Communications in Mathematics and Statistics (2022) 10:543–564
https://doi.org/10.1007/s40304-021-00253-8

Isogeometric Topology Optimization Based on Deep Learning

Taining Zheng[1] · **Xin Li**[1]

Received: 5 February 2021 / Revised: 20 May 2021 / Accepted: 9 June 2021 /
Published online: 9 July 2022

Abstract
Topology optimization plays an important role in a wide range of engineering applications. In this paper, we propose a novel isogeometric topology optimization algorithm based on deep learning. Unlike the other neural network-based methods, the density distributions in the design domain are represented in the B-spline space. In addition, we use relatively novel technologies, U-Net and DenseNet, to form the neural network structure. The 2D and 3D numerical experiments show that the proposed method has an accuracy rate of over 97% for the final optimization results. After training, the new approach can save time greatly for the new topology optimization compared with traditional solid isotropic material with penalization method and IGA method. The approach can also overcome the checkerboard phenomenon.

Keywords Topology optimization · Deep learning · Isogeometric analysis · B-splines · SIMP

Mathematics Subject Classification 65D07 · 65M15

1 Introduction

Topology optimization is a mathematical method to solve a material layout problem for the given design domain, loading, and boundary conditions. It plays a huge role in aerospace, automobile manufacturing, bridge construction, and other fields. Due to the importance in the engineering field, many methods have emerged in the past few

✉ Xin Li
lixustc@ustc.edu.cn

Taining Zheng
zhengtn@mail.ustc.edu.cn

[1] School of Mathematical Science, University of Science and Technology of China, Hefei, People's Republic of China

decades. In 1988, Bendsøe and Kikuchi [2] proposed a numerical method for structural topology optimization. Then, the research of topology optimization has entered a new situation and lots of classic methods have been proposed, such as the solid isotropic material with penalization (SIMP) method [3], the evolutionary structural optimization method [37,38], the level-set method [32,35], and the moving morphable components method [12], topology optimization based on phase field [6], and isogeometric analysis (IGA)-based topology optimization [13,26]. The most popular method is the SIMP method, which assumes that the material properties are constant within each element used to discretize the design domain, and the element densities act as design variables. In practical applications, the method has disadvantages of high computational cost, checkerboard phenomenon, and mesh dependency. Thus, some researchers applied IGA to topology optimization [17,31]. The main idea of IGA is to use the basis functions from computer-aided design (CAD), such as non-uniform rational B-spline (NURBS) [7], T-splines [1], and subdivision [21], to perform numerical analysis. IGA-based topology optimization can improve calculation accuracy and overcome the checkerboard phenomenon. Hassani et al. [13] applied IGA to structural topology optimization, in which the NURBS basic functions represent the material density. Qian [26] used tensor product B-spline functions to represent the density field, which have inherent filters. This method can effectively eliminate numerical artifacts and obtain high-resolution topological structures. Kang and Youn [18] expressed the shell structure by introducing the NURBS trimming curve and performed isogeometric topology optimization. The isogeometric analysis of NURBS was also combined with level set method [36] and moving morphable components method [15,41].

In recent years, machine learning techniques have been widely used in speech recognition, self-driving cars, computer vision, medical, financial, and other fields. These machine learning techniques have also been applied in engineering [25], including topology optimization. Sosnovik and Oseledets [33] converted the topology optimization into an image segmentation problem and mapped the previous topology optimization structure to the optimal design. The approach can speed up the process and has very good generalization ability. However, the authors did not consider the topological structure constraints. Then, [27,28] introduced generative adversarial networks (GAN) technology into the topology optimization. They also proposed a novel method based on conditional wasserstein generative adversarial networks (CWGAN). Saurabh et al. [30] used the deep learning method of encoder–decoder convolutional neural network architecture in 3D topology optimization. Yu et al. [39] proposed an algorithm of combining two neural networks for topology optimization, where the first network is based on convolutional neural networks (CNN) to obtain a low-resolution structure and the second one is a conditional generative adversarial networks to get the high-resolution near-optimal structure. Deng and To [8] proposed a topology optimization method based on geometry deep learning, which can significantly reduce design variables.

In this paper, we propose a novel IGA topology optimization based on deep learning. Unlike other deep neural networks, we use the U-Net and DenseNet to form the neural network architecture. And the proposed approach uses neural networks to directly learn the spline representation of the topology. The approach can effectively eliminate the checkerboard phenomenon without redividing the design domain to obtain a high-

resolution topology structure. Due to the representation of the spline function, we can reduce design variables. The 2D and 3D numerical experiments show that the proposed method has an accuracy rate of over 97% for the final optimization results. Our proposed method can save time greatly compared with the traditional topology optimization methods after training.

The rest of the paper is organized as follows. Section 2 presents a short introduction to B-splines and neural networks. Section 3 introduces the conventional topology optimization approaches. Section 4 proposes our deep learning methods in detail, and Sect. 5 shows some 2D and 3D numerical experiments. The last section is the conclusions and future work.

2 Preliminary

This section provides some preliminary concepts related with the present paper, including B-splines and neural networks.

2.1 B-Splines

B-spline basis functions $B_i^p(\xi)$ of degree p for a knot vector $\Xi = \{\xi_0, \xi_1, \ldots, \xi_{n+p+1}\}$ are defined recursively as Eq. (2.2).

For $p = 0$,

$$B_i^0(\xi) = \begin{cases} 1, & \text{if } \xi_i \le \xi < \xi_{i+1}, \\ 0, & \text{otherwise}, \end{cases} \tag{2.1}$$

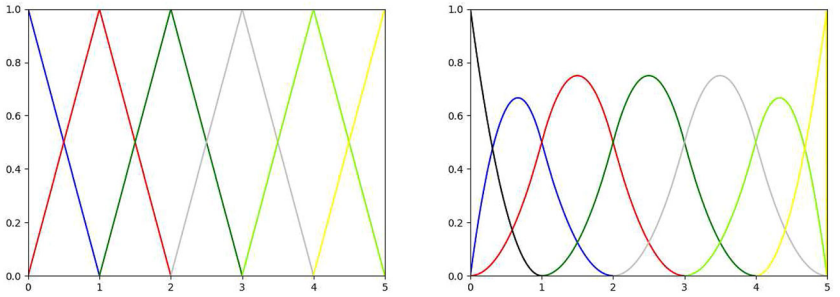
and for $p \ge 1$,

$$B_i^p(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} B_i^{p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} B_{i+1}^{p-1}(\xi). \tag{2.2}$$

B-spline basis functions $B_{i,j}^{p,q}(\xi, \eta) = B_i^p(\xi) B_j^q(\eta)$, where $B_i^p(\xi)$ and $B_j^q(\eta)$ are B-spline basis functions of degree p and q, corresponding to knot vectors $\Xi = \{\xi_0, \xi_1, \ldots, \xi_{n+p+1}\}$ and $\mathcal{H} = \{\eta_0, \eta_1, \ldots, \eta_{m+q+1}\}$, respectively. Some examples of B-spline basis functions for degree $p = 1, 2$ formed from knot vectors $\Xi = \{0, 0, 0, 1, 2, 3, 4, 5, 5, 5\}$ are shown in Fig. 1.

2.2 Neural Networks

The research of artificial neural network started in the 1940s [23]. After making progress in the study of biological neurons, the researchers used mathematical and physical methods to simulate the human brain nervous system model. An artificial neuron is the basic processing unit of a neural network. It is a structural unit with multiple inputs and an output value through an activation function. The artificial neural networks are nonlinear, self-adaptive, and have strong learning ability. At present,



(a) $p = 1$ (b) $p = 2$

Fig. 1 B-spline basis functions of order $p = 1$ (**a**) and $p = 2$ (**b**), the knot vectors $\Xi = \{0, 0, 0, 1, 2, 3, 4, 5, 5, 5\}$

neural networks are used in various aspects, including signal processing, pattern recognition, optimized combination, expert systems.

Convolutional neural networks [20] are feed-forward neural networks with excellent performance in large-scale image processing. It has been widely adopted in many fields such as image classification, positioning, and other fields. Compared with other neural network architectures, the convolutional neural networks require relatively few parameters. The modern CNN architecture contains many techniques, such as pooling layer and dropout for regularization, to improve the robustness [11]. Thus, in CNN, the convolutional layer, pooling layer, and full connection layer are commonly used, which form a multi-layer network and perform feature learning on the input data. Neural networks have powerful approximation capabilities [14], and the universal approximation theorem shows that it can approximate continuous functions on compact subsets of R, under mild assumptions on the activation function. Experience shows that deeper networks can work better than single hidden layer networks.

3 Topology Optimization

In this section, we briefly introduce the standard topology optimization methods mentioned in introduction. Topology optimization is the process to design the material distribution, which aims to minimize the amount of materials under various realistic conditions. We restrict the design domain to be D, and set the area occupied by the structure as Ω. We hope to obtain the design variable $\boldsymbol{x}$, which can satisfy various constraints and minimize the objective function $c(\boldsymbol{x})$. The optimization problem can be written into the form as

$$\begin{cases} \text{Find} \quad \boldsymbol{x} \in \mathcal{U}_{\boldsymbol{x}} \\ \text{Minimize}: \quad c = c(\boldsymbol{x}) \\ \text{s.t.} \quad G_i(\boldsymbol{x}) \leq 0, \ i = 0, 1, 2, \ldots, m, \\ \qquad H_j(x_j) \leq 0, \ j = 1, 2, 3, \ldots, n, \end{cases} \tag{3.1}$$

where $G_i(\boldsymbol{x})$ represents design constraints, such as stress, strain, and volume constraints. H_j is the constraint on the design variable x_j itself, which has a total of n design variables. $\mathcal{U}_{\boldsymbol{x}}$ is the set of values of the design variables $\boldsymbol{x}$. Particularly, the volume constraint is one of the most important constraints, and we use G_0 to specifically refer to the volume constraint. Many different approaches have been proposed in these years, including homogenization method [2], evolutionary structural optimization method [37], solid isotropic material with penalization method [3], level set method [32], morphable components method [12].

One of the most widely used models in the density-based approach is the SIMP. The design variables are material density ρ_e of each element, and the penalty method is used to limit the density between 0 and 1, which force solutions to a suitable 0/1 problem. Then, the above stiffness tensor in classical SIMP can be written as $E_{ijkl} = \rho(\boldsymbol{x})^p E^0_{ijkl}$. The tensor E^0_{ijkl} is the stiffness tensor for the given isotropic material, and $p > 1$ is the penalty factor. The formulation for minimum compliance problem based on finite element method is provided below.

$$\begin{aligned} \underset{\boldsymbol{\rho},\mathbf{u}}{\text{Minimize}}: \quad & c(\boldsymbol{\rho},\mathbf{u}) = \mathbf{f}^T\mathbf{u} \\ \text{s.t.}: \quad & K(\boldsymbol{\rho})\mathbf{u} = \mathbf{f}, \\ & \sum_{e=1}^{N_e} \rho_e V_e \le V_0, \\ & 0 < \rho_{\min} \le \rho_e \le 1, \quad e = 1, 2, \ldots, N_e, \end{aligned} \tag{3.2}$$

where $\mathbf{u}$ and $\mathbf{f}$ are the displacement and force vectors. K is the global stiffness matrix, and the number of elements is N_e, which used to discretize the design domain. V_e is the volume of element e; $\rho_{\min}$ is a very small positive number to avoid problems related to vanishing stiffness, usually taken as 0.001.

In actual work, the problem of topology optimization is extremely large. For example, in the variable density method, a three-dimensional problem has easily the design variables of one hundred thousand or one million scale. Most optimization algorithms and optimization solvers may be helpless in the face of such a large-scale problem. In most cases, only the first derivative (the so-called sensitivity) information can be obtained. This determines that it is unlikely to adopt methods that strongly rely on the second derivative. Every time the objective function/constraint condition is calculated, a finite element analysis needs to be performed completely, which consumes too much time. Machine learning, especially deep learning, can solve specific topology optimization problems and achieve outstanding results. It has been recorded that the new method saves a lot of time than the traditional method and can obtain the result of topology optimization in a very short time. For example, in [39], two networks are used, CNN and the other the CGAN, with an accuracy rate of over 97% and time-saving of over 99%. Besides, as in [28], the calculation time of the traditional method is significantly reduced to 10^2 orders of magnitude, although the generalization ability of neural networks is not better.

4 Deep Learning Methods

In this section, we introduce the isogeometric topology optimization method based on deep learning in details.

4.1 Topology Optimization in the B-Spline Space Based on Neural Network

We first review the IGA topology optimization based on B-spline space [26]. In this method, the design domain D is restricted to the B-spline space. Therefore, the density distributions are represented as a continuous B-spline function $g(\boldsymbol{x})$ that is parameterized by a finite number of B-spline coefficients. In two dimensions, $g(x, y) = \sum_{i=0}^{m} \sum_{j=0}^{n} B_i^p(x) B_j^q(y) p_{ij}$, where p_{ij} is the B-spline coefficient. And Eq. (3.2) can be expressed as follows.

$$\begin{aligned} \underset{P}{\text{Minimize}}: \quad & c(\boldsymbol{\rho}, \mathbf{u}) = \mathbf{f}^T \mathbf{u} \\ \text{s.t.}: \quad & \sum_{e=1}^{N_e} \int_{\Omega^e} \rho_e(\boldsymbol{x}) \mathrm{d}V \leq V_0, \quad \boldsymbol{x} \in D, \\ & K(\boldsymbol{\rho})\mathbf{u} = \mathbf{f}, \\ & \rho_e(\boldsymbol{x}) = h_e(g(\boldsymbol{x})), \end{aligned} \tag{4.1}$$

where P represents all the B-spline coefficients, which are design variables in the method. $\mathrm{f}, K, \mathrm{u}$ are the same as in Eq. (3.2). For the design domain, the density distribution of the element e in the D is obtained by the function h_e that represents the analysis approximation of continuous B-spline function $g(\boldsymbol{x})$. ρ_e represents the density distribution of each element e in the discrete case.

In (4.1), the stiffness matrix K depends on the stiffness E_e in each element e, i.e., $K = \sum_{e=1}^{N_e} K_e$, where K_e is the element stiffness matrix. The Youngs modulus is parameterized as $E_e(\rho_e) = \rho_e(\boldsymbol{x})^p E^0$, where E^0 is the original Youngs modulus of the material in the solid phase and p is the penalization parameter. ρ_e has a lower bound as the above formula (3.2). Thus, c can be written as $c = \mathbf{f}^T \mathbf{u} = \mathbf{u}^T K \mathbf{u} = \sum_{e=1}^{N_e} \mathbf{u}_e^T (E_e(\rho_e) K_e^0) \mathbf{u}_e$. Here $\mathbf{u}_e$, K_e^0 are the element displacement vector and the element stiffness matrix of the solid material.

In the gradient-based optimization method, sensitivity analysis of the objective function and constraints is required. Referring to [4,9,24], the sensitivity of the objective function c to the variable ρ_e is

$$\frac{\partial c}{\partial \rho_e} = -\mathbf{u}^T \frac{\partial K}{\partial \rho_e} \mathbf{u} = -p \rho_e^{p-1} E^0 \mathbf{u}_e^T K_e^0 \mathbf{u}_e. \tag{4.2}$$

In this paper, we use the central density value x_e^{center} of the eth element as the value of this element. Therefore, the element density is $\rho_e(x) = g(x_e^{\text{center}})$. And then we have

$$\frac{\partial c}{\partial P_i} = \sum_e \frac{\partial c}{\partial \rho_e}\frac{\partial \rho_e}{\partial P_i} = \sum_e \frac{\partial c}{\partial \rho_e}\frac{\partial \sum N_i(x_e^{\text{center}})P_i}{\partial P_i} = \sum_e \frac{\partial c}{\partial \rho_e} N_i(x_e^{\text{center}}), \quad (4.3)$$

where e indicates that the density distribution of this element is affected by P_i. $g(x_e^{\text{center}})$ is composed of a finite number of B-spline tensor product N_i and the corresponding coefficient P_i.

Different from updating P through iterations, we find the best P^* by updating the parameters θ in the neural networks. And then we express the density distribution through the B-spline functions to generate the optimal structure. Based on the above analysis, we can get the sensitivity analysis of the objective function to network parameters θ, and $\frac{\partial c}{\partial \theta} = \frac{\partial c}{\partial P}\frac{\partial P}{\partial \theta}$. Here $\partial P/\partial\theta$ can be obtained by the backpropagation algorithm of the neural networks. P^{pred} can be defined as a function of P, P_0 and θ, denoted as $N(P, P_0, \theta)$, where P, P_0 are input and θ can be updated as

$$\hat{\theta} = \arg\min_{\theta} L(P^{target}, N(P, P_0, \theta))) + \lambda J(N). \quad (4.4)$$

In the equation, L, J are loss function and the function of model complexity, respectively. The parameter λ controls the balance between the goal of fitting training and the goal of keeping the parameter values small. L function is used to estimate the gap between the outputs of the neural networks P^{pred} and P^{target}, which guide the optimization of the model. $J(N)$ measures the complexity of the model and keeps the parameters small to prevent over-fitting, also called regularization technology.

4.2 The Architecture of the Neural Networks

In this section, we introduce the architecture of the neural network in details. The existing optimizing neural networks are based on the idea of backpropagation, which lead to some problems, including vanishing/exploding gradients [5]. In order to prevent such problems and find a suitable neural network, we construct a deep convolutional neural network [19], which is a combination of U-Net [29] and DenseNet [16]. On the one hand, it can increase the robustness to some small disturbances of the inputs and reduce the risk of overfitting, and the U-shaped architecture can be restored to its original size. On the other hand, the dense blocks in DenseNet can reduce the vanishing gradient problem.

As illustrated in Fig. 2, the deep fully convolutional neural networks are proposed. The networks could be divided into two parts: encoder and decoder. The encoder mainly includes the down-sampling process, and the decoder includes the up-sampling process. As we known, P_{ij} affects several elements in the design domain D. If P_{ij} is relatively large, it can affect the density value of e. Therefore, in the down-sampling process, the larger P_{ij} can be retained in the deep feature map. And dense blocks are used to better display these features.

In a standard convolutional neural network, the output of the lth layer is denoted as x_l, which is obtained from the output x_{l-1} of the previous layer through a nonlinear transformation H_l, $x_l = H_l(x_{l-1})$, where H_l generally includes batch normalization, activation function, and convolution operation. Referring to Fig. 3, in a densely con-

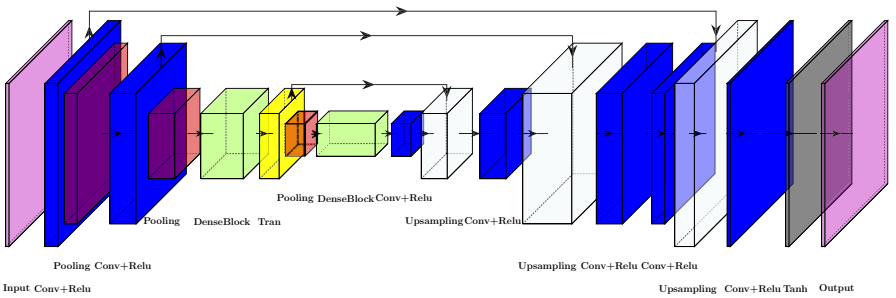


Fig. 2 Architecture of the proposed neural network for topology optimization. The first purple block represents the input layer. The remaining purple blocks are all convolutional layers (the activation function ReLU follows); the red block represents the pooling layer; the green block represents the dense block; the white block is the up-sampling layer, finally through the activation function Tanh

nected network, all feature outputs are concatenated in a feed-forward manner, i.e., $x_l = H_l([x_0, x_1, \ldots, x_{l-1}])$, where $[x_0, x_1, \ldots, x_{l-1}]$ means to concatenation the outputs feature maps from the 0th layer to the $(l-1)$th layer.

In the neural networks, all convolutional layers are followed by batch normalization and ReLU nonlinearities, which the 3×3 kernels are used except in the dense block. The decoder comprises classic convolutional layers and reverse convolutional layers. There is a convolutional layer between dense block and before the decoding blocks. The purpose is to map the feature with deep channels to the feature with shallow channels and keep the number of data channels consistent. We use max-pooling in the decoding part, which the window size of 2×2. Transition layers are used between two dense blocks, aiming to reduce the number of feature maps. In the dense block, each function H_l produces k feature maps; it follows that the lth layer has $k_0 + k \times (l-1)$ input feature maps, where k_0 is the number of channels in the input layer. The growth rate for dense block is $k = 64$. In order to prevent over fitting, we adopt the dropout, and the parameter is set to 0.2, indicating that there is a 20% random probability of "discarding neurons."

4.3 Inputs and Outputs

For a specific topology optimization problem, it is difficult to use a completely accurate B-spline to obtain the corresponding topology structure. Under the same conditions, using methods (3.2) and (4.1), the results are slightly different. Therefore, in order to be consistent with other papers, we approximate the data generated by the SIMP method with the uniform B-spline functions to obtain the coefficients in the two-dimensional situation.

In the two-dimensional situation, the optimal structure is the results after 100 iterations in the SIMP for most of the topology optimization problems can get better solutions after 100 iterations. Therefore, we can use the optimal structure to get the corresponding coefficients as the target P^{target}. The inputs of the neural networks includes two parts: P and P_0. P refers to the B-spline coefficients after n iterations in the early stages of the traditional method. P_0 denotes the difference between the coef-

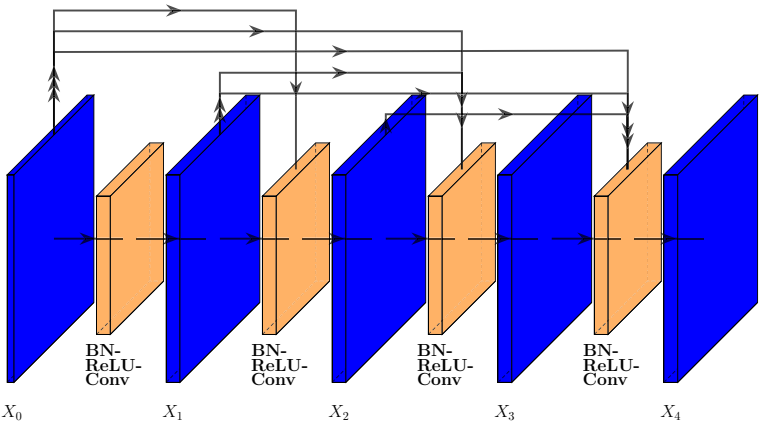


Fig. 3 A 5-layer dense block with a growth rate of $k = 4$. Each layer takes all preceding feature maps as input

ficients after the nth and $(n-1)$th iterations. Since we must obtain the initial topology to get the training data, the strategy we used is that the number of initial iteration stops $\boldsymbol{n}$ obeys the Poisson distribution, in which the parameter $\lambda = 10$. We multiply P_0 by m ($m = 1, 2, 4, \ldots$) to make the difference more obvious. P_0 is the change value of the coefficients P, which can help the networks recognize the direction of changes. Our experiments show that if there is no P_0, the overall results will be poor. It is also possible to obtain prediction results directly through P without using P_0. Thus, the inputs of the neural network are $[P, 2 \times P_0]$ (see Fig. 5), the outputs are the predicted coefficients, and the target is P^{target} in 2D case. In 3D case, the number of iterations $I = 120$, and the number of initial iteration stops n obeys the Poisson distribution, where $\lambda = 12$.

4.4 Dataset

In this section, we illustrate the format of datasets. We used the SIMP method to generate the training, validation, and testing data for our proposed models. In Eq. (4.1), a topology optimization problem is mainly determined by volume constraint and boundary conditions. We design the following sampling strategy to define the volume, load, and boundary conditions of the topology optimization problem.

(1) The volume fraction is sampled from the uniform distribution between 0.2 and 0.8.
(2) The number of nodes with fixed x and y translations is sampled from the Poisson distribution: $N \sim P(\lambda = 2)$, and the probability of these nodes appearing at the boundary of the design domain is ten times that of the interior.
(3) The number of loads is sampled from the uniform distribution between 0 and 6, and values are chosen as -1.
(4) The penalty factor is set 3.

Fig. 4 **a** Topology optimization problem of a cantilever beam. **b** Result after 100 iterations using the SIMP method. **c** Continuous B-spline functions approximated in each element in (**b**)

Fig. 5 **a** and **b** are inputs of the neural networks. **a** is the coefficients distribution corresponding to the 10th iteration. **b–d** are difference between the P obtained after the 10th iteration and 9th iteration

We generated 10,000 samples based on the above conditions divided into three parts, training set, validation set, and test set. The ratio of the three sets is 8:1:1. Figure 4a shows an example of an optimization problem with the volume fraction 0.355, (b) shows the result after 100 iterations using the SIMP method, and (c) represents a continuous topological structure for the problem. The design domain size of the sample is 40×40. The degree of the B-splines basis function is bi-cubic. The coefficients are the same size as the number of elements that can be obtained because B-splines domain can be constructed from the bounding box of the design domain. Figure 5 shows the inputs of the neural network, where (a) represents the coefficients and (b) is the difference between the P obtained after the 10th iteration and 9th iteration ($2 \times P_0$); the rest represents $m \times P_0$, where $m = 1, 4$. In our paper, we choose $m = 2$. For high-precision requirements, our method can significantly accelerate and have no checkerboard phenomenon.

4.5 Training

Our network parameters are optimized via Adam with $\beta 1$ and $\beta 2$ equal to 0.9 and 0.999, respectively, and training is run during 200 epochs. The learning rate is initially set to 0.001 and after 100 epochs is 0.0005. Mini-batch size is 16 to improve computing efficiency. The proposed architectures are implemented in pytorch. The optimization of the parameter θ in the above formula (4.4) requires L and J. In order to reduce the distance between the predicted data P^{pred} and the target data P^{target}, each data

P_i^{pred} tend to the corresponding target data P_i^{target}; the squared loss function is used, see (4.5). For the J function, we use the most widely used L_2 norm,

$$\begin{aligned} L(P^{\text{target}}, P^{\text{pred}}) &= \sum (P_i^{\text{target}}, P_i^{\text{pred}})^2, \\ J(N) &= \frac{1}{2} \sum_i \theta_i^2. \end{aligned} \tag{4.5}$$

Experiments are performed on a computer with an Intel(R) Core(TM) i7-9700k CPU and an NVIDIA GeForce GTX 2080 Ti GPU. While training took around 4 h, predict that all samples take 36.06 s on average. After training, to compute the density in current research, it takes only about 298.42 s for 1000 cases, which all time consumption is less than 5% of the traditional methods. See Table 1 for more details. The time consumed includes the time to calculate the accuracy (pixel accuracy) and the time to obtain and save the topology optimized structures.

5 Results and Discussions

5.1 2D Numerical Examples

In this section, we provide the performance of the proposed neural networks and comprehensive comparison with the traditional method. To facilitate the comparison with the optimal structure, we make the predicted results in a continuous topological structure and then take the value of the corresponding element $\boldsymbol{e}$ in the design domain. The typical indicator pixel accuracy (PA) is adopted to evaluate the results

$$\text{PA} = \frac{\sum_{i=0}^{k} p_{ii}}{\sum_{i=0}^{k} \sum_{j=0}^{k} p_{ij}}, \tag{5.1}$$

where $i, j \in \{0, 1\}, k = 1$. p_{ii} is the total number of pixels with real pixel category i being predicted to be category i and p_{ij} being the total number of pixels whose real pixel category is i being predicted to be category j. The higher of the p_{ii}, the better the prediction effect, and the higher the p_{ij}, the poorer prediction effect. We adopted such evaluation is based on pixelwise comparison. In our paper, not only pixel-based comparisons and the efficiency of the method, but also important constraints in topology optimization are introduced, including the comparison of volumes and, more importantly, the comparison of objective functions.

Tables 1, 2 , 3 and Fig. 6 reflect the performance of the proposed method. In Table 1, the second and third columns record the total time of the test samples and the average time of each sample of the two methods. On average, the time consumed by the traditional method for a sample is 7.84 s, while in our method, it is only 0.33 s. The last column of Table 1 is the time-saving ratio of the proposed method compared to the SIMP method. It can be seen from the table that the proposed method can indeed improve efficiency. We know that the efficiency of deep learning is based on training and processing data. The training process is actually a mathematical modeling

Table 1 Comparison of computational time between SIMP method and the proposed method

Method	Time consumed by all test samples	Average time consumed by test samples	Calculation time saved
SIMP method	7843.34 s	7.84 s	–
Proposed method	36.06 s+298.42 s	0.33 s	95.8%

Table 2 Comparison of pixel accuracy between SIMP method and the proposed method

Average PA	Worst PA	Best PA	Proportion of bad samples (PA $<$ 90%)
97.2%	83.9%	100%	1.6%

Table 3 Comparison of the proposed method and SIMP method with volume constraint

	SIMP method (%)	Proposed method (%)
Exceeding volume constraint	10.9	29.1
Between 100 and 105% of the volume constraint	10.9	22.6
Between 105 and 110% of the volume constraint	0.0	3.5
Exceeding 110% of volume constraint	0.0	3.0

The data in the table are the proportion of the test set

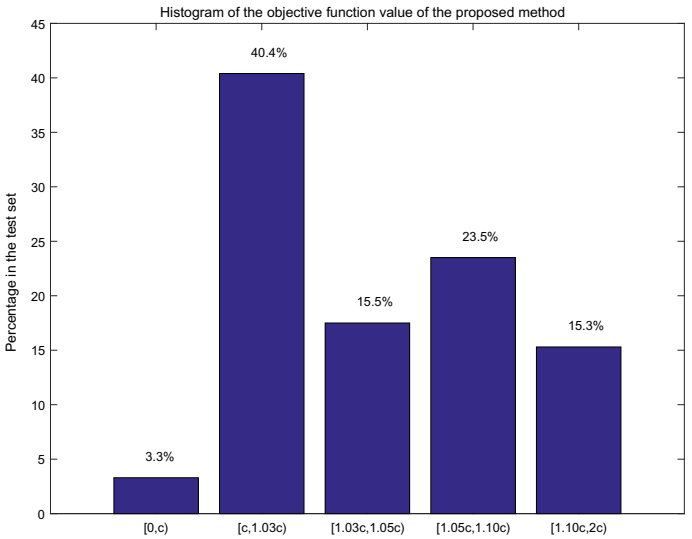


Fig. 6 Figure shows the comparison result of the objective function value obtained by the proposed method and SIMP method. The percentage in the figure represents the proportion of the total samples, and c represents the objective function value obtained by the SIMP method. For example, 40.4% means 404 samples that the objective function value obtained by proposed method is between 100 and 103% of the value obtained by the SIMP method

process for the present method. Once the model is completed, we can solve the topology optimization problem very efficiently. Thus, in the application of the topology optimization, we do not need the training step.

Table 2 records the accuracy of the test samples. In all tested samples, our results show that the average PA is 97.2%. In the worst sample, the accuracy is only 83.9%, but in the best case, it can get the same results as the SIMP method. What is gratifying is that our extremely bad result (accuracy rate of less than 90%) only accounts for 1.6%.

Table 3 reflects the volume constraint. It can be seen that the results obtained by the SIMP method almost meet the constraint, although nearly 10% of the samples exceed the constraint. However, the effect of our proposed method is poor. Almost 30% of the samples exceed the volume constraint, and nearly 7% of the samples exceed the volume constraint by more than 5%. In other words, the volume fraction error of nearly 93% of the samples can be controlled within 5%. This will be left for future work.

In addition, we are aiming to minimize the compliance problem of topology optimization. From Fig. 6, the result shows that 3.3% of the samples have lower compliance than the SIMP method. This result seems unacceptable. There is more than 50% (40.4% + 15.5%) of the samples whose objective function value is between 100 and 105% of the value in the SIMP method. This shows that in the minimize compliance problem, our proposed method does not have the SIMP method to obtain a lower objective function value. But the result is very close to the objective function value obtained by the SIMP. It can also be seen from the above PA results that our pixel accuracy can reach 97.2%, and the structure between the two methods is similar. Therefore, the objective function value is also quite close. Unfortunately, we still have 15.3% of the samples whose objective function value exceeds 10% of the SIMP objective function value.

We select some validation results randomly to evaluate our model. Figure 7 shows some cases computed by the proposed model. Each row is one case, and the first image is an optimal structure. The image in the second column is predicted by B-spline continuous functions is composed of the value of the element e in the design domain, which is the same size as the first column. The third column is the continuous expression of the predicted results. It can be seen that the predictions are closer to the optimal structure, and some examples are almost the same as the optimal structure without any difference by naked eyes. In addition, from the results of continuous predictions, we not only clearly see the trend of topology changes, but also can effectively remove the checkerboard phenomenon. Therefore, a higher-precision topology can be obtained directly from the continuous structure than in the design domain.

As expected, some results are not very satisfactory. Most of these results are concentrated in places with very thin structures or small holes, as shown in Fig. 8. Therefore, the accuracy of these examples is not very high. The possible reason is that the area near the "thin" structures is material-free, and the continuous nature of B-spline basis functions makes this area smooth, which leads to the predictions of the neural networks more biased toward the intermediate value of [0, 1]. The same is true for the small holes structures. But from the third column, the continuous B-spline basis functions can fit the contour of topology, which shows that the neural networks do have a certain learning ability for these subtle structures.

(a) c=17.57,vol=0.59,V_0=0.60 (b) c=18.07,vol=0.59,PA=97.7% (c) Continuous structure

(d) c=28.14,vol=0.37,V_0=0.38 (e) c=26.57,vol=0.39,PA=98.6% (f) Continuous structure

Fig. 7 Some random examples results of the proposal method. From left to right: (1): the ground truth; (2): predicted results; (3): predicted continuous results. In the subscript, c presents compliance, vol represents volume fraction, and V_0 represents volume constraint, PA denotes pixel accuracy. c, vol, and V_0 retain two decimal places

(a) c=22.54,val=0.55,V_0=0.56 (b) c=22.87,val=0.54,PA=95.4% (c) Continuous structure

(d) c=8.09,val=0.44, V_0=0.44 (e) c=8.17,val=0.43,PA=97.7% (f) Continuous structure

Fig. 8 Some examples of poor performance from the proposed model. The first column are the ground truth, the middle are predicted results and the last ones are continuous results

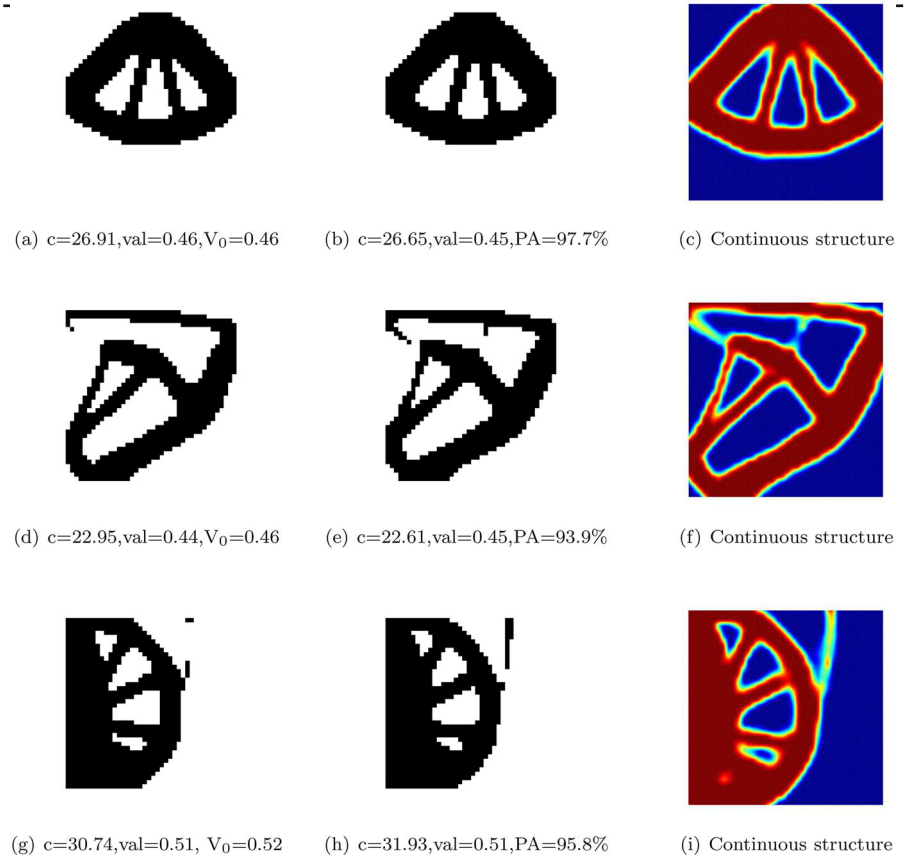


Fig. 9 Some examples of great performance from the proposed model. The first column are the ground truth, the middle are predicted results and the last ones are continuous results

We also have some satisfactory findings. Some of the predictions are better than the structures of the SIMP method. Mainly consider structural stability, in addition to volume constraint and compliance. From the examples in the first and second rows of Fig. 9, the prediction accuracy is not as good as the optimal structure, but under the volume constraint, the prediction results have lower compliance. We also can see that the support rods are not stable and even break easily in the results generated by the traditional method. A stable triangular structure is formed from the structure obtained by our neural network when the constraints are met. Although some examples are not obvious, judging from the third column, there is indeed a tendency to generate this stable structure (the example in the second row has already generated this stable structure).

5.2 3D Numerical Examples

For topology optimization that solves practical problems, some related papers on 3D topology optimization problems can be referred to [10,22,40]. We extend the

Table 4 Comparison of computational time between IGA and the proposed method (3D)

Method	Time consumed by all test samples	Average time consumed by test samples (s)	Calculation time saved
IGA method	> 24 h	352.82	–
Proposed method	12.50 s+1640.71 s	5.51	98.4 %

Table 5 Comparison of pixel accuracy between IGA method and the proposed method (3D)

Average PA	Worst PA	Best PA	Bad samples (PA < 90%)
98.7%	93.9%	99.8%	0%

Table 6 Comparison of the IGA method and the proposed method with volume constraint

	IGA method (%)	Proposed method (%)
Exceeding volume constraint	69.7	70.7
Between 100 and 105% of the volume constraint	68.7	70.0
Between 105 and 110% of the volume constraint	1.0	0.7

The data in the table are the proportion of the test set (3D)

above-mentioned method to the three-dimensional situation, and the overall method is similar to the two-dimensional situation. For the three-dimensional situation, we do not compare with the SIMP method; our results are compared with the IGA method [34].

The size of the design domain D is (40, 40, 8), and the degree of the B-splines basis function is equal to 4 in x, y, and z directions. The sampling strategy of three-dimensional case is slightly different from that of two-dimensional, including load nodes and fixed nodes which are taken from the surface of the given three-dimensional design domain D and the sampling method that meets the specified distribution. We generated 3000 three-dimensional samples and divided into three parts, including training set (2400), validation set (300), and test set (300). The neural networks adopt the above architecture, except that the data are increased by one dimension, and the other is almost unchanged.

Tables 4, 5 and 6 reflect the performance of the proposed method in three-dimensional situation. In Table 4, we can see that the time consumed to generate a three-dimensional structure is much greater than that of in two-dimensional case. It takes more than 300 s to generate a three-dimensional sample. Although the proposed method also takes more than 5 s, most of the time is spent on obtaining and saving the structure. In addition, compared with the two-dimensional situation, it can highlight that our method can save time greatly. The result of the efficiency comparison is based on the test set. Such results show that if a large number of three-dimensional topology optimization results are required in practical applications, then our method can indeed improve efficiency more than in the two-dimensional situation. Conversely, if several

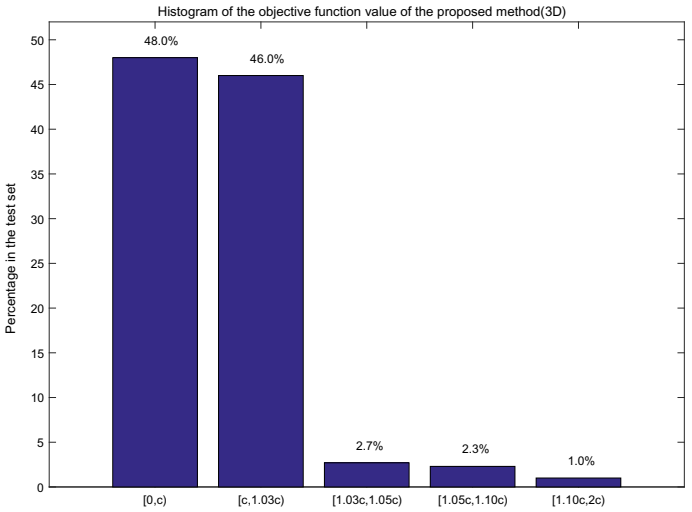


Fig. 10 Figure shows the comparison result of the objective function value obtained by the proposed method and IGA method. The percentage in the figure represents the proportion of the total sample, and c represents the objective function value obtained by the IGA method. For example, 46.0% means that the objective function value of 138 samples is between 100 and 103% of the value obtained by the IGA method

3D topology optimization results need to be obtained, then the traditional method may be more advantageous.

In Table 5, the average PA of the test samples reached an astonishing 98.7%, and the worst case PA can also reach 93.9%. The three-dimensional PA is much better than the two-dimensional. The reason is that the ground truth is generated by the IGA method, rather than the SIMP method.

In Table 6, we notice that the samples exceed the volume constraint for more than half in the IGA method. For only 0.7% of the samples, their volume fraction exceeds the prescribed volume limit of 5%, while the IGA method has 1% of the samples. This shows that in the IGA method and our proposed method, more than 99% of the samples have the volume fraction error within 5%. An important reason may be that we have too few iterations in the IGA method. In the 3D topology optimization, we chose $I = 120$. For complex three-dimensional examples, choosing $I = 120$ is still far from enough. It takes hundreds or even thousands of times to achieve satisfactory results.

As with the two-dimensional situation, we are aiming to minimize the compliance problem of topology optimization. From Fig. 10, we can see that 48.0% of the test samples have lower compliance than the IGA method. This is a good result than the 2D topology optimization. Only 6% of the samples, their objective function value error is higher than 3% of the IGA method. After comparing the results of the two methods, the values of the two are very close. This also shows that the higher the PA, the closer the objective function value. Although the accuracy in the two-dimensional situation is only about 1.5% lower than that in the three-dimensional case, it has a great influence on the objective function value.

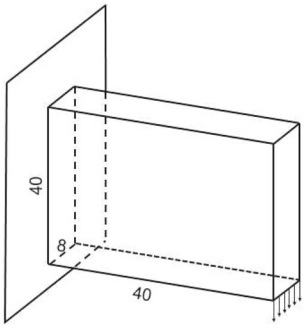
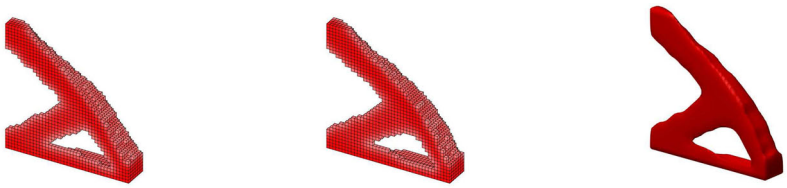


Fig. 11 A three-dimensional minimum compliance problem for the cantilevered beam. According to the unit settings in this paper, we divide it into $40 \times 40 \times 8$



(a) c=466.9,vol=0.21,V_0=0.22 (b) c=477.5,vol=0.21,PA=97.8% (c) Continuous structure

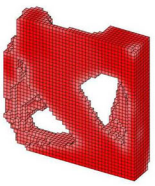
Fig. 12 Solution to the problem mentioned in Fig.11. The first column are the ground truth, the middle are predicted results and the last ones are continuous results

The reason why we compare the IGA method in the 3D topology optimization is that we directly obtain the data we need through the IGA method. And it is closer to the essence of topology optimization than the data obtained by fitting in SIMP method in the two-dimensional situation. From the analysis of the results, the fitting method does cause certain errors in the two-dimensional situation. Although we can obtain a continuous structure in the two-dimensional situation, this sacrifices some of the properties of topology optimization (such as volume fraction, objective function value, etc.). On the other hand, it is possible to analyze the pros and cons of the method proposed in this paper from multiple angles through different comparison objects.

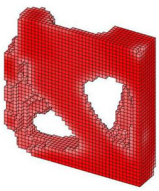
Here we show a three-dimensional minimum compliance problem for the cantilevered beam in Fig. 11. We directly gave the optimal topological structure obtained by IGA method and the structure obtained by our method, as well as the continuous structure form (see Fig. 12). We can see that when the volume constraint is satisfied, the objective function value obtained by our proposed method is about 2% higher than that of the IGA method. In addition, Fig. 13 shows some examples of 3D topology optimization.

6 Conclusions

In this paper, we proposed a neural network that combined the well-known UNet and DenseNet for solving the topology optimization problem. In this way, the networks can accept the learned features of previous layers during the encoder part. In the decoder

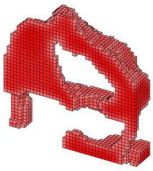


(a) c=112.0,vol=0.47,V_0=0.46

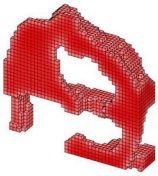


(b) c=112.8,vol=0.46,PA=98.1%

(c) Continuous structure

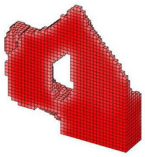


(d) c=28.0,vol=0.25, V_0=0.27

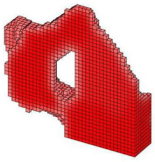


(e) c=27.7,vol=0.26,PA=97.9%

(f) Continuous structure

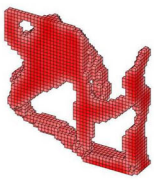


(g) c=30.7,vol=0.36, V_0=0.37

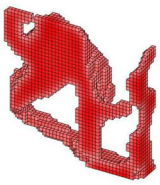


(h) c=29.6,vol=0.36,PA=99.2%

(i) Continuous structure

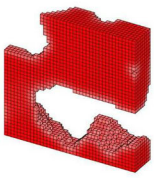


(j) c=101.1,vol=0.26, V_0=0.27

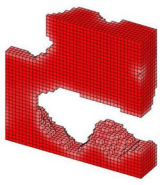


(k) c=95.7,vol=0.26,PA=96.5%

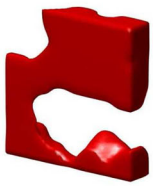
(l) Continuous structure



(m) c=133.8,vol=0.46, V_0=0.45



(n) c=133.8,vol=0.45,PA=99.5%



(o) Continuous structure

Fig. 13 Some examples of 3D topology optimization. The first column are the ground truth, the middle are predicted results and the last ones are continuous results

part, we provide the skip connection to help restore information loss caused by downsampling. The 2D and 3D experimental results show that it reaches an acceptable accuracy for minimal compliance problems. Besides, our proposed method consumes less time than traditional topology optimization methods. Compared to other neural networks used in topology optimization, the inputs are the coefficients of uniform B-splines and the change of the coefficients from the initial structure with the traditional method, not the density distribution. The reason is that the B-spline's continuous nature is used, and the checkerboard phenomenon can be effectively eliminated. The proposed method is mainly to quickly obtain a rough structure and thus has great advantages for engineering problems that require a large number of topology optimization results.

At present, the method we proposed is a new attempt with deep learning in topology optimization. Therefore, there are some problems with the method. First, this work's limitations are that rely on the first few iterations and choice of B-spline degrees and knot spans. We have to go through the initial iteration to get the intermediate results and approximate each element with the continuous B-spline functions. There is also a critical shortcoming, which is also mentioned in other papers by Yu et al. [39]. Some examples have structural disconnection phenomenon. It is also not to be ignored that we do not take the volume constraint into consideration in the proposed method. In addition, the time consumption of training and preparing data in deep learning is also a factor that needs to be considered.

In future work, we must overcome the shortcomings mentioned above, primarily the phenomenon of structural disconnection. We can take advantage of the spline function properties to handle topology optimization problems with different design domain sizes. In real life, topology optimization's design variables are enormous; especially for three-dimensional problems, the design variables may reach millions. We will conduct further research to optimize the method we proposed, reduce the number of variables, and speed up the process of topology optimization, hoping to be applied in practice in the future.

Acknowledgements The authors would like to thank the scholars for their valuable suggestions and useful comments contributed to the final version of this paper. The authors were supported by the National Key R&D Program of China (2020YFB1708900), NSF of China (No. 61872328), and the Youth Innovation Promotion Association CAS.

References

1. Bazilevs, Y., Calo, V.M., Cottrell, J.A., Evans, J.A., Hughes, T.J.R., Lipton, S., Scott, M.A., Sederberg, T.W.: Isogeometric analysis using T-splines. Comput. Methods Appl. Mech. Eng. **199**(5–8), 229–263 (2010)
2. Bendsøe, M.P., Kikuchi, N.: Generating optimal topologies in structural design using a homogenization method. Comput. Methods Appl. Mech. Eng. **71**(2), 197–224 (1988)
3. Bendsøe, M.P., Sigmund, O.: Material interpolation schemes in topology optimization. Arch. Appl. Mech. **69**(9–10), 635–654 (1999)
4. Bendsøe, M.P., Sigmund, O.: Topology Optimization: Theory, Method and Applications. Springer, Berlin (2003)
5. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Netw. **5**(2), 157–166 (1994)

6. Bourdin, B., Chambolle, A.: Design-dependent loads in topology optimization. ESAIM Control Optim. Calc. Var. **9**(9), 19–48 (2003)
7. Cottrell, J.A., Hughes, T.J.R., Bazilevs, Y.: Isogeometric Analysis: Toward Integration of CAD and FEA. Wiley (2009)
8. Deng, H., To, A.C.: Topology optimization based on deep representation learning (DRL) for compliance and stress-constrained design. Comput. Mech. **66**, 449–469 (2020)
9. Dunning, P.D., Kim, H.A.: A new hole insertion method for level set based structural topology optimization. Int. J. Numer. Methods Eng. **93**(1), 118–134 (2013)
10. Fu, Y.F., Rolfe, B., Chiu, L.N.S., Wang, Y., Ghabraie, K.: Smooth topological design of 3D continuum structures using elemental volume fractions. Comput. Struct. **231**, 106213 (2020)
11. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. The MIT Press (2015)
12. Guo, X., Zhang, W., Zhong, W.: Doing topology optimization explicitly and geometrically—a new moving morphable components based framework. J. Appl. Mech. **81**(8), 081009 (2014)
13. Hassani, B., Khanzadi, M., Tavakkoli, S.M.: An isogeometrical approach to structural topology optimization by optimality criteria. Struct. Multidiscip. Optim. **45**(2), 223–233 (2012)
14. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Netw. **2**(5), 359–366 (1989)
15. Hou, W., Gai, Y., Zhu, X., Wang, X., Zhao, C., Xu, L., Jiang, K., Hu, P.: Explicit isogeometric topology optimization using moving morphable components. Comput. Methods Appl. Mech. Eng. **326**, 694–712 (2017)
16. Huang, G., Liu, Z., Maaten, L.V.D., Weinberger, K.Q.: Densely connected convolutional networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 1, pp. 2261–2269 (2017)
17. Hughes, T.J.R., Cottrell, J.A., Bazilevs, Y.: Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. Comput. Methods Appl. Mech. Eng. **194**(39–41), 4135–4195 (2005)
18. Kang, P., Youn, S.K.: Isogeometric topology optimization of shell structures using trimmed NURBS surfaces. Finite Elem. Anal. Des. **120**, 18–40 (2016)
19. Krizhevsky, A., Sutskever, I., Hinton, G.: ImageNet classification with deep convolutional neural networks. Commun. ACM **60**(6), 84–90 (2017)
20. Lecun, Y., Bottou, L.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
21. Li, X., Wei, X., Zhang, Y.: Hybrid non-uniform recursive subdivision with improved convergence rates. Comput. Methods Appl. Mech. Eng. **352**, 606–624 (2019)
22. Liu, K., Tovar, A.: An efficient 3D topology optimization code written in Matlab. Struct. Multidiscip. Optim. **50**(6), 1175–1196 (2014)
23. Mcculloch, W.S., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. Bull. Math. Biophys. **5**(4), 115–133 (1943)
24. Nguyen, T.H., Paulino, G.H., Song, J., Le, C.H.: A computational paradigm for multiresolution topology optimization (MTOP). Struct. Multidiscip. Optim. **41**(4), 525–539 (2010)
25. Nie, Z., Jiang, H., Kara, L.B.: Stress field prediction in cantilevered structures using convolutional neural networks. J. Comput. Inf. Sci. Eng. **20**(1), 011002 (2019)
26. Qian, X.: Topology optimization in B-spline space. Comput. Methods Appl. Mech. Eng. **265**, 15–35 (2013)
27. Rawat, S., Shen, M.H.H.: A novel topology design approach using an integrated Deep Learning network architecture. e-Print Archive. arXiv:1808.02334 (2018)
28. Rawat, S., Shen, M.H.H.: A novel topology optimization approach using conditional deep learning. e-Print Archive. arXiv:1901.04859 (2019)
29. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. Med. Image Comput. Comput.-Assist. Interv. **2015**, 234–241 (2015)
30. Saurabh, B., Harsh, G., Sanket, B., Sagar, P., Levent, K.: 3D topology optimization using convolutional neural networks. e-Print Archive. arXiv:1808.07440v1 (2018)
31. Seo, Y.D., Kim, H.J., Youn, S.K.: Isogeometric topology optimization using trimmed spline surfaces. Comput. Methods Appl. Mech. Eng. **199**(49–52), 3270–3296 (2010)
32. Sethian, J.A., Wiegmann, A.: Structural boundary design via level set and immersed interface methods. J. Comput. Phys. **163**(2), 489–528 (2000)

33. Sosnovik, I., Oseledets, I.: Neural networks for topology optimization. Russ. J. Numer. Anal. Math. Model. **34**(4), 215–223 (2019)
34. Wang, M., Qian, X.: Efficient filtering in topology optimization via B-Splines. J. Mech. Des. **137**(3), 031402 (2015)
35. Wang, M.Y., Wang, X., Guo, D.: A level set method for structural topology optimization. Comput. Methods Appl. Mech. Eng. **192**(1–2), 227–246 (2003)
36. Wang, Y., Benson, D.J.: Isogeometric analysis for parameterized LSM-based structural topology optimization. Comput. Mech. **57**(1), 19–35 (2016)
37. Xie, Y., Steven, G.P.: A simple evolutionary procedure for structural optimization. Comput. Struct. **49**(5), 885–896 (1993)
38. Xie, Y., Yang, X., Steven, G.P., Querin, O.: The theory and application of evolutionary structural optimization method. Eng. Mech. **16**(6), 70–81 (1999)
39. Yu, Y., Hur, T., Jung, J.: Deep learning for determining a near-optimal topological design without any iteration. Struct. Multidicip. Optim. **59**, 787–799 (2019)
40. Zhang, W., Li, D., Yuan, J., Song, J., Guo, X.: A new three-dimensional topology optimization method based on moving morphable components (MMCs). Comput. Mech. **59**(4), 1–19 (2016)
41. Zheng, R., Kim, C.: An enhanced topology optimization approach based on the combined MMC and NURBS-curve boundaries. Int. J. Precis. Eng. Manuf. **21**(2), 1529–1538 (2020)