



HAL
open science

Deep Learning Architecture for Topological Optimized Mechanical Design Generation with Complex Shape Criterion

Waad Almasri, Dimitri Bettebghor, Fakhreddine Ababsa, Florence Danglade,
Faouzi Adjed

► **To cite this version:**

Waad Almasri, Dimitri Bettebghor, Fakhreddine Ababsa, Florence Danglade, Faouzi Adjed. Deep Learning Architecture for Topological Optimized Mechanical Design Generation with Complex Shape Criterion. The 34th International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems (IEA/AIE 2021), Jul 2021, Kuala Lumpur, Malaysia. pp.222-234, 10.1007/978-3-030-79457-6_19. hal-03368343

HAL Id: hal-03368343

<https://hal.science/hal-03368343v1>

Submitted on 6 Oct 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Deep Learning architecture for topological optimized mechanical design generation with complex shape criterion

Waad Almasri^{1,2[0000-0002-0648-7206]*}, Dimitri Bettebghor^{1[0000-0001-5793-6393]}, Fakhreddine Ababsa^{2[0000-0003-3862-2449]}, Florence Danglade^{2[0000-0002-8281-1004]}, and Faouzi Adjed^{1[0000-0002-0100-9352]}

¹ Expleo France, Montigny-Le-Bretonneux, France
`firstName.lastName@expleogroup.com`

² Laboratoire d'Ingénierie des Systèmes Physiques et Numériques (LISPEN), Arts et Métiers, Cluny, France
`firstName.lastName@ensam.eu`

Abstract. Topology optimization is a powerful tool for producing an optimal free-form design from input mechanical constraints. However, in its traditional-density-based approach, it does not feature a proper definition for the external boundary. Therefore, the integration of shape-related constraints remains hard. It requires the experts' intervention to interpret the generated designs into parametric shapes; thus, making the design process time-consuming. With the growing role of additive manufacturing in the industry, developing a design approach considering mechanical and geometrical constraints simultaneously becomes an interesting way to integrate manufacturing and aesthetics constraints into mechanical design. In this paper, we propose to generate mechanically and geometrically valid designs using a deep-learning solution trained via a dual-discriminator Generative Adversarial Network (GAN) framework. This Deep-learning-geometrical-driven solution generates designs very similar to traditional topology optimization's outputs in a fraction of time.

Moreover, it allows an easy shape fine-tuning by a simple increase/decrease of the input geometrical condition (here the total-bar-count), a task that a traditional topology optimization cannot achieve.

Keywords: Topology Optimization (TO) · Deep Learning (DL) · Generative Adversarial Networks (GAN).

1 Introduction

In the late 20th century, the advent of additive manufacturing (AM) allowed the production of organic shapes that were costly and complex with conventional

* Corresponding author: Waad Almasri, waad.almasri@ensam.eu

shaping processes. On the other hand, given a set of parameters such as loads, boundary conditions, and volume fraction (i.e. the percentage of material volume used), topology optimization (TO) allows the generation of smooth and organic shapes. Its synergy with AM made it further attractive in the research areas. Despite the success of AM, not all the designs could be manufactured. Steep curvatures, overhanging patterns, the need for supports, and other geometrical constraints are still a hurdle[2]. Therefore, an engineer manually reconstructs a shape inspired by TO’s suggestion, considering the geometric and manufacturing constraints implicitly. This re-interpretation phase is not straightforward, can compromise the initial design’s optimality, and can be time-consuming; it depends on the engineer’s experience and expertise. Moreover, a recent survey[26] has shown that half of TO practitioners regret the absence of geometric and manufacturing-related plug-ins in TO’s software.

Thus, to accelerate the design process, [12,15,31,13,29] integrated overhangs and building directions into the formulation of TO to minimize the need for supports during manufacturing. Nevertheless, TO is an iterative, finite-element-based optimization method, hence computationally expensive. Its efficiency depends on the design space (mesh size), the complexity of the input conditions, and the resolution of discretized linear elasticity equations (e.g Finite Elements). Consequently, other research has focused on accelerating the TO process via machine and Deep Learning (DL) techniques.

DL architectures have proven efficiency and robustness in learning complex spatial correlations and extracting high-level features (including geometrical or shape-related features) from real-world images [8,20,33].

The DL-TO methods found in the literature can be divided into two parts. The first part used DL to assist traditional finite-element-TO [25,11,17] while the second part tried to replace completely TO’s formulation by DL [28,30,22,19,1,14,6,10]. None of these DL-based methods included any geometrical constraints; on the contrary, they were left to the re-interpretation phase.

In this work, the primary objective is not to accelerate TO via DL but to take advantage of DL’s capability to learn spatial correlations to facilitate the integration of geometrical constraints at the conceptual level of TO. The geometrical constraint considered in this work is the total-bar-count referred to as the design’s complexity. This DL-geometrical-driven TO paves the way to handle and tailor different complexities during design generation.

Our approach consists of a dual-discriminator Generative Adversarial Network (GAN)[7]. The generator (the DL-TO) encodes the mechanical and geometrical conditions and outputs the 2D design. The first discriminator penalizes the generator over the mechanical constraints, while the second penalizes it over the geometrical one. The GAN is chosen for its flexible training framework. New discriminators can be easily appended to pass new knowledge to the generator during training. For example, one can add a build-time predictor, a thermal distortion predictor, etc. as discriminators to generate designs accounting for a short build time, thermal distortion, etc. Besides, these discriminators can always be used separately (to predict the build-time of a design or its thermal

distortion). To the best of our knowledge, related literature on automatic design generation only focuses on the generated designs’ aesthetics. In our work, we developed an objective evaluation, which considers not only mechanical criteria (compliance, volume) but also an objective measure of complexity.

The major contributions of this paper can be summarized as follows: (1) The integration of mechanical and geometrical constraints simultaneously at the conceptual level via a DL-based TO. (2) The ability to easily tailor the geometry of a design by a simple change of the input geometrical condition (the total-bar-count in this case).

The rest of the paper is organized as follows: sections 2.1 and 2.2 provide a theoretical overview of TO and GANs respectively. In section 3, the dual-discriminator GAN approach is explained. Section 4 details the consolidation of the training and test datasets used to train and evaluate the DL-TO. Generated designs are shown and evaluated in Section 5. Finally, section 6 summarizes the methodology and its outcomes and discusses future works.

2 Theoretical Overview

2.1 Topology Optimization

Topology optimization seeks to find the optimal layout within a design space for a specific set of boundary conditions, load configurations, and volume fraction. It gained its success in the industrial world for its intrinsic characteristics: it allows effective use of the material and has a higher degree of freedom when addressing the topology, shape, and sizing problems altogether. In the literature, several approaches were developed to solve the TO problem: density-based[4], level-set[3] and others. The topmost common commercial approach is the Solid Isotropic Material with Penalization (SIMP) method[4]. SIMP is a density gradient-based iterative method that uses penalization of the intermediate non-binary values of density material to converge to an optimal binary design. SIMP represents a design as a distribution of discretized square material elements e (material properties are assumed constant within each element e). The variables are the element-relative-densities x_i such that $x_i = 1/0$ represents presence/absence of material at point i of the design domain.

A TO problem where the objective is to minimize the compliance $c(x)$ can be written as the following:

$$\min_x = U^T K U = \sum_{e=1}^N x_e^p u_e^T k_0 u_e \quad s.t. \quad K U = F, \quad \frac{V(x)}{V_0} \leq f, \quad 0 < x_0 \leq x \leq 1 \quad (1)$$

where U and u_e are the global and element-wise displacements, F the forces vector, K and k_e are the global and element-wise stiffness matrices and $N =$ number of elements used to discretize the design domain. x is the design variables vector i.e. the density material and x_0 the minimum relative densities (non-zero to avoid singularity), p penalization power (typically 3 for Poisson’s

ratio = 1/3 [5]). V_0 and $V(x)$ are the design domain volume and material volume respectively and f the volume fraction. To efficiently solve the problem stated above, several approaches were proposed: the Optimality Criteria (OC) methods, Sequential Linear Programming (SLP) methods, the Method of Moving Asymptotes (MMA), etc. In [23], Sigmund used the OC method and added a mesh-independency filter to ensure the existence of solutions to the problem and avoid checker-board patterns[24]. In this study, a modified Python version of the 99-line-of-code of the SIMP method written by Sigmund[23] is used to generate the training and test datasets of 2D designs (section 4).

2.2 Generative Adversarial Networks

Generative adversarial network (GAN) was first introduced by Goodfellow[7]. This method learns to mimic any input data distribution. A GAN consists of two neural networks the generator $G(z, \theta_g)$ and the discriminator $D(x, \theta_d)$, where θ_g and θ_d are the parameters of the generator’s and discriminator’s networks respectively. $G(z, \theta_g)$ would like to generate from a latent space z (z follows a noise prior distribution p_z) samples with a distribution p_g similar to the original ones p_{data} . However, $D(x, \theta_d)$ tries to discriminate real sample (p_{data}) from synthesized ones (p_g). Both networks are trained in a minimax framework to improve the same loss function: the cross entropy loss $L(G, D)$. The optimization is successful when the generator starts to output data samples following the same distributions as the real sample (i.e. $p_g = p_{data}$). On the other hand, a conditional GAN (cGAN) [16] is an extension of the GAN network enabling the generation to be oriented by a specific input condition c . In this framework, the basics of cGAN become: the conditional generator as $G((z/c), \theta_g)$, the conditional discriminator as $D((x/c), \theta_d)$ and the loss function as:

$$L(G, D) = \min_G \max_D \mathbb{E}_{x \sim p_{data}(x)} [\log(D(x/c))] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z/c)))] \quad (2)$$

The approach adopted in this work is based on the cGAN framework.

3 Methodology

In this work, a conditional [16] convolutional[18] dual-discriminator GAN has been adopted. This approach consists of a deep ResUnet generator[32] (the DL-based TO), a Residual-based discriminator to differentiate between the real designs (SIMP-based) and the generated ones, and an inception-based[27] bar counter (the 2nd discriminator) to quantify the complexity of the generated/real designs. The training procedure is detailed in Fig. 1.

3.1 Architecture of the Generator

The generator is a deep ResUnet[32] network. It takes as input the mechanical and geometrical conditions and outputs the 2D design. It’s an encoder-decoder

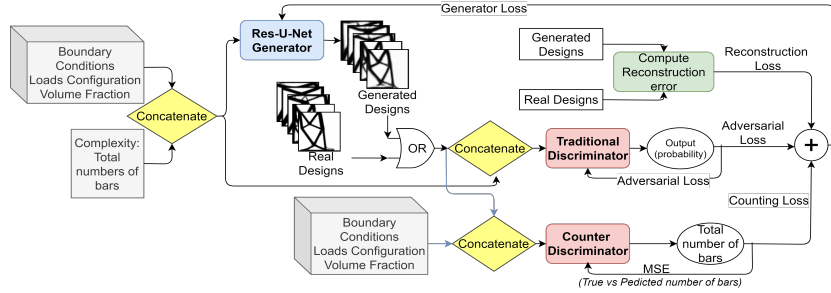


Fig. 1: Training Procedure.

convolutional architecture with residual and skip-connections between the outputs of encoder layers and the inputs of decoder layers or what is called U-Net. This architecture benefits from the U-Net[21] and residual[9] advantages. U-Net connections ensures that high-frequency details are not lost in the decoding phase, and residual connections allow a deeper network without any performance degradation (usually due to vanishing gradients). The network can be divided into three parts: an encoder, a bridge, and a decoder. The encoder is formed of 4 blocks, each consisting of a down-sampling layer (a convolution of stride 2) and a residual unit³. The bridge connection has the same architecture as an encoder’s block and combines the encoder to the decoder. The decoder is formed of 5 blocks, each consisting of an up-sample layer (a transpose convolution of stride 2) and a residual unit, followed by a convolution of kernel size 1×1 and a sigmoid activation.

3.2 Architecture of the Discriminators

The first one, the traditional discriminator, takes as input the design along with the geometrical and mechanical conditions and outputs the probability that the design comes from the real data distribution. The second one is a regression inception-based DL counter. It takes as input the design and only its corresponding mechanical conditions and outputs the total number of bars present in the design.

The traditional discriminator’s network consists of seven blocks of down-sample and residual units followed by a dropout, then a fully connected layer. It outputs a probability p regarding the design being real ($p \approx 1$) or fake ($p \approx 0$). It helps the generator improve the generated designs’ quality and conformity to boundary conditions and load configurations.

The counter network consists of a stem, an Inception/Reduction Resnet-v1-block-A, an Inception/Reduction Resnet-v1-block-B, an Inception Resnet-v1-block-C followed by an average pooling layer, a dropout layer, and a fully

³ The architecture of the residual unit block used in this work is detailed in [32].

connected layer⁴. In this work, 4347 SIMP-designs were manually labeled (*we manually counted the total number of bars present in each design*). The counter-discriminator is pre-trained on these labeled SIMP designs before the full training of the GAN, for this procedure improves the generator’s convergence. The counter-discriminator is pre-trained using 3885 labeled designs and tested over the remaining 462 designs. The counter-discriminator predicts 94.9% of the time a total-bar-count within an error margin of ± 2 bars. Additionally, even if we restrict further the error margin to ± 1 bar, its accuracy slightly drops to 85.4%; knowing that the range of total-bar-count is very wide ([3, 31]). This pre-trained counter is also used to predict the total-bar-count on unlabeled train designs to augment the training dataset.

3.3 Loss Function

This dual-discriminator GAN aims to train a generator embracing two aspects: the reconstructed 2D designs’ quality and their conformity to the mechanical and geometrical conditions. Thus, the original adversarial loss function used to train the generator (Eq. 2) was altered to consider both aspects. A reconstruction loss (L_r) and a counting loss (L_{count}) were added to the generator’s loss. The modified generator loss function L_G adapted in the training process is the following:

$$L_G = \lambda_1 L_r + \lambda_2 L_{adv} + \lambda_3 Acc_{count} L_{count} \quad (3)$$

Where $L_r = \frac{1}{n} \sum_{i=1}^n (x_i - \hat{x}_i)^2$, $L_{count} = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2$, with x_i , \hat{x}_i the true and predicted 2D design, y_i the input total bar-count, \hat{y}_i the predicted total-bar-count in the generated designs and n the batch size. The accuracy of the counter discriminator $Acc_{count} = \frac{1}{N} \sum_{i=1}^N (\hat{t}_i == y_i)$ with y_i , \hat{t}_i the true and predicted total-bar-count in the real designs and N the total number of training samples; this accuracy is updated at the end of each epoch. λ_1 , λ_2 and λ_3 are the penalization weights of L_r , L_{adv} and L_{count} respectively. The adversarial loss ensures the generation of creative and varied 2D structures. The reconstruction loss boosts the aesthetics and the reproduction of high-frequency details in the generated samples. The counting loss penalizes the generator every time the geometrical constraint on the total number of bars is not respected.

In this work, stabilizing the loss function was a challenge, especially that it consists of different types of losses having different orders of magnitude: $0 \leq L_r \leq 1$, $0 \leq L_{adv} \leq 100$; due to Pytorch Implementation of the Binary Cross Entropy Loss, $0 \leq Acc_{count} \leq 1$ and $0 \leq L_{count} \leq 961$; in this work, the maximum total-bar-count is 31. Additionally, during training, L_r tends to decrease sharply after only few iterations ($0 \leq L_r \leq 0.1$). The same behavior is noticed with L_{adv} ($0 \leq L_{adv} \leq 1$). While $L_{count} \times Acc_{count}$ seems to vary between 0 and 30, especially that, as mentioned earlier, the counter discriminator is pre-trained before the

⁴ The stem and inception/reduction blocks used defers from the original paper [27] only by the number of input/output feature maps.

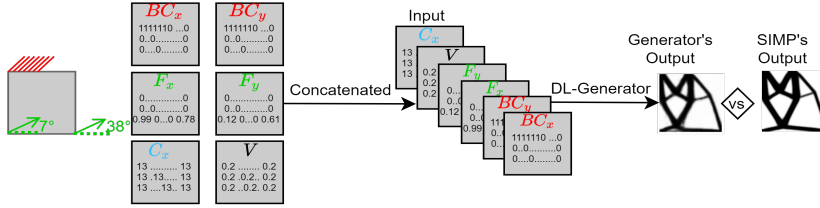


Fig. 2: Input of the DL-generator. Boundary conditions (BC_x , BC_y), load configurations (F_x , F_y), volume fraction V and complexity C_x (i.e. total-bar-count) are formulated as a six-channel image forming the generator’s input.

training ($Acc_{count} \approx 0.8$). Thus, to ensure that the generator is equally penalized over the three losses, λ_1 , λ_2 and λ_3 were set to 10, 1 and 0.1 respectively.

















4 Data Generation

To train the model, 21 538 2D-designs of size 100×100 pixels were generated following the SIMP method explained in section 2.1 via a modified Python version of the academic open-source TO code written by Sigmund[23]⁵. The geometrical constraint (the complexity) is added by manual labeling over 4347 samples (which are used to pre-train the counter discriminator) and then the complexity of the remaining samples is predicted using the counter discriminator (section 3.2).

















A 2D structure of size $n_x \times n_y$ can be discretized into a mesh of $(n_x + 1) \times (n_y + 1)$ nodes and is subject to 2 major constraints: the boundary conditions i.e. the fixed nodes and the loads i.e. the loaded nodes. Boundary conditions BC are represented as $(n_x + 1) \times (n_y + 1)$ matrices with null values everywhere except for the fixed nodes set to 1.0; we only consider encastrated designs i.e. BC along the x axis (BC_x) and y axis (BC_y) are similar. Loads F_x and F_y are represented as $(n_x + 1) \times (n_y + 1)$ matrices with null values everywhere except for the loaded nodes⁶. Since BC and F are 101×101 matrices, hence, to concatenate the latter with the rest of the inputs altogether: the 2D design, the volume fraction, and the complexity are reshaped into a 101×101 image. An example of an input to the generator is shown in Fig. 2. The dataset was separated into train (17230 samples) and test (4308 samples). The test set is used to evaluate the generator’s performance.

⁵ This code is available on the GitHub repository: https://github.com/dbetteb/TOP_OPT.git.

⁶ A loaded node n_e located at line i and column j tilted θ degrees has $F_x(i, j) = \cos(\theta)$ and $F_y(i, j) = \sin(\theta)$; the magnitude of the loads were set to 1.0 N .

Design	Original Design								
	Generated Design								
Metrics	$e_{V\%}$	-3.9	-2.07	-3.64	-7.94	-1.58	2.37	-3.73	-1.11
	$e_{C\%}$	20.12	12	2.41	39.27	9.39	6.94	44.24	37.7
	ΔC_x	0	0	0	+2	+1	-1	0	+1
	nT	1833	1017	14002	974	1903	3703	2838	3923

(a) Without Threshold

Design with Threshold	Original Design (Threshold=0.4)								
	Generated Design (Threshold=0.4)								
Metrics With Threshold	$e_{V\%}$	-1.36	-0.16	-3.72	-7.16	-1.34	3.45	0.17	-3
	$e_{C\%}$	3.75	-2	2.83	31.75	-2.53	0.48	4.31	21.1
	ΔC_x	0	0	0	+1	0	0	0	-1
	nT	1833	1017	14002	974	1903	3703	2838	3923

(b) With Threshold

Fig. 3: In this figure, we compare the aesthetics, volume fraction ($e_{V\%}$), compliance ($e_{C\%}$), complexity (ΔC_x) and generation speed (nT) of the original (SIMP-based) versus generated (DL-based) designs with and without Threshold. In both cases, DL-designs are barely indistinguishable, in terms of shape, from SIMP-designs, achieve lower Volume Fractions ($e_{V\%} \leq 0$), respect the Complexity constraint ($|\Delta C_x| \leq 2$) and are generated thousand of times of faster than the SIMP-designs i.e. DL-TO generates the first design in Fig.3a 1833 times faster than SIMP-TO. However, while the Compliance of DL-designs is higher than SIMP-designs before threshold, it is significantly reduced after threshold.

5 Experiments and Results

As mentioned above, TO finds the optimal material distribution in a design space that minimizes compliance and meets the boundary conditions, the loads, and the volume fraction constraint. Still, an optimal design for manufacturing is a compromise between mechanical performance and geometrical constraints. Thus, to evaluate the generated designs, we will examine their compliance values, volume fractions, and complexities (geometrical constraint). The metrics used for the volume fraction V and compliance C are the relative errors $e_{V\%} = \frac{V_g - V_o}{V_o} \times 100$ and $e_{C\%} = \frac{C_g - C_o}{C_o} \times 100$ respectively. The metric of Complexity C_x is the barcount difference $\Delta C_x = C_{x_g} - C_{x_o}$. Where X_g, X_o refer to generated and original respectively and $\{X_g, X_o : X \in \{V, C, C_x\}\}$. The metric used to compare the generation speed between SIMP and DL-TO is $nT = \frac{\text{Generation Time of SIMP-TO}}{\text{Generation Time of DL-TO}}$, which refers to DL-TO is faster nT times than SIMP-TO.

Figure 3a displays a sample of original (i.e. SIMP-based) versus generated (i.e.

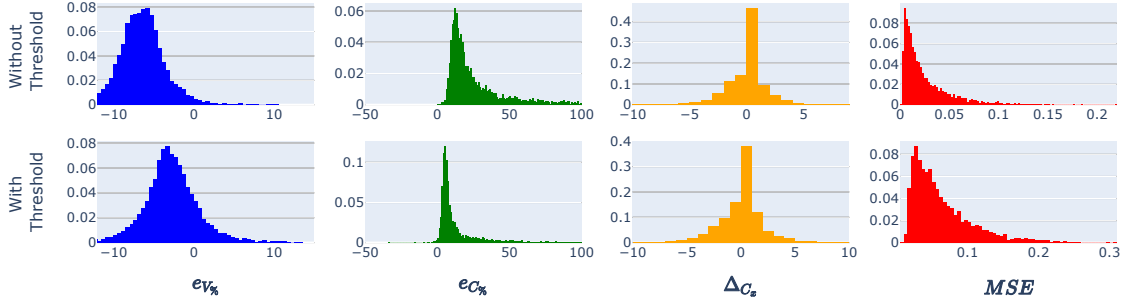


Fig. 4: Distributions of Volume Fraction ($e_{V\%}$), Compliance ($e_{C\%}$), Complexity (Total bar count, Δ_{C_x}) and Reconstruction error (MSE) between the original (SIMP-based) and generated (DL-based) designs with and without threshold in the Test Set. The generated designs show conformity with all constraints except for the compliance, which is improved with threshold.

DL-based) designs from the test set. It also shows their relative errors $e_{V\%}$, $e_{C\%}$ and bar-count differences Δ_{C_x} .

The DL-designs are aesthetically plausible. The fixed and loaded bars⁷ respect the input boundary conditions and load configurations.

In the majority of cases, $e_{V\%}$ is negative, e.g. the DL-TO (the generator) tends to find a lower volume fraction bound than that found by the traditional SIMP formulation. However, $e_{C\%}$ is relatively higher ($e_{C\%} \geq 10\%$ in most cases); showing that the DL-designs exhibit greater external stresses. It would be interesting to note here that SIMP-TO and DL-TO output continuous non-binary designs and the compliance is very sensitive to intermediate-density-pixel values. Hence, to account for this drawback, a threshold of 0.4 is applied to the test designs. This particular threshold improved the compliance of test designs globally compared to others. However, a better approach would be to choose an adapted global threshold per design. Results are reported in Fig. 3b. In most cases, the $e_{C\%}$ dropped significantly, $e_{C\%} \leq 5\%$, i.e. after threshold, DL and SIMP designs tend to have similar compliance values ($C_g \leq 1.05 \times C_o$). We point out that the application of threshold increased the volume fraction. Yet, the DL-designs still achieved lower volume fractions than the SIMP-designs. In other words, after threshold, in the majority of cases, DL-designs tend to present a better mechanical performance: lower volume fraction and similar compliance.

Additionally, the geometrical constraint is mainly respected. Complex DL-designs tend to display additional or fewer internal bars; the maximum bar-difference is

⁷ A fixed bar is a bar where boundary conditions are applied. A loaded bar is a bar where a load is applied.

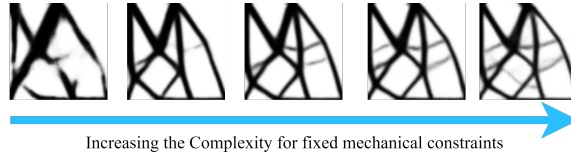


Fig. 5: A sample of generated designs with similar mechanical constraints but different complexities. From left to right, the complexities are the following: 10, 18, 22, 25 and 30 bars.

more or less two bars (Fig. 3a, Fig. 3b).

Figure 4 summarizes the overall performance of the generator. The average reconstruction error (MSE) is 0.025, manifesting the aesthetic aspect of the DL-designs. The volume fraction constraint is respected with 94% of the DL-designs having a volume fraction lower than that achieved by SIMP. Moreover, 86% of the DL-designs have, at most, 2 additional/fewer bars ($|\Delta C_x| \leq 2$). Nevertheless, the DL-designs tend to show higher external stresses. 50% of the DL-designs score a compliance greater than that computed over the SIMP designs ($e_{C\%} \geq 20\%$). One of the reasons is that the generator was not penalized explicitly on the compliance during the training. The integration of a compliance predictor as a third discriminator into our GAN could improve the generated designs' compliance. We note that the DL-designs comply better with complexity and volume fraction constraints; the generator was penalized on the reconstruction error (it embeds the volume fraction constraint implicitly) and complexity error during training. We also compare the designs after the application of threshold. As expected, the compliance dropped; ($e_{C\%} \leq 20\%$) in 70% of the cases.

We underline that the design's mechanical and geometrical performance depends on the threshold-value's choice: the lower the threshold, the higher the volume fraction, the higher the threshold, the lower the bar-count, and possibly the appearance of discontinuities in the design. Consequently, we need to prioritize the constraints to find the best compromise.

To validate the generator's understanding of complexity (total-bar-count), we fixed the mechanical conditions of designs, changed the complexity and reported the generator's response to such change as shown in Figure 5. The number of bars increases with the complexity. However, the additional bars are blurry. This is due to the volume fraction constraint. The volume fraction and complexity of a design are proportional. A better approach would be to increase/decrease the complexity and volume fraction together.

Finally, in terms of computational time, it would be interesting to highlight that the DL-TO generates a 2D design in 0.047s from specific mechanical conditions and an additional geometrical one (the total-bar-count in our case), while a SIMP-TO requires 221s on average (i.e. ≈ 5000 times slower) for only the same mechanical conditions and needs supplementary post-processing to integrate the geometrical one. Moreover, a DL-TO's generation time and computational complexity are independent of the input constraints, unlike traditional

Table 1: Generation Time (in seconds s) and Computational Complexity (in Gega Floating Point Operations per Second $GFLOPS$) of SIMP (FE-based) versus DL-TO.

Input Constraints	Generation Time (s)		Computational Complexity ($GFLOPS$)	
	SIMP	DL-TO	SIMP	DL-TO
1 Load	68	0.02	62.81	2.27
2 Loads	132	0.02	125.89	2.27
10 Loads	656	0.02	620.28	2.27

TO approaches as shown in Tab.1. For SIMP-TO, the computational time and complexity increase with the complexity of the input constraint (here, the number of loads) while they remain unchanged for DL-TO.

To sum up, the DL-TO proposed in this study generates mechanically valid designs, indistinguishable from those generated by SIMP-TO to the naked eye, and is a thousand times faster than SIMP-TO. Moreover, while SIMP-TO needs post-processing to account for a geometrical condition, DL-TO integrates it at the conceptual level. Finally, DL-TO enables TO users to easily tailor the design’s complexity as the first step towards making it manufacturable.

6 Conclusion

In this paper, we build an original approach to generate topologically optimized designs with the help of advanced DL architectures. This DL-TO (generator) not only generates mechanical designs faster but also tailors the design’s complexity (total-bar-count), as the first step towards making designs manufacturable. We demonstrate the DL-TO’s ability to adjust the complexity, a task that is hardly feasible with conventional TO methods. Lastly, we leverage the generative capability of GANs; the DL-TO generates creative valid designs.

Moreover, the adopted training strategy enables the addition of future developed modules (as discriminators) to the generation approach such as: (a) a build-time module to identify structures rapidly manufactured, (b) a thermal distortion module, (c) and other complex modules to validate a design mechanically and geometrically. In future works, we will integrate more complex additive manufacturing constraints particularly, a build-time module, a geometry-evaluator (to check for overhangs, bar curvatures, minimum length/width, etc.), and supplementary mechanical validation modules into the generation process.

Acknowledgement. This work was supported by Expleo France.

References

1. Abueidda, D.W., Koric, S., Sobh, N.A.: Topology optimization of 2d structures with nonlinearities using deep learning. *Computers & Structures* **237**, 106283 (2020)

2. Adam, G.A., Zimmer, D.: Design for additive manufacturing—element transitions and aggregated structures. *CIRP Journal of Manufacturing Science and Technology* **7**(1), 20–28 (2014)
3. Allaire, G., Jouve, F., Toader, A.M.: A level-set method for shape optimization. *Comptes Rendus Mathématique* **334**(12), 1125–1130 (2002)
4. Bendsoe, M.P.: Optimal shape design as a material distribution problem. *Structural optimization* **1**(4), 193–202 (1989)
5. Bendsoe, M.P., Sigmund, O.: Material interpolation schemes in topology optimization. *Archive of applied mechanics* **69**(9-10), 635–654 (1999)
6. Bi, S., Zhang, J., Zhang, G.: Scalable deep-learning-accelerated topology optimization for additively manufactured materials. arXiv preprint arXiv:2011.14177 (2020)
7. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: *Advances in neural information processing systems*. pp. 2672–2680 (2014)
8. Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S., Lew, M.S.: Deep learning for visual understanding: A review. *Neurocomputing* **187**, 27–48 (2016)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 770–778 (2016)
10. Hoyer, S., Sohl-Dickstein, J., Greydanus, S.: Neural reparameterization improves structural optimization. arXiv preprint arXiv:1909.04240 (2019)
11. Kallioras, N.A., Kazakis, G., Lagaros, N.D.: Accelerated topology optimization by means of deep learning. *Structural and multidisciplinary optimization*,(under review) (2020)
12. Leary, M., Merli, L., Torti, F., Mazur, M., Brandt, M.: Optimal topology for additive manufacture: A method for enabling additive manufacture of support-free optimal structures. *Materials & Design* **63**, 678–690 (2014)
13. Li, S., Yuan, S., Zhu, J., Wang, C., Li, J., Zhang, W.: Additive manufacturing-driven design optimization: Building direction and structural topology. *Additive Manufacturing* p. 101406 (2020)
14. Malviya, M.: A systematic study of deep generative models for rapid topology optimization (2020)
15. Mass, Y., Amir, O.: Topology optimization for additive manufacturing: Accounting for overhang limitations using a virtual skeleton. *Additive Manufacturing* **18**, 58–73 (2017)
16. Mirza, M., Osindero, S.: Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014)
17. Nie, Z., Lin, T., Jiang, H., Kara, L.B.: Topologygan: Topology optimization using generative adversarial networks based on physical fields over the initial domain. arXiv preprint arXiv:2003.04685 (2020)
18. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)
19. Rawat, S., Shen, M.H.H.: A novel topology optimization approach using conditional deep learning. arXiv preprint arXiv:1901.04859 (2019)
20. Rawat, W., Wang, Z.: Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation* **29**(9), 2352–2449 (2017)
21. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: *International Conference on Medical image computing and computer-assisted intervention*. pp. 234–241. Springer (2015)

22. Sharpe, C., Seepersad, C.C.: Topology design with conditional generative adversarial networks. In: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. vol. 59186, p. V02AT03A062. American Society of Mechanical Engineers (2019)
23. Sigmund, O.: A 99 line topology optimization code written in matlab. *Structural and multidisciplinary optimization* **21**(2), 120–127 (2001)
24. Sigmund, O., Petersson, J.: Numerical instabilities in topology optimization: a survey on procedures dealing with checkerboards, mesh-dependencies and local minima. *Structural optimization* **16**(1), 68–75 (1998)
25. Sosnovik, I., Oseledets, I.: Neural networks for topology optimization. *Russian Journal of Numerical Analysis and Mathematical Modelling* **34**(4), 215–223 (2019)
26. Subedi, S.C., Verma, C.S., Suresh, K.: A review of methods for the geometric post-processing of topology optimized models. *Journal of Computing and Information Science in Engineering* **20**(6) (2020)
27. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: Thirty-first AAAI conference on artificial intelligence (2017)
28. Ulu, E., Zhang, R., Kara, L.B.: A data-driven investigation and estimation of optimal topologies under variable loading configurations. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization* **4**(2), 61–72 (2016)
29. Wang, C., Qian, X.: Simultaneous optimization of build orientation and topology for additive manufacturing. *Additive Manufacturing* p. 101246 (2020)
30. Yu, Y., Hur, T., Jung, J., Jang, I.G.: Deep learning for determining a near-optimal topological design without any iteration. *Structural and Multidisciplinary Optimization* **59**(3), 787–799 (2019)
31. Zhang, W., Zhou, L.: Topology optimization of self-supporting structures with polygon features for additive manufacturing. *Computer Methods in Applied Mechanics and Engineering* **334**, 56–78 (2018)
32. Zhang, Z., Liu, Q., Wang, Y.: Road extraction by deep residual u-net. *IEEE Geoscience and Remote Sensing Letters* **15**(5), 749–753 (2018)
33. Zhao, Z.Q., Zheng, P., Xu, S.t., Wu, X.: Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems* **30**(11), 3212–3232 (2019)