TopologyGAN: Topology Optimization Using Generative Adversarial Networks Based on Physical Fields Over the Initial Domain

Zhenguo Nie
Department of Mechanical Engineering
Tsinghua University
Beijing, China 100084
Email: zhenguonie@tsinghua.edu.cn

Tong Lin
Department of Mechanical Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
Email: tongl1@andrew.cmu.edu

Haoliang Jiang
School of Computer Science
Georgia Institute of Technology
Atlanta, GA 30332
Email: hjiang321@gatech.edu

Levent Burak Kara*
Department of Mechanical Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
Email: lkara@cmu.edu

In topology optimization using deep learning, the load and boundary conditions represented as vectors or sparse matrices often miss the opportunity to encode a rich view of the design problem, leading to less than ideal generalization results. We propose a new data-driven topology optimization model called TopologyGAN that takes advantage of various physical fields computed on the original, unoptimized material domain, as inputs to the generator of a conditional generative adversarial network (cGAN). Compared to a baseline cGAN, TopologyGAN achieves a nearly 3× reduction in the mean squared error and a 2.5× reduction in the mean absolute error on test problems involving previously unseen boundary conditions. Built on several existing network models, we also introduce a hybrid network called U-SE(Squeeze-and-Excitation)-ResNet for the generator that further increases the overall accuracy. We publicly share our full implementation and trained network.

*Address all correspondence to this author.

1 Introduction

Topology optimization of solid structures involves generating optimized shapes by minimizing an objective function such as compliance or mass within a material domain, subject to a set of displacement and load boundary conditions (Figure 1). With rapid advances in additive manufacturing and the associated design tools, topology optimization is becoming increasingly more prevalent as it allows optimized structures to be designed automatically. Existing methods include the density based approaches such as the Solid Isotropic Material with Penalization (SIMP) method [1,2,3,4,5,6,7,8,9,10], grid based approaches [11, 12, 13, 14, 15, 16], moving boundary based approaches [17, 18, 19, 20, 21, 22, 23, 24, 25], load-path based approaches [26, 27, 28], and optimal partitioning approaches [29, 30, 31]. Although significant efforts have been made to improve solution efficiency, topology optimization methods remain to be computationally demanding and are not readily suited to be used inside other design optimization modules such as layout or configuration design tools [25, 32, 33].

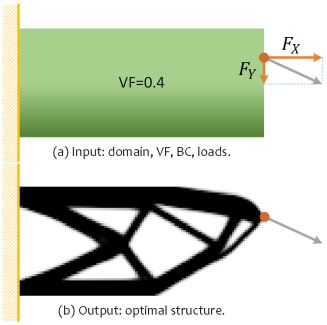

Fig. 1. 2D topology optimization.

In recent years, new data-driven methods for topology optimization have been proposed to accelerate the process. Deep learning methods have shown promise in efficiently producing near-optimal results with respect to shape similarity as well as compliance with negligible run-time cost [34,35,36,37,38,39,40,41]. Theory-guided machine learning methods use domain-specific theories to establish the mapping between the design variables and the external boundary conditions [42, 43, 44]. However, a significant challenge in topology optimization is learning an accurate and generalizable mapping from the boundary conditions to the resulting optimal structure. As such, approaches that involve establishing this map directly often have to severely restrict the displacement and external load configurations, as the results are difficult to generalize to novel boundary conditions.

As one step toward addressing this issue, we propose a new deep learning based generative model called TopologyGAN for topology optimization. TopologyGAN is based on conditional Generative Adversarial Networks (cGAN). The main hypothesis we pursue is that rather than trying to map the boundary conditions to the resulting optimal shapes directly, various physical fields computed on the initial, unoptimized domain subject to the prescribed boundary conditions may hold useful information that allows the network to learn more accurate maps. This approach has been motivated by our observation that in deep learning based approaches, the input displacement and load boundary conditions are often represented as sparse vectors or matrices[1]. This sparsity prevents the network from exploiting useful spatial variations and physical phenomena that occur within the material domain. By contrast, in this work, we propose to take advantage of such variations.

To this end, in TopologyGAN (*i*) the input channels of the generator are related to the physical fields (non-sparse matrices) computed in the initial, unoptimized domain such as the von Mises stress fields, strain energy fields, and displacement fields rather than the original boundary conditions alone, and (*ii*) the condition for the discriminator involve both the physical fields and the initial inputs. We use ground truth data generated by the SIMP method, although our approach is applicable to ground truth data obtained from other topology optimization methods.

[1]Unless, of course, they are fixed, in which case the network cannot account for variations in these conditions.

Based on this formulation and the model selection studies we conduct, we find that the von Mises stress field and the strain energy density fields are the most useful channels of information to augment with the original displacement and load boundary conditions. Our results show that compared to a baseline cGAN model that does not take advantage of such fields, TopologyGAN achieves lower *test* errors than that of the *training* error of the baseline method on previously unseen boundary conditions. These results are encouraging in that they may prove useful for other researchers who may wish to explore the use of such fields in similar topology optimization scenarios.

We publicly share our full implementation and trained network (`https://github.com/zhenguonie/2020_TopologyGAN`). The entire data set used in this work is available freely upon request.

Our main contributions are:

- A new method, TopologyGAN, for topology optimization using deep learning models.
- A new design of the input matrices involving the initial physical fields. This input complements the original problem input matrices.
- A hybrid neural network architecture, namely U-SE-ResNet, as the generator for TopologyGAN.

2 Related Work

Our review focuses on studies that highlight topology optimization, deep learning for topology optimization, generative adversarial networks (GAN), and two network architectures closely related to our work.

2.1 Topology Optimization and SIMP

Topology optimization seeks an optimal subset $\Omega_{mat} \subset \Omega$, where Ω is the design domain. To formulate this problem, an objective function $f(y)$ to be minimized is defined as in Eq.(1), in which y denotes the structural design (material distribution)[2] and $h(y)$ is a resulting physical outcome such as stress, strain, or displacement.

$$f(y) = \begin{cases} \min\limits_{y} & f(y, h(y)) \\ \texttt{s.t.} & \begin{cases} \text{design constraint on } y \\ \text{state constraint on } h(y) \\ \text{equilibrium constraint} \end{cases} \end{cases} \tag{1}$$

In this work, we use the density-based SIMP method ([1,45]), which is widely implemented in many commercial design software [46, 47]. The SIMP method discretizes the design domain into a grid of finite elements called isotropic solid microstructures. It models the material density y_e to vary between zero (void) and one (full solid). This representation allows the assignment of intermediate densities to the

[2]We use y for material distribution for consistency with the cGAN output presented in the following sections.

elements. The Young's modulus E_e for each grid element e is given as:

$$E_e(y_e) = E_{min} + y_e^p(E - E_{min}) \tag{2}$$

where E is the material stiffness, E_{min} is an infinitesimal stiffness and p is a penalization factor to favor binary outputs avoiding intermediate densities. The optimization works toward minimizing the compliance $C(y)$ ([48,49]) as follows:

$$\left.\begin{aligned} \min_y: \quad & C(y) = \mathbf{U}^T\mathbf{K}\mathbf{U} = \sum_{e=1}^{N}(y_e)^p\mathbf{u}_e^T\mathbf{k}_e\mathbf{u}_e \\ \texttt{s.t.}: \quad & \tfrac{V(y)}{V_0} = \mathrm{VF} \\ : \quad & \mathbf{K}\mathbf{U} = \mathbf{F} \\ : \quad & 0 \le y_e \le 1 \end{aligned}\right\} \tag{3}$$

where y is the density-based structural design tensor, $\mathbf{U}$ and $\mathbf{F}$ are the global displacement and force vectors, $\mathbf{K}$ is the stiffness matrix, $\mathbf{u}_e$ is the elemental displacement vector, $\mathbf{k}_e$ is the elemental stiffness matrix, and N is the number of total elements.

While the above existing methods can provide optimized solutions, our work aims to accelerate the iterative nature of these solvers using a data-driven approach.

2.2 Deep Learning for Topology Optimization

With recent advances in computer hardware, deep neural networks have been widely applied in various fields, including autonomous vehicles, robotics, medical diagnosis, bio-medicine, material design, machine health monitoring, mechanical design, and manufacturing. Deep neural networks have proven to be effective at learning complex mappings between problem input variables and constraints, and target design objectives. Supervised machine learning techniques have proven to be effective for engineering design exploration and optimization, and for mapping out feasible regions of the design space [50]. Guo et al. [34] propose a data-driven design representation where an augmented variational autoencoder is used to encode 2D topologies into a lower-dimensional latent space and to decode samples from this space back into 2D topologies. Oh et al. [51] propose a deep generative adversarial framework capable of generating numerous design alternatives that are both aesthetic and optimized for engineering performance. In 3D topology optimization, Rawat and Shen [36] integrate Wasserstein GAN and a convolutional neural network (CNN) into a coupled model to generate new 3D structures using limited data samples. To speed up convergence in SIMP based topology optimization for thermal conduction, Lin et al. [52] introduce a deep learning approach using U-Nets. Through deep learning, only the early results obtained through SIMP are fed into the network to directly produce the final outputs. Sosnovik and Oseledets [53] use CNNs to accelerate topology optimization from two halfway grayscale images to the final binary image generated through SIMP. However, these networks focus either on latent candidate generation or accelerated optimization and do not establish an end-to-end mapping from the boundary conditions to the optimized topologies.

To realize an end-to-end topology optimization from prescribed boundary conditions, Yu et al. [35] propose a CNN-based encoder-decoder for the generation of low-resolution structures, which are then passed through a super-resolution GAN to generate the final results. Sharpe and Seepersad [37] explore the use of cGANs as a means of generating a compact latent representation of structures resulting from topology optimization. However, only a few boundary conditions and optimization settings are considered. Extending data-driven topology optimization to novel displacement and external load conditions remains a major challenge.

2.3 Generative Adversarial Networks

Generative Adversarial Networks (GAN) [54], is a generative model formulated as a minimax two-player game between two models. It consists of: (1) A generator G whose aim is to learn a generative density function that models the training data and, (2) a discriminator D that aims to discern if an input sample is part of the original training set or a synthetic one generated by G. The structure of GAN is shown in Figure 2. The input to G is random noise z sampled from a distribution $p_z(z)$. The output of G, $y_g = G(z)$, is a fake data. A real y_r or a fake sample y_g is then fed into D to obtain an output $D(y)$, which generates a probability estimate of the true nature of y. In this case, D is trained to maximize the probability of assigning the correct label to both the real samples y_r and fake samples y_g. G is simultaneously trained to minimize $\log(1 - D(G(z)))$. The training loss functions $\mathcal{L}_D^{GAN}$ and $\mathcal{L}_G^{GAN}$ are:

$$\max_D \mathcal{L}_D^{GAN} = \mathbb{E}_{y_r \sim p_{data}(y)}[\log D(y_r)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))] \tag{4}$$

$$\min_G \mathcal{L}_G^{GAN} = \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))] \tag{5}$$

Conditional GAN Generative adversarial networks can be extended to a conditional model when the generator and the discriminator are conditioned on prescribed information [55]. A conditional GAN (cGAN) learns a mapping from an input x to an output y, as shown in Figure 3. The cGAN loss ([56]) is:

$$\begin{aligned} \mathcal{L}_{G,D}^{cGAN} = \mathbb{E}_{(x,y)\sim p_{data}(x,y)}[\log D(x,y)] + \\ \mathbb{E}_{x \sim p_{data}(x), z \sim p_z(z)}[\log(1 - D(x, G(x,z)))] \end{aligned} \tag{6}$$

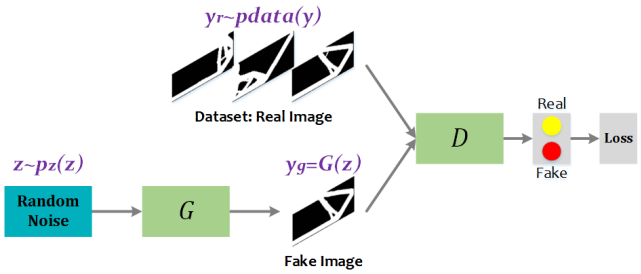

Fig. 2. Schematic diagram of GAN.

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x,z)\|_1] \tag{7}$$

$$G^* = \arg\min_G \max_D \mathcal{L}^{cGAN}_{G,D} + \lambda \mathcal{L}_{L1}(G) \tag{8}$$

where G^* is the final optimized generative model. In this model, the inputs x are composed of the original (full) domain, the desired volume fraction, the displacement boundary conditions, and the external loads. These are utilized as conditions that the generator has to attune to. The ground truth final optimized structure (real structure) y is computed through the SIMP method and provided as input to the discriminator, alongside x for training.

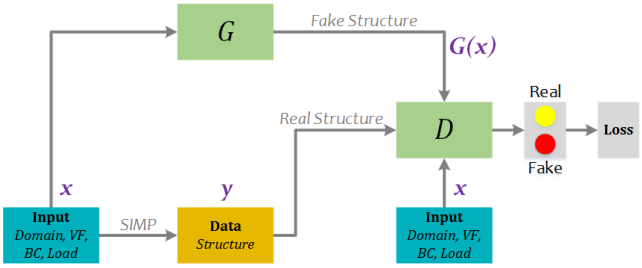

Fig. 3. Baseline: cGAN for topology optimization.

2.4 U-Net and SE-ResNet

The U-Net architecture ([57]) shown in Figure 4 allows the network to propagate context information from the downsampling layers to the upsampling layers at various resolutions. To predict the output around the border of the image, the missing context is extrapolated by reflecting the input image. This tiling strategy is important when the network is applied to large images, as otherwise the resolution would be limited by the GPU capacity [58].

SE-ResNet is a CNN-based ResNet [59] enhanced by SE-blocks [60] as shown in Figure 5. Each SE-ResNet module contains two CNN blocks and one SE block [61]. The

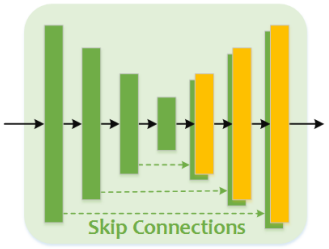

Fig. 4. Architecture of the U-Net. The skip connections between networks are illustrated as dotted arrows.

distinguishing feature of ResNet is the skip connection which simply performs identity mapping added to the output of the stacked layers. As such, ResNet can dynamically select the layer depth for the desired underlying mapping. The final output of the SE-ResNet module is computed by a feedforward neural network with a shortcut connection: $w = v + x$.

SE block is used in SE-ResNet to improve the representational capacity of the network by enabling it to perform dynamic channel-wise feature recalibration. The input data $u \in R^{H\times W\times C}$ is shrunk to $S(u) \in R^C$ through the global average-pooling layer. Then two fully connected layers are used to downsample and upsample the linear array $S(u)$. A reshape operation produces the excitation output $E(u)$ that has the same dimension as the initial input u. The final output of the SE block is a Hadamard product of $E(u)$ and u through the element-wise matrix multiplication: $v = E(u) \otimes u$.

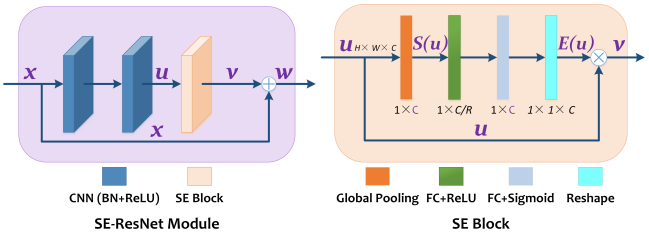

Fig. 5. ResNet enhanced by SE block [61].

3 Technical Approach

Topology optimization using deep learning has difficulties in extending to previously unseen boundary conditions because of the high discreteness of the boundary conditions and the sparsity of the input matrices. The high sparsity of the input matrices leads to high variance of the mapping function.

As one step toward overcoming this challenge, we propose a new model called TopologyGAN. The method is based on the use of various physical fields such as the strain energy field, von Mises stress field, displacement fields, and strain fields computed on the *initial* (unoptimized) design domain as a way to augment the baseline cGAN with this extra information. In this work, we denote these as the *initial fields* f. An illustrative schematic of how TopologyGAN works is shown in Figure 6. The horizontal axis is composed of the

problem input matrices encoding the desired VF, displacement BC, and load. The vertical axis is the resulting structure where three structural designs are generated respectively by the SIMP method as the ground truth, by the cGAN as the baseline, and by TopologyGAN as the proposed model. The baseline cGAN directly maps the inputs to the output structure: $x \xrightarrow{\text{cGAN}} y$. Our proposed TopologyGAN, on the other hand, builds a mapping from the inputs x to the initial fields f, followed by a mapping from f to the output structure y as follows: $x \xrightarrow{\text{FEM}} f + \text{VF} \xrightarrow{\text{TopologyGAN}} y$. The inputs of the generator in TopologyGAN mainly include two parts: the initial physical field f (e.g., strain energy density, stress field) and VF. Note that during run time, the initial fields are computed only once.

Our hypothesis in utilizing such initial fields is that they provide useful information regarding the physical state of the original domain under the inputs x that can be further exploited by the network. For instance, as shown in Figure 6, both the initial strain energy density and the von Mises stress maps produce scalar fields that are richer in information compared to the original problem input matrices x alone. In particular, the initial fields are hypothesized to produce information f (green solid curve) that correlates well with the final structure y (yellow dashed curve), thereby making the network's remaining task less daunting compared to the scenario of mapping x to y directly. As will be shown in Section 5.1, the initial fields indeed help TopologyGAN attain significantly higher training and test accuracies over the baseline cGAN.

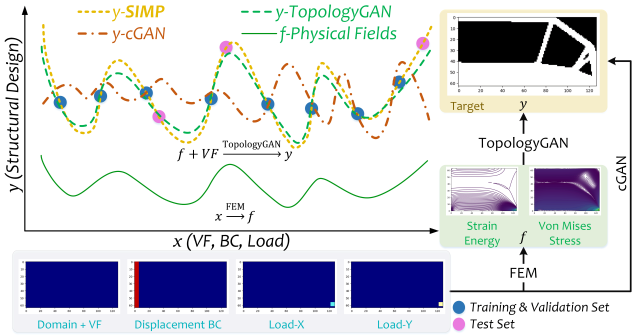

Fig. 6. Schematic diagram of TopologyGAN.

Both the generator and the discriminator utilize CNN blocks as shown in Figure 8. For the generator, we propose a hybrid architecture called U-SE-ResNet. U-SE-ResNet is a U-Net with the SE-ResNet module in a downsampling-upsampling structure. In the upsampling, transposed convolution layers are used instead of pooling layers. The Sigmoid function is used in the generator's last layer for outputting a grayscale structure $\hat{y}$. According to [56], PatchGAN discriminator is adopted because the problem of blurry images caused by failures at high-frequency features such as edges and textures can be alleviated by restricting the GAN discriminator to only model high frequencies. The discriminator is conditioned on the extra information r, where $r = x$ for cGAN, and $r = x + f(x)$ for TopologyGAN. The structure, y or $G(f)$, concatenates with the conditions r together as the inputs to the discriminator: $[y \vee G(f)] + r \Rightarrow D$, then the discriminator learns to determine whether the structure is real or fake. To fairly compare the performance of cGAN and TopologyGAN, the same generator and discriminator architectures are implemented in the two deep learning models.

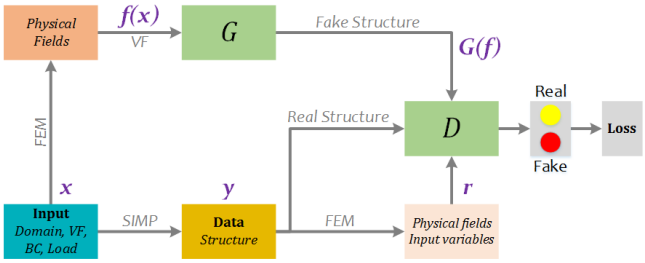

Fig. 7. TopologyGAN approach to topology optimization.

Fig. 8. Architectures of the generator G and discriminator D.

3.1 Network Architecture of TopologyGAN

TopologyGAN is based on the conditional Pixel2Pixel GAN architecture [56] and incorporates the FEM and the SIMP solver, as shown in Figure 7. The generator of TopologyGAN establishes a mapping from the initial physical fields f to the estimated optimized topology $\hat{y} = G(f(x), \text{VF})$. The ground truth output y is generated by the SIMP method, as was the case with the baseline cGAN shown in Figure 3. In TopologyGAN, both the problem inputs x and the initial fields $f(x)$ are used as the condition: $r(x) = [x, f(x)]$.

3.2 TopologyGAN Loss Function

As shown in Eq.(9) - Eq.(12), the loss consists of three parts: (1) the loss of TopologyGAN: $\mathcal{L}_{G,D}^{TGAN}$, (2) the L2 loss of the generator: $\mathcal{L}_{L2}(G)$, and (3) the absolute error of the volume fraction of the generator: AE^{VF}.

$$\mathcal{L}_{G,D}^{TGAN} = \mathbb{E}_{(x,y)\sim p_{data}(x,y)}[\log D(r(x), y)] + \mathbb{E}_{x\sim p_{data}(x)}[\log(1 - D(r(x), G(f(x))))] \tag{9}$$

$$\mathcal{L}_G^{TGAN} = \mathbb{E}_{x\sim p_{data}(x)}[\log(1 - D(r(x), G(f(x))))] \tag{10}$$

$$\mathcal{L}_{L2}(G) = \mathbb{E}_{x,y}[\|y - G(f(x))\|_2] \tag{11}$$

$$\mathrm{AE}_G^{\mathrm{VF}} = \mid \mathrm{VF} - \widehat{\mathrm{VF}} \mid = \frac{1}{N} \mid \sum_{e=1}^{N} (y_e - G(f(x))_e) \mid \tag{12}$$

$$G^* = \arg\max_D \min_G \mathcal{L}_{G,D}^{TGAN} + \lambda_1 \mathcal{L}_{L2}(G) + \lambda_2 \mathrm{AE}_G^{\mathrm{VF}} \tag{13}$$

where the generator G tries to minimize the combined loss $\mathcal{L}_G^{TGAN} + \lambda_1 \mathcal{L}_{L2}(G) + \lambda_2 \mathrm{AE}_G^{\mathrm{VF}}$. At the same time, the adversarial D tries to maximize $\mathcal{L}_{G,D}^{TGAN}$. $r(x)$ is the condition that contains the problem inputs x and the initial physical fields $f(x)$. n is pixel count of the output image. Scalars λ_1 and λ_2 are used to balance the three loss functions. The two scalars are determined empirically using parametric studies. In this paper, $\lambda_1 = 10{,}000$, $\lambda_2 = 1$.

We select the cGAN as our baseline model. Its structure is shown in Figure 3. Its loss is shown in Eq.(8).

4 Experiments

The proposed TopologyGAN and the baseline cGAN are trained on the same data set that is generated by the SIMP method, and then are evaluated and compared on their prediction performance.

4.1 Experiment Design

In addition to a comparison of TopologyGAN and the baseline cGAN, we also study the impact of different physical fields as inputs. Additionally, we compare different generator structures, namely U-Net, SE-ResNet, and the proposed U-SE-ResNet.

4.2 Data set

A SIMP-based topology optimization solver called ToPy [62] is used to generate the data set. A total of 49,078 optimized structures are generated with the randomized conditions as follows:

- volume fraction: $[0.3 : 0.02 : 0.5]$
- displacement BCs: 42 scenarios
- load position: nodes on the boundary of the domain
- load direction: $\left[0 : \frac{\pi}{6} : \pi\right]$
- SIMP penalty: 2
- SIMP filter radius: 1.5

The 2D domain consists of a 64×128 FEM node consisting of squared elements. BCs and loads are applied to the nodes. We use SolidsPy [63] to obtain the physical fields over the initial domain. The stress and strain fields are integrated on the Gauss points and then extrapolated to the nodes in the same way. All the physical fields are represented in the form of the node matrix.

The data set is divided into training, validation, and test sets as follows: 4 displacement BCs of the total 42 are randomly selected as the test set. All data samples from the remaining 38 displacement boundary conditions are shuffled and split into training (80%) and validation (20%) sets. Note that the boundary conditions in the test set will not have been seen by the network during training.

Six samples are randomly selected from the entire data set for an illustration of the input and outputs in Figure 9. Images in each row form a sample for TopologyGAN. The first four images in each row are input variables x = (VF,BC, F), including volume fraction (VF), displacement boundary condition (BC), and external loads (Load) acting along the x-axis and the y-axis: (1) VF is input as a 2D matrix that has the same dimension as the design domain. The value of each element in the matrix is equal to the VF value, and colored in dark blue. (2) Displacement BC is represented as a 2D matrix by assigning one of the four integers to each element: 0 represents unconstrained (colored in dark blue), 1 represents $u_x = 0$ (colored in red), 2 represents $u_y = 0$ (colored in green), and 3 represents $u_x = u_y = 0$ (colored in light blue). (3) External loads acting along the x-axis and the y-axis are respectively represented as two 2D matrices, where 0 represents zero force (colored in dark blue), non-zero entry represents external load magnitude and location (colored in red). Based on the input variables, the initial stress ($\sigma = [\sigma_{11}, \sigma_{22}, \sigma_{12}]$), strain ($\varepsilon = [\varepsilon_{11}, \varepsilon_{22}, \varepsilon_{12}]$), and displacement ($u = [u_x, u_y]$) fields are computed by FEM. Strain energy density and von Mises stress are shown as the example physical fields, which are expressed in Eq(15) and Eq(14). The last image is the output structure y from SIMP.

$$W = \frac{1}{2}(\sigma_{11}\varepsilon_{11} + \sigma_{22}\varepsilon_{22} + 2\sigma_{12}\varepsilon_{12}) \tag{14}$$

$$\sigma_{\mathrm{vm}} = \sqrt{\sigma_{11}^2 - \sigma_{11}\sigma_{22} + \sigma_{22}^2 + 3\sigma_{12}^2} \tag{15}$$

where W is strain energy density, σ_{vm} is von Mises stress, σ_{ij} are the components of the stress tensor.

4.3 Evaluation Metrics

To assess the performance of TopologyGAN, we use the following four metrics: mean absolute error (MAE), mean squared error (MSE), relative error of volume fraction ($\mathrm{RE}^{\mathrm{VF}}$)

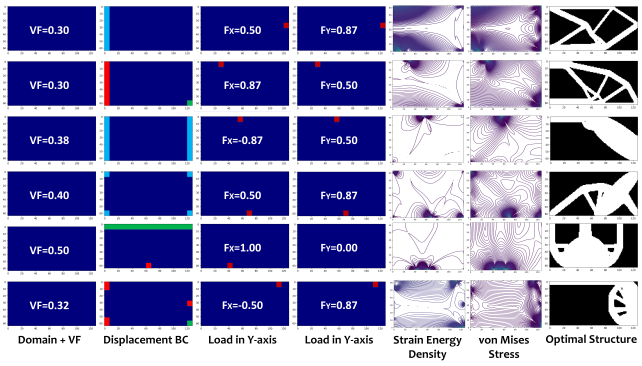

Fig. 9. Data samples generated by FEM solver and SIMP solver. Images in each row form one sample. The first four images in each row are input variables x(VF,BC,Load), the last image is output y. Strain energy density and von Mises stress are shown in level sets with contour lines.

and its absolute value ($|\,\mathrm{RE}^{\mathrm{VF}}\,|$), relative error of the compliance (RE^{C}) and its absolute value ($|\,\mathrm{RE}^{\mathrm{C}}\,|$). Denote $\hat{y}$ as the prediction from the generator G:

$$\hat{y} = G(f(x)), \tag{16}$$

and then both the ground truth y and the prediction $\hat{y}$ are reshaped into vectors, with the length of 8,192, from 2D matrices, with the size of 64×128.

MAE shown in Eq.(17) and MSE shown in Eq.(18) are used to evaluate the model. MAE measures the average magnitude of the absolute differences between the prediction values and the ground truth. MSE measures the average squared difference between the estimated values and the ground truth.

$$\mathrm{MAE} = \frac{1}{M}\sum_{i=1}^{M} |\, y^{(i)} - \hat{y}^{(i)} \,| = \frac{1}{M}\sum_{i=1}^{M}\frac{1}{N}\sum_{e=1}^{N} |\, y_e^{(i)} - \hat{y}_e^{(i)} \,| \tag{17}$$

$$\mathrm{MSE} = \frac{1}{M}\sum_{i=1}^{M}(y^{(i)} - \hat{y}^{(i)})^2 = \frac{1}{M}\sum_{i=1}^{M}\frac{1}{N}\sum_{e=1}^{N}(y_e^{(i)} - \hat{y}_e^{(i)})^2 \tag{18}$$

where M is the total number of data samples, and $N = 8{,}192$ is the number of grid elements.

In addition to these commonly used metrics, we define two other metrics for evaluation: (1) $\mathrm{RE}^{\mathrm{VF}}$ is the relative error of the volume fraction between prediction output and ground truth output, and (2) RE^{C} is the ratio between the compliance of the predicted structure and the ground truth structure. These are defined as follows:

$$\mathrm{VF} = \frac{1}{N}\sum_{e=1}^{N} y_e \tag{19}$$

$$\mathrm{RE}^{\mathrm{VF}} = \frac{\widehat{\mathrm{VF}} - \mathrm{VF}}{\mathrm{VF}} = \frac{\sum_{e=1}^{N}(\hat{y}_e - y_e)}{\sum_{e=1}^{N} y_e} \tag{20}$$

$$C(y) = \mathbf{U}^T\mathbf{K}\mathbf{U} = \sum_{e=1}^{N}(y_e)^p \mathbf{u}_e^T \mathbf{k}_0 \mathbf{u}_e \tag{21}$$

$$\mathrm{RE}^{\mathrm{C}} = \frac{C(\hat{y}) - C(y)}{C(y)} \tag{22}$$

where $C(y)$ is the compliance of the predicted structure under the given loads.

5 Results and Discussions

Our code is written in TensorFlow, and trained and tested on an NVIDIA GeForce GTX 2080Ti GPU. Adam [64] is used for optimization, combining the advantages of AdaGrad [65] and RMSProp [66]. It takes 158 seconds for the trained TopologyGAN to generate the topology images for the set of 4,356 test samples, while it takes 105 seconds for the trained cGAN. In contrast, it takes approximately 145 hours for the topology optimization software ToPy to accomplish the computation of the same number of test samples (on Intel i7-6500U CPU).

We find that TopologyGAN outperforms the baseline cGAN in both the training and test. Furthermore, we analyze the physical field selection and the network architecture for the generator. We find that the VF+σ_{vm}+W combination performs significantly better than all other combinations. Moreover, to find the best generator structure, we compare the U-SE-ResNet, U-Net, and SE-ResNet. A comparative experiment is conducted using different generators on the same discriminator and the data set. The results show that the proposed U-SE-ResNet outperforms the other two structures.

5.1 Model Evaluation

Our training results of TopologyGAN and the baseline cGAN are summarized in Table 1. MSE and MAE of TopologyGAN in training are shown in Figure 10. It can be seen that TopologyGAN achieves a 3× lower MSE error and 2.5× lower MAE error than that of the cGAN. More surprisingly, both the MSE and MAE of TopologyGAN on the *test* set are lower than those of cGAN on the *training* set.

Loss functions of TopologyGAN are shown in Figure 11, where $\mathcal{L}_{G,D}^{TGAN}$ is the discriminator loss of TopologyGAN, $\mathcal{L}_{G}^{TGAN}$ is the generator loss of TopologyGAN, and G^* is the whole objective of the generator. As the training progresses, it can be seen that $\mathcal{L}_{G}^{TGAN}$ and G^* decrease gradually, and $\mathcal{L}_{G,D}^{TGAN}$ oscillates and tends to balance.

Table 1. Comparison of results between TopologyGAN and cGAN.

Model	MAE			MSE		
	Training	Validation	Test	Training	Validation	Test
TopologyGAN	0.001808	0.019133	0.070128	0.001340	0.018022	0.059943
Baseline: cGAN	0.088257	0.100565	0.181095	0.085916	0.097966	0.175226

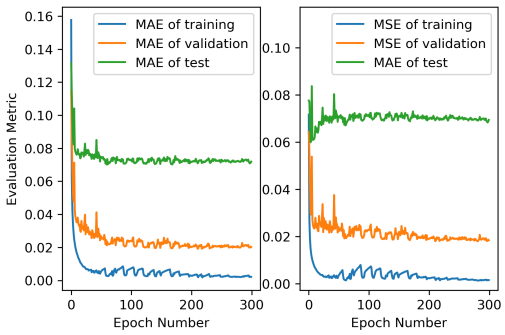

Fig. 10. MAE and MSE of TopologyGAN in training.

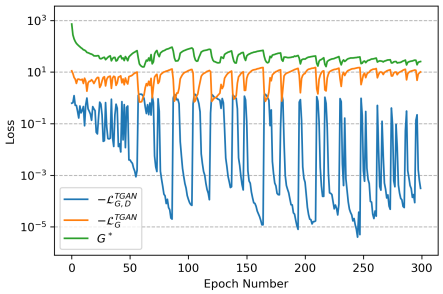

Fig. 11. Loss functions of TopologyGAN, Discriminator and Generator in training.

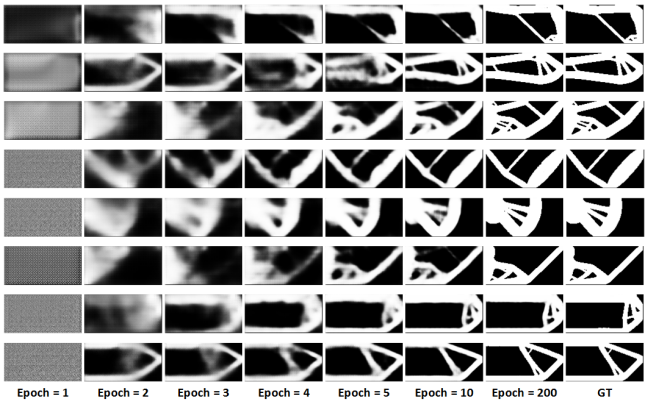

Fig. 12. Evolution of TopologyGAN predictions during training.

5.2 Accuracy and Performance

The accuracy and performance of the trained TopologyGAN are discussed to show the prediction performance. As shown in Figure 12, the generated structures from TopologyGAN become increasingly more refined over the training epochs. Each row denotes a randomly selected sample from the training set. After two-hundred epochs, the predictions become virtually indistinguishable from the ground truth (GT) structures.

To visually compare the prediction accuracy of the fully trained TopologyGAN on the training, validation, and test sets, Figure 13 shows the computed results and the corresponding ground truth structures. By comparing each set of images, we find that TopologyGAN performs well on the training and validation sets. The performance of TopologyGAN on the test set is expectedly lower. The generated structures may exhibit disconnected fragments, which may not impact the volume fraction but may have a considerable effect on the resulting compliance.

As such, to further quantify the performance of cGAN and TopologyGAN, we compute the VF and the compliance of the resulting structures and compare them to those of the ground truth structures. Based on the fully trained TopologyGAN, we randomly select 640 samples from each of the training and test sets. For each data sample, this results in two structures: a prediction $\hat{y}$ and a ground truth y. The relative error of the volume fraction $\mathrm{RE}^{\mathrm{VF}}$ and the relative error of compliance RE^{C} are computed respectively using Eq.(20) and Eq.(22).

The sorted $\mathrm{RE}^{\mathrm{VF}}$ for the training and test sets (as well as their histograms) for the cGAN and TopologyGAN are shown in Figure 14 and Figure 15, respectively. Specifically for the test sets (orange), both cGAN and TopologyGAN are zero-centered, but the latter having a smaller variance which aligns well with the results in Table 1 (an F-test rejects the null hypothesis that the two variances are equivalent with $p < 0.05$). RE^{C} shows a similar pattern as shown in Figure 16 and Figure 17, with the TopologyGAN producing a smaller variance in the compliance error (an F-test rejects the null hypothesis that the two variances are equivalent with $p < 0.05$).

5.3 Comparison and Selection of Physical Fields

In addition to the combination of von Mises stress σ_{vm} and strain energy density W, various physical field combinations are studied as the inputs. The comparison is shown in Table 2. The results indicate that №8 - $[\mathrm{VF} + U + \sigma_{\mathrm{vm}} + W]$ has the best performance on the training and validation sets, but №4 - $[\mathrm{VF} + \sigma_{\mathrm{vm}} + W]$ shows a better prediction performance on the test set.

Table 2. Comparison and selection of physical fields as GAN inputs.

Metrics		MAE			MSE		
№	Physical Fields	Training	Validation	Test	Training	Validation	Test
0	Baseline	0.088257	0.100565	0.181095	0.085916	0.097966	0.175226
1	VF+U	0.001941	0.039551	0.105560	0.002391	0.030464	0.099863
2	VF+W	0.001781	0.039145	0.100626	0.002343	0.033687	0.094903
3	VF+σ_{vm}	0.001758	0.040411	0.098619	0.002333	0.035467	0.081702
4	VF+σ_{vm}+W	0.001808	0.019133	0.070128	0.001340	0.018022	0.059943
5	VF+σ	0.002339	0.037105	0.101626	0.002382	0.031802	0.095526
6	VF+ε	0.001729	0.034620	0.093073	0.002306	0.029235	0.087086
7	VF+σ+ε	0.010823	0.037518	0.122126	0.001976	0.032496	0.092162
8	VF+U+σ_{vm}+W	0.000942	0.019519	0.088748	0.001099	0.031628	0.079184
9	VF+LP	0.001914	0.033079	0.139781	0.00328	0.037652	0.100326

Note: VF is volume fraction, F is external loads, U is displacement field, σ_{vm} is von Mises stress, W is strain energy density, σ is stress field, ε is strain field, and LP is load path vector. Number marked in yellow is the minimum in each column.

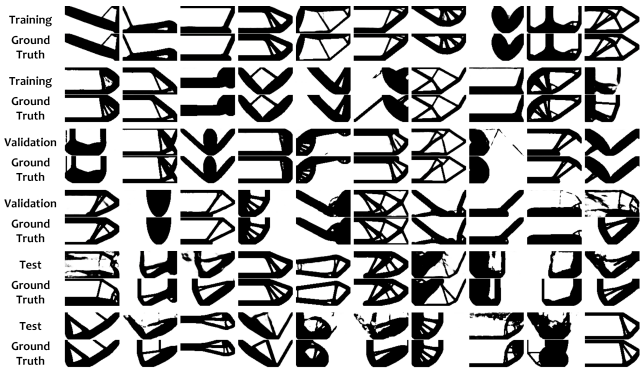

Fig. 13. Comparison of the predictions of the fully trained TopologyGAN on training, validation and test sets.

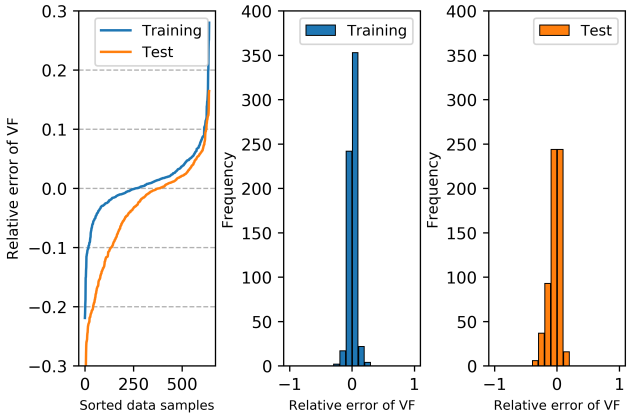

Fig. 14. Comparison of volume fraction on training and test sets in cGAN

5.4 Generator Architecture

The most important feature of the U-Net is the skip connections between mirrored downsampling and upsampling layers to transfer local information and to merge features at the various resolution levels. The most distinct feature of SE-ResNet is the shortcut connection which performs identity mapping added to the output of the stacked layers to dynamically select the layer depth for the desired underlying mapping. Served as the generator of TopologyGAN, the proposed architecture U-SE-ResNet in this article combines U-Net and SE-ResNet, as shown in Figure 18. We train the TopologyGAN model with different generator architectures, which are U-Net, SE-ResNet, and U-SE-ResNet. The training results are shown in Table 3. U-SE-ResNet has the best performance of the three. U-Net allows the networks to propagate context information from the downsampling layers to the upsampling layers at various resolutions, which makes the output take advantage of a spatially broader view combining localization and context for a better prediction. SE-ResNet improves the representational capacity and dynamically selects the layer depth for the desired underlying mapping. U-SE-ResNet combines the advantages of U-Net and SE-ResNet, which improve the model flexibility in the local information transfer and the adjustable network depth.

5.5 Limitations and Future Work

TopologyGAN exhibits good performance compared to the baseline cGAN and generalizes to previously unseen boundary conditions. However, there are several limitations. First, there is no in-built mechanism that ensures a single con-

Table 3. Training results of the three generator architectures

Architecture	MAE			MSE		
	Training	Validation	Test	Training	Validation	Test
U-Net	0.002908	0.027434	0.101455	0.002471	0.029133	0.098439
SE-ResNet	0.008597	0.100675	0.142915	0.058231	0.089755	0.157362
U-SE-ResNet	0.001808	0.019133	0.070128	0.001340	0.018022	0.059943

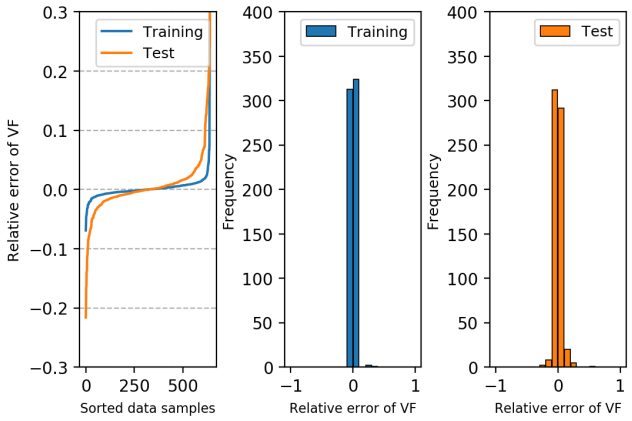

Fig. 15. Comparison of volume fraction on training and test sets in TopologyGAN

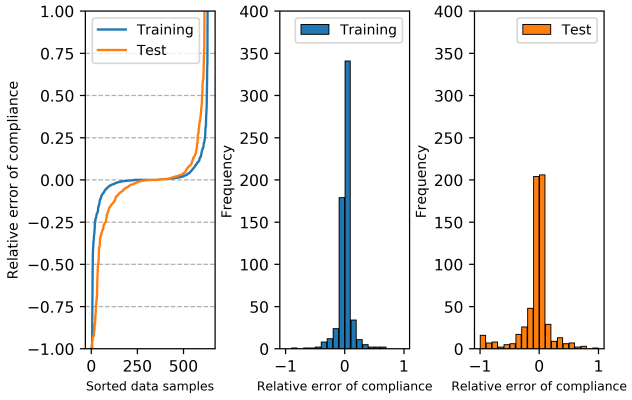

Fig. 16. Comparison of compliance on training and test sets in cGAN.

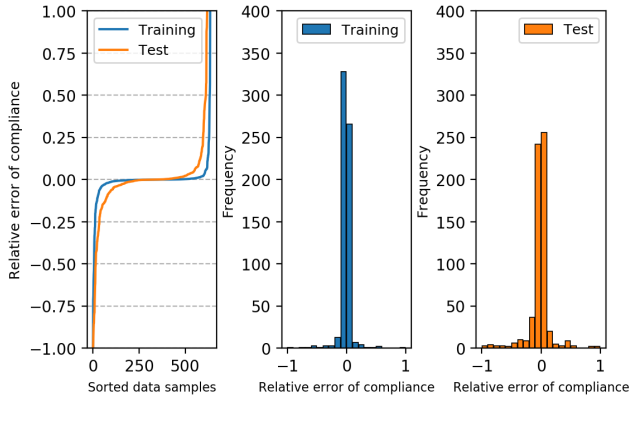

Fig. 17. Comparison of compliance on training and test sets in TopologyGAN.

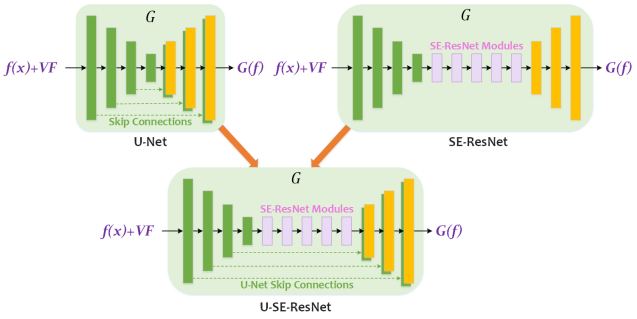

Fig. 18. Architectures of U-Net, SE-ResNet, and U-SE-ResNet

nected component (i.e., avoidance of disconnected regions or a checkerboard pattern) outside of the penalization enhanced SIMP-based ground truth training data the network observes. Second, while VF is typically employed as an upper bound for the amount of material that SIMP can utilize, TopologyGAN treats VF as a condition to match as closely as possible (minimizing the absolute error of the target and generated VF) rather than a true inequality constraint. Third, TopologyGAN is implemented for 2D topology optimization and, while the approach will be similar in nature, will require modifications to extend to 3D for generalization. Fourth, TopologyGAN was trained on the 64×128 2D domain and cannot be applied directly to other size domains. To make the transition to a different size, TopologyGAN needs a retraining on the new data set.

TopologyGAN only takes advantage of the fields computed on the original, unoptimized domain. However, theoretically, there is no restriction on which fields can be used to augment the network. In particular, a new network could be devised where the structure generated by TopologyGAN is reassessed to compute the fields of interest and this information can be fed back to the network for improved predictions in several iterative (yet fast) steps. These limitations and observations will be the subject of our future work.

6 Conclusions

We present a new deep learning based generative model called TopologyGAN for topology optimization where the displacement and load boundary conditions, and the target volume fraction are prescribed as inputs. TopologyGAN uses dense initial fields computed over the original, unoptimized domain to augment the conventional inputs. These new fields prove to be helpful for the network to outperform a baseline cGAN in significant ways. In addition, we propose a new hybrid generator architecture called U-SE-ResNet that combines the features of U-Net and SE-ResNet. The use of the initial fields, as well as the U-SE-ResNet model, allows TopologyGAN to significantly reduce the test errors on problems involving previously unseen boundary conditions.

References

[1] Bendsøe, M. P., and Kikuchi, N., 1988. "Generating optimal topologies in structural design using a homogenization method". *Computer methods in applied mechanics and engineering,* ***71***(2), pp. 197–224.

[2] Suzuki, K., and Kikuchi, N., 1991. "A homogenization method for shape and topology optimization". *Computer methods in applied mechanics and engineering,* ***93***(3), pp. 291–318.

[3] Bendsøe, M. P., 2009. *Topology optimization*. Springer.

[4] Diaz, A., and Bendsøe, M., 1992. "Shape optimization of structures for multiple loading conditions using a homogenization method". *Structural optimization,* ***4***(1), pp. 17–22.

[5] Bendsøe, M. P., 1989. "Optimal shape design as a material distribution problem". *Structural optimization,* ***1***(4), pp. 193–202.

[6] Bendsøe, M. P., and Sigmund, O., 1999. "Material interpolation schemes in topology optimization". *Archive of applied mechanics,* ***69***(9-10), pp. 635–654.

[7] Sigmund, O., 2007. "Morphology-based black and white filters for topology optimization". *Structural and Multidisciplinary Optimization,* ***33***(4-5), pp. 401–424.

[8] Biyikli, E., and To, A. C., 2015. "Proportional topology optimization: a new non-sensitivity method for solving stress constrained and minimum compliance problems and its implementation in matlab". *PloS one,* ***10***(12), p. e0145041.

[9] Nie, Z., Jung, S., Kara, L. B., and Whitefoot, K., 2019. "Optimization of part consolidation for minimum production costs and time using additive manufacturing". *Journal of Mechanical Design*, pp. 1–16.

[10] Suresh, K., 2010. "A 199-line matlab code for pareto-optimal tracing in topology optimization". *Structural and Multidisciplinary Optimization,* ***42***(5), pp. 665–679.

[11] Xie, Y. M., and Steven, G. P., 1993. "A simple evolutionary procedure for structural optimization". *Computers & structures,* ***49***(5), pp. 885–896.

[12] Chu, D. N., Xie, Y., Hira, A., and Steven, G., 1997. "On various aspects of evolutionary structural optimization for problems with stiffness constraints". *Finite Elements in Analysis and Design,* ***24***(4), pp. 197–212.

[13] Querin, O., Steven, G., and Xie, Y., 1998. "Evolutionary structural optimisation (eso) using a bidirectional algorithm". *Engineering computations,* ***15***(8), pp. 1031–1048.

[14] Young, V., Querin, O. M., Steven, G., and Xie, Y., 1999. "3d and multiple load case bi-directional evolutionary structural optimization (beso)". *Structural optimization,* ***18***(2-3), pp. 183–192.

[15] Reynolds, D., McConnachie, J., Bettess, P., Christie, W., and Bull, J., 1999. "Reverse adaptivity—a new evolutionary tool for structural optimization". *International journal for numerical methods in engineering,* ***45***(5), pp. 529–552.

[16] Liu, J., Parks, G., and Clarkson, P., 2000. "Metamorphic development: a new topology optimization method for continuum structures". *Structural and Multidisciplinary Optimization,* ***20***(4), pp. 288–300.

[17] Eschenauer, H. A., Kobelev, V. V., and Schumacher, A., 1994. "Bubble method for topology and shape optimization of structures". *Structural optimization,* ***8***(1), pp. 42–51.

[18] Eschenauer, H., and Schumacher, A., 1997. "Topology and shape optimization procedures using hole positioning criteria". In *Topology optimization in structural mechanics*. Springer, pp. 135–196.

[19] Allaire, G., Jouve, F., and Toader, A.-M., 2002. "A level-set method for shape optimization". *Comptes Rendus Mathematique,* ***334***(12), pp. 1125–1130.

[20] Allaire, G., Jouve, F., and Toader, A.-M., 2004. "Structural optimization using sensitivity analysis and a level-set method". *Journal of computational physics,* ***194***(1), pp. 363–393.

[21] Allaire, G., De Gournay, F., Jouve, F., and Toader, A.-M., 2005. "Structural optimization using topological and shape sensitivity via a level set method". *Control and cybernetics,* ***34***(1), p. 59.

[22] Wang, M. Y., Wang, X., and Guo, D., 2003. "A level set method for structural topology optimization". *Computer methods in applied mechanics and engineering,* ***192***(1-2), pp. 227–246.

[23] Challis, V. J., and Guest, J. K., 2009. "Level set topology optimization of fluids in stokes flow". *International journal for numerical methods in engineering,* ***79***(10), pp. 1284–1308.

[24] Suresh, K., and Takalloozadeh, M., 2013. "Stress-constrained topology optimization: a topological level-set approach". *Structural and Multidisciplinary Optimization,* ***48***(2), pp. 295–309.

[25] Ulu, E., Mccann, J., and Kara, L. B., 2017. "Lightweight structure design under force location uncertainty". *ACM Transactions on Graphics (TOG),* ***36***(4), pp. 1–13.

[26] Kelly, D., and Elsley, M., 1995. "A procedure for determining load paths in elastic continua". *Engineering Computations,* ***12***(5), pp. 415–424.

[27] Kelly, D., Hsu, P., and Asudullah, M., 2001. "Load paths and load flow in finite element analysis". *Engineering*

Computations, ***18***(1/2), pp. 304–313.

[28] Kelly, D., Reidsema, C., Bassandeh, A., Pearce, G., and Lee, M., 2011. "On interpreting load paths and identifying a load bearing topology from finite element analysis". *Finite elements in Analysis and design,* ***47***(8), pp. 867–876.

[29] Sobieszczanski-Sobieski, J., 1999. "Multidisciplinary design optimisation (mdo) methods: their synergy with computer technology in the design process". *The Aeronautical Journal,* ***103***(1026), pp. 373–382.

[30] Allison, J. T., Kokkolaras, M., and Papalambros, P. Y., 2009. "Optimal partitioning and coordination decisions in decomposition-based design optimization". *Journal of Mechanical Design,* ***131***(8).

[31] Allison, J. T., and Herber, D. R., 2014. "Special section on multidisciplinary design optimization: multidisciplinary design optimization of dynamic engineering systems". *AIAA journal,* ***52***(4), pp. 691–710.

[32] Ulu, E., McCann, J., and Kara, L. B., 2019. "Structural design using laplacian shells". In Computer Graphics Forum, Vol. 38, Wiley Online Library, pp. 85–98.

[33] Sigmund, O., and Maute, K., 2013. "Topology optimization approaches". *Structural and Multidisciplinary Optimization,* ***48***(6), pp. 1031–1055.

[34] Guo, T., Lohan, D. J., Cang, R., Ren, M. Y., and Allison, J. T., 2018. "An indirect design representation for topology optimization using variational autoencoder and style transfer". In 2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, p. 0804.

[35] Yu, Y., Hur, T., Jung, J., and Jang, I. G., 2019. "Deep learning for determining a near-optimal topological design without any iteration". *Structural and Multidisciplinary Optimization,* ***59***(3), pp. 787–799.

[36] Rawat, S., and Shen, M. H., 2019. Application of adversarial networks for 3d structural topology optimization. Tech. rep., SAE Technical Paper.

[37] Sharpe, C., and Seepersad, C. C., 2005. "Topology design with conditional generative adversarial networks". In ASME 2019 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers Digital Collection.

[38] Ulu, E., Zhang, R., and Kara, L. B., 2016. "A data-driven investigation and estimation of optimal topologies under variable loading configurations". *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization,* ***4***(2), pp. 61–72.

[39] Raina, A., McComb, C., and Cagan, J., 2019. "Learning to design from humans: Imitating human designers through deep learning". *Journal of Mechanical Design,* ***141***(11).

[40] Zhang, W., Yang, Z., Jiang, H., Nigam, S., Yamakawa, S., Furuhata, T., Shimada, K., and Kara, L. B., 2019. "3d shape synthesis for conceptual design and optimization using variational autoencoders". In ASME 2019 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers Digital Collection.

[41] Banga, S., Gehani, H., Bhilare, S., Patel, S., and Kara, L., 2018. "3d topology optimization using convolutional neural networks". *arXiv preprint arXiv:1808.07440*.

[42] Cang, R., Yao, H., and Ren, Y., 2019. "One-shot generation of near-optimal topology through theory-driven machine learning". *Computer-Aided Design,* ***109***, pp. 12–21.

[43] Lei, X., Liu, C., Du, Z., Zhang, W., and Guo, X., 2019. "Machine learning-driven real-time topology optimization under moving morphable component-based framework". *Journal of Applied Mechanics,* ***86***(1), p. 011004.

[44] Kumar, T., and Suresh, K., 2019. "A density-and-strain-based k-clustering approach to microstructural topology optimization". *Structural and Multidisciplinary Optimization*, pp. 1–17.

[45] Rozvany, G. I., Zhou, M., and Birker, T., 1992. "Generalized shape optimization without homogenization". *Structural optimization,* ***4***(3-4), pp. 250–252.

[46] Eschenauer, H. A., and Olhoff, N., 2001. "Topology optimization of continuum structures: a review". *Applied Mechanics Reviews,* ***54***(4), pp. 331–390.

[47] Plocher, J., and Panesar, A., 2019. "Review on design and structural optimisation in additive manufacturing: Towards next-generation lightweight structures". *Materials & Design*, p. 108164.

[48] Ambrosio, L., and Buttazzo, G., 1993. "An optimal design problem with perimeter penalization". *Calculus of Variations and Partial Differential Equations,* ***1***(1), pp. 55–69.

[49] Petersson, J., 1999. "Some convergence results in perimeter-controlled topology optimization". *Computer Methods in Applied Mechanics and Engineering,* ***171***(1-2), pp. 123–140.

[50] Sharpe, C., Wiest, T., Wang, P., and Seepersad, C. C., 2019. "A comparative evaluation of supervised machine learning classification techniques for engineering design applications". *Journal of Mechanical Design,* ***141***(12).

[51] Oh, S., Jung, Y., Kim, S., Lee, I., and Kang, N., 2019. "Deep generative design: Integration of topology optimization and generative models". *Journal of Mechanical Design,* ***141***(11).

[52] Lin, Q., Hong, J., Liu, Z., Li, B., and Wang, J., 2018. "Investigation into the topology optimization for conductive heat transfer based on deep learning approach". *International Communications in Heat and Mass Transfer,* ***97***, pp. 103–109.

[53] Sosnovik, I., and Oseledets, I., 2019. "Neural networks for topology optimization". *Russian Journal of Numerical Analysis and Mathematical Modelling,* ***34***(4), pp. 215–223.

[54] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y., 2014. "Generative adversarial nets". In Advances in neural information processing systems, pp. 2672–2680.

[55] Mirza, M., and Osindero, S., 2014. "Conditional generative adversarial nets". *arXiv preprint arXiv:1411.1784*.

[56] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A., 2017. "Image-to-image translation with conditional adversarial networks". In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1125–1134.

[57] Long, J., Shelhamer, E., and Darrell, T., 2015. "Fully convolutional networks for semantic segmentation". In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3431–3440.

[58] Ronneberger, O., Fischer, P., and Brox, T., 2015. "U-net: Convolutional networks for biomedical image segmentation". In International Conference on Medical image computing and computer-assisted intervention, Springer, pp. 234–241.

[59] He, K., Zhang, X., Ren, S., and Sun, J., 2016. "Deep residual learning for image recognition". In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778.

[60] Hu, J., Shen, L., and Sun, G., 2018. "Squeeze-and-excitation networks". In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7132–7141.

[61] Nie, Z., Jiang, H., and Kara, L. B., 2020. "Stress field prediction in cantilevered structures using convolutional neural networks". *Journal of Computing and Information Science in Engineering,* ***20***(1).

[62] Hunter, W., et al., 2017. Topy-topology optimization with python. `https://github.com/williamhunter/topy`.

[63] Gómez, J., and Guarín-Zapata, N., 2018. Solidspy: 2d-finite element analysis with python. `https://github.com/AppliedMechanics-EAFIT/SolidsPy`.

[64] Kingma, D. P., and Ba, J., 2014. "Adam: A method for stochastic optimization". *arXiv preprint arXiv:1412.6980*.

[65] Duchi, J., Hazan, E., and Singer, Y., 2011. "Adaptive subgradient methods for online learning and stochastic optimization". *Journal of Machine Learning Research,* ***12***(Jul), pp. 2121–2159.

[66] Tieleman, T., and Hinton, G., 2012. "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude". *COURSERA: Neural networks for machine learning,* ***4***(2), pp. 26–31.