] M. Greco, F. Gesualdo, W. Venturini, H. Coda, Nonlinear positional formulation for space truss analysis, Finite Elem. Anal. Des. 42 (2006) 1079–1086.
[49] H.B. Coda, R.R. Paccola, 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 (2014) 1–15.
[50] E. Riks, The Application of Newton's Method to the Problem of Elastic Stability, 1972.
[51] E. Riks, An incremental approach to the solution of snapping and buckling problems, Int. J. Solid Struct. 15 (1979) 529–551.
[52] R.G. Regis, A survey of surrogate approaches for expensive constrained black-box optimization, in: Optimization of Complex Systems: Theory, Models, Algorithms and Applications, 2020, pp. 37–47.
[53] C.B. Niutta, E. Wehrle, F. Duddeck, G. Belingardi, Surrogate modeling in design optimization of structures with discontinuous responses, Struct. Multidiscip. Optim. 57 (2018) 1857–1869.
[54] F. Boukouvala, M.M.F. Hasan, C.A. Floudas, Global optimization of general constrained grey-box models: new method and its application to constrained PDEs for pressure swing adsorption, J. Global Optim. 67 (2017) 3–42.
[55] G. Arpat, F. Gümrah, B. Yeten, The neighborhood approach to prediction of permeability from wireline logs and limited core plug analysis data using backpropagation artificial neural networks, J. Petrol. Sci. Eng. 20 (1998) 1–8.
[56] K.M. Hamdia, X. Zhuang, T. Rabczuk, An efficient optimization approach for designing machine learning models based on genetic algorithm, Neural Comput. Appl. (2020) 1–11.
[57] C. Nwankpa, W. Ijomah, A. Gachagan, S. Marshall, Activation Functions: Comparison of Trends in Practice and Research for Deep Learning, 2018. arXiv preprint arXiv 1811.03378.
[58] Q. Wang, Y. Ma, K. Zhao, Y. Tian, A comprehensive survey of loss functions in machine learning, Annals of Data Science (2020) 1–26.
[59] L. Deng, J. Li, J.-T. Huang, K. Yao, D. Yu, F. Seide, M. Seltzer, G. Zweig, X. He, J. Williams, et al., Recent advances in deep learning for speech research at microsoft, in: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2013, pp. 8604–8608.
[60] J. Duchi, E. Hazan, Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization, J. Mach. Learn. Res. 12 (2011).
[61] M.D. Zeiler, Adadelta: an Adaptive Learning Rate Method, 2012. arXiv preprint arXiv 1212.5701.
[62] G. Hinton, N. Srivastava, K. Swersky, Neural Networks for Machine Learning, Coursera, Video Lectures 264 (????).
[63] D.P. Kingma, J. Ba, Adam, A Method for Stochastic Optimization, 2014. arXiv preprint arXiv 1412.6980.
[64] D.T. Do, D. Lee, J. Lee, Material optimization of functionally graded plates using deep neural network and modified symbiotic organisms search for eigenvalue problems, Compos. B Eng. 159 (2019) 300–326.
[65] S. Lee, S. Park, T. Kim, Q. X. Lieu, J. Lee, Damage quantification in truss structures by limited sensor-based surrogate model, Appl. Acoust. 172 (????) 107547.
[66] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (2014) 1929–1958.
[67] V. Ho-Huu, T. Nguyen-Thoi, T. Vo-Duy, T. Nguyen-Trang, An adaptive elitist differential evolution for optimization of truss structures with discrete design variables, Comput. Struct. 165 (2016) 59–75.
[68] V. Ho-Huu, T. Nguyen-Thoi, L. Le-Anh, T. Nguyen-Trang, An effective reliability-based improved constrained differential evolution for reliability-based design optimization of truss structures, Adv. Eng. Software 92 (2016) 48–56.
[69] V. Ho-Huu, T. Nguyen-Thoi, M. Nguyen-Thoi, L. Le-Anh, An improved constrained differential evolution using discrete variables (d-icde) for layout optimization of truss structures, Expert Syst. Appl. 42 (2015) 7057–7069.
[70] J. Bonet, A.J. Gil, R.D. Wood, Worked Examples in Nonlinear Continuum Mechanics for Finite Element Analysis, Cambridge University Press, 2012.
[71] C. Krishnamoorthy, G. Ramesh, K. Dinesh, Post-buckling analysis of structures by three-parameter constrained solution techniques, Finite Elem. Anal. Des. 22 (1996) 109–142.
[72] K. Choong, Y. Hangai, Review on methods of bifurcation analysis for geometrically nonlinear structures, Bulletin of the International Association for Shell and Spatial structures 34 (1993) 133–149.
[73] Q.X. Lieu, D.T. Do, J. Lee, An adaptive hybrid evolutionary firefly algorithm for shape and size optimization of truss structures with frequency constraints, Comput. Struct. 195 (2018) 99–112.