Front. Mech. Eng. 2021, 16(1): 80–96
https://doi.org/10.1007/s11465-020-0614-2

RESEARCH ARTICLE

Liang XUE, Jie LIU, Guilin WEN, Hongxin WANG

Efficient, high-resolution topology optimization method based on convolutional neural networks

Abstract Topology optimization is a pioneer design method that can provide various candidates with high mechanical properties. However, high resolution is desired for optimum structures, but it normally leads to a computationally intractable puzzle, especially for the solid isotropic material with penalization (SIMP) method. In this study, an efficient, high-resolution topology optimization method is developed based on the super-resolution convolutional neural network (SRCNN) technique in the framework of SIMP. SRCNN involves four processes, namely, refinement, path extraction and representation, nonlinear mapping, and image reconstruction. High computational efficiency is achieved with a pooling strategy that can balance the number of finite element analyses and the output mesh in the optimization process. A combined treatment method that uses 2D SRCNN is built as another speed-up strategy to reduce the high computational cost and memory requirements for 3D topology optimization problems. Typical examples show that the high-resolution topology optimization method using SRCNN demonstrates excellent applicability and high efficiency when used for 2D and 3D problems with arbitrary boundary conditions, any design domain shape, and varied load.

Keywords topology optimization, convolutional neural network, high resolution, density-based

Received June 29, 2020; accepted October 8, 2020

Liang XUE, Guilin WEN (✉), Hongxin WANG
State Key Laboratory of Advanced Design and Manufacturing for Vehicle Body, Hunan University, Changsha 410082, China
E-mail: glwen@gzhu.edu.cn

Liang XUE, Jie LIU, Guilin WEN
Center for Research on Leading Technology of Special Equipment, School of Mechanical and Electric Engineering, Guangzhou University, Guangzhou 510006, China

1 Introduction

The work of Bendsøe and Kikuchi [1] is the basis of topology optimization, and it has been followed by several excellent topological optimization methods to solve material distribution problems under given goals and constraints; examples include solid isotropic material with penalization (SIMP) [2–4], evolutionary structural optimization/bi-directional evolutionary structural optimization (BESO) [5–9], level-set-based topology optimization [10–14], moving morphable components/void [15–17], and bubble method [18]. As topology optimization approaches mature and gradually shift their application from mathematical theory to practical engineering [19–21], the desire for developing existing topological optimization methods to achieve the capacity for dealing with large-scale or high-precision structures increases. Typical large-scale objects include those in architecture and aerospace [19,22,23]. Bionic bones [24], convection radiators [25], and mechanical metamaterials [26] are examples of objects that require high-precision design. Topology optimization is an excellent design tool in engineering, but it is only suitable for the design of normal-sized structures. Resolution is the main obstacle in dealing with large-scale or high-precision structural design because improving the design resolution inevitably increases the computational cost. At present, with an acceptable computing cost, people can only deal with structures with millions of resolutions. Although supercomputers have powerful computing power, they are not available to everyone. Therefore, the application of the density-based topology optimization method is limited by resolution.

Several studies have attempted to improve the resolution of topology optimization designs at an acceptable computational cost. Several mainstream density-based topology optimization methods that were developed to address the aforementioned issue are presented below.

1) *Designing the microstructure of elements*. Groen and Sigmund [27] and Wu et al. [28] used the homogenization method to design the microstructure of elements, and Zhu

et al. [29] and Wang et al. [30] utilized a pre-designed microstructure for topology optimization. This method improves the resolution of the results, but it fails to change the jagged boundaries caused by coarse elements. Li et al. [31,32] advocated the use of density-based and level-set-based methods to design macrostructures and microstructures, respectively, to ensure the continuity of boundaries.

2) *Adaptive adjustment of a discretized mesh*. For tetrahedral meshes, Christiansen et al. [33] used adaptive theory to move the cell nodes and make the structure pleasant in appearance. Wang et al. [34] extended the method to the case with hexahedral meshes. This low-cost, high-efficiency post-processing method is easy to implement in common CAD/CAE software. However, it is highly dependent on the mesh. In addition, the quality of the final configuration relies on the selected node adjustment criteria.

3) *Multi-resolution topology optimization*. This idea was proposed by Nguyen et al. [35], who separated a design variable mesh from a displacement mesh and refined it. Subsequent studies on this subject can be divided into two main groups depending on the research direction. (i) Nguyen-Xuan [36] used a hierarchical data structure called a polytree to refine boundary elements selectively; the structure improves the quality of the boundary. Leader et al. [37] and Chin et al. [38] interpolated the node design variables under a coarse mesh to obtain a refined displacement mesh that can solve multi-material and frequency response optimization problems in large-scale designs. In these methods, the resolution of the finite element mesh (displacement mesh) is not lower than that of the design variable mesh, which means that finite element analysis (FEA) devotes a large amount of time to optimization. (ii) Meanwhile, other studies opted to increase the resolution of the design variable mesh and maintain a coarse finite element mesh. In the work of Nguyen [35], high-order finite elements were used to improve algorithm accuracy and efficiency [39]. An adaptive refinement/coarsening criterion was introduced to increase the speed of the multi-resolution topology optimization method [40]. Using the multi-grid method in the approximate analysis of large problems is an effective solution [41]. In addition, because the non-uniform rational basis spline has high continuity, several studies used the isogeometric analysis method to map the design variables and the finite element mesh accurately [42,43] or improve algorithm efficiency [44]. Recently, Wang et al. [45] proposed a novel multi-resolution topology optimization method. In this method, extended finite element method is introduced into the BESO method to establish a multi-resolution framework that can capture non-smooth displacement fields across material interfaces.

4) *Adaptive mesh refinement*. Kim and Yoon [46] used separable wavelet transform to improve mesh resolution continuously during optimization, which can reduce the initial calculation amount. Stainko [47] proposed a solution method that involves adaptive multilevel techniques; the method focuses only on elements near the boundary. Liao et al. [48] designed a partial update rule that concentrates all computational power on changeable elements. These studies improved the computational efficiency of large-scale optimization problems to a certain extent. The addition of adaptive criteria also improves mesh independence.

The aforementioned studies have provided valuable algorithmic-level contributions to the goal of maximizing design resolution at an acceptable cost. Owing to the rapid development of computer science, several studies have used advanced algorithms or hardware (including GPU and supercomputers) to achieve the required high-resolution design [49–52]. Algorithm development and hardware update are complementary, not contradictory. An excellent algorithm and efficient hardware can produce desirable results.

Notably, computational methods in the form of machine learning perform well in the development of algorithms and hardware. Machine learning can be used to extract implicit rules from a large amount of historical data, and its excellent generality comes from training objectives that contain implicit laws about unknown data. Machine learning has a wide range of current applications, which include data mining, computer vision, and medical diagnosis. Deep learning, which belongs to machine learning, has become popular in recent years due to its high efficiency, plasticity, and universality.

Several scholars have attempted to use data-driven concepts to complete shape [53,54] and topology optimization design. Sosnovik and Oseledets [55] inputted the configuration during optimization and the corresponding gradient information into the neural network to obtain the final optimized configuration directly. This method can effectively improve optimization speed. Banga et al. [56] used a 3D encoder–decoder convolutional neural network, which is more efficient than the usual convolutional neural network, to perform 3D topology optimization. The number of design variables in this study was $12\times24\times12$, which is small for 3D samples. This number of design variables indicates the huge sample demand and calculation cost of 3D convolutional neural networks. Zhang et al. [57] directly obtained the final configuration by inputting the displacement and stress information of the initial design into the neural network at a definitive mesh resolution. This method has good versatility at a given mesh resolution but requires retraining the neural network for other structures with a different resolution. Li et al. [58] implemented topology optimization without iteration by using a generative adversarial network (GAN) and used another GAN as a post-processing method to improve the resolution of the final configuration.

However, the traditional multi-resolution topology optimization method has two shortcomings, namely, problems in arbitrary mapping and computational

efficiency. Mapping must be performed between the FEA mesh and the design variable mesh in the multi-resolution topology optimization method. Given that mapping from low resolution to high resolution differs for distinct structural features, artificially designing mappings for different features is difficult. Deep learning can intelligently extract structural features. In the subsequent process of extracting structural features, different structural features possess different mappings. Most neural networks have a fixed amount of input layer data, which directly affects the versatility of topology optimization methods.

This study selects the super-resolution convolutional neural network (SRCNN) framework [59] with good applicability to enhance the resolution of topology optimization design. Two strategies, namely, a pooling strategy for mesh balance and a combined treatment method using 2D SRCNN, are developed to ease the computational burden in the high-resolution topology optimization (HRTO) method. The proposed method demonstrates high versatility for 2D and 3D problems with any design domain shape, arbitrary boundary conditions, and any load case.

The remainder of this paper is organized as follows. Section 2 describes the density-based topology optimization method. Section 3 introduces SRCNNs. Section 4 presents the implementation of the proposed methodology, and Section 5 shows some numerical examples. The conclusions are provided in Section 6.

2 Density-based topology optimization methods

This work is performed under the framework of density-based topology optimization methods. Under the guidance of the classic SIMP method [2,3,60], a mathematical description of the topology optimization problem and the entire topology optimization process are presented in this section.

Density-based topology optimization is a method for solving the 0/1 value of the relative density ρ of each element in a given design domain Ω that has been discretized into finite elements. The aim is to find the minimum compliance structure under the target volume constraints. The mathematical description of the problem is as follows:

$$\begin{aligned} \min_{x} \quad & c(x) = \boldsymbol{U}^{\mathrm{T}}\boldsymbol{K}\boldsymbol{U}, \\ \text{s.t.} \quad & V(x) \leqslant V^{*}, \\ & \boldsymbol{K}\boldsymbol{U} = \boldsymbol{F}, \\ & 0 < x_{\min} \leqslant x < 1, \end{aligned} \tag{1}$$

where $\boldsymbol{U}$ and $\boldsymbol{F}$ are the global displacement and force vectors, respectively, $\boldsymbol{K}$ is the global stiffness matrix, x is the design variable, $x_{\min}$ is the minimum relative density, and $V(x)$ and V^{*} are the material and target volumes, respectively. In this study, the following penalty interpolation method of SIMP [3] is used to combine the element stiffness matrix into a global stiffness matrix:

$$E_i = x_i^p E_0, \tag{2}$$

where E_i and E_0 are the Young's modulus of element i and the basic material, respectively. The penalization exponent is usually set to $p = 3$ to push the median value of the design variable close to the 0–1 solution.

$$\boldsymbol{K} = \sum_{i=1}^{N} \boldsymbol{k}_i(E_i), \tag{3}$$

where N represents the total number of elements in the design domain and $\boldsymbol{k}_i$ refers to the stiffness matrix of ith element. Sensitivity can be obtained by deriving the objective function for the design variables of each element. To suppress numerical instabilities, we use the filter scheme proposed by Sigmund [3]. Then, the optimality criteria (OC) method is utilized to solve this topological optimization problem. The following text shows the basic concepts of SRCNN and how SRCNN is implemented in the proposed topology optimization framework.

3 Super-resolution convolutional neural network

In the SRCNN framework in the present work, four classical topological optimization models are selected as training samples, and each model contains 20 different cases. In the training process, several samples are selected from the training sample set at each iteration step to ensure the rationality of the convolutional neural network. Sections 3.1 and 3.2 focus on the network architecture and training process, respectively. The implementation of SRCNN combined with topology optimization is illustrated in the following section.

3.1 Network architecture

A complete process of increasing the resolution requires four steps, namely, refinement, path extraction and representation, nonlinear mapping, and image reconstruction. Refinement is a pre-processing process that uses quadratic interpolation to upscale the original low-resolution image. The processed image reaches the targeted pixel value, but its quality is still not good enough; hence, it is still regarded as a low-resolution image and marked as L. This low-resolution image participates in SRCNN as an input sample. Such a refinement step can improve the versatility of the algorithm. Next, a series of convolutional neural network operations are implemented. Path extraction and representation is a process of extracting features from low-resolution images by using multiple convolution

kernels. Nonlinear mapping combines these features and maps them to the next maps. Then, the image reconstruction process reconstructs the mapped features into a high-resolution image, which is denoted by H.

Figure 1 shows the four steps of SRCNN from left to right. The upper left corner of Fig. 1 shows the size of three convolution kernels (W_1, W_2, and W_3) and their corresponding operation results. The changes in the data structure are shown below each image. Specifically, $nelx \times nely$ is the design variable resolution of the topology optimization model; USF is the upscaling factor; f_1, f_2, and f_3 are the size of convolution kernels; and n_1 and n_2 are the depth of the neural network. From the connection in Fig. 1, we can understand the transmission path of data in the neural network. The structural components of each step are specified separately in the subsequent subsections. Refinement, as a pre-processing method, is not the focus of neural networks. We select bicubic interpolation as the solution of the refinement process because the bicubic interpolation method is convenient to use, and its computational efficiency is high. This operation is no longer described separately.

3.1.1 Patch extraction and representation

The main function of this part is to extract features from low-resolution image L. For easy understanding, each convolution kernel can be imagined as a filter. The features extracted by each convolution kernel constitute the low-resolution feature map $F_1(L)$ of SRCNN. This operation can be expressed as follows:

$$F_1(L) = \max(0,\ \boldsymbol{W}_1 * L + \boldsymbol{B}_1), \tag{4}$$

where $\boldsymbol{W}_1$ and $\boldsymbol{B}_1$ are convolution kernels and biases, respectively. $\boldsymbol{W}_1$ contains n_1 convolution kernels of size $f_1 \times f_1$. We assume that the size of L is $X \times Y$, and the size of $F_1(L)$ is $X \times Y \times n_1$. Each convolution kernel is independently convolved with L. $\boldsymbol{B}_1$ is a vector of length n_1, and it corresponds to the convolution kernels of $\boldsymbol{W}_1$ in order. Notably, the operator "$*$" represents the "same" type of convolution operation, which needs to expand $(f_1 - 1)/2$ circles (f_1 generally takes an odd number) around the periphery of L. This type of convolution ensures that the dimensions of maps will not decrease in the next process. Unless otherwise specified in the subsequent operations, the operator "$*$" defaults to the "same" type of convolution operation. Then, the rectified linear unit (ReLU, $\max(0, x)$) is used as an activation function to map the calculation results to the low-resolution feature maps.

3.1.2 Nonlinear mapping

The second convolution layer maps the low-resolution feature maps $F_1(L)$ to the high-resolution feature maps $F_2(L)$. In accordance with the work of Dong et al. [59], we select $\boldsymbol{W}_2$ with a size of 5×5. The operational formula for this layer is expressed as follows:

$$F_2(L) = \max\left(0,\ \boldsymbol{W}_2 * F_1(L) + \boldsymbol{B}_2\right), \tag{5}$$

where $\boldsymbol{W}_2$ contains n_2 convolution kernels of sizes $f_2 \times f_2 \times n_1$ and $\boldsymbol{B}_2$ is a bias vector with a length of n_2. The size of $F_1(L)$ is $X \times Y \times n_1$, and the size of $F_2(L)$ is $X \times Y \times n_2$. The nonlinearity of this convolutional layer is strong and results in a significantly increased training time for neural networks.

3.1.3 Image reconstruction

To obtain a high-resolution design from the high-resolution feature map $F_2(L)$, we need a convolutional layer to reconstruct these features. The formula of the last layer is

$$H = \min\left(1,\ \max\left(0,\ \boldsymbol{W}_3 * F_2(L) + \boldsymbol{B}_3\right)\right), \tag{6}$$

where $\boldsymbol{W}_3$ is a convolution kernel of size $f_3 \times f_3 \times n_2$ and $\boldsymbol{B}_3$ is a bias value of size 1×1. Modifications are made to the ReLU activation function, as shown in Eq. (7), to match the results to the topology-optimized design variable range. The reconstructed map value range is limited to between 0 and 1.

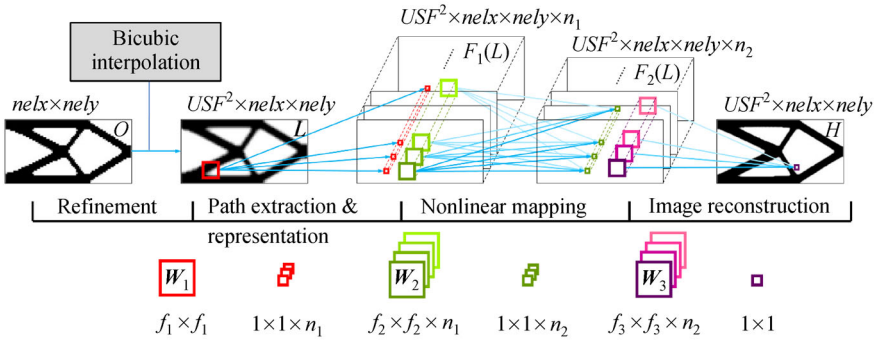

Fig. 1 Relative position and connection state of SRCNN operations. SRCNN: Super-resolution convolutional neural network.

$$\mathrm{ReLU}(x) = \min\Big(1,\ \max(0,\ x)\Big). \tag{7}$$

3.1.4 Network calculation steps

In the SRCNN framework, the "refinement" part uses bicubic interpolation to increase the pixels of the input image in accordance with the upscaling factor. The "path extraction and representation" part uses n_1 convolution cores to extract data from the image. Each convolution kernel is similar to the filter used in topology optimization, and the calculation process does not change the image size. The $F_1(L)$ obtained in this step contains n_1 feature maps with the same size as the input image. In the "nonlinear mapping" part, n_2 convolution kernels are utilized to convolve n_1 feature maps in $F_1(L)$. Then, n_2 feature maps with a constant size are obtained to form $F_2(L)$. In the final "image reconstruction" part, $F_2(L)$ is scanned with a convolution kernel. The obtained H contains only one image with the same size as the input image. Regardless of the changes in image size, the convolution kernel scans each pixel and its neighborhood in order. Images of different sizes generate feature images of different sizes. The output image is of the same size as the input image.

The following shows the step-by-step calculation process of SRCNN:

1) Using bicubic interpolation, the number of pixels of the input low-resolution image with a size of $nelx \times nely$ is increased to $USF^2 \times nelx \times nely$, which is L, according to the upscaling factor.

2) n_1 convolution kernels of size $f_1 \times f_1$ are used to perform convolution calculations on low-resolution images L with a size of $USF^2 \times nelx \times nely$.

3) The ReLU activation function is utilized to process the result of the previous step. We obtain n_1 feature maps of size $USF^2 \times nelx \times nely$, namely, $F_1(L)$.

4) A convolution kernel with a size of $f_2 \times f_2 \times n_1$ is adopted to perform convolution calculation on the feature map $F_1(L)$ of the n_1 layer with a size of $USF^2 \times nelx \times nely$. n_1 layers exist in total, and the convolution calculation is performed layer by layer. The results are added together to form a feature map with a size of $USF^2 \times nelx \times nely$.

5) n_2 convolution kernels of size $f_2 \times f_2 \times n_1$ are available, so Step 3 needs to be repeated n_2 times. Each repetition produces a new feature map. After each feature map is processed by the ReLU activation function, and n_2 feature maps of size $USF^2 \times nelx \times nely$, namely $F_2(L)$, are derived.

6) A convolution kernel of size $f_3 \times f_3 \times n_2$ is used to perform convolution calculation on the feature map $F_2(L)$ of the n_2 layer of size $USF^2 \times nelx \times nely$. n_2 layers exist in total, and the convolution calculation is performed layer by layer.

7) The modified activation function (Eq. (7)) is used to process the results of Step 6. The results are added to obtain a high-resolution image of size $USF^2 \times nelx \times nely$.

3.2 Training

The neural network needs to be trained with the network architecture to learn the entire process from low resolution to high resolution. Training is the process of estimating and adjusting network parameters $\{W_1,\ W_2,\ W_3,\ B_1,\ B_2,\ B_3\}$. To distinguish between reconstructed and real high-resolution configurations, we adopt 4 classical topological optimization models, namely, cantilever beam (Fig. 2(a)), L-bracket (Fig. 2(b)), T-bracket (Fig. 2(c)), and MBB beam (Fig. 2(d)). We also use different design domain sizes and

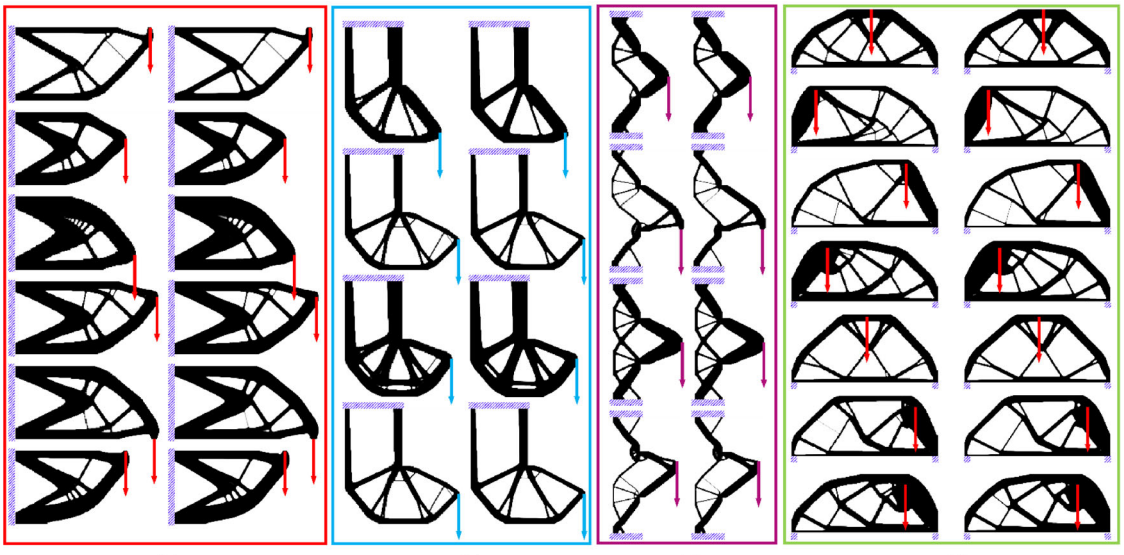

Fig. 2 Presentation of several training samples. This illustration shows 4 types of classical topological optimization models: (a) Cantilever beam, (b) L-bracket, (c) T-bracket, and (d) MBB beam.

random load positions. The traditional topology optimization method is utilized to calculate training samples of high and low resolution in 20 different cases for each of the 4 classical models. Only a part of the sample is shown in Fig. 2. The area filled with slashes in Fig. 2 indicates the boundary constraints, and the load is replaced by arrows. Notably, although only these 4 models are used as the base samples, the versatility of the SRCNN architecture is not affected. It still works for any size and dimension model, including non-convex shapes.

With the training sample, the mean square error is selected as the loss function (*Loss*) to characterize the difference between the reconstructed and real configurations.

$$Loss = \frac{1}{n}\sum_{N=1}^{n} \|H_N(L,\ W_l,\ B_l) - H_N^{\text{real}}\|^2, \tag{8}$$

where n is the number of training samples selected for each iteration. The value of l is 1, 2, and 3 corresponding to three convolution operations. H is the high-resolution configuration of L reconstructed by SRCNN, and H^{real} is the high-resolution training sample matched with L. The random gradient descent method is used to minimize the loss value during standard backpropagation. The formula for updating each convolution kernel weight and bias is as follows:

$$\begin{cases} \Delta W_{l(i+1)} = \gamma \times \Delta W_{li} - \eta \times \dfrac{\partial Loss}{\partial W_{li}}, \\ W_{l(i+1)} = W_{li} + \Delta W_{l(i+1)}, \\ \Delta B_{l(i+1)} = \gamma \times \Delta B_{li} - \eta \times \dfrac{\partial Loss}{\partial B_{li}}, \\ B_{l(i+1)} = B_{li} + \Delta B_{l(i+1)}, \end{cases} \tag{9}$$

where γ is the inheritance coefficient with a value from 0 to 1. η is the learning rate. In this study, γ and η take 0.9 and 10^{-4}, respectively, to ensure network convergence. $\dfrac{\partial Loss}{\partial W_{li}}$ and $\dfrac{\partial Loss}{\partial B_{li}}$ are the derivatives of the loss for each member of the convolution kernel W and the biases B, respectively, and l and i are the number of layers and the iteration step, respectively. The convolution kernel W and biases B of each layer take a random number between –1 and 1 as an initialization value.

The network architecture and training process of SRCNN are discussed above. Although SRCNN is a short neural network, using it is enough to establish a link between high and low resolutions after training. Therefore, SRCNN has high efficiency and a good mapping effect. After deriving the SRCNN for topology optimization, we focus on combining the convolutional neural network with the topology optimization process in the next section. The 2D SRCNN is then extended to solve 3D HRTO.

4 Details of the optimization

The two preceding sections introduced the classical topology optimization method and the network architecture and training method of SRCNN. This section shows the incorporation of SRCNN into the topology optimization process, but several details need to be discussed in advance. First, the difference between large scale and high precision needs to be clarified. Second, the conversion between small-sized configuration elements and large-sized FEA elements should be solved. Lastly, a 2D to 3D transformation method for SRCNN is proposed to help extend the 2D HRTO method to 3D cases.

4.1 High-resolution filter

As mentioned in Section 2, topology optimization involves a sensitivity filter defined by filter radius $r_{\min}$. For ease of explanation, this study uses the number of elements as the unit of the filter radius and defines the actual radius length as the true radius. The filter radius links mesh size to model size. Both large scale and high precision are closely related to the improvement of resolution (i.e., high resolution), but model size, element size, and filter radius vary. For example, the output resolution of a case with a size of $X \times Y$ is increased to $USF \times X \times USF \times Y$. As described in Section 3.1, *USF* is the upscaling factor. Compared with the low-resolution model, the large-scale model increases only the model size and retains the element size and filter radius. The high-precision model's size does not change, but as the element size decreases, the filter radius increases. For a specific case, Table 1 shows a base model with a size of 200×100 and a filter radius of 3 (the number of elements is used as the unit of the filter radius). The changes in large scale and high precision when *USF* has a value of 4 are displayed in Table 1. The element edge length is used as the filter radius. The value of the filter radius needs to be changed for different meshes and enhancement modes to keep the absolute radius constant. Table 1 indicates that the high-precision filter radius selected intuitively is 12, but the filter radius used for the high-precision images in the training set is 15. We use Fig. 3 to explain this difference.

Figure 3 shows how the filter radius changes with increasing resolution. In Fig. 3, the blue circular area and the large blue element represent the filter region and the filtered element of the low-resolution base model with a filter radius of 3, respectively. Table 1 indicates that the model size and finite element mesh are increased to 800×400 when the intuitive choice deals with large-scale models. The filter radius is maintained at 3, which corresponds to the yellow area in Fig. 3. For high-precision models, the model size remains to be 200×100, but the finite element mesh is increased to 800×400 and the filter radius is increased to 12 (red area in Fig. 3).

We found that the high-precision filter region of the

Table 1 Model data of different high-resolution transformations

Method	Enhancement mode	Model size	FEA mesh	Filter radius	Output resolution
Low-resolution	Basic model	200×100	200×100	3	200×100
Intuitive choice	Large-scale	800×400	800×400	3	800×400
	High-precision	200×100	800×400	12	800×400
Training set	Large-scale	800×400	800×400	3	800×400
	High-precision	200×100	800×400	15	800×400

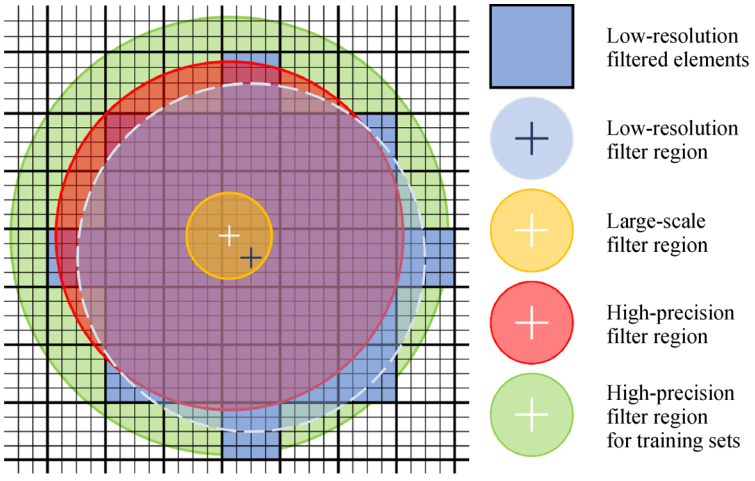

Fig. 3 Filter region and filtered elements of different high-resolution transformations.

intuitive choice in Fig. 3 does not include all the filtered elements with low resolution. SRCNN relies on data. Therefore, to ensure that the training set of the neural network contains all the necessary data, we used the following filter radius conversion formula for the SRCNN training set:

$$\begin{cases} r_{\min}^{H} = (r_{\min}^{L} + 1) \times USF - 1, \\ r_{\min}^{L} = (r_{\min}^{H} + 1)/USF - 1, \end{cases} \tag{10}$$

where $r_{\min}^{H}$ and $r_{\min}^{L}$ are the filter radius of the high- and low-resolution meshes, respectively, and USF is the upscaling factor. As shown in Table 1, the training set using the filter radius conversion formula has a high-precision filter radius of 15, which corresponds to the green area in Fig. 3. In this way, the filter area contains all the necessary data.

4.2 Pooling strategy

Pooling is a concept in convolutional neural networks that aims to reduce network dimension. Inspired by this concept, we attempted to balance the number of FEAs and output meshes in the optimization process. The pooling of convolutional neural networks considers reverse operations and generally uses average pooling or maximum pooling. Topology optimization does not require reverse pooling operations, so the options for pooling are numerous. In this study, we used the mean of the fine elements corresponding to the corners of the coarse cells as the pooling value, as shown in Fig. 4. The computational cost can be controlled well through this pooling method.

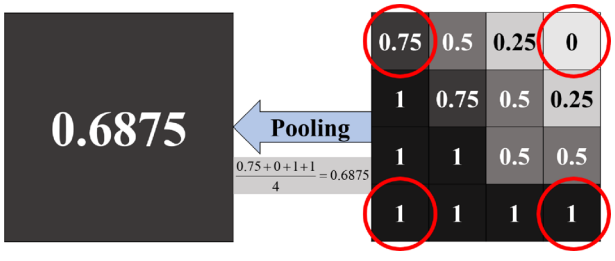

Fig. 4 Pooling strategy with corner mean sampling.

4.3 Numerical implementation

A HRTO method can be established by implementing SRCNN and pooling strategies, as shown in Fig. 5. HRTO separates the output configuration mesh from the finite element mesh through SRCNN, and pooling connects them. Therefore, the high-resolution information obtained

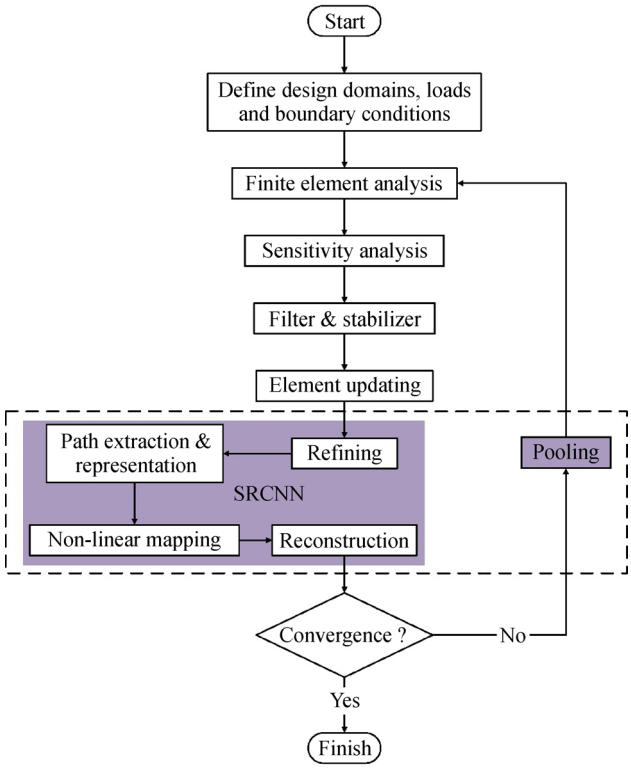

Fig. 5 High-resolution topology optimization method.

by SRCNN can still affect the FEA while maintaining an efficient computation. In the HRTO process, the stabilizer and convergence criteria used in the BESO method [7] are introduced to improve convergence. If the filter scheme [3] can smooth the sensitivity in space, then the stabilizer will smooth the sensitivity in time.

The procedure of the presented HRTO method is given as follows:

1) The FEA mesh of the design domain and its load and boundary conditions are defined. Then, an initial design variable value (0 or 1) is assigned to each element.

2) The equilibrium equation is calculated using FEA.

3) The sensitivity of individual elements is calculated using the adjoint method.

4) A filter [3] is used to smooth the sensitivity spatially and a stabilizer [7] to smooth the sensitivity temporally.

5) The design variables are updated with the OC method.

6) The resolution of the design variables is increased with SRCNN.

7) Whether the optimization design satisfies the convergence condition $error = \frac{\left|\sum_{i=1}^{N}(c_{k-i+1}-c_{k-N-i+1})\right|}{\sum_{i=1}^{N} c_{k-i+1}} \leqslant 0.001$ is determined. If it does not, then pooling will be executed to reduce the design resolution.

8) Steps 2–7 are repeated until the convergence condition is satisfied.

4.4 Combination treatment of 3D models

We attempted to extend the proposed strategy to deal with 3D cases. Mathematically, a 3D convolutional neural network is achievable. In reality, huge training set and high computational cost are the main obstacles to the strategy's application. Therefore, we used 2D SRCNN instead of 3D SRCNN operations to process 3D elements from three directions. Figure 6 shows the refinement process for an element in a 3D model with a *USF* of 4; the arrow represents the normal direction of the 2D SRCNN processing plane. After one processing has been performed in each of the three directions, an element of $1\times1\times1$ is gradually shifted to $16\times16\times16$ elements. When SRCNN is used to process 3D examples in three directions, the model can be regarded as the accumulation of multi-layer images. SRCNN processes each layer independently in three directions in sequence. For example, for an $l\times m\times n$ 3D model, the front to back direction can be regarded as the accumulation of n images of size $l\times m$. After each image is processed by SRCNN, it becomes a $4l\times4m$ image, and the total of these images is n. At this time, the size of this 3D model is $4l\times4m\times n$, which can be regarded as a stack of $4l$ images of size $4m\times n$ from top to bottom. After these images are processed by SRCNN, the size becomes $16m\times4n$. Currently, the size of this 3D model is $4l\times16m\times4n$. From right to left, it can be viewed as a superposition of $16m$ images with a size of $4l\times4n$. After each layer of images is processed by SRCNN, the final size of this model becomes $16l\times16m\times16n$. The role of SRCNN in topology optimization and the different processing methods of 2D and 3D models have been determined at this point. The next section shows how the method works by using several numerical examples.

5 Numerical examples

This section uses the HRTO method to solve the topology optimization problem. We adopted the efficient topology optimization code presented by Andreassen et al. [61] as the base program. We referred to this efficient topology optimization code based on the SIMP method as the conventional method. All examples were implemented on the same computer, and the hardware included an Inter(R) Xeon(R) CPU E5-2689 v4 @ 3.10 GHz with 256 G of RAM. None of them used GPU parallelism.

5.1 2D numerical examples

Figure 7 shows the two 2D examples we prepared. The load forms, boundary conditions, and design domains of the two examples differ from those in the training set. Figure 7(a) shows a barrier structure. An irregular, empty, non-design domain can be seen at the bottom left, and it makes the structure appear like a shape of the number 7. A

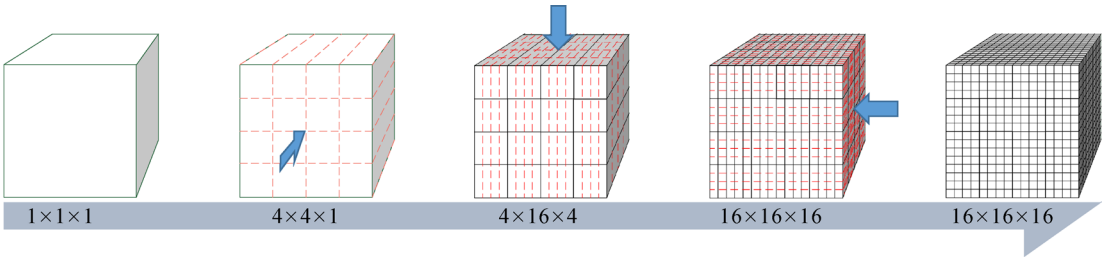

Fig. 6 Combination treatment of 3D models using 2D SRCNN.

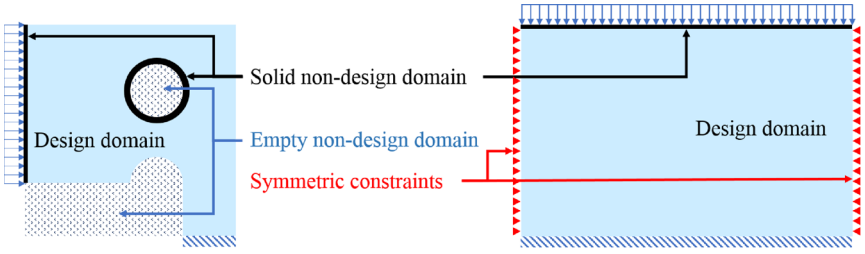

Fig. 7 Design domain and boundary conditions of two 2D examples: (a) A barrier structure with a hole and (b) a sandwich structure with symmetrical boundary conditions.

uniform load is applied to the left, and all the degrees of freedom at the bottom node are fixed. The middle of the structure has a ringed, non-design domain, which transforms the structure into a non-convex figure. Figure 7(b) presents a sandwich structure with all degrees of freedom on the lower surface fixed. Symmetry conditions are set on the left and right boundaries of the design domain. The upper surface has a layer of solid non-design domain simulation splint, and uniform load is applied on the splint.

The low-resolution image of the barrier structure from Fig. 7(a) shown in Fig. 8(a) has a 210×210 mesh and a filter radius of 4, and the objective function value of the low-resolution structure is 91.495. In Fig. 8, the filter radius and objective function are replaced by $r_{\min}$ and c_{obj}, respectively. Figures 8(b)–8(f) present high-resolution images under various strategies, and they have the same 840×840 high-resolution mesh. Figures 8(b) and 8(c) show high-precision and large-scale images calculated using the conventional method. For the conventional method, the filter radius of the high-precision image is 19 according to Eq. (10). This filter radius makes the high-precision configuration almost consistent with the low-resolution images. The high-precision objective function value calculated by the conventional method is 97.613. However, the large-scale graphic with a filter radius of 4 has clear boundaries and fine features, and the objective function value is only 84.179. As shown in Fig. 8(d), when only SRCNN is used as a post-processing solution to calculate high-resolution graphics, the resulting structure is destroyed. This post-processing solution is unavailable. Figures 8(e) and 8(f) present the high-precision and large-scale structures calculated by HRTO, respectively. The filter radius is 4 because their FEA uses a coarse mesh. HRTO high-precision graphics have many materials arranged near the ring because SRCNN concentrates the gray elements around the ring in the coarse mesh, and the features of other areas are reduced. By contrast, the HRTO large-scale configuration has many detailed features near the left-end face. Given that HRTO can effectively concentrate gray-scale elements, the two HRTO cases have clear boundaries, and their objective function values are smaller than the corresponding structures calculated by conventional methods. The objective function of the HRTO high-precision structure is 81.168, and the objective function of the HRTO large-scale structure is 83.348.

We also tested the sandwich structure in Fig. 7(b). In Fig. 9, $r_{\min}$ and c_{obj} are the filter radius and objective function, respectively. The low-resolution image shown in Fig. 9(a) has a 200×140 mesh, a filter radius of 4, and objective function value of 2.0214. Figures 9(b)–9(f) show high-resolution images under the various strategies, and they all have a high-resolution mesh of 800×560. Figures 9(b) and 9(c) present the high-precision and large-scale images calculated by a conventional method, respectively. When the conventional method is used, the high-precision image with a filtering radius of 19 becomes similar to the low-resolution image, and the objective function value of the high-precision structure is 2.0715. The large-scale graph with 4 as the filter radius has clear boundaries, fine features, and an objective function value of 1.9335. Figure 9(d) shows a high-resolution image calculated using SRCNN as a post-processing scheme. The image has

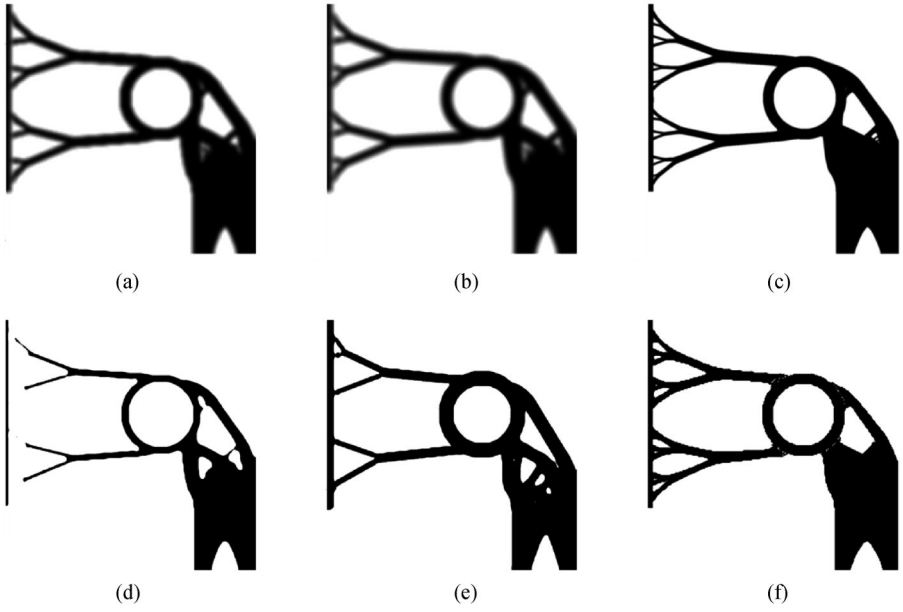

Fig. 8 High-resolution images of the barrier structure under various strategies. Results obtained by traditional methods: (a) Low-resolution, 210×210, r_{min}= 4, c_{obj} = 91.495; (b) high-precision, 840×840, r_{min} = 19, c_{obj} = 97.613; (c) large-scale, 840×840, r_{min} = 4, c_{obj} = 84.179. (d) Result post-processed by SRCNN, 840×840, r_{min} = 4, c_{obj} = 1.2×10^8. Results obtained by HRTO: (e) High-precision, 840×840, r_{min} = 4, c_{obj} = 81.168; (f) large-scale, 840×840, r_{min} = 0.25, c_{obj}= 83.348. r_{min} and c_{obj} represent the filter radius and the objective function, respectively.

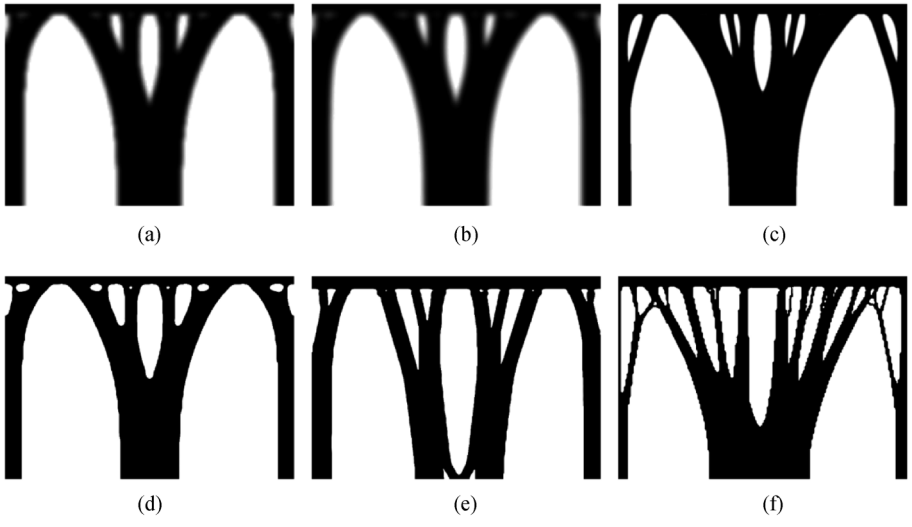

Fig. 9 High-resolution images of the sandwich structure under various strategies. Results obtained by traditional methods: (a) Low-resolution, 200×140, r_{min} = 4, c_{obj}= 2.0214; (b) high-precision, 800×560, r_{min} = 19, c_{obj} = 2.0715; (c) large-scale, 800×560, r_{min} = 4, c_{obj} = 1.9335. (d) Result post-processed by SRCNN, 800×560, r_{min} = 4, c_{obj}= 2.1632. Results obtained by HRTO: (e) High-precision, 800×560, r_{min} = 4, c_{obj} = 1.8352; (f) large-scale, 800×560, r_{min} = 0.25, c_{obj} = 1.9201. r_{min} and c_{obj} indicate the filtering radius and objective function, respectively.

numerous details, but the objective function is increased to 2.1632. Figures 9(e) and 9(f) show the high-precision and large-scale structures calculated by HRTO, respectively. Their filter radius is 4 because their FEA uses a coarse mesh. The high-precision structure of HRTO is different from the result of conventional methods. The HRTO high-precision structure has more branches and straighter struts. The large-scale configuration of HRTO has many detailed features. However, due to the instability of numerical calculations, the structure exhibits asymmetry. The

objective function values of the HRTO high-precision and large-scale structures are 1.8352 and 1.9201, respectively, which are lower than the values for the corresponding conventional schemes.

To investigate the influence of each optimization parameter on HRTO efficiency, we selected an MBB beam as the base model, half of which is used as an optimization model in Fig. 10. Its Young's modulus E equals to 1, the load F equals to 2, and the optimization parameters included a base resolution of 140×70, a target volume of 0.5, a filter radius of 3, and an upscaling factor of 4. Therefore, the output resolution would reach 560×280. Table 2 lists alternative optimization parameters for a parametric analysis.

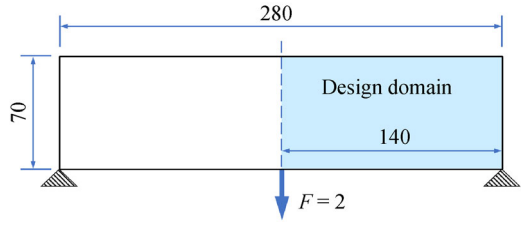

Fig. 10 MBB beam basic model.

Table 2 Alternative optimization parameters

Basic resolution	Target volume	Filter radius	Upscaling factor
100×50	0.3	1	2
120×60	0.4	2	3
140×70	0.5	3	4
160×80	0.6	4	–
180×90	0.7	5	–

In accordance with Table 2, we tested the influence of each alternative parameter on design compliance by using the control variable method. In Fig. 11, HRTO_FEA refers to the compliance of the coarse mesh in HRTO, and HRTO_output is the compliance of the high-resolution image output in each iteration. Figures 11(a)–11(d) present the effects of each optimization parameter on design compliance under the high-precision model. The effects of each optimization parameter on design compliance under the large-scale model are shown in Figs. 11(e)–11(h). The optimization parameters indicated in Fig. 11 are the number of elements, volume fraction, filter radius, and upscaling factor (from top to bottom). As can be seen in Figs. 11(a)–11(d), the FEA compliance of the HRTO high-precision model deviates from that of the conventional method, but this error can be corrected and optimized by SRCNN. The objective function of HRTO's design is approximately 0.5% better than that of the conventional method. Only one large deviation can be seen in Fig. 11(c) where the filter radius $r_{\min}$ equals to 1. The performance of HRTO is good under other high-precision conditions. In contrast, the HRTO method is not good enough in large-scale models. The error of the objective function decreases with increasing individual optimization parameters in Figs. 11(e), 11(f), and 11(h) (except for the filter plate radius $r_{\min}$ in Fig. 11(g)), but it is only reduced to around 5%. It can also be observed in Figs. 11(a)–11(h) that the high-resolution design sacrifices about 5% performance compared to the low-resolution design not processed by SRCNN.

An MBB beam whose design domain shape, boundary conditions, and load position are similar to those indicated in Fig. 10 was selected to test the efficiency of the HRTO method. Half of the base model size was 200×100, target volume V^* was 50%, filter radius $r_{\min}$ was 3, Young's modulus E was 1, external load F was 2, and SRCNN upscaling factor USF was 4. We calculated large-scale and high-precision models by using traditional and HRTO methods. Figure 12 shows the convergence history of the base model and the two types of high-resolution models calculated using the conventional method and the HRTO method. The black solid line denotes the basic model convergence history. The high-resolution and large-scale models of the conventional method are represented by a dashed line and a short-dashed line, respectively. The high-resolution and large-scale models of the HRTO method are indicated by a dashed–dotted line and a dashed double-dotted line, respectively. The curve in Fig. 12 represents the compliance calculated by FEA. According to the test results in Fig. 11, the output compliance of the HRTO method is about 5% higher than that of FEA. Figure 12 shows that the convergence of the HRTO method is much better than that of the conventional method. Table 3 presents some specific data on the convergence history of Fig. 12. Table 3 shows the efficiency of conventional methods and HRTO, and Table 4 calculates the reduction ratio of various data between conventional methods and HRTO. As can be seen from Table 4, HRTO is an efficient method. For the high-precision models, because of the low resolution of the FEA mesh and the small filter radius, HRTO only needs to add a small amount of memory for neural network operations, thereby reducing the initial time (I.T.) by 99.95% and the step time (S.T.) by 86.18%. In addition, SRCNN includes filter characteristics. Thus, the iteration number (It.) was also reduced (about 86.64%). The computational efficiency for the large-scale models was also improved. I.T., S.T., and It. decreased by 98.45%, 88.80%, and 63.44%, respectively. The combination of these improved values can reduce time and computing costs on the side of users. The conventional method requires a large amount of running memory in the high-precision case, whereas the HRTO method needs only a small amount. Comparison of the large-scale cases of the two methods revealed that the conventional method requires a large memory space to calculate the filter in

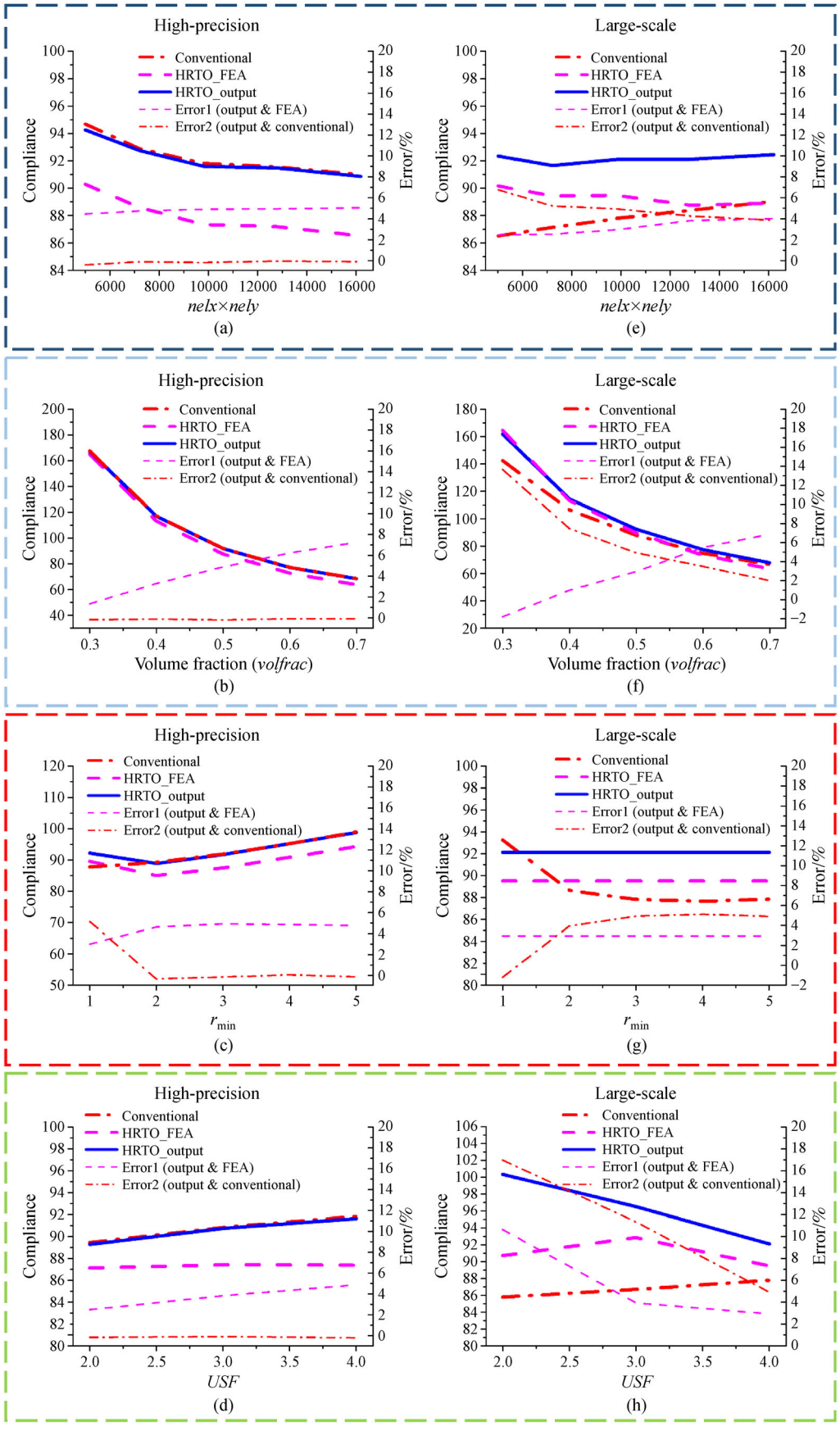

Fig. 11 The influence of each optimization parameter of 2D designs on the objective. The influence of (a) number of elements, (b) volume fraction, (c) filter radius, and (d) upscaling factor on the objective under the high-precision situation. The influence of (e) number of elements, (f) volume fraction, (g) filter radius, (h) upscaling factor on the objective under the large-scale situation.

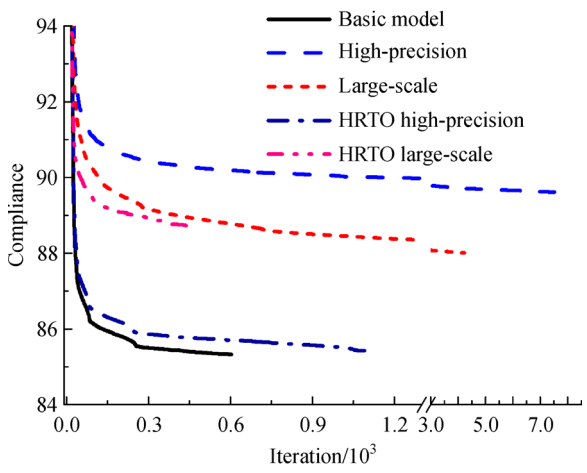

Fig. 12 MBB beam convergence history of the conventional method and HRTO method.

the high-precision case, and much of HRTO's memory is devoted to neural network computation.

Table 5 lists the efficiencies of the HRTO method at different resolutions. The design domain shape, boundary conditions, and load position of the basic model in Table 5 are similar to those of the MBB beam in Fig. 10. In addition to the change in resolution, the unlisted optimization parameters (target volume of 0.5, filter radius of 3, and upscaling factor of 4) were constants. The data in the table show that HRTO has a stable capability to accelerate high-precision models, and the acceleration capability of large-scale models increases with resolution.

5.2 3D numerical examples

Figure 13 shows the computing power of the HRTO method for the 3D models. The base resolution of the cantilever beam was 100×20×10, volume fraction *vol* was 0.35, filter radius $r_{\min}$ was 2, and upscaling factor *USF* was 4. With the help of the combination treatment of 3D models, the output resolution was increased to 16 times in all three dimensions (i.e., 1600×320×160), and the output designs included a total of 81920000 elements. This example does not require strong hardware support, and it can be implemented on a typical computer. This computer used for this case had an Intel(R) Core (TM) i5-7500 CPU @ 3.40 GHz and 4 GB of RAM. Only 0.57 s was required for initialization and 651.41 s for a single step. The conventional method is difficult to run in the same hardware environment. Therefore, the design of the conventional method is not shown here. Again, the high computational efficiency of the proposed method is highlighted.

To reveal the advantages of the HRTO method, we compared its efficiency with that of two similar algorithms. As shown in Table 6, the first compared method is a higher-order multi-resolution topology optimization approach using the voxel version of the finite cell method (FCM) [39]. The second method is an efficient structure topology optimization approach using the multiscale finite element method (MsFEM) [62]. According to the two above-mentioned previous studies, the acceleration rates of the FCM-based method for 2D and 3D models are 2.9 and 32, respectively. The MsFEM-based method increases the computational efficiency of the 2D model by 17 times and improves the calculation efficiency of the 3D model by 50 times. The proposed HRTO method can increase the computational efficiency of the 2D large-scale model by 24 times and can achieve an acceleration rate of 54 times for 2D high-precision design. In 3D model design, the acceleration rate of HRTO can even reach 79 times. In addition, Table 6 indicates that the acceleration effect of the three methods on the 3D model is greater than that on the 2D model. The reason is that the FEA of the 3D model is time consuming, and the FEA time is almost equivalent to the time of each iteration. The longer FEA is, the better the acceleration effects of the methods are. The HRTO

Table 3 MBB beam efficiency of the conventional method and HRTO method

Method	Enhancement mode	Output resolution	I.T./s	It.	S.T./s	Max. ram/GB
Low-resolution	Basic model	200×100	0.0994	606	0.3328	0.0100
Conventional	High-precision	800×400	209.4000	8174	20.0261	2.5303
	Large-scale	800×400	1.6530	4231	7.5468	0.1529
HRTO	High-precision	800×400	0.1138	1092	2.7686	0.1621
	Large-scale	800×400	0.0256	474	2.7592	0.1621

Table 4 Efficiency data reduction ratio of the conventional method and HRTO method

Enhancement mode	Reduction ratio/%			
	I.T.	It.	S.T.	Max. ram
High-precision	99.95	86.64	86.18	93.59
Large-scale	98.45	88.80	63.44	–6.08

Table 5 Efficiencies of the HRTO method at different resolutions

Basic resolution	Output resolution	Efficiency of conventional method		Efficiency of HRTO			
		I.T./s	S.T./s	I.T./s	Reduction ratio/%	S.T./s	Reduction ratio/%
High-precision							
100×50	400×200	15.19	4.591	0.023	99.85	0.660	85.63
120×60	480×240	24.89	7.166	0.039	99.84	0.905	87.37
140×70	560×280	37.55	7.068	0.054	99.86	1.254	82.25
160×80	640×320	58.13	9.790	0.073	99.88	1.719	82.44
180×90	720×360	121.0	14.14	0.088	99.93	2.115	85.04
200×100	800×400	209.4	20.03	0.114	99.95	2.769	86.18
Large-scale							
100×50	400×200	0.539	1.710	0.006	98.85	0.695	59.37
120×60	480×240	0.731	2.607	0.007	99.00	0.899	65.51
140×70	560×280	0.746	2.794	0.010	98.66	1.230	56.00
160×80	640×320	1.015	3.935	0.014	98.66	1.631	58.54
180×90	720×360	1.342	5.089	0.021	98.41	2.015	60.40
200×100	800×400	1.653	7.547	0.026	98.45	2.759	63.44

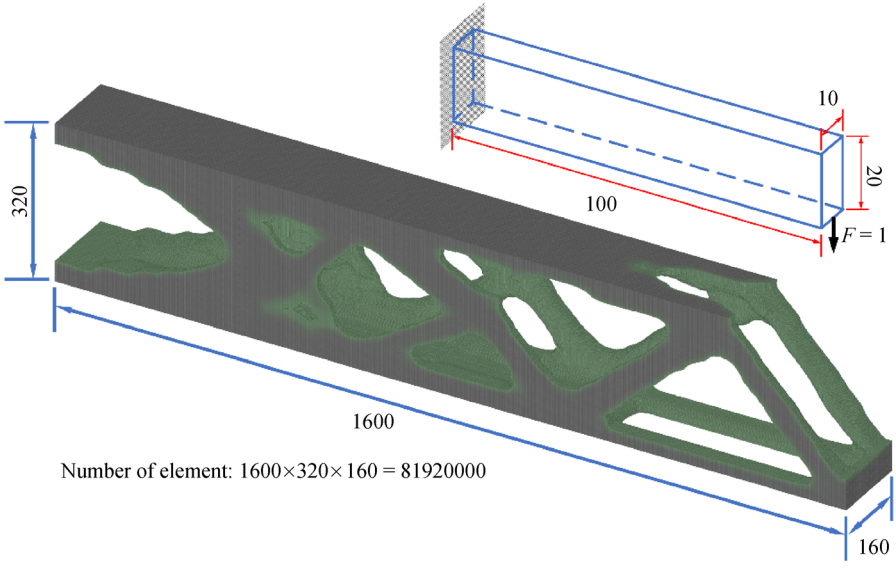

Fig. 13 Design domain and HRTO topology solution of a 3D cantilever beam.

Table 6 Comparison of acceleration ratios of three algorithms

Method	Acceleration rate	
	2D model	3D model
FCM-based	2.9	32
MsFEM-based	17	50
HRTO	24–54	79

upscaling factor for the 2D model is 4. However, for the 3D model, the upscaling factor is increased to 16 due to the algorithm. This condition makes HRTO advantageous for the calculation of 3D models.

6 Conclusions and remarks

This study established an efficient HRTO method using SRCNN. Two strategies, namely, a pooling strategy for mesh balance and a combined treatment method using 2D SRCNN, were developed in this framework. The method allows 3D HRTO to eliminate high computational costs. The following conclusion were obtained from a comprehensive comparison.

1) In terms of resolution, the data used in this study increased the resolutions of the 2D and 3D models from 200×100 to 800×400 and from 100×20×10 to

1600×320×160, respectively. The HRTO method could make the design achieve any resolution by flexibly combining SRCNN and pooling modules. For example, if we were to nest three SRCNNs and their corresponding pooling with the *USF* of 4 in a model with a resolution of 200×100, we would obtain a result with a resolution of $4^3\times200\times4^3\times100$. In the case without nesting, if we were to choose any three iterative steps without performing pooling, we would also obtain a result with a resolution of $4^3\times200\times4^3\times100$.

2) In terms of efficiency, HRTO exhibited higher efficiency than the traditional algorithms. In the high-precision design, the iteration number was reduced from 8174 to 1092, and the step time decreased from 20.026 to 2.7686 s. Further testing revealed that the acceleration effect became increasingly apparent as the number of meshes in the design domain increased.

3) In terms of versatility, HRTO benefits from the extensive application of SRCNN, and it is more versatile than other topology optimization methods using neural networks. HRTO can be used for any design domain, number of elements, arbitrary boundary conditions, and loads. Notably, the FEA mesh, rather than the output mesh, affects the mesh independence of HRTO.

Acknowledgements This work was supported by the National Natural Science Foundation of China (Grant Nos. 11672104 and 11902085), the Key Program of National Natural Science Foundation of China (Grant No. 11832009), and the Chair Professor of Lotus Scholars Program in Hunan Province, China (Grant No. XJT2015408).

References

1. Bendsøe M P, Kikuchi N. Generating optimal topologies in structural design using a homogenization method. Computer Methods in Applied Mechanics and Engineering, 1988, 71(2): 197–224
2. Bendsøe M P. Optimal shape design as a material distribution problem. Structural Optimization, 1989, 1(4): 193–202
3. Sigmund O A. 99 line topology optimization code written in Matlab. Structural and Multidisciplinary Optimization, 2001, 21(2): 120–127
4. Rozvany G I N, Zhou M, Birker T. Generalized shape optimization without homogenization. Structural Optimization, 1992, 4(3–4): 250–252
5. Xie Y M, Steven G P. A simple evolutionary procedure for structural optimization. Computers & Structures, 1993, 49(5): 885–896
6. Querin O M, Steven G P, Xie Y M. Evolutionary structural optimisation (ESO) using a bidirectional algorithm. Engineering Computations, 1998, 15(8): 1031–1048
7. Huang X, Xie Y M. Convergent and mesh-independent solutions for the bi-directional evolutionary structural optimization method. Finite Elements in Analysis and Design, 2007, 43(14): 1039–1049
8. Rozvany G I N. A critical review of established methods of structural topology optimization. Structural and Multidisciplinary Optimization, 2009, 37(3): 217–237
9. Xia L, Zhang L, Xia Q, et al. Stress-based topology optimization using bi-directional evolutionary structural optimization method. Computer Methods in Applied Mechanics and Engineering, 2018, 333: 356–370
10. Wang M Y, Wang X, Guo D. A level set method for structural topology optimization. Computer Methods in Applied Mechanics and Engineering, 2003, 192(1–2): 227–246
11. Wei P, Li Z, Li X, et al. An 88-line MATLAB code for the parameterized level set method based topology optimization using radial basis functions. Structural and Multidisciplinary Optimization, 2018, 58(2): 831–849
12. Xia Q, Shi T, Xia L. Stable hole nucleation in level set based topology optimization by using the material removal scheme of BESO. Computer Methods in Applied Mechanics and Engineering, 2019, 343: 438–452
13. Xia Q, Shi T. Generalized hole nucleation through BESO for the level set based topology optimization of multi-material structures. Computer Methods in Applied Mechanics and Engineering, 2019, 355: 216–233
14. Liu H, Zong H, Shi T, et al. M-VCUT level set method for optimizing cellular structures. Computer Methods in Applied Mechanics and Engineering, 2020, 367: 113154
15. Guo X, Zhang W, Zhong W. Doing topology optimization explicitly and geometrically—A new moving morphable components based framework. Journal of Applied Mechanics, 2014, 81(8): 081009
16. Zhang W, Chen J, Zhu X, et al. Explicit three dimensional topology optimization via moving morphable void (MMV) approach. Computer Methods in Applied Mechanics and Engineering, 2017, 322: 590–614
17. Lei X, Liu C, Du Z, et al. Machine learning-driven real-time topology optimization under moving morphable component-based framework. Journal of Applied Mechanics, 2019, 86(1): 011004
18. Cai S, Zhang W. An adaptive bubble method for structural shape and topology optimization. Computer Methods in Applied Mechanics and Engineering, 2020, 360: 112778
19. Zhu J H, Zhang W H, Xia L. Topology optimization in aircraft and aerospace structures design. Archives of Computational Methods in Engineering, 2016, 23(4): 595–622
20. Fu Y F, Rolfe B, Chiu L N S, et al. Design and experimental validation of self-supporting topologies for additive manufacturing. Virtual and Physical Prototyping, 2019, 14(4): 382–394
21. Meng L, Zhang W, Quan D, et al. From topology optimization

design to additive manufacturing: Today's success and tomorrow's roadmap. Archives of Computational Methods in Engineering, 2020, 27(3): 805–830
22. Chin T W, Kennedy G J. Large-scale compliance-minimization and buckling topology optimization of the undeformed common research model wing. In: Proceedings of the 57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference. San Diego: AIAA, 2016
23. Liu J, Ou H, He J, et al. Topological design of a lightweight sandwich aircraft spoiler. Materials, 2019, 12(19): 3225
24. Sutradhar A, Park J, Carrau D, et al. Designing patient-specific 3D printed craniofacial implants using a novel topology optimization method. Medical & Biological Engineering & Computing, 2016, 54(7): 1123–1135
25. Alexandersen J, Sigmund O, Aage N. Large scale three-dimensional topology optimisation of heat sinks cooled by natural convection. International Journal of Heat and Mass Transfer, 2016, 100: 876–891
26. Ye M, Gao L, Li H. A design framework for gradually stiffer mechanical metamaterial induced by negative Poisson's ratio property. Materials & Design, 2020, 192: 108751
27. Groen J P, Sigmund O. Homogenization-based topology optimization for high-resolution manufacturable microstructures. International Journal for Numerical Methods in Engineering, 2018, 113(8): 1148–1163
28. Wu Z, Xia L, Wang S, et al. Topology optimization of hierarchical lattice structures with substructuring. Computer Methods in Applied Mechanics and Engineering, 2019, 345: 602–617
29. Zhu B, Skouras M, Chen D, et al. Two-scale topology optimization with microstructures. ACM Transactions on Graphics, 2017, 36(4): 120b
30. Wang Y, Xu H, Pasini D. Multiscale isogeometric topology optimization for lattice materials. Computer Methods in Applied Mechanics and Engineering, 2017, 316: 568–585
31. Li H, Luo Z, Gao L, et al. Topology optimization for concurrent design of structures with multi-patch microstructures by level sets. Computer Methods in Applied Mechanics and Engineering, 2018, 331: 536–561
32. Li H, Luo Z, Xiao M, et al. A new multiscale topology optimization method for multiphase composite structures of frequency response with level sets. Computer Methods in Applied Mechanics and Engineering, 2019, 356: 116–144
33. Christiansen A N, Bærentzen J A, Nobel-Jørgensen M, et al. Combined shape and topology optimization of 3D structures. Computers & Graphics, 2015, 46: 25–35
34. Wang H, Liu J, Wen G. An efficient evolutionary structural optimization method with smooth edges based on the game of building blocks. Engineering Optimization, 2019, 51(12): 2089–2018
35. Nguyen T H, Paulino G H, Song J, et al. A computational paradigm for multiresolution topology optimization (MTOP). Structural and Multidisciplinary Optimization, 2010, 41(4): 525–539
36. Nguyen-Xuan H. A polytree-based adaptive polygonal finite element method for topology optimization. International Journal for Numerical Methods in Engineering, 2017, 110(10): 972–1000
37. Leader M K, Chin T W, Kennedy G J. High-resolution topology optimization with stress and natural frequency constraints. AIAA Journal, 2019, 57(8): 3562–3578
38. Chin T W, Leader M K, Kennedy G J. A scalable framework for large-scale 3D multimaterial topology optimization with octree-based mesh adaptation. Advances in Engineering Software, 2019, 135: 102682
39. Groen J P, Langelaar M, Sigmund O, et al. Higher-order multi-resolution topology optimization using the finite cell method. International Journal for Numerical Methods in Engineering, 2017, 110(10): 903–920
40. Gupta D K, van Keulen F, Langelaar M. Design and analysis adaptivity in multi-resolution topology optimization. 2018, arXiv:1811.09821v1
41. Xiao M, Lu D, Breitkopf P, et al. Multi-grid reduced-order topology optimization. Structural and Multidisciplinary Optimization, 2020, 61: 2319–2341
42. Lieu Q X, Lee J. Multiresolution topology optimization using isogeometric analysis. International Journal for Numerical Methods in Engineering, 2017, 112(13): 2025–2047
43. Xu M, Xia L, Wang S, et al. An isogeometric approach to topology optimization of spatially graded hierarchical structures. Composite Structures, 2019, 225: 111171
44. Wang Y, Liao Z, Ye M, et al. An efficient isogeometric topology optimization using multilevel mesh, MGCG and local-update strategy. Advances in Engineering Software, 2020, 139: 102733
45. Wang H, Liu J, Wen G. Achieving large-scale or high-resolution topology optimization based on modified BESO and XEFM. 2019, arXiv:1908.07157
46. Kim Y Y, Yoon G H. Multi-resolution multi-scale topology optimization—A new paradigm. International Journal of Solids and Structures, 2000, 37(39): 5529–5559
47. Stainko R. An adaptive multilevel approach to the minimal compliance problem in topology optimization. Communications in Numerical Methods in Engineering, 2006, 22(2): 109–118
48. Liao Z, Zhang Y, Wang Y, et al. A triple acceleration method for topology optimization. Structural and Multidisciplinary Optimization, 2019, 60(2): 727–744
49. Suresh K. Generating 3D topologies with multiple constraints on the GPU. In: Proceedings of the 10th World Congress on Structural and Multidisciplinary Optimization. Orlando, 2013, 1–9
50. Challis V J, Roberts A P, Grotowski J F. High resolution topology optimization using graphics processing units (GPUs). Structural and Multidisciplinary Optimization, 2014, 49(2): 315–325
51. Aage N, Andreassen E, Lazarov B S, et al. Giga-voxel computational morphogenesis for structural design. Nature, 2017, 550 (7674): 84–86
52. Long K, Gu C, Wang X, et al. A novel minimum weight formulation of topology optimization implemented with reanalysis approach. International Journal for Numerical Methods in Engineering, 2019, 120(5): 567–579
53. Wang Y, Liao Z, Shi S, et al. Data-driven structural design optimization for petal-shaped auxetics using isogeometric analysis. Computer Modeling in Engineering & Sciences, 2020, 122(2): 433–458
54. Zhou Y, Zhan H, Zhang W, et al. A new data-driven topology optimization framework for structural optimization. Computers &

Structures, 2020, 239: 106310
55. Sosnovik I, Oseledets I. Neural networks for topology optimization. Russian Journal of Numerical Analysis and Mathematical Modelling, 2019, 34(4): 215–223
56. Banga S, Gehani H, Bhilare S, et al. 3D topology optimization using convolutional neural networks. 2018, arXiv:1808.07440v1
57. Zhang Y, Chen A, Peng B, et al. A deep convolutional neural network for topology optimization with strong generalization ability. 2019, arXiv:1901.07761v1
58. Li B, Huang C, Li X, et al. Non-iterative structural topology optimization using deep learning. Computer-Aided Design, 2019, 115: 172–180
59. Dong C, Loy C C, He K, et al. Image super-resolution using deep convolutional networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2016, 38(2): 295–307
60. Bendsøe M P, Sigmund O. Topology Optimization: Theory, Methods, and Applications. Berlin: Springer, 2013, 37–40
61. Andreassen E, Clausen A, Schevenels M, et al. Efficient topology optimization in MATLAB using 88 lines of code. Structural and Multidisciplinary Optimization, 2011, 43(1): 1–16
62. Liu H, Wang Y, Zong H, et al. Efficient structure topology optimization by using the multiscale finite element method. Structural and Multidisciplinary Optimization, 2018, 58(4): 1411–1430