16th World Congress in Computational Mechanics (WCCM)
21-26 July 2024, Vancouver, Canada
A. Korobenko, M. Laforest, S. Prudhomme, R. Vaziri (Eds)

STRUCTURAL OPTIMIZATION THROUGH GENERATIVE ADVERSARIAL NETWORKS

Lucas M. S. Pereira[1], **Larissa Driemeier**[2]

[1] Department of Mechatronic Engineering and Mechanical Systems, Polytechnic School of São Paulo University
Av. Prof Mello Moraes, 2231, São Paulo, 05508-900, São Paulo, Brazil
lucas.lmspereira@usp.br

[2] Department of Mechatronic Engineering and Mechanical Systems, Polytechnic School of São Paulo University
Av. Prof Mello Moraes, 2231, São Paulo, 05508-900, São Paulo, Brazil
driemeie@usp.br

Key words: Finite Element Method, Machine Learning, Generative Adversarial Network, Topology Optimization.

Abstract. The finite element method (FEM) is a well known approach to solve partial differential equations. It has important applications in structural engineering, such as in topology optimization (TO). TO involves, at each iteration, the solution of structural problems via FEM, which can add up to a high computational cost. Therefore, a line of research to accelerate TO emerged over the years focusing on machine learning (ML) approaches. Particularly, Artificial Neural Networks (ANNs) have been proposed to significantly speed-up the process by eliminating the iterative algorithm, which is intrinsic to TO. Since ANN is a supervised ML method, first a dataset is generated, containing finite element analysis (FEA) inputs, volume fraction, post-processing, and final topologies. Then, with the Wasserstein Generative Adversarial Networks (WGANs) is trained on this dataset to map fields of physical quantities, such as the von Mises stress, to the final optimized structure. The final designs obtained via ML are quantitatively analyzed according to the metrics.

1 INTRODUCTION

In industry, topology optimization has been applied for a long time to design structures that maximize performance. Its goal is to find an optimal structure according to certain objective(s) and constraint(s). Common examples of objectives are weight minimization and structural natural frequency maximization. And common constraints are maximum volume and stress. The main steps in topology optimization start at an initial guess of the material distribution (usually uniform on the domain). Then, FEA and optimization steps are repeated until the process converges to an optimal design. In general, a high computational cost comes from the repetition of

simulations.

Another very useful class of numerical methods is machine learning, which has been applied to a variety of contexts in structure engineering, from constructing design spaces to optimizing metamaterials. [6] focused on replacing the entire topology optimization procedure by an ML model. They state that the sparsity of the boundary condition arrays hinders the training of the model. Alternatively, they included the following fields of physical quantities in the inputs to the generator network of the proposed Generative Adversarial Network (GAN): von Mises stress, and strain energy density. [7] developed two CNNs to predict the von Mises stress field from FEA inputs. Many other works focusing on the interaction between ML and TO can be found in a recent review study [11]. Table 1 places the current work in comparison to other recent studies. "Supports" summarizes the variation of supports present in the dataset; so does "Load" for loading conditions. "Train/Test" indicates the difference between the samples in train and test splits. "Training samples" is the size of the training split. "2D/3D" indicates how many spatial dimensions the problems had in the study.

Work	**Supports**	**Mesh**	**Load**	**Train/Test**	**Training samples**	**2D/3D**
Present	4 clamps + 3 random pins	140x50	2 loads. Random components and positions	1 of the clamps separated for testing	44,661	2D
[6]	42 cases	128x64	On boundary. Discrete direction from 0 to π	4 BCs separated for testing	$\sim$35,524	2D
[10]	Poisson distribution	40x40	Poisson distribution	Equal	$<$ 10,000	2D
[9]	3 cases	Up to 65x45x6	Random point in 4 regions	Larger x and y dimensions for testing	6300	3D

Table 1: Comparison between recent studies applying ML to TO for direct design. Source: authors.

In the current work, the authors take a step towards ML models that alleviate the computational cost of topology optimization. In the end, the model developed is capable of quickly suggesting an optimal design for a fixed-mesh 2D linear structural problem. The authors begin by generating a dataset with the required information: FEM inputs (position and components of loads, mechanical supports), physical fields (von Mises stress, and strain energy density), volume fraction, and final topology. This data is then used to train generative adversarial networks.

Starting from a reimplementation of the architecture proposed in [6], a more recent alternative from [4] is also considered, which is trained with the Wasserstein loss. The topologies generated by the models are visually promising, and their numerical and mechanical qualities is quantitatively assessed. To detail the background, methods, experiments and results the text is divided as follows. Sections 2 and 3 briefly present the fundamental theoretical background in topology optimization and machine learning related to the present work, respectively. In section 4, the proposed approach is detailed. This comprises of subsection 4.1 to detail dataset generation (4.1.1) and analysis (4.1.2); subsection 4.2 focusing on the ML models used; and subsection 4.3 detailing the training procedures. In section 5, the experiments and results are discussed. This also entails studying the interpretability of the trained ML models in subsection 5.2. Finally, in section 6, final remarks are made, and possible extensions of the current work are suggested.

2 TOPOLOGY OPTIMIZATION

In engineering, often one seeks to alter the shape of a body in order to minimize/maximize an objective function, such as structural compliance. A very effective approach for TO is Solid Isotropic Material with Penalty (SIMP) [1]. In this method, a pseudo-density $\rho_e \in [0, 1]$ is associated to each element. A value close to zero indicates that it should be removed from the structure. Furthermore, this value is raised to the power of $p \geqslant 3$, the penalty coefficient. This penalization pushes values around 0.5 towards 0, alleviating the problem of intermediary pseudo-densities, whose interpretation is unintuitive in practice. The fundamental formulation of this problem is given by Equation 1.

$$
\begin{aligned}
\min_{\mathbf{u},\rho_e} \quad & \mathbf{f}^T\mathbf{u} \\
& \text{s.t.:} \\
& \left(\sum_{e=1}^{N} \rho_e^p \mathbf{K}_e\right)\mathbf{u} = \mathbf{f} \\
& \sum_{e=1}^{N} v_e \rho_e \leq V \\
& 0 < \rho_{min} \leq \rho_e \leq 1 \text{ for } i = 1, 2, \ldots, N
\end{aligned}
\tag{1}
$$

where: C is the structure's mechanical compliance; $\mathbf{f}$ is the load vector; $\mathbf{u}$ is the displacement vector; $\mathbf{K}$ is the structure's stiffness matrix; and v_e and V are the volumes of a single element and the whole structure, respectively. In Equation 1, the design variable is the pseudo-density ρ_e, and the cost function is related to the compliance. Furthermore, the three constraints are respectively related to equilibrium, upper bound for the structure's volume, and a lower bound ρ_{min} to the pseudo density, to avoid numerical problems with the stiffness matrix.

3 GENERATIVE ADVERSARIAL NETWORKS

Machine learning (ML) is a branch of AI that, by processing datasets, optimizes parametric statistical models to emulate intelligent behaviour. The general ML project pipeline is fairly constant. First the training data must be acquired. For the purposes of the current work, a supervised ML approach is used. Therefore, each sample must contain both the model's input and the respective expected output. Then the quality of this dataset is evaluated through exploratory analysis, assessing, for example, the variety of samples. Afterwards, the data is separated in training, validation, and test samples. Finally, training begins by feeding the data to the model to get predictions. The error in relation to the actual label is used to adjust the system's parameters. Iterating in this procedure, the optimizer changes the model to the set of parameters that minimizes the prediction error (i.e. loss function). Among ML models, the most popular and versatile are ANNs, since they enable an endless variety of architectures. [4] includes a rich review of this line of research. For the purposes of the current work, a specially important ML framework was proposed in [2]: generative adversarial networks (GAN). It involves training two adversarial networks at the same time. The generator G tries to forge samples similar to the ones in the dataset. Meanwhile, the discriminator (or "critic" in WGAN) D must be able to discern false samples from real ones (that is, from the actual dataset).

4 METHODOLOGY

The approach developed by the authors is based on recent advances in dealing with machine learning methods for topology optimization, such as [6]. Figure 1 summarizes the general logic. The end goal is to have a network (the trained generator) that is able to quickly predict the final topology given the initial physical fields over the physical domain of 2D linear FEA structural problems. All code implementations are carried out in the Julia[1] programming language.

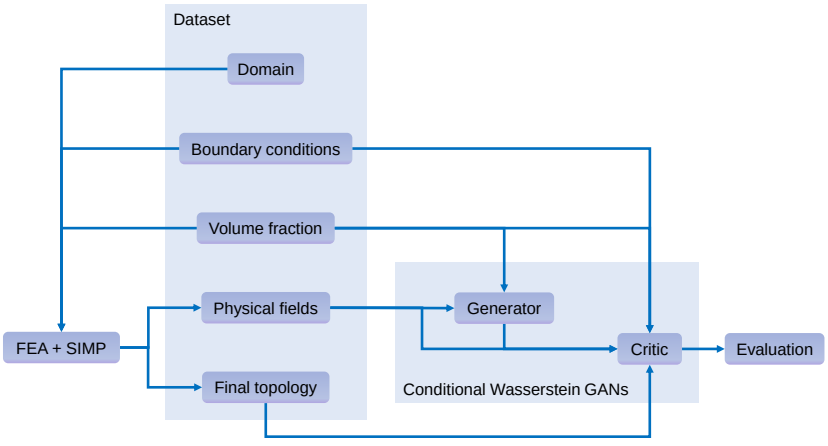

Figure 1: Approach developed in the current work. Source: authors.

[1] https://julialang.org

4.1 Dataset

4.1.1 Generation

Following the basic ML project pipeline (subsection 3), first a dataset[2] is needed for training and validating the ML system. In the current work, the data is created through FEA, and it is based on randomized 2D problems. Procedures related to topology optimization are handled by the TopOpt.jl[3] package. In a fixed rectangular mesh of 140x50 linear square elements of unitary sides, two point loads are placed, with random x and y components ([-90; 90] N) and positions. Additionally, there are two types of supports. In the first, all elements of one random boundary are clamped. In the second, three random elements have their nodes pinned. Figure 2 includes representations of both types of supports. A random volume fraction (VF) between 0.3 and 0.9 is also chosen.

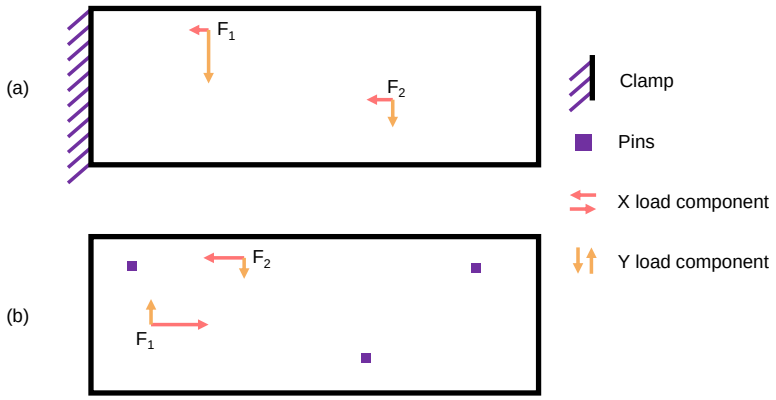

Figure 2: Types of samples generated. (a) Sample with simple pins, clamping a random side of the mesh. (b) Sample with three random elements pinned. Source: authors.

Then, a single FEM analysis is performed on the homogeneous domain with the chosen VF everywhere. With post-processing routines, the physical fields can be determined, namely von Mises stress, and strain energy density. As usual in FEM practice, these fields are obtained in the Gaussian points of the mesh. Afterwards, the standard SIMP procedure is carried out to obtain the final topology for this particular FEA problem. Finally, by repeating these steps, a dataset is built. Each sample is defined by 51x141 matrices (dimensions of nodal mesh) with: volume fraction, von Mises field, strain energy density field, binary support definition, x components of loads, y components of loads, final topology (see Figure 1). The dataset has an initial (unfiltered) size of 138,600 samples.

[2] https://zenodo.org/record/8191138
[3] https://github.com/JuliaTopOpt/TopOpt.jl

4.1.2 Feature Engineering

With the dataset in place, the important step of data analysis must be performed. This is an opportunity to: 1) define data metrics; 2) assess the quality of the dataset; 3) filter the dataset according to targets of these metrics. This is crucial to avoid feeding the network with low quality samples that would mislead, hinder or slow down training. Another issue to avoid is the violation of the linear behavior of 2D FEA. The analysis and filtering of the dataset is done in steps, each focusing on a different type of problem. With the goal of only feeding the GAN with high quality samples, many problems were identified, and samples suffering from those were removed. Table 2 contains the size of the dataset in each *sequential* stage of analysis. In the end, the filtered dataset contains 121,840 samples from the initial size of 138,600 samples, meaning that 16,760 samples were removed. It is equivalent to a 12.1% reduction in size. Only the filtered dataset is used to train the GAN.

Stage	Total	Absolute variation	Percentage variation
Initial	138,600	-	-
Plasticity	137,649	-951	-0.7%
Non-binary	128,867	-8782	-6.4%
Geometrical nonlinearity	128,862	-5	-0.004%
Structural disconnections	127,855	-1007	-0.8%
Isolated Features	121,840	-6015	-4.7%

Table 2: Summary of dataset size variation after each sequential analysis. Source: authors.

4.2 GAN models

In this work, GAN is used for the prediction of final topologies. During training, the generator improves its ability to create faithful topologies. And the discriminator receives either a real topology (from the dataset) or a suggested one (created by the generator), improving its ability to distinguish both. The generator is conditioned to the physical fields; and the discriminator, to the physical fields, volume fraction and FEA inputs.

When it comes to architectures, among the three types of generator used in [6], one was always superior, the "U-SE-ResNet" model. The architecture here proposed is called "QuickTO"[4], and is based on the "ConvNeXt" model from [4]. Their final architecture is used for classification tasks, and has a performance similar to that of the Swin Transformer, a recent and very successful machine learning model for computer vision. However, the state-of-the-art accomplishments of transformers are frequently attributed to their attention mechanism alone, which

[4] `https://github.com/LucasMSpereira/QuickTO`

ConvNeXt does not include. Comparing both generator architectures, QuickTO has 220M parameters, less than half of U-SE-ResNet (470M). Network size is very important, since a smaller model can be trained in less time, uses fewer computational resources, and processes inputs faster (inference time).

4.3 Training

The training scheme included a validation epoch after every 2 training epochs, of which there were 9. 60% of the dataset was used, resulting in 73,103 samples, which were split as: 44,661 for training; 19,140 for validating; and 9,302 for testing. The generator's parameters are updated by the optimizer to minimize the loss given in Equation 2 (which includes the formulation from [5], and empirical weights), as suggested in [6].

$$\mathcal{L}^{G}_{WGAN} = \mathbb{E}\{-30D[G(g_{in})] + 4000L2\} \tag{2}$$

Where $y^* = G(g_{in})$ is the topology suggested by the generator; and $D(\cdot)$ is the discriminator's response. Meanwhile, the critic's parameters are updated by the optimizer to minimize the loss in Equation 3. Pairs (g_{in}, y) are sampled from the dataset, where y is a target topology.

$$\mathcal{L}^{D}_{WGAN} = \mathbb{E}\{D[G(g_{in})]\} - \mathbb{E}[D(y)] \tag{3}$$

As discussed in [11], an important proof of generalization capability of ML models applied in TO is the ability to generate topologies for FEA boundary conditions never seen before by the ML model. Taking this into consideration, samples with clamped top boundary condition were separated for model testing, so that the GAN never sees them during training. Apart from testing, the data was always split in 70% for training, and 30% for validation.

5 DISCUSSION

Figure 3 depicts the loss history of the generator, including the individual terms and the total loss. During experiments, the generator showed very high sensitivity to the empirical weight of the first term in Equation 2, related to the adversarial role of the critic. If this coefficient was too high, the created topologies included no grayscale. At the same time, however, strong mode collapse occurred (low output variety). Meanwhile, high values of the weight of the second term (MSE) in Equation 2 resulted in topologies with the correct general contours, but still diffuse and without fine details, such as small holes and thin structural members.

Figure 4 depicts the critic's loss (and components) during training, validation and test. The high amplitude noise during training epochs is a result of the Dropout layers included for regularization. In this figure, "fake" refers to the critic's output for samples from the generator; and "real", from the dataset (refer to equation 3).

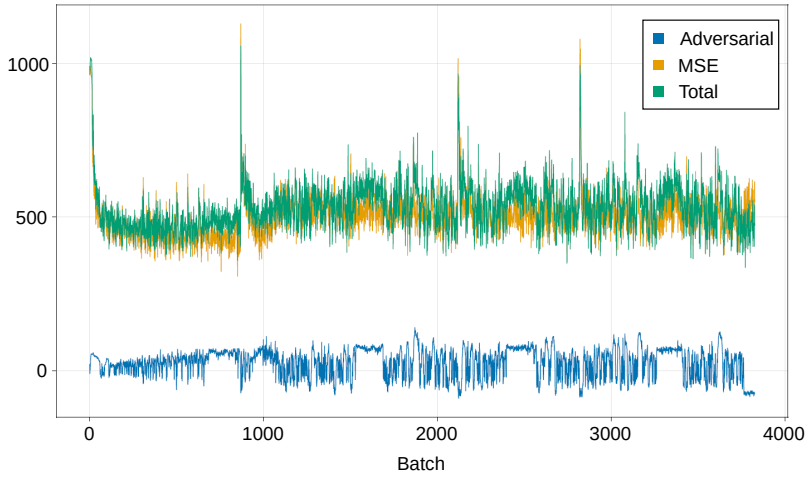

Figure 3: Loss (and its components) history of QuickTO generator. Plot includes training, validation and test epochs. Source: authors.

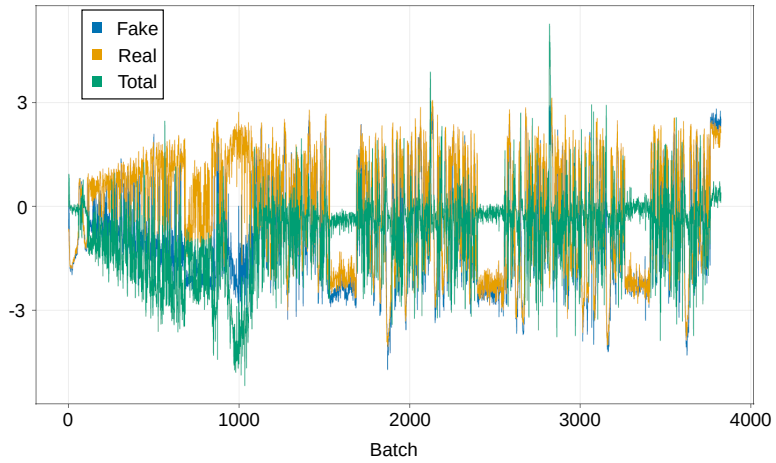

Figure 4: Critic's loss and its components during training, validation and test. Source: authors.

5.1 Comparison

Figure 5 presents normalized histograms of the errors from the trained generators in the test set (structures with clamped top boundary condition). The first row refers to the U-SE-ResNet model suggested in [6]; and the second, to QuickTO, proposed in the present work. The first column includes the distribution of mean squared error (MSE). And the second column depicts the distribution of relative volume fraction error $VF_R = \frac{|VF^* - VF|}{VF}$, where VF^* and VF are the volume fractions from suggested and real topologies, respectively.

The relative compliance error in the test results was also determined, using $C_R = \frac{|C^* - C|}{C}$, where C^* and C are the compliances of the suggested and FEA topologies, respectively. The

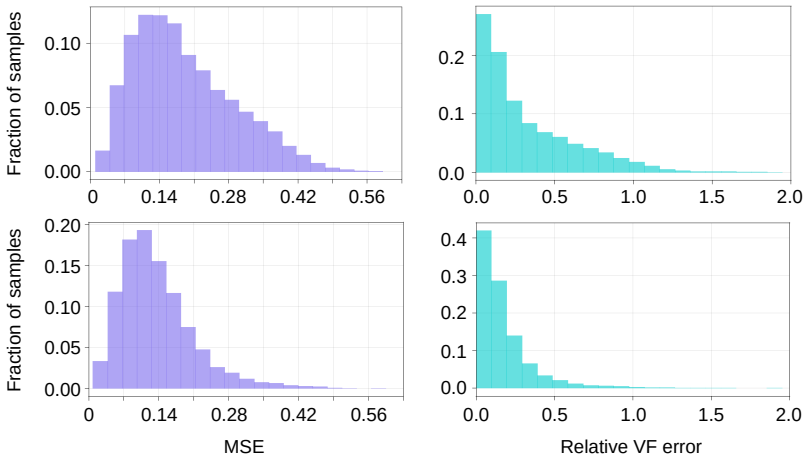

Figure 5: Normalized histograms describing error distribution among samples from test set. First row refers to U-SE-ResNet model, and second row to QuickTO. First column includes topology MSE; and second column, the relative VF error. Source: authors.

suggestion of improper designs was alleviated by dataset filtering stage, but not completely solved. And, inside FEA, these problems can lead to numerically unstable compliances that are very high. To visualize, Figure 6 includes violin plots of the base 10 logarithm of the relative compliance errors in the test split. As shown, the errors are concentrated in -2, referring to 0.01 (i.e. 10%) relative difference in compliance. In summary, QuickTO performed better in topology MSE and relative VF error, but slightly worse in compliance error and disconnections. However, it is important to remember that the design resulting from the standard TO method usually goes through post-processing steps. Therefore, a perfect match of suggested topologies is not necessary.

Table 3 compares the techniques here discussed regarding computational performance, containing averages of processing time to obtain the optimal design. The averages are taken over 150 random FEA structural problems. From these values, on average, QuickTO is more than 4 times faster than SIMP, and 1.5 times faster than U-SE-ResNet. Figure 7 includes a few examples of topologies generated by QuickTO, and their respective FEM inputs, von Mises field distributions, and real designs.

	Avg. Time (s)
SIMP	4.37
U-SE-ResNet	1.66
QuickTO	1.06

Table 3: Computational performance of SIMP, U-SE-ResNet and QuickTO to obtain optimal design. Source: authors.

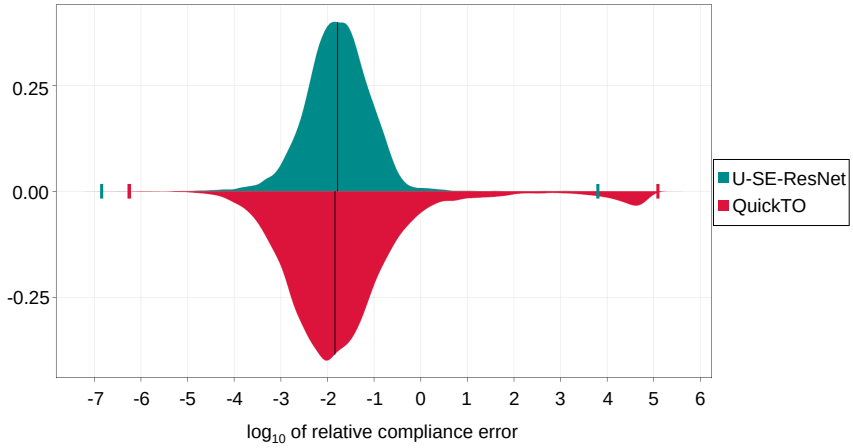

Figure 6: Violin plots of base 10 logarithm of relative compliance error for both models. Metric evaluated on test split. Vertical short bars indicate maxima and minima of error for the model of corresponding color. Vertical black lines indicate median. Source: authors.

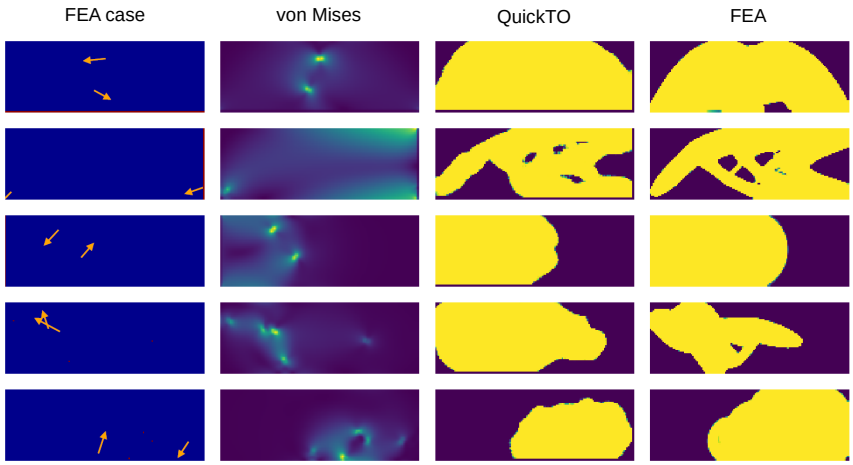

Figure 7: Batch of samples from validation data split. Designs suggested by QuickTO approach. Source: authors.

5.2 Model interpretability

The field of ”interpretable AI” (XAI) has been developed to ”[...] augment the training process, the learned representations, and the decisions with human-interpretable explanations” [8]. This opposes the application of an ML model as a ”black-box”. Accordingly, the behavior of the trained ML models was studied to better understand how they were processing the input information. Starting by the QuickTO generator, a basic question to be answered is ”how much did each input information influence the topologies suggested?”, specifically the strain energy density and von Mises fields. For this purpose, noise is applied only to the specific channel being evaluated, and the corresponding drop in generator performance is measured. When applying

50% of binary noise to the VM channel, a drop of 36% in MAE was observed. As for the strain energy channel, the drop was of 29%. These results indicate that the VM channel is on average more significant to determining the final design.

Of equal importance, one can also attempt to explain the critic's behaviour. Figure 8 includes the relevance distribution of some input channels, namely volume fraction, von Mises field, support definition, x load components, and input topology. "Relevance" here is defined as the gradient multiplied pointwise by the input. In the VF relevance plot, sharp variations are noticeably clustered away from the support (clamped top row); and checkerboard-like patterns are formed all over the channel. In the von Mises relevance plot, similar behaviors are present, but concentrated around the loads. In both the support and load relevance plots, very few elements seem to be of importance to the network. This was expected, since these channels are sparse. Lastly, in the relevance plot for the target topology (third row, first column), the shape of the topology is clearly delineated. Since the critic is trained to distinguish between real and artificial topologies, the high relevance of this channel is expected.

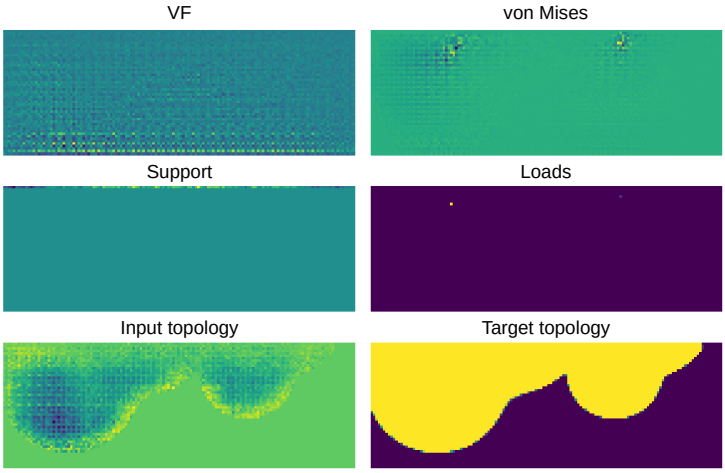

Figure 8: Distribution of relevance across some input channels (alongside target topology). Plots defined by input values multiplied by the gradient. Technique applied to critic (QuickTO approach). Source: authors.

6 CONCLUSIONS

As with any other technological field, there's an ever growing demand for computational performance in topology optimization analyses, with which ML can help. The current work is the first application of both a ConvNeXt-inspired model and the Wasserstein loss in the field. A very large dataset of samples was generated, and filtered against a handful of possible problems. Furthermore, these samples faced very few restrictions, exposing the models to a great variety of FEA problems. Additionally, the model's generalization capabilities were evaluated in a class of FEA boundary conditions never seen by the model, being therefore a more demanding test. Also, AI interpretability techniques were applied to both the generator and discriminator, aiming at a better understanding of their behaviors. However, there are still many challenges

and possible directions. One example is to include the compliance error in the generator's loss. Additionally, consideration of not only statistical similarity, but also diversity, novelty, constraint satisfaction, design performance, and conditioning. More extensions of the current work are FEA non-linearities, 3D problems, WGAN with gradient penalty, sliced-wasserstein loss, geometry encoding, other generative models (e.g. diffusion, normalizing flow, energy), spectral normalization, and geometric ML based on the domain mesh.

References

[1] Bendsoe M. P., Sigmund O (2004) Topology optimization theory, methods and applications. Springer.

[2] Goodfellow I. J. et al (2014) Generative Adversarial Networks. Advances in neural information processing systems.

[3] He K., Zhang X., Ren S., Sun J (2020) Deep Residual Learning for Image Recognition. Conference on Computer Vision and Pattern Recognition. IEEE.

[4] Z. Liu, H. Mao, C. Y. Wu, C. Feichtenhofer, T. Darrel, S. Xie (2022) A ConvNet for the 2020s. Conference on Computer Vision and Pattern Recognition. IEEE.

[5] Martin A., Soumith C., Léon B (2017) Wasserstein Generative Adversarial Networks. International Conference on Machine Learning.

[6] Nie Z., T. Lin, H. Jiang, L. B. Kara (2021) TopologyGAN: Topology Optimization Using Generative Adversarial Networks Based on Physical Fields Over the Initial Domain. Journal of Mechanical Design. ASME.

[7] Nie Z. et al (2020) Stress field prediction in cantilevered structures using convolutional neural networks. Journal of Computing and Information Science in Engineering. American Society of Mechanical Engineers.

[8] Samek W. et al (2021) Explaining Deep Neural Networks and Beyond: A Review of Methods and Applications. Proceedings of the IEEE. IEEE.

[9] Zheng S., He Z., Liu H. (2021) Generating three-dimensional structural topologies via a U-Net convolutional neural network. Thin-Walled Structures. Elsevier.

[10] Sosnovik I., Oseledets I (2019) Neural networks for topology optimization. Russian Journal of Numerical Analysis and Mathematical Modelling. De Gruyter.

[11] R. V. Woldseth, N. Aage, J. A. Baerentzen, O. Sigmund. (2022) On the use of artificial neural networks in topology optimisation. Structural and Multidisciplinary Optimization. Springer.