POLYPUBLIE
Polytechnique Montréal

POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

Titre: A Novel Framework for Accelerating Topology Optimization : Merging Machine Learning with Feature-Mapping Methods
Title:

Auteur: Thomas Rochefort-Beaudoin
Author:

Date: 2024

Type: Mémoire ou thèse / Dissertation or Thesis

Référence: Rochefort-Beaudoin, T. (2024). A Novel Framework for Accelerating Topology Optimization : Merging Machine Learning with Feature-Mapping Methods [Thèse de doctorat, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/61717/
Citation:

Document en libre accès dans PolyPublie
Open Access document in PolyPublie

URL de PolyPublie: https://publications.polymtl.ca/61717/
PolyPublie URL:

Directeurs de recherche: Aurelian Vadean, & Sofiane Achiche
Advisors:

Programme: Génie mécanique
Program:

Ce fichier a été téléchargé à partir de PolyPublie, le dépôt institutionnel de Polytechnique Montréal
This file has been downloaded from PolyPublie, the institutional repository of Polytechnique Montréal

https://publications.polymtl.ca

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

A Novel Framework for Accelerating Topology Optimization: Merging Machine Learning with Feature-Mapping Methods

THOMAS ROCHEFORT-BEAUDOIN

Département de génie mécanique

Thèse présentée en vue de l'obtention du diplôme de *Philosophiæ Doctor*

Génie Mécanique

Décembre 2024

© Thomas Rochefort-Beaudoin, 2024.

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Cette thèse intitulée :

A Novel Framework for Accelerating Topology Optimization: Merging Machine Learning with Feature-Mapping Methods

présentée par **Thomas ROCHEFORT-BEAUDOIN**

en vue de l'obtention du diplôme de *Philosophiæ Doctor*

a été dûment acceptée par le jury d'examen constitué de:

Sébastien LECLAIRE, président
Aurelian VADEAN, membre et directeur de recherche
Sofiane ACHICHE, membre et codirecteur de recherche
Walid JOMAA, membre
Ilmar Ferreira SANTOS, membre externe

DEDICATION

To my parents, Marie and Martin, for their endless support, encouragement, and love.

To my sisters, Maude and Cloé, for always inspiring me in countless ways.

And to Laurence, my love, for standing by my side through every triumph and hardship.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deepest gratitude to my thesis co-directors, Professor Aurelian Vadean and Professor Sofiane Achiche. Your unwavering support, insightful guidance, and constant encouragement have been invaluable throughout this journey. I am immensely grateful to have had the opportunity to learn and grow under your mentorship.

I would also like to extend my sincere thanks to Professor Niels Aage from the Technical University of Denmark for the profound collaboration in advancing various aspects of this thesis. Your innovative ideas and constructive feedback have significantly enriched this research. I am also grateful to the entire *TopOpt* group for hosting me during my visit, for providing a stimulating and welcoming environment that fostered both professional and personal growth.

Most of all, I would like to thank my family and friends for their unwavering support and encouragement throughout the years. I am particularly grateful for the immeasurable support my parents have given me every day. Being engineers themselves, they have always inspired me with their knowledge and dedication, providing guidance and practical support that has been invaluable during this journey. I also want to express my deepest gratitude to Jean and Annick for their unwavering support. I began this PhD while quarantined in their home at the onset of the Covid-19 pandemic, and they welcomed me as a part of their family from the very start.

And finally, my deepest gratitude goes to my better half, Laurence. Your love, understanding, and belief in me have been a constant source of strength and motivation. I'm excited to see what the future holds for us.

RÉSUMÉ

Cette thèse explore l'intégration de l'apprentissage automatique et des méthodes de projection d'entités géométriques (*feature-mapping methods*) pour l'optimisation topologique. La recherche aborde deux défis principaux: le coût computationnel élevé associé à l'optimisation topologique à grande échelle et le post-traitement laborieux requis pour convertir les distributions de densité optimisées en géométries paramétriques.

La première partie de la thèse présente un modèle d'apprentissage profond supervisé qui génère des variables de conception géométriques indépendamment de la résolution du maillage d'analyse par éléments finis. Malgré des performances de régression de pointe, les prédictions du modèle présentent une rigidité moyenne inférieure de 11.48 % par rapport aux conceptions conventionnelles. Cependant, lorsque utilisée pour générer des solutions initiales, cette approche réduit le temps de calcul de 36.84 % sans compromettre les performances mécaniques.

Pour surmonter les limites des modèles basés sur la régression, la deuxième partie de la thèse présente le *Structural Optimization gym* (SOgym), un environnement d'apprentissage par renforcement intégrant la physique de l'optimisation topologique dans la fonction de récompense. L'agent DreamerV3-100M produit des structures avec une rigidité 54% inférieure par rapport aux méthodes conventionnelles, et a démontré un taux de déconnexion structurelle de 0 %.

La troisième partie de la thèse présente YOLOv8-TO, un modèle de segmentation d'instance qui automatise la rétro-ingénierie des structures optimisées. YOLOv8-TO surpasse les techniques de squelettisation, réalisant une amélioration moyenne de 13.84 % du coefficient de Dice, avec un maximum de 20.78 %. La méthode démontre une bonne généralisation aux géométries complexes tout en maintenant des temps d'inférence rapides.

Les résultats de cette recherche contribuent à l'avancement des applications de l'apprentissage automatique dans l'optimisation topologique, offrant des pistes de recherche pour les applications de conception directe et de post-traitement. Les méthodologies proposées non seulement améliorent l'efficacité du processus d'optimisation, mais facilitent également l'intégration dans les flux de travail de conception assistée par ordinateur conventionnels. Les travaux futurs se concentreront sur l'élargissement de l'applicabilité de ces méthodes aux structures tridimensionnelles et l'intégration de phénomènes physiques plus complexes dans les processus d'apprentissage.

ABSTRACT

This thesis investigates the integration of machine learning (ML) and feature-mapping methods for Topology Optimization (TO). The research addresses two primary challenges: the high computational cost associated with large-scale TO and the labor-intensive post-processing required to convert optimized density distributions into parametric geometries.

The first part of the thesis presents a supervised deep learning model based on the Moving Morphable Components (MMC) framework, which generates geometric design variables independent of finite element analysis (FEA) mesh resolution. Despite achieving state-of-the-art regression performance, the model's predictions resulted in structures with an average stiffness 11.48% lower than conventional MMC designs. However, when used to generate initial designs for conventional optimization, this approach reduced computation time by 36.84% without compromising the mechanical performance of the final topology.

To overcome the limitations of regression-based models, the second part of the thesis introduces the Structural Optimization gym (SOgym), a novel reinforcement learning (RL) environment that incorporates the physics of TO into the reward function. The DreamerV3-100M agent trained within SOgym produced designs with compliance within 54% of that achieved by traditional optimization methods and demonstrated a 0% disconnection rate, a notable improvement over existing ML methods that often struggle with disconnected structures.

The third part of the thesis presents YOLOv8-TO, an instance segmentation model that automates the reverse engineering of optimized structures into interpretable geometric parameters. YOLOv8-TO outperformed traditional skeletonization techniques, achieving an average 13.84% improvement in the Dice coefficient and a peak enhancement of 20.78%. The method demonstrated strong generalization to complex geometries, while maintaining fast inference times suitable for integration into design workflows.

The findings of this research contribute to the advancement of ML applications in TO, providing scalable research directions for direct design and post-processing applications. The proposed methodologies not only enhance the efficiency of the optimization process but also facilitate the integration of optimized designs into conventional computer-aided design (CAD) workflows. Future work will focus on expanding the applicability of these methods to three-dimensional structures and incorporating more complex physical phenomena into the training processes.

TABLE OF CONTENTS

DEDICATION III

ACKNOWLEDGEMENTS IV

RÉSUMÉ V

ABSTRACT VI

TABLE OF CONTENTS VII

LIST OF TABLES XII

LIST OF FIGURES XIII

LISTE OF SYMBOLS AND ABBREVIATIONS XIX

LIST OF APPENDICES XXI

CHAPTER 1 INTRODUCTION 1

1.1 Context 1

1.2 Topology Optimization 2

1.2.1 MMA Optimization 2

1.3 Problematic 4

1.4 Machine Learning for Topology Optimization 5

1.5 Research Objectives 8

1.6 Thesis Structure and Overview 9

CHAPTER 2 LITERATURE REVIEW 10

2.1 Topology Optimization Algorithms 11

2.1.1 Solid Isotropic Material with Penalization 12

2.1.2 Method of Moving Morphable Components 13

2.2 Post-Processing of Optimal Structures 15

2.3 Machine Learning for Topology Optimization 18

2.3.1 Image-Based Regression Learning....18
2.3.2 Generative Modeling Approaches....20
2.3.3 Machine Learning for Feature-Mapping Methods....22
2.3.4 'Physics-Informed' Approaches....23
2.3.5 Reinforcement Learning....24
Proximal Policy Optimization....27
DreamerV3....28
Reinforcement Learning for Structural Design and Optimization....31
2.3.6 Machine Learning for Post-Processing of Topology Optimization Outputs....32
2.4 Conclusion on the State-of-the-Art of Machine Learning for Topology Optimization....33
CHAPTER 3 NEW APPROACHES IN MACHINE LEARNING FOR DIRECT DESIGN AND POST-PROCESSING TOPOLOGY OPTIMIZATION APPLICATIONS....35
3.1 Feature-Mapping Methods for Machine Learning Topology Optimization....36
3.2 Summary of Articles....36
CHAPTER 4 ARTICLE 1: SUPERVISED DEEP LEARNING FOR THE MOVING MORPHABLE COMPONENTS TOPOLOGY OPTIMIZATION FRAMEWORK....39
4.1 Introduction....40
4.1.1 The Moving Morphable Components Topology Optimization Method....40
4.1.2 Machine Learning for Topology Optimization....43
4.2 Method....47
4.2.1 Benchmark Problems....48
4.2.2 Network Architecture....50
4.3 Numerical Results....52
4.3.1 Performance of the Model's Prediction in a Direct Design Strategy....52
4.3.2 Performance of the Model as an Initial Layout Generator for MMC....56

4.3.3 Out-of-Distribution Boundary Conditions Performance 60
4.4 Discussion 63
4.5 Conclusion 65
CHAPTER 5 ARTICLE 2: STRUCTURAL DESIGN THROUGH REINFORCEMENT LEARNING 67
5.1 Introduction 68
5.2 Method 70
5.2.1 Action Space 71
5.2.2 Boundary Conditions 73
5.2.3 Observation Space 74
5.2.4 Reward Function 77
5.2.5 RL Algorithms 78
5.2.6 Model Architecture 78
5.3 Results 79
5.3.1 Evaluation of Performance Across Different Observation Spaces 80
5.3.2 DreamerV3 vs PPO 81
5.3.3 Learning Rate Comparison 83
5.3.4 Structural Performance Comparison with Baseline 84
5.4 Discussion 88
5.4.1 Algorithm Performance and Structural Quality 88
5.4.2 Breakeven Threshold 88
5.4.3 Richer Observation Space Brings Training Efficiency and Performance 89
5.4.4 Model-Based DreamerV3 Outperforms Model-Free PPO 89
5.4.5 A “Physical” RL Training Objective Solves the Disconnection Problem 90
5.4.6 Comparing Learning Rates With Engineering Students 90

5.5 Conclusion 91
5.6 Data & Code Availability 92
5.7 Usage of Generative AI 92
5.8 Acknowledgements 93
CHAPTER 6 ARTICLE 3: FROM DENSITY TO GEOMETRY: INSTANCE SEGMENTATION FOR REVERSE ENGINEERING OF OPTIMIZED STRUCTURES 94
6.1 Introduction 95
6.2 Method 98
6.2.1 Dataset 98
MMC Dataset 99
Random Assembly Dataset 100
SIMP Dataset 101
Low Volume Fraction SIMP Dataset ($SIMP_{5\%}$) 101
Out-of-Distribution Dataset 102
6.2.2 The YOLOv8-TO Model 103
Pixel-Based Bounded Reconstruction Loss 105
Segmentation Mask Non-Max Suppression 106
6.2.3 Comparing YOLOv8-TO to Skeletonization 107
6.2.4 Disconnected Load Path in Predicted Structures 108
6.3 Results 108
6.4 Discussion 117
6.4.1 Improvement in Precision 118
6.4.2 Small Impact From Using Random Structures 118
6.4.3 Scaling Yields a Significant Performance Boost 119
6.4.4 Faster Inference Speed on Common Workspace Hardware 119

6.4.5 Sensitivity Issues 120
6.5 Conclusion 122
CHAPTER 7 DISCUSSION 125
7.1 Contributions and Importance of Research 128
7.2 Limitations 130
CHAPTER 8 CONCLUSION AND RECOMMENDATIONS 132
DECLARATION OF GENERATIVE AI AND AI-ASSISTED TECHNOLOGIES IN THE WRITING PROCESS 135
REFERENCES 136
APPENDICES 149

LIST OF TABLES

Table 4.1: Summary of the current ML accelerated MMC literature. 45

Table 4.2: Description of the input description vector used to describe the topology optimization problem. 51

Table 4.3: Mechanical performance evaluation of the direct MMC process, the model's predictions and the direct MMC process using the model's predictions as the initial layout. 53

Table 4.4: Computing time savings in terms of iteration of the conventional MMC optimization with the model-generated initial layout. 58

Table 4.5: Randomly chosen OOD boundary conditions used. 61

Table 5.1: Minimum and maximum boundaries of the design variables. 73

Table 5.2: SOgym environment parameters 74

Table 5.3: Supported observation space configurations of the SOgym environment. 75

Table 5.4: Composition of the β description vector. 76

Table 5.5: Performance of the RL-generated structures compared to the corresponding baseline design obtained via conventional MMC optimization. 85

Table 5.6: Processing time comparison between conventional optimization and RL algorithms. 86

Table 6.1: Parameters defining the boundary conditions distribution. 99

Table 6.2: Aggregated results of YOLOv8-TO vs skeletonization on all test datasets. 110

Table 6.3: Comparison of inference cost of YOLOv8-TO vs Skeletonization method. 120

Table 7.1: Summary of major contributions to the direct design and post-processing fields of the machine learning for topology optimization community. 129

Table 7.2 : Academic contributions completed during this PhD. 130

LIST OF FIGURES

Figure 1.1: Comparison of the MMA and GCMMA approximations at the timestep k for an arbitrary function fx at the current design variable point $x(k)$. Figure recreated from [6]. .. 3

Figure 1.2: Integration of machine learning applications in the topology optimization process. ... 6

Figure 1.3: Illustration of a U-Net CNN model trained to generate optimal density fields in a direct design, image-to-image approach, with problem parameters represented as images as input. The U-Net CNN is made up of trainable CNN layers with residual connections between the various layers (in blue). The SIMP method is illustrated here as the ground truth sample generation method. 7

Figure 2.1: Number of publications per year from 2010 to 2023 on *Web of Science* returned by a 'All fields' search for the terms 'topology optimization' AND ('machine learning' OR 'deep learning'). *Certain data included herein are derived from Clarivate™ (Web of Science™). © Clarivate 2024. All rights reserved.* 10

Figure 2.2: Comparison of density-based methods representation of the structure (left) vs feature-mapping methods (right). 12

Figure 2.3 : Illustration of an MMC component with linearly varying thickness. Variables $t1$ and $t2$ represent the two thicknesses. $x0$ and $y0$ are the centroid coordinates. L and θ are the length and orientation of the component respectively. 13

Figure 2.4 : Illustration of a linearly varying MMC component. a) The hyperelliptic TDF. b) the level-set operation defining the material domain based on the 0 level. c) Heaviside projection on the density space..... 15

Figure 2.5: Illustration of post-processing of TO outputs from a SIMP method. a) Remeshing technique. b) Parametric skeletonization extraction converts the topology into a high-level skeleton representation. 16

Figure 2.6: Intermediate and final iterations of the classic cantilever beam optimization obtained from the 88-line Matlab code [59] with the command *top88(240,80,0.5,3.0,7,1)*. 18

Figure 2.7: Fully convolutional direct design approach. The input is a multi-channel image where the first channel represents the support condition, the second channel represents the desired volume fraction, and the last channel represents the location of a unit downward load. 19

Figure 2.8: Overview of generative approaches for TO. a) Conditional GAN: The generator creates images from noise and problem parameters to maximize the discriminator error. b) Diffusion model: The forward process adds noise, and the reverse process iteratively removes it to reconstruct the original. 21

Figure 2.9: Three samples of generated structures for a single random load case obtained via TopoDiff [74]. 22

Figure 2.10 : Illustration of the interaction between the agent and environment in a MDP [85]. 25

Figure 2.11: Recurrent State-Space Model (RSSM) of DreamerV3. The world model encodes sensory inputs into latent representations (zt) using encoders (Enc.) and reconstructs inputs with decoders (Dec.). Recurrent states (ht) predict future latent states using the past states and the actions (at), enabling the agent to imagine and plan future trajectories. Figure recreated from [95]. 29

Figure 2.12: Actor-Critic Interaction in DreamerV3. The actor selects actions at to maximize returns (rt), while the critic evaluates state values $v\psi st$. Imagined trajectories from the world model enable efficient learning. Figure recreated from [95]. 30

Figure 4.1: Design variables defining the shape of a linearly varying thickness component of the MMC framework 41

Figure 4.2: Flowchart illustrating the supervised learning pipeline used for the proposed method. 48

Figure 4.3: Loading and support conditions used. In red: the boundary where the load is placed. In blue: the fully supported boundary. 49

Figure 4.4: Example of a generated support/loading boundary condition and its corresponding optimal topology obtained via MMC. Each component's contour is represented by a unique colour. 50

Figure 4.5: Comparison of results obtained from the neural network model (right column) and the conventional MMC optimization (left column). 54

Figure 4.6: Comparison of results obtained from the neural network model (right column) and the conventional MMC optimization (left column) for the 4 worst direct design prediction performances. 55

Figure 4.7: Two samples from the test set showing how a lower RMSE does not translate to lower compliance. A good-performing model prediction (upper row) has a higher RMSE loss value than a bad-performing model prediction (lower row). 56

Figure 4.8: Uniform initial solution used for MMC. 56

Figure 4.9: Model predictions as initial layout for conventional MMC optimization. 59

Figure 4.10: Performance profiles of the final compliance (a) and the total iterations until convergence (b) of the conventional MMC with uniform initial solution, and the model's initial solution accelerated MMC (MMC+DL). It can be observed that the model-accelerated method shows a clear reduction in total iterations by placing itself in the upper left corner of the graph (b), with a clear gap with the conventional MMC method. 60

Figure 4.11: Topologies obtained for the OOD boundary conditions test. 62

Figure 5.1: MMC component defined by endpoint coordinates and thicknesses. 72

Figure 5.2: Boundary conditions parameters defining the random distribution 74

Figure 5.3: Illustration of the TopOpt observation space. a) Example of a random episode leading to a connected final structure. b) Example of a random episode leading to a disconnected structure where the strain energy is not shown. 77

Figure 5.4: Squashed SOgym reward using the inverse of the natural logarithm of the compliance. 78

Figure 5.5: Actor-critic IMPALA-inspired architecture of the PPO model. 79

Figure 5.6: Performance comparison of different observation spaces for DreamerV3-12M model. a) Mean evaluation reward: TopOpt (blue) achieves the highest reward of 0.29 in around 2.3M episodes. Image (green) achieves a maximum reward of 0.28 in around 3.75M episodes, while Vector (orange) reaches 0.27 after nearly 6M episodes. b) Training Frames Per Second (FPS):

Vector (orange) maintains the highest FPS at 279, followed by Image (green) at 173 FPS. TopOpt (blue) has the lowest FPS at 112. 81

Figure 5.7: Comparison of evaluation rewards for DreamerV3-12M (orange), DreamerV3-100M (green), and PPO (blue). PPO reaches the highest evaluation performance while the two DreamerV3 show the fastest pace of improvement early in training and then rapidly top off without improving much. 82

Figure 5.8*:* PPO Training Dynamics: KL Divergence and Evaluation Reward. This figure shows the Kullback-Leibler (KL) divergence in logarithmic scale (in blue) and evaluation reward (in orange) during PPO training. The KL divergence generally rises, indicating learning instability, with a sharp spike around the 4.6 million episodes that aligns with a significant drop in evaluation performance, highlighting instability in the training process. 83

Figure 5.9: Normalized inverse compliance over episodes for PPO (blue), DreamerV3-12M (orange), and DreamerV3-100M (green). The DreamerV3-100M demonstrated the highest learning rate of 1.78 -07, just under 4 orders of magnitude smaller than then the learning rate of the TopOpt game in-class experiment (2.40e-03, black vertical line). 84

Figure 5.10: Comparison of generated structures from the RL agents and the corresponding baseline. The samples were randomly chosen from the 1000 test samples. 87

Figure 6.1: Compression algorithm analogy. a) YOLOv8-TO as a learnable geometrical encoder with an explicit geometry decoding defined by the MMC hyperelliptic functions. b) Skeleton extraction heuristic as a geometrical encoder with a heuristic-based thickness estimation decoder. 97

Figure 6.2: Reverse-engineering of an optimal density distribution as an instance segmentation approach. 98

Figure 6.3: Illustration of the boundary conditions distribution 99

Figure 6.4 : MMC dataset training sample. a) Output from MMC optimization. b) Black and white input for the instance segmentation model. c) Instance segmentation labels. 100

Figure 6.5: Random MMC training samples. 100

Figure 6.6 : SIMP test dataset samples. 101

Figure 6.7: Low volume fraction SIMP samples with volume constraint of 5%. 101

Figure 6.8: Out-of-distribution test samples. a): 2D femur with total volume constraint. b) 2D femur with local volume constraints, reproduced with permission from [164]. c) Cantilever beam partially optimized with the level set method, obtained with [30]. d) Compliant mechanism force inverter, obtained with [59]. The red arrow shows the optimization objective to maximize. 102

Figure 6.9: Compliant mechanism force inverter TO problem definition with symmetry conditions. 103

Figure 6.10: Architecture of the YOLOv8-TO model. The input density field is processed by a CNN CSPDarknet feature extractor (left). The neck architecture (middle) passes three different layers from the backbone to separate YOLOv8-TO heads and a segmentation mask prototype network. Each YOLOv8-TO head (right) generates predictions for classification, bounding box, mask coefficients, and design variables. The segmentation mask prototype network is a fully convolutional network that generates prototype segmentation masks. .. 104

Figure 6.11: MMC Component formulation used in [40] 105

Figure 6.12: Endpoint MMC formulation used in this paper. 105

Figure 6.13: Illustration of the MMC component. a) The contour surface of the level set function. Everything above the '0^{th}' level is highlighted in red. b) The corresponding sigmoid projection. c) 2D view of the sigmoid projection corresponding to the density field of the single MMC component. 105

Figure 6.14: Pixel-based reconstruction loss algorithm. 106

Figure 6.15: a) Orthogonally intersecting pair of equal length MMC component. b) Model detection results with NMS on the bounding box. c) Model detection results with NMS based on the Dice coefficient of masks. 107

Figure 6.16: Skeleton thickness estimation algorithm. 107

Figure 6.17: Comparative training curves for bounding box mean Average Precision (mAP) with a range of 50-95 and regression head loss across a) datasets incorporating MMC, random, and a combination of both, and b), various model scales. 111

Figure 6.18: Results sample from the MMC test dataset. 112

Figure 6.19: Results sample from the SIMP dataset. 112

Figure 6.20: Results sample from the $SIMP_{5\%}$ dataset. 113

Figure 6.21: Results for the 2D femur case with total volume constraint. 114

Figure 6.22: Results for the 2D femur case with local volume constraint. 115

Figure 6.23: Results for the level set cantilever beam after 20 iterations. 116

Figure 6.24: Results for the compliant mechanism force inverter. 117

Figure 6.25: Effect of confidence and IoU threshold on detected components for the OOD level set example with YOLOv80TO x-large. 122

Figure 7.1: Positioning of the three articles presented in this thesis in relation to the ML-TO applications presented in Chapter 1. Chapter 4 and 6 focus on addressing key limitations of direct-design applications. Chapter 5 present a novel framework to leverage robust computer vision models for the post-processing of TO outputs. 125

LISTE OF SYMBOLS AND ABBREVIATIONS

AI	Artificial Intelligence
AM	Additive Manufacturing
ANN	Artificial Neural Network
CAD	Computer-Aided Design
CNN	Convolutional Neural Network
CSG	Constructive Solid Geometry
CV	Computer Vision
DL	Deep Learning
FEA	Finite Element Analysis
GAN	Generative Adversarial Network
GCMMA	Globally Convergent Method of Moving Asymptotes
HPC	High-Performance Computing
KKT	Karush-Kuhn-Tucker
MMA	Method of Moving Asymptotes
MMC	Moving Morphable Components
ML	Machine Learning
ML-TO	Machine Learning for Topology Optimization
MO	Main Objective
MSE	Mean Squared Error

NLP	Natural Language Processing
NMS	Non-Max Suppression
PP-TO	Post-Processing of Topology Optimization
RL	Reinforcement Learning
RMSE	Root Mean Squared Error
SOgym	Structural Optimization Gym
SO1	Secondary Objective 1
SO2	Secondary Objective 2
SO3	Secondary Objective 3
SIMP	Solid Isotropic Material with Penalization
TO	Topology Optimization
YOLO	You Only Look Once

LIST OF APPENDICES

APPENDIX A SUPPORTING MATERIAL FOR CHAPTER 4 149

APPENDIX B SUPPORTING MATERIAL FOR CHAPTER 5 153

APPENDIX C SUPPORTING MATERIAL FOR CHAPTER 6 154

CHAPTER 1 INTRODUCTION

1.1 Context

Designing structures often symbolizes the essence of engineering, with gravity-defying bridges and high-rise towers standing tall as symbols of the ingenuity and creativity of humanity. Central to this design process is the task of structural optimization, which can be defined as “making an assemblage of material sustain loads in the best way” [1]. Historical examples of structural optimization are evident in iconic structures such as the Eiffel Tower, where mathematical modeling of wind forces led to its characteristic inward tapering base, eliminating the need for cross-bracing truss supports [2]. Today, structural optimization involves the application of modern mathematical and computational methods to improve the performance of structures, typically aiming to minimize material usage while maximizing specific performance criteria. Structural optimization can be classified based on the degree of geometric freedom allowed by the optimization algorithm. In increasing order of geometric freedom, the classifications are size optimization, shape optimization, and topology optimization.

Size optimization seeks the optimal value for design variables that represent specific features of a structure, such as the radius of a circle or the thickness and length of truss members. Shape optimization involves modifying the boundaries and contours of a structure without altering its fundamental connectivity. The design variables in shape optimization typically represent the structure's boundary through differential equations. Both size and shape optimization restrict the geometrical freedom of the resulting design, forcing the optimization algorithm to explore a narrower region of the design space. These methods are typically employed later in the design process when the structure's conceptual design and connectivity have been established [3].

Topology Optimization (TO) is the most general and flexible form of structural optimization. It involves variables representing material distribution within a given domain, allowing for the creation or removal of material. This capability leads to changes in the connectivity and layout of the structure, making topology optimization the most complex and versatile approach.

1.2 Topology Optimization

For a structural mechanics problem, the typical TO problem formulation under linear elasticity assumptions takes the form presented in Equation 1.1:

$$\begin{aligned} \min_{\rho} \quad & : \quad \Phi\big(U(\rho)\big) \\ s.t. \quad & : \quad g_i(\rho) \le g_i^*, \quad i = 1, \dots, M \\ & : \quad \rho_{min} \le \rho \le \rho_{max} \\ & : \quad K(\rho)U = F \end{aligned} \tag{1.1}$$

Where $\Phi(U(\rho))$ is the objective function defined by the state variables U, corresponding to the displacement field, and the design variables ρ, corresponding to the density distribution of the structure. The problem needs to respect certain design constraints $g_i(\rho) \le g_i^*$, the bounds of the design variables $\rho_{min} \le \rho \le \rho_{max}$, as well as the state equation $K(\rho)\, U = F$ where K and F are the stiffness and force matrices respectively.

A sensitivity analysis is performed to obtain the derivative of the objective function with respect to the design variables.

1.2.1 MMA Optimization

To update the design variables iteratively, a non-linear iterative optimizer, the Method of Moving Asymptotes (MMA) is typically used [4]. MMA is a robust and efficient optimization algorithm widely used in structural optimization due to its capability to handle many design variables and constraints. MMA constructs a sequence of convex sub-problems that approximate the original non-linear optimization problem. At each iteration, these sub-problems are formulated using first-order Taylor series expansions of the objective and constraint functions, combined with specially chosen moving asymptotes to ensure convexity. This approach allows MMA to navigate complex design spaces effectively and converge to an optimal solution. The moving asymptotes adapt dynamically based on the current design point, which enhances the stability and convergence rate of the algorithm. For each iteration k, the sub-problem in MMA is formulated in Equation 1.2:

$$\begin{aligned} min \quad & f_0(\boldsymbol{x}) + a_0 z + \sum_{i=1}^{m} c_i y_i + \frac{1}{2} d_i y_i^2 \\ s.t. \quad & f_i(\boldsymbol{x}) - a_i z - y_i \le 0 && i = 1, \dots, m \\ & \alpha_j^k \le x_j \le \beta_j^k && \mathrm{j} = 1, \dots, n \\ & y_i \ge 0, z \ge 0 && i = 1, \dots, m \end{aligned} \tag{1.2}$$

Where $\boldsymbol{x}$ is the vector of design variables with lower and upper limits α_j^k and β_j^k which are updated at each iteration according to heuristics. Artificial optimization variables $y_1, \ldots, y_m$ and z enable the formulation of various optimization problems with coefficients that respects: $a_0 > 0, a_i \geq 0, c_i \geq 0, d_i \geq 0 \; and \; c_i + d_i > 0$ for all i.

To solve these sub-problems, a "primal-dual interior-point approach" is commonly used, where a sequence of relaxed Karush-Kuhn-Tucker (KKT) conditions are solved using Newton's method. By solving these sub-problems iteratively, MMA refines the design variables step by step, making it particularly suitable for problems with non-linear and non-monotonous behaviors. MMA only uses one asymptote at a time for a monotonous approximation, as illustrated in Figure 1.1. The Globally Convergent MMA version (GCMMA) modifies the approximating functions to use both asymptotes at the same time for a non-monotonous approximation and adds inner iterations to provide more conservative variable updates. More details on the MMA and GCMMA subproblem formulation and its primal-dual interior-point approach can be found in [5].

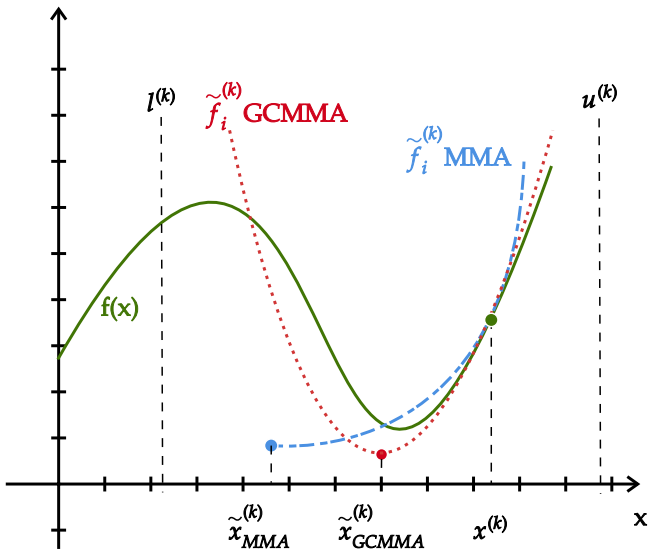


Figure 1.1: Comparison of the MMA and GCMMA approximations at the timestep k for an arbitrary function $f(x)$ at the current design variable point $x^{(k)}$. Figure recreated from [6].

1.3 Problematic

TO is a powerful tool that enables engineers to design products with a reduced carbon footprint, playing a crucial role in addressing global warming. High-resolution TO is essential for large-scale structures. Given the commercial software's maximum resolution of 5 million elements, this means that for an airplane wing's dimensions, the mesh elements would have a minimum size of $5cm^3$, which is far too large to accurately capture the physics of the structure [7]. Advances in research have introduced methods to perform TO at giga-scale resolutions (in the billions of elements) built on parallel computing libraries like the Portable, Extensible Toolkit for Scientific Computation (PETSc) [8]. Applied to full-scale airplane wing structures, the resulting weight saving allowed for a theoretical reduction in fuel consumption of 40 to 200 tons per plane annually [9]. For suspension bridges, large-scale TO of bridge girder sections can reduce the weight of the required structural steel by 28.4% [10], resulting in a potential reduction of 18.45% in CO2 emissions from materials usage. Even with simplifying the physics by only considering linear elasticity, these two experimental studies required a minimum of 192,000 and 1,360,000 CPU hours respectively on large compute clusters. Although highly efficient implementation for FEA and adjoint sensitivity analysis exist to treat such large structures on massively parallel computing systems [11], access to large computer clusters limits the widespread usage of large-scale TO among engineering professionals.

Adding to the large compute requirement of TO is the necessary post-processing stage which involves interpreting the elemental density distribution to respect the requirements of the further design stages [12]. The manual intervention needed in post-processing arises from the necessity to translate the implicit, often pixel or voxel-based outputs of density-based topology optimization into smooth, parametric geometries that can be used in conventional CAD software. This step involves dealing with issues such as rough surfaces, unmanageable geometries, and elements that do not align with manufacturing constraints. Engineers need to refine and simplify these geometries while maintaining the integrity and the performance gains of the optimized design, which is a tedious and detail-oriented task. Although feature-mapping methods help the transition to CAD software [13], they show worse convergence performance [14] and density-based methods are usually the workhorse of industrial TO applications.

The scope of this thesis will cover research addressing the following two challenges:

Challenge 1 The high computational cost of running large-scale topology optimization limits its applicability, requiring extensive computational resources that are often only available in specialized research environments

Challenge 2 The post-processing stage of TO is labor-intensive and time-consuming, involving the conversion of pixel or voxel-based optimization results into parametric, manufacturable CAD models, which requires significant manual intervention and refinement by engineers.

1.4 Machine Learning for Topology Optimization

As the advantages of large-scale TO have been demonstrated, research has expanded into areas such as High-Performance Computing (HPC), multi-grid methods, and developing methods like Machine Learning (ML) [15]. The emergence of ML, and especially Deep Learning (DL) [16] in the past decade, as a state-of-the-art tool in areas like Computer Vision (CV) and Natural Language Processing (NLP) [17], has inevitably attracted an emerging research field in ML applied to TO (ML-TO). ML-TO aims to leverage large computing power and utilize past optimization solutions to learn patterns and structures in the data, with the long-term goal of accelerating the TO process and making large-scale TO more accessible and efficient for engineering professionals. The literature on ML applications for TO can be sorted based on its applications in the topology optimization process as is illustrated in Figure 2.4 [18] :

- Reduction: Reparametrized the TO problem using fewer design variables to reduce the size of the design space.
- Direct design: Directly generate an optimal topology from a set of input parameters that describe the TO problem.
- Acceleration: Accelerate some part of the iterative optimization process, for example by using surrogate models for the FEA or sensitivity analysis.
- Post-processing: Models applied on the output of the conventional TO methods to ensure specific design requirements such as manufacturability or interpretability.
- Design diversity: Generate several alternative solutions to the same TO problems.

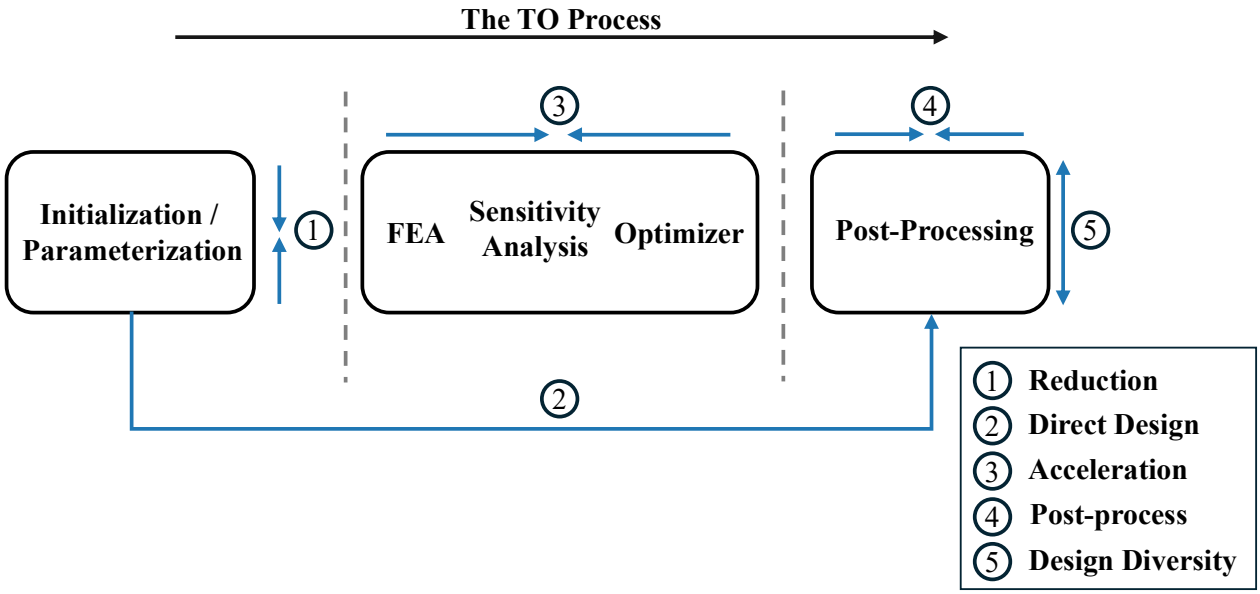


Figure 1.2: Integration of machine learning applications in the topology optimization process.

However, there are three key gaps in current ML-TO literature:

1. **Mesh Dependence of Image-to-Image Approaches:** Most of the research in ML-TO focuses on exploiting the architectural compatibility between the elemental density distribution of density-based TO methods and popular CV models like Convolutional Neural Networks (CNNs). This compatibility arises because the elemental density distribution in TO can be represented as pixels or voxels, similar to the data structures used in CV. This has driven recent studies to explore the application of image-to-image models within a direct design approach. Figure 1.6 demonstrates a typical pipeline involving a U-Net CNN [19], which generates optimal density fields from images that represent the problem parameters. These models are then typically trained via a regression loss like the Mean Squared Error (MSE), to minimize the difference between the predicted density field and a corresponding ground-truth image obtained via conventional TO. Image-to-image approaches enable the use of fully convolutional model architectures [20], which allows the model to run inference on variable image resolutions. The computational complexity of these models increases also with the resolution of the FEA mesh used, growing the memory requirement and the processing time significantly. As the resolution of the images and the corresponding FEA mesh increases, the number of computations required for each layer of

the network grows exponentially, leading to increased demand for GPU memory and computational resources. This can become a bottleneck in real-time applications or when processing very high-resolution data.

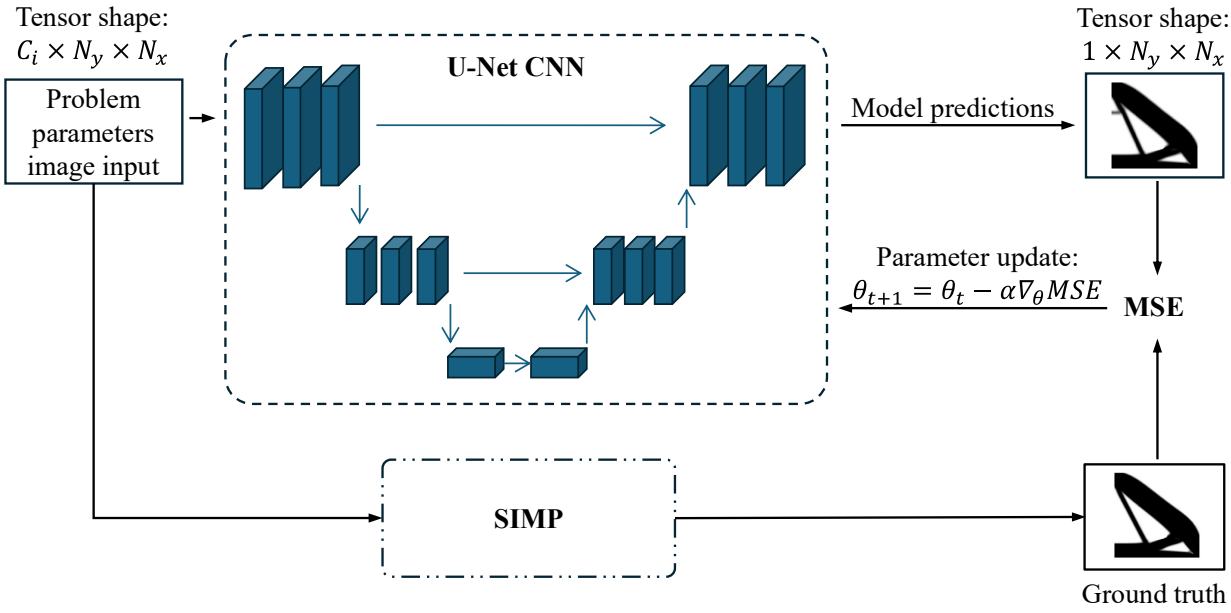


Figure 1.3: Illustration of a U-Net CNN model trained to generate optimal density fields in a direct design, image-to-image approach, with problem parameters represented as images as input. The U-Net CNN is made up of trainable CNN layers with residual connections between the various layers (in blue). The SIMP method is illustrated here as the ground truth sample generation method.

2. **Image similarity loss functions do not optimize for the physics of the TO problem**: This can cause suboptimal design outputs that do not adequately reflect the true physical behavior and constraints of the structures being optimized [18]. The primary focus of image similarity loss functions, such as MSE, is on minimizing the difference between the predicted and actual images in a pixel-wise manner. A few pixels difference can however change drastically the mechanical response of the structure if a load or support condition is disconnected from the structure by example. No literature studies the inclusion of the actual physics in direct design approaches.

3. **Lack of Literature on Post-Processing with ML:** While there has been significant research on using ML to accelerate the TO process itself, there is limited work on applying machine learning for the post-processing of topology optimization results. Recent advances in computer vision and pattern recognition using state-of-the-art ML approaches have shown remarkable performance in interpreting complex data structures. However, these techniques have not yet been extensively explored for automating the interpretation and conversion of TO outputs into manufacturable designs, leaving a gap in leveraging ML's full potential in this context.

1.5 Research Objectives

To address the challenges of high computing costs in TO and the tedious manual work involved in post-processing, this research thesis will seek to answer the following question:

> *How can ML be leveraged to accelerate TO for direct design and post-processing applications?*

From this research question and the three gaps in current ML-TO approaches discussed above, the following research objectives are defined:

Main Objective (MO): Develop scalable ML models for both the direct design generation and post-processing of optimal topologies.

Secondary Objective 1 (SO1): Develop a direct design model with mesh-independent inputs and outputs.

Secondary Objective 2 (SO2): Develop a direct design ML approach that includes the physical objective of TO in the training loss function.

Secondary Objective 3 (SO3): Develop an automatic geometric interpretation tool of elemental density distributions.

1.6 Thesis Structure and Overview

- Chapter 2 presents the literature review, mapping the current knowledge frontier in ML-TO for direct design and post-processing applications, highlighting the research gaps that form the basis of the three articles included in this thesis.
- Chapter 3 summarizes the methods and research results presented in the articles, providing an overview of the approaches and key findings that contribute to the thesis.
- Chapter 4 details the development of a supervised learning approach based on the MMC framework and includes a study of the performance of this approach as a direct design and acceleration application, demonstrating its potential and limitations.
- Chapter 5 introduces an open-source deep reinforcement learning environment designed for the development of autonomous agents optimized to enhance physical feedback while integrating the TO objective, offering a novel approach to the optimization process.
- Chapter 6 frames the task of post-processing optimal discretized density fields as a reverse-engineering problem, similar to instance segmentation in CV, and presents the development of a CNN-based automatic algorithm to extract geometry from black-and-white topologies, aiming to streamline the post-processing stage.
- Chapter 7 provides a general discussion on ML-TO, synthesizing insights from the research and discussing the broader implications and contributions of the work.
- Chapter 8 offers insights into the scientific importance of the research, its limitations, and potential future directions, reflecting on the significance of the findings and suggesting areas for further investigation and development.

CHAPTER 2 Literature Review

In the mid-2010s, a significant increase in public dataset sizes and access to high-performance computing hardware led to incredible advances in the field of DL [21]. DL models have become state-of-the-art in fields such as CV [22], NLP [17, 23], and search engines [24]. Building on these successes, researchers have begun exploring the potential of DL in other areas, including the nascent field of ML-TO, which began attracting interest around 2018, as illustrated in Figure 2.1.

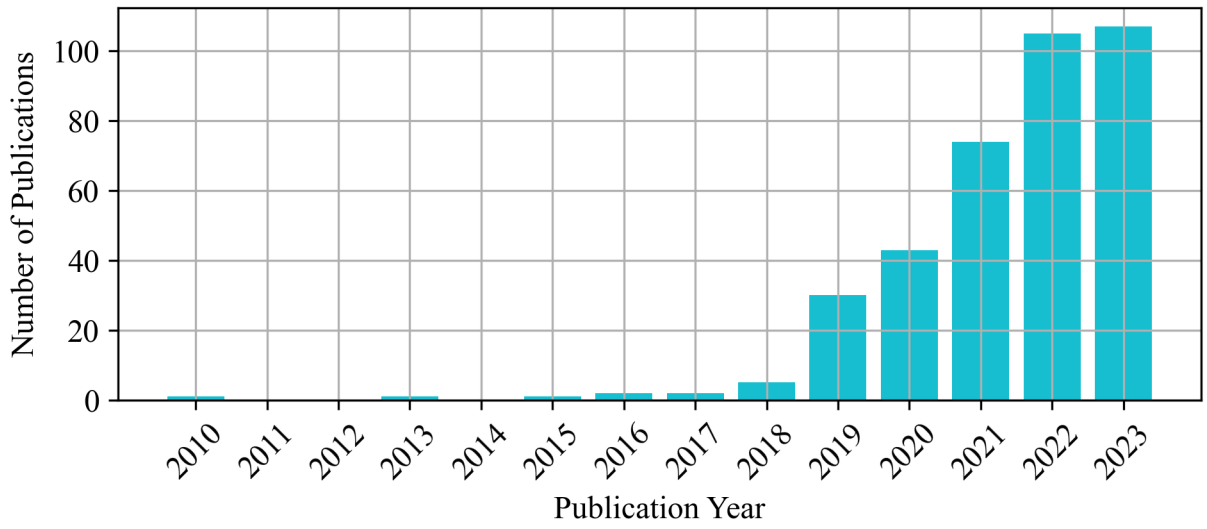


Figure 2.1: Number of publications per year from 2010 to 2023 on *Web of Science* returned by a 'All fields' search for the terms 'topology optimization' AND ('machine learning' OR 'deep learning'). *Certain data included herein are derived from Clarivate™ (Web of Science™). © Clarivate 2024. All rights reserved.*

This surge in research interest coincides with several highly publicized breakthroughs that continue to fuel today's wave of DL research and development. In 2016, DeepMind's AlphaGo beat the world's best Go player [25], marking the first time a computer achieved this feat. In 2017, the transformer architecture was introduced [17], followed by the first version of the Generative Pretrained Transformer (GPT) in 2018 [23], which sparked the Large Language Model (LLM) frenzy of recent years.

This growing attention to ML-TO has however resulted in numerous poorly designed experiments and overhyped results, often exaggerating the current capabilities of ML-TO. A significant challenge in this field is the lack of large datasets of optimized structures, in contrast to other fields

where DL has thrived due to the availability of extensive datasets. This scarcity has led to many early results demonstrating poor performance and limited generality across the diverse problems encountered in TO. Despite these challenges, ML has a proven track record of exceeding expectations and pushing the boundaries of what is possible in many areas, and there is a lot of research interest aiming to show that TO will not be an exception to the rule.

This literature review aims to explore the promising research areas of ML-TO. The review begins by presenting the two TO algorithms used in the subsequent chapters: the Solid Isotropic Material with Penalization (SIMP), which is the most used TO method, and the Moving Morphable Components (MMC) method. Following this, the current literature on the Post-Processing of TO (PP-TO) is examined. The review then delves into the ML literature specific to TO, covering various approaches such as image-based regression, generative modeling, physics-informed methods, and reinforcement learning and its application to TO. Finally, the role of ML in PP-TO is addressed, highlighting the potential for future advancements and innovative solutions in this evolving field. A summary of the research state-of-the-art and the current limitations will conclude this section.

2.1 Topology Optimization Algorithms

TO algorithms aim to find the best distribution of material inside a given domain and under specific design constraints. A classification can be made based on how the assembly of material is represented. The parameters defining this assembly correspond to the optimization variables and will be referred to as the design variables.

The most prevalent family of TO algorithms are the so-called density-based methods [26], which represents the material distribution through the individual densities of the FEM mesh. The SIMP method is the most widely used TO approach.

A drawback of density-based methods is that they offer an implicit description of the structure, representing the geometry as an assembly of elemental densities. This often requires interpretation when the design needs to be constructed using predefined stock components or when design iterations necessitate direct integration with CAD software. To address this limitation, feature-mapping methods propose to represent the topology using geometric design variables defining high-level geometric features which are then mapped onto the FEM mesh [27]. One of the most widely used feature-mapping methods is the MMC method [28], which proposes to use

hyperelliptic functions defined by geometric design variables such as coordinates, orientation, length, and thicknesses.

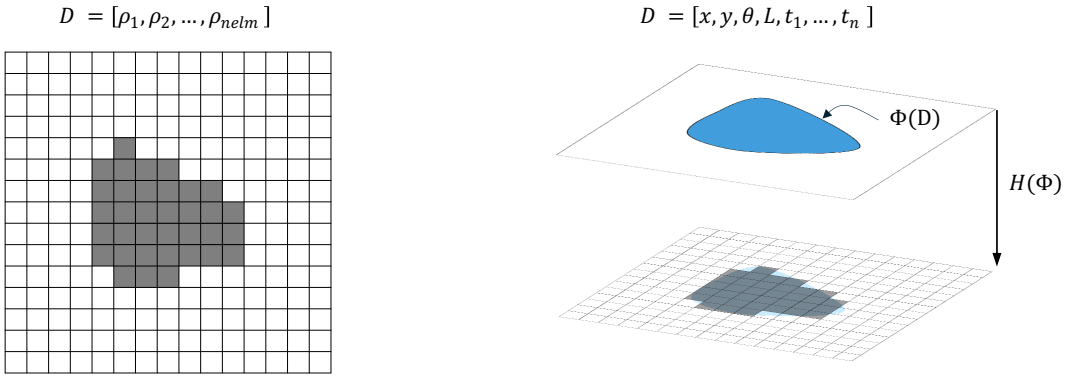


Figure 2.2: Comparison of density-based methods representation of the structure (left) vs feature-mapping methods (right).

2.1.1 Solid Isotropic Material with Penalization

The SIMP method [29] parameterizes the topology by assigning a continuous density value between 0 (void) and 1 (material) to each mesh element. To push solutions toward binary material distribution, a power law formulation is used to represent the material property. For example, for a linear elasticity problem, the Young's modulus of each element is formulated as:

$$E(\rho_i) = E_{min} + \rho_i^p (E_0 - E_{min}) \tag{2.1}$$

Where $E(\rho_i)$ is the continuous value of the Young's modulus assigned to the i^{th} element with density ρ_i, E_0 is the Young's modulus of an isotropic material, E_{min} is a small positive value ($E_{min} \in [10^{-9}, 10^{-6}]$) to avoid numerical errors when dealing with void densities, and p is the penalization parameter. For a typical compliance minimization problem under volume constraint, the optimization problem described in Equation 1.1 becomes:

$$\begin{aligned} \min_{\rho} \quad & : \quad F^T U \\ s.t. \quad & : \quad \sum_{e=1}^{N} v_e \rho_e = v^T \rho \leq V^* \\ & : \quad 0 \leq \rho \leq 1 \\ & : \quad K(\rho) U = F \end{aligned} \tag{2.2}$$

Where $F^T U$ is the structural compliance, $\sum_{e=1}^{N} v_e \rho_e$ is the global volume of the N elements, and V^* is the volume constraint. When using SIMP, one will choose values of $p > 1$ to make intermediate densities contribute less stiffness compared to the volume they occupy. A value of $p = 3$ is typically used in practice.

The SIMP method is the most widely used TO algorithm, deployed in most CAD software [7] due to its simplicity of implementation [30-32], and the rise of additive manufacturing (AM) which allows to leverage the complexity of the design space that can be explored by TO [33]. The freedom allowed by AM however is not free of manufacturing constraints [34]. Overhang [35] and minimum length-scale [36] constraints can be integrated directly into the optimization process, removing the need to attend to these in a post-processing step. Multi-physics applications that can be effectively discretized with numerical methods like FEA can also be approached using the SIMP method. Heat transfer [37], fluid problems [38], and electromagnetism [39] are examples of popular applications.

2.1.2 Method of Moving Morphable Components

The MMC feature-mapping method parameterizes the topology using an assembly of Topology Description Functions (TDF). The i^{th} component of the topology can be defined by many shapes and functions, but in this work, a 2D linearly varying thickness component is used, as illustrated in Figure 2.3.

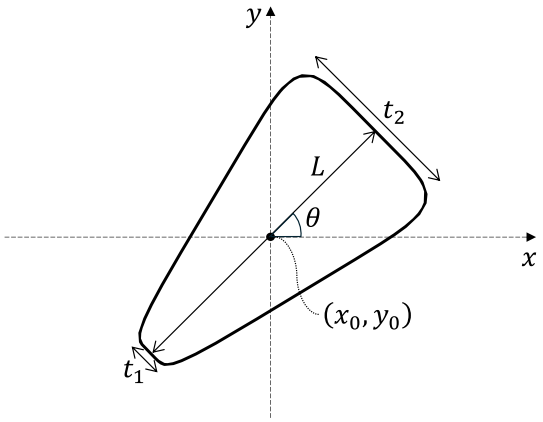


Figure 2.3 : Illustration of an MMC component with linearly varying thickness. Variables t_1 and t_2 represent the two thicknesses. x_0 and y_0 are the centroid coordinates. L and θ are the length and orientation of the component respectively.

The component is defined by a hyperelliptic TDF defined by the design variables $d = (x_0, y_0, L, \theta, t_1, t_2)$. The TDF is presented in Equation 2.3, and an illustration of the function is present in Figure 2.4a).

$$\phi_i(x,y) = 1 - \left(\left(\frac{x'}{\frac{L}{2}} \right)^6 + \left(\frac{y'}{t(x')} \right)^6 \right) \tag{2.3}$$

Where $t(x')$ is the variable thickness profile of the component, which for a linearly varying thickness is:

$$t(x') = \frac{t_1 + t_2}{2} + \frac{t_2 - t_1}{2L} x' \tag{2.4}$$

$$x' = \cos(\theta)\,(x - x_0) + \sin(\theta)(y - y_0) \tag{2.5}$$

$$y' = -\sin(\theta)\,(x - x_0) + \cos(\theta)\,(y - y_0) \tag{2.6}$$

To define the subset of a design domain D occupied by a single component material (Ω_i), we define the 0-th level as the boundary of the hyperelliptic TDF. This level is illustrated in Figure 2.4b):

$$\begin{cases} \phi_i(x) > 0, & if \quad x \in \Omega_i \\ \phi_i(x) = 0, & if \quad x \in \partial\Omega_i \\ \phi_i(x) < 0, & if \quad x \in D \backslash \Omega_i \end{cases} \tag{2.7}$$

A regularized Heaviside projection is used to obtain a binary material distribution, as illustrated in Figure 2.4c), according to the following equation:

$$Acad \tag{2.8}$$

Where α is a small positive value to avoid numerical singularities, and ϵ is a regularization parameter. The material property of the underlying FEA mesh is obtained from the TDF and corresponding Heaviside projection of the 4 elemental nodes of the element e ($H(\phi_i^e)$) using the ersatz material model described in Equation 2.9:

$$E^e = \frac{E\left(\sum_{i=1}^{4}\left(H(\phi_i^e)\right)^2\right)}{4} \tag{2.9}$$

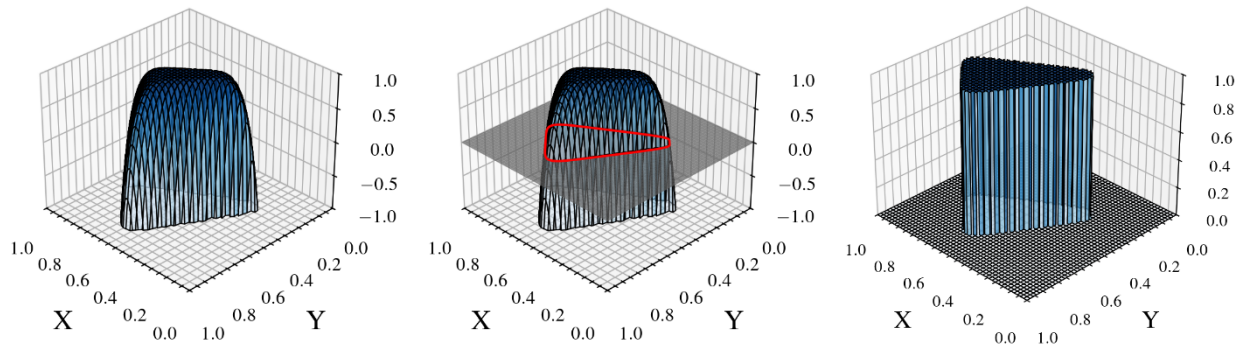


Figure 2.4 : Illustration of a linearly varying MMC component. a) The hyperelliptic TDF. b) the level-set operation defining the material domain based on the 0 level. c) Heaviside projection on the density space

The optimization problem of Equation 1.1 becomes:

$$\begin{aligned} \min_{d} \quad &: \quad \int_D H\big(\phi^s(x;d)\big)\, F\ U\ dV \\ s.t. \quad &: \quad \int_D H\big(\phi^s(x;d)\big) dV \leq V^* \\ &: \quad d \subset u_d \\ &: \quad K(d)U = F \end{aligned} \tag{2.10}$$

Where u_d is the admissible set that d belongs to and can be defined by constraints on the design variables such as minimum and maximum thicknesses.

Despite the limited literature on the MMC method due to its relatively recent emergence in 2016, notable advancements have been made, including its application to three-dimensional problems [40] and the development of multi-resolution approaches [41]. For an overview of the current research on MMC, refer to [42].

2.2 Post-Processing of Optimal Structures

PP-TO aims at modifying the output of optimal density distribution to make it compatible with further applications in the design process: either prototyping, design analysis, or design exploration. For prototyping, post-processing methods ensure that the optimized design meets various fabrication constraints, such as those imposed by AM or traditional manufacturing techniques. In design analysis, the goal is to convert the often simplified meshes used in TO into high-quality meshes suitable for accurate numerical simulation, as presented in Figure 2.5a). For

design exploration, post-processing involves transforming the optimal density distribution into parametric geometries, allowing engineers to interact with and modify the design, usually within CAD software, facilitating iterative design improvements [12]. An example of this is illustrated in Figure 2.5b).

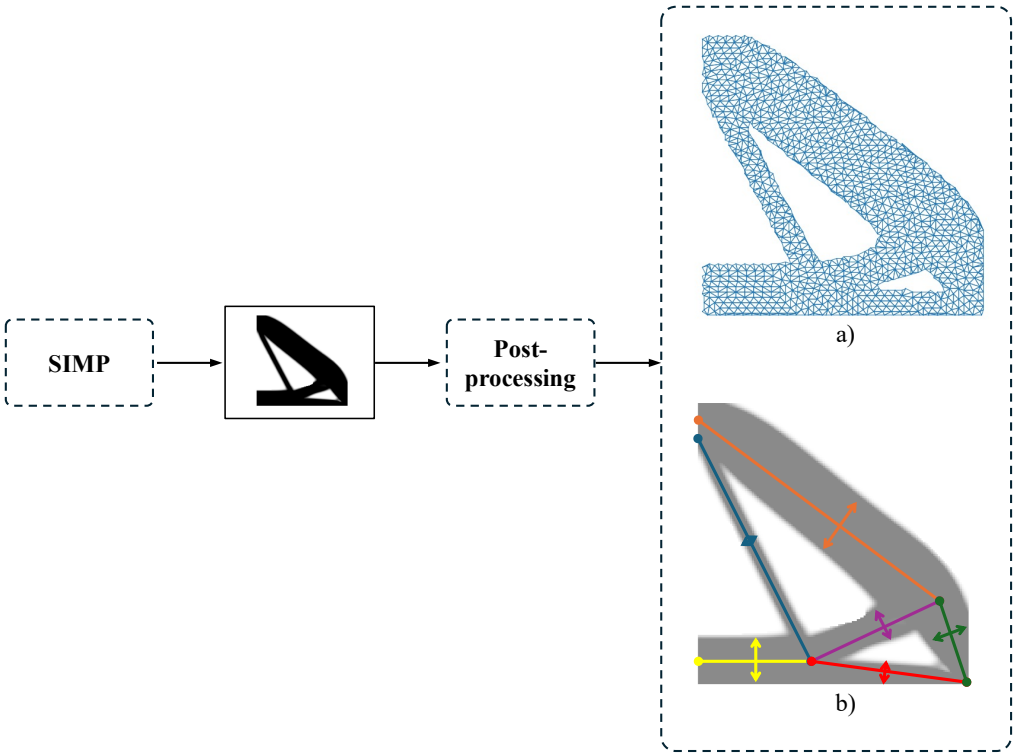


Figure 2.5: Illustration of post-processing of TO outputs from a SIMP method. a) Remeshing technique. b) Parametric skeletonization extraction converts the topology into a high-level skeleton representation.

Most of the literature on PP-TO focuses on constrained optimization, addressing the various manufacturing requirements directly within the optimization process. These constraints typically include fabrication limitations such as minimum feature sizes, overhang angles in AM, and material properties, ensuring that the optimized designs are practical and manufacturable. Similarly, design exploration can be enhanced through constrained optimization by parameterizing the design variables using higher-level geometries, as seen in feature-mapping methods. However, incorporating additional constraints into the optimization process can limit design freedom and increase the computational cost of TO.

The simplest post-processing method for density-based TO for CAD-compatible structure representation is a 1D skeletonization-based method, where typically the central axis of each skeleton branch is estimated and then a cross-section algorithm is used to reconstruct each branch [43]. Boundary condition information can also be included in the skeletonization extraction process to ensure the interpreted structure respects the problem parameters [44]. For a thorough survey of skeletonization methods, the readers are referred to [45].

Skeletonization however only works on beam-like structures, usually the outcome of low volume fraction constraints. For more complex topologies, surface-based methods aim at generating 2D smooth, high-quality meshes that can be manipulated and interacted with via commercial engineering software. This includes re-meshing techniques [46] of noisy meshes, surface-fitting where parametric surfaces like Basis-splines (B-spline) and Non-Uniform Rational Basis Spline (NURBS) [47, 48] are fitted to the boundaries of the material distribution, or subdivision surfaces [49, 50]. These surfaces allow for post-processing of arbitrary complexity but are defined via control points on the respective topology boundaries, which does not allow for intuitive control or modifications by the engineer.

Engineers interact with CAD software via parametric models defined by dimensions and geometrical operations typically represented as Constructive Solid Geometry (CSG) trees. Literature on parametric CAD model extraction from TO outputs is small and mostly focuses on skeletonization-based approaches for beam-like structures [51, 52], or geometries capable of being approximated via swept-volumes [53].

2.3 Machine Learning for Topology Optimization

The challenge of accelerating TO can be addressed through multiple ML approaches. This section explores the various strategies proposed, highlighting image-based regression learning, generative modeling, physics-informed methods, reinforcement learning, and ML for PP-TO.

2.3.1 Image-Based Regression Learning

As can be observed in Figure 2.6, intermediate solutions during the iterative updates of conventional TO can provide a lot of information about the final black-and-white structure. CNN models can be utilized to predict the final topology from intermediate solutions [54-58] to minimize the error between the predicted structure and the final optimal one. Using fully convolutional CNN architectures allows a trained model to be used for inference on density fields of various resolutions without retraining. These 'accelerated convergence' methods assume that the information from the N^{th} intermediate iteration is sufficient to extrapolate to the final optimal structure. However, this assumption and the value of N are highly problem-dependent and will fail for more complex problems where sub-structures can shift in later iterations. Super-resolution approaches where low-resolution outputs of conventional TO are used to predict a higher-resolution structure are also trained in a similar fashion, with intermediate or final density fields as their input [41]. Again, these methods assume that low-resolution density fields show enough information to predict accurately the equivalent structure at higher resolution.

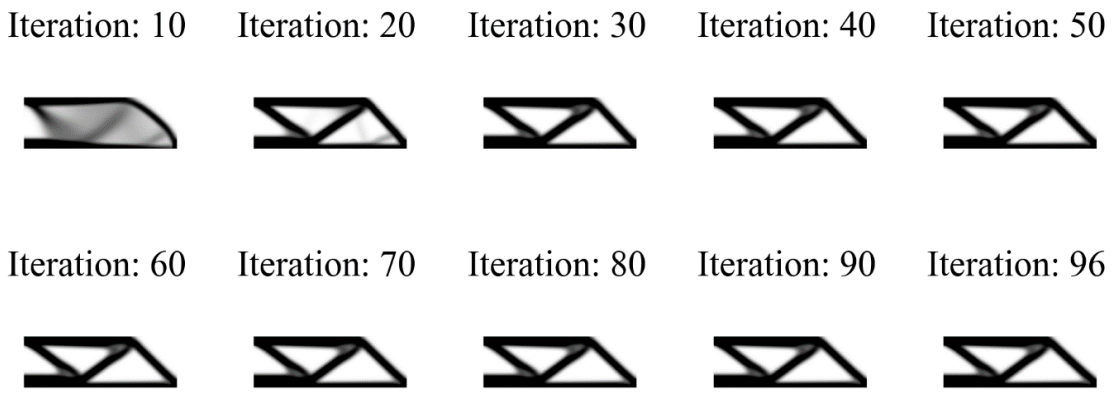


Figure 2.6: Intermediate and final iterations of the classic cantilever beam optimization obtained from the 88-line Matlab code [59] with the command *top88(240,80,0.5,3.0,7,1)*.

Ideally, a direct design approach would enable predictions of the optimal structure from a representation of the boundary conditions and other parameters defining the TO problem using a single forward pass through the model. Like ‘accelerated convergence’ approaches, a training dataset of optimal topologies is required and must first be generated using conventional optimization such as SIMP. To keep the resolution invariance property of fully convolutional networks, the problem parameters are then formatted into multi-channel images, and an image model is trained to recreate the corresponding optimal topologies using a regression loss such as a MSE [60-62]. This type of approach is illustrated in Figure 2.7.

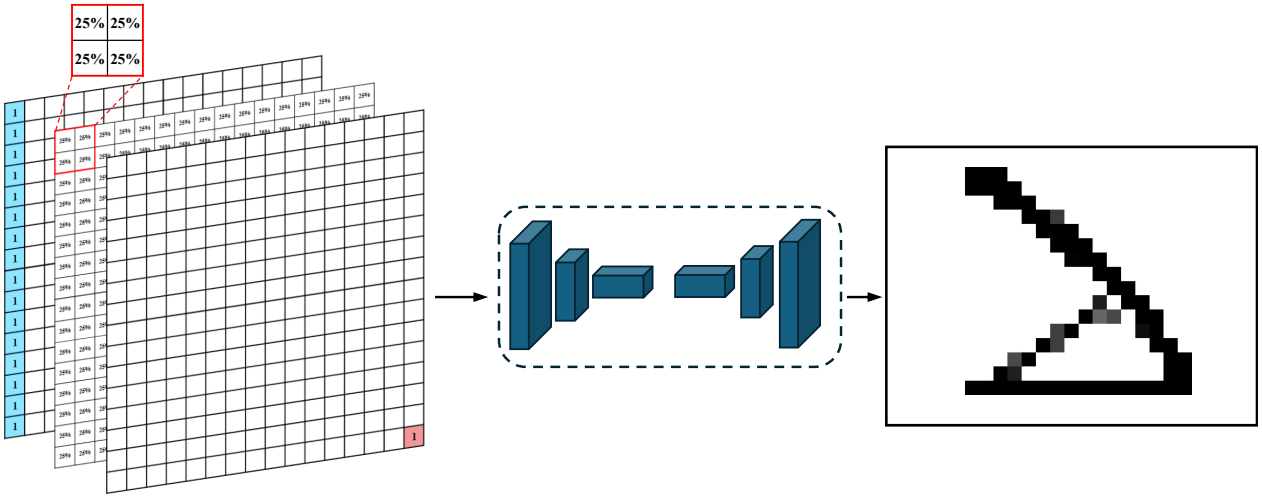


Figure 2.7: Fully convolutional direct design approach. The input is a multi-channel image where the first channel represents the support condition, the second channel represents the desired volume fraction, and the last channel represents the location of a unit downward load.

Representing boundary conditions as images means that the input is very sparse, and a lot of computation must be spent on domain regions with no load or support. Using the output of a FEA from a full material distribution is a solution to represent the problem with better information, at the cost of computing the additional physical fields [63]. Various state-of-the-art CV CNN models have been applied to direct design TO problems, such as Unets [64], pyramid networks [65], attention-based models [66]. While image-based direct design approaches are the most common approaches seen in literature, pure image similarity training brings several limitations:

- Direct design regression models generate deterministic outputs meaning that visually different designs for the same input fields cannot be created.
- The trained models are directly dependent on the FEA resolution they have been trained on. Although fully convolutional architectures are commonly used, there is no research investigating the generalization capacity of these architectures to many resolutions [18].
- Most importantly, the image similarity optimization objective does not consider the physics of the problems. This leads to the very common issue with direct design ML approaches of generating discontinued structures, where the load path is broken by a few pixels which are incorrectly assigned with a void density. This failure mode is widely reported in [60, 67, 68].

2.3.2 Generative Modeling Approaches

After the introduction of generative models for image generation like the Generative Adversarial Networks (GAN) in 2014 [69] and now the recent diffusion models [70], researchers have explored the usage of such generative approaches to generate optimal topologies. These two training approaches are illustrated in Figure 2.8. To take into account the problem parameters defining the TO problem, Conditional GANs (cGANs) [71] generate designs conditioned on these parameters, allowing for context-specific design generation. While simple conditional vectors containing the desired volume fractions and additional 1D information for the boundary conditions have been used [72], physical fields such as displacement and stress have outperformed these approaches, and they also enable the usage of fully convolutional generators [73]. Conditioned diffusion models based on the physical fields have been shown to outperform cGANs on direct design tasks, producing lower compliance errors and fewer disconnected structures, at the expense of a 360x increase in computation time [74]. To tackle the issue of disconnected structures, state-of-the-art generative approaches include topological information in the loss function to penalize structures with a disconnected load path [74, 75]. The advantage of generative modeling for TO is the ability to generate variations of output for the same input. An example of generated structures by TopoDiff for one load case is shown in Figure 2.9.

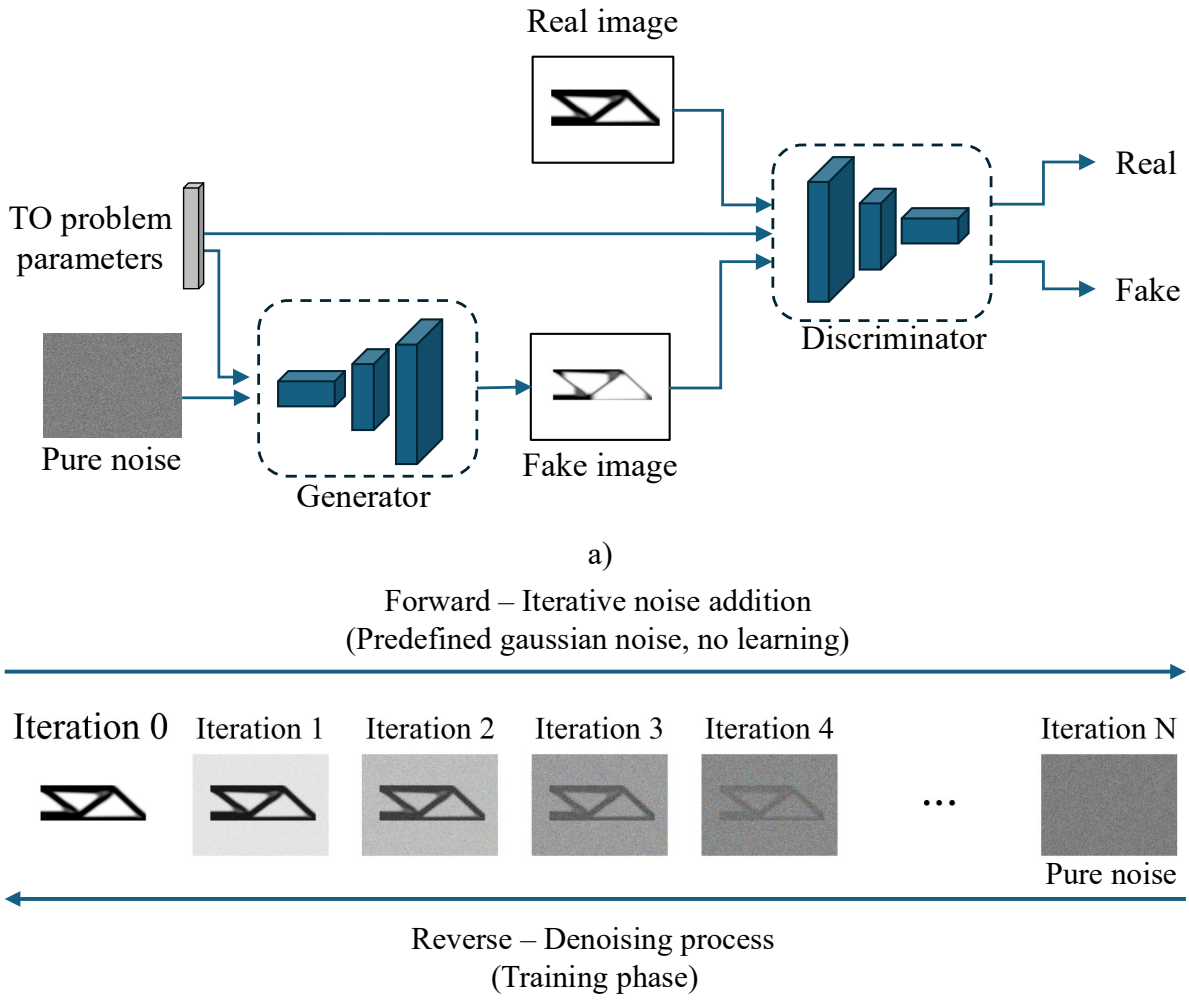


Figure 2.8: Overview of generative approaches for TO. a) Conditional GAN: The generator creates images from noise and problem parameters to maximize the discriminator error. b) Diffusion model: The forward process adds noise, and the reverse process iteratively removes it to reconstruct the original.



a) b) c)

Figure 2.9: Three samples of generated structures for a single random load case obtained via TopoDiff [74].

2.3.3 Machine Learning for Feature-Mapping Methods

Leveraging feature-mapping methods in ML-TO allows direct design models to generate design variable outputs decoupled from the FEA mesh. An early demonstration of this involved using a Support Vector Regression (SVR) model to generate a vector of MMC design variables based on the position of a single load on a 2x1 cantilever beam [76]. This model's prediction performance was subpar, but it was suggested that it could serve as an initial solution generator, potentially reducing the conventional optimization time by 90%.

Unfortunately, the quality of research applying neural networks to the MMC method has been consistently poor, mirroring the broader issues in the ML-TO literature. Most studies rely on fully convolutional architecture and both image similarity [77-79] and generative modeling approaches [80] have been explored to generate images of optimal structures trained on optimal structures obtained via conventional MMC optimization. This approach undermines the advantages of feature-mapping methods, as it fails to utilize the parametric design variables representing the higher-level geometric components. Additionally, experiments in this subfield are often inadequately designed and reported. For instance, all reviewed papers mentioned above focus exclusively on a 2 x 1 cantilever beam, varying only the position of a single unit load on the right boundary under a fixed volume constraint. Some studies neglect to even specify the model architecture or input composition.

This lack of rigour leaves many questions unanswered about the potential of feature-mapping methods in ML-TO. The narrow focus and poor experimental quality limit the generalizability and

applicability of the findings, highlighting a gap in the research that needs to be addressed to fully explore the benefits of mesh-independent feature-mapping methods in ML-TO.

2.3.4 'Physics-Informed' Approaches

The issue of disconnected structures has brought a lot of research into the integration of the physics of the TO problem in the training objective of supervised learning models with the goal of penalizing model predictions which are 'non-physical'. One simple solution, if one assumes that the model-predicted structure is close to optimal or is in a 'desirable' region of the design space, is to do a few iterations of conventional optimization using the predicted design as a starting point. Using just 5 SIMP iterations after inference from a trained CNN managed to completely solve the disconnection issue while maintaining a healthy 5x time reduction over conventional TO [81].

Another method is to switch from a regression training paradigm to a self-supervised training pipeline where the loss function becomes the TO objective directly as illustrated in the loss function from [82]:

$$L(\theta) = \frac{u^T K u}{J^0} + \alpha \left(\frac{\sum_e \rho_e v_e}{V^*} - 1 \right)^2 \tag{2.11}$$

Where θ are the model parameters, u and K are the displacement and stiffness matrices, $\sum_e \rho_e v_e$ is the total volume taken by the individual elements, V^* is the desired volume constraint, J^0 is a compliance scaling constant, and α is a parameter balancing the impact of the physical term with the design constraint term. Two issues arise from such an approach. The first challenge is that the implementation of FEA needs to be done in auto-differentiation software. Although this can be currently achieved using fast auto-differentiation software like Jax with dense solvers [83], the major deep learning development libraries do not officially support sparse solvers, which is necessary for this kind of problems. The second issue is that the physical term and the design constraints need to be carefully balanced in the loss function to obtain stable results. Choosing a value for J^0 and α that are reasonable for a wide range of boundary conditions is a non-trivial problem. These issues have severely limited the scope of direct TO optimization using neural networks, and such methods have only been used in iterative processes in a similar fashion to conventional optimization.

2.3.5 Reinforcement Learning

The task of engineering design can be viewed as a decision-making process, where engineers need to decide among a set of options, within a fixed allocation of resources, all within the conditions of uncertainty and risk [84]. Furthermore, the dimensionality of an engineering design task is so large, that exploring every possible design choice is infeasible or economically impractical.

The ML methods discussed so far in this thesis all belong to the field of supervised learning, where labeled samples are provided to the model, which for an engineering design task, would correspond to the expected design decision. Collecting such samples often requires expensive physics simulation which makes it difficult to collect a representative set of behaviors for the large design space of the engineering task. Reinforcement Learning (RL) instead focuses on "computational methods for learning from interaction" [85]. It focuses on training a learner model through trial and error to maximize a reward signal.

A RL task can be idealized in the form of a Markov Decision Process (MDP), where an agent interacts with an environment through a series of timesteps, receiving a representation of the environment called the state, and selecting an action that will cause the environment to update its state, and send a reward evaluating the 'goodness' of the action. This interaction illustrated in Figure 2.10 takes the form of the following trajectory:

$$\tau = (S_0, A_0, R_1, S_1, A_1, \dots, S_t, A_t, R_t) \tag{2.12}$$

Where S_t represents the state of the environment at timestep t, A_t represents the action taken by the agent, and R_t represents the reward received after taking the action. The 'brain' of an RL agent is a policy that parameterizes a probability distribution of taking actions knowing the state of the environment:

$$\pi(s, a) = P\ (A_t = a | S_t = s) \tag{2.13}$$

The goal of the agent is to learn a policy that maximizes the expected return for a given trajectory, denoted $R(\tau)$, which is the sum of rewards obtained along this trajectory. Therefore, the optimization problem in RL can be expressed as:

$$J(\pi) = E[R(\tau)] \tag{2.14}$$

$$\pi^* = argmax\ J(\pi) \tag{2.15}$$

Where $J(\pi)$ is the expected return over the trajectory when acting according to the policy, and $\pi^*$is the optimal policy which is the goal of the optimization problem.

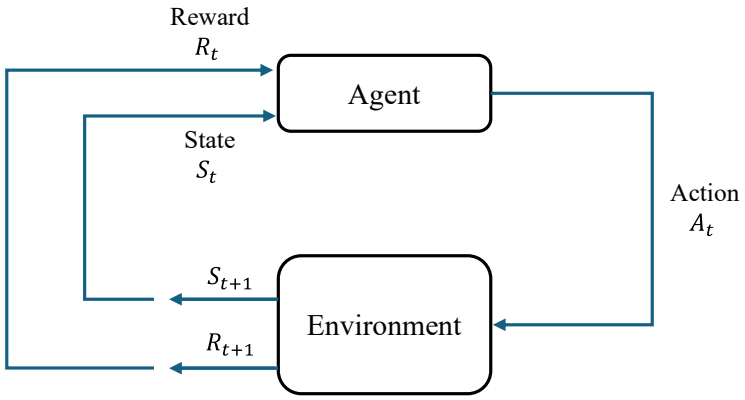


Figure 2.10 : Illustration of the interaction between the agent and environment in a MDP [85].

A fundamental concept of RL is the Markov Property, which states that the future state of an environment depends only on the current state and the action taken, not on the sequence of events that preceded it. This can be expressed as:

$$P(s_{t+1}, R_{t+1} | s_t, r_t, s_{t-1}, r_{t-1}, \dots) = P(s_{t+1}, R_{t+1} | s_t, r_t) \quad (2.16)$$

The Markov Property provides the intuition that all the information relevant to decision-making is contained in the state representation and is a basis of RL where decisions and values are typically assumed to be a function of the current state only. In practice, most modern research in RL deals with intractably large state spaces, and with an incomplete description of a state, requiring the usage of function approximators to learn a good approximation of the optimal behaviour. Deep Reinforcement Learning (DRL) refers to the family of methods leveraging DL models to approximate various parts of the RL optimization problem, such as the policy or the value of a state. The field of DRL has led to impressive feats of generalization, defeating best-of-the-world players in the games of Go [25], chess [86] and the real-time strategy game StarCraft II [87] or multiplayer game Dota II [88].

DRL algorithms can be classified based on whether the agent learns a model of the environment (model-based RL) which predicts the transition of the environment, or if they do not (model-free RL). Model-free RL either takes the shape of policy optimization or Q-learning.

Q-learning aims to learn an approximator of the optimal action-value function $Q^*(s,a)$:

$$Q^{\pi}(s,a) = E[R(\tau)|s_0 = s, a_0 = a] \tag{2.17}$$

And then choosing the action according to:

$$a(s) = argmax\ Q_{\theta}(s,a) \tag{2.18}$$

The development of the Deep Q-Network (DQN) was a pivotal moment for DRL, demonstrating human-level performance on a suite of Atari 2600 games [89]. This Q-learning algorithm revolutionized the field by integrating deep neural networks with Q-learning, enabling it to handle high-dimensional sensory inputs such as raw pixels from video games. The DQN approach showcased the potential of DL to manage complex, high-dimensional state spaces effectively, leading to significant advancements in the capabilities of RL algorithms.

Policy Optimization seeks to optimize the policy parameters θ to maximize the objective of equation 2.14 via the update:

$$\theta_{new} = \theta_{old} + \alpha \nabla_{\theta} J(\pi_{\theta}) \tag{2.19}$$

The policy gradient ($\nabla_{\theta} J$) is expressed as:

$$\nabla_{\theta} J(\pi_{\theta}) = E\left[\sum_{t=0}^{T} \nabla_{\theta} \log \pi_{\theta}(a_t|s_t)\, R(\tau)\right] \tag{2.20}$$

Since this is an expectation, it can be estimated with a sample mean from a number (D) of collected trajectories obtained from interactions between the agent and the environment:

$$\nabla_{\theta} J(\pi_{\theta}) \approx \frac{1}{D} \sum_{\tau \in D} \sum_{t=0}^{T} \nabla_{\theta} \log \pi_{\theta}(a_t|s_t)\, R(\tau) \tag{2.21}$$

The policy gradient update therefore focuses on increasing the probability of action that led to high returns and decreasing the probability of taking actions that led to worse-than-expected returns.

Model-based RL can either include the task of learning a world model or have it given intrinsically as is the case for games like chess and Go where the dynamics of the world are known clearly in advance and algorithms like Monte-Carlo Tree Search can be used to plan [90]. In a lot of use cases, a ground truth model of the environment is not available, and instead, the agent must learn it alongside its policy. A popular method of doing so is to augment the training of model-free

algorithms using fictitious experiences obtained from a learned world model [91]. Model-based methods exhibit higher sample efficiency than model-free methods on tasks where an accurate world model can be learned [92]. Some tasks can however be more difficult to learn an accurate model of the environmental dynamics, compared to learning an effective policy, and in such cases, agents suffer from world model bias [93].

This thesis uses the model-free Proximal Policy Optimization (PPO) [94] and DreamerV3 [95] model-based algorithms in Chapter 5.

Proximal Policy Optimization

Due to its simplicity of implementation and robustness, the most widely used DRL algorithm today is the Proximal Policy Optimization (PPO). PPO is based on an actor-critic architecture, which involves two neural networks: the actor (policy) network and the critic (value) network. The actor network determines the actions to take, while the critic network evaluates these actions by estimating the value function. The value function represents the expected return (sum of future rewards) from state s under the current policy:

$$V^{\pi}(s) = E_{\pi}[R(\tau)|s_0 = s] \tag{2.22}$$

The key innovation of PPO is the use of a clipped surrogate objective function that restricts the policy update to be within a specified range, thus preventing large updates that could destabilize training. This clipping mechanism ensures that the policy does not change too much in a single update, which helps in maintaining stable learning and improving sample efficiency.

$$L^{CLIP}(\theta) = \hat{E}_t\left[\min\left(\frac{\pi_{old}(a_t|s_t)}{\pi_{\theta}(a_t|s_t)}\hat{A}_t, clip\left(\frac{\pi_{old}(a_t|s_t)}{\pi_{\theta}(a_t|s_t)}, 1-\epsilon, 1+\epsilon\right)\hat{A}_t\right)\right] \tag{2.23}$$

Where $\frac{\pi_{old}}{\pi_{\theta}}$ is the probability ratio between the current and old policies, ϵ is the clipping parameter which limit the extent of policy updates and $\hat{A}_t$ is the advantage estimate, which represents how much better or worse the taken action $a_t$was compared to the average action at state s_t. The advantage function $\hat{A}_t$ is estimated using the Generalized Advantage Estimation (GAE) as described in Equation 2.24:

$$\hat{A}_t = \sum_{k=0}^{\infty}(\gamma\lambda)^k(\delta_{t+k}) \tag{2.24}$$

where γ is the discount factor, λ is a parameter that controls the bias-variance tradeoff in advantage estimation, and δ_{t+k} is the Temporal Difference (TD) residual at time t+k, defined in Equation 2.25:

$$\delta_{t+k} = r_{t+k} + \gamma V(s_{t+k+1}) - V(s_{t+k}) \tag{2.25}$$

Here, $V(s)$ is the critic's estimated value of state s. The critic network updates its value function by minimizing these TD residuals, providing more accurate feedback to guide the actor's policy updates.

The actor network, which represents the policy π_θ, is updated by maximizing the clipped surrogate objective using gradient ascent as defined in:

$$\theta_{new} = \theta_{old} + \alpha \nabla_\theta L^{CLIP}(\theta) \tag{2.26}$$

Where α is the learning rate. This update encourages the actor to take actions that lead to higher advantages while ensuring that the updated policy does not deviate too far from the previous one.

DreamerV3

DreamerV3 is designed to solve a diverse set of tasks across multiple domains using a single configuration of hyperparameters. It achieves this through an actor-critic framework combined with a world model to imagine future scenarios and improve its policy based on these "imagined" outcomes. The actor-critic learning is done using the world model by generating imagined trajectories of states and rewards. The actor network learns to choose actions that maximize the expected return of these trajectories, while the critic network predicts the value of the imagined states. Both the actor and critic are trained concurrently from the trajectories provided by the world model, enabling robust and efficient learning across diverse tasks. The integration of the actor-critic framework with the world model allows DreamerV3 to optimize performance effectively across various domains.

The world model is implemented as a Recurrent State-Space Model (RSSM), as illustrated in Figure 2.11 and composed of the following components:

- Encoder: maps inputs (x_t) to stochastic discrete representation (z_t).

$$z_t \sim q_\phi(z_t | x_t) \tag{2.27}$$

- Sequence model: uses recurrent state (h_t) to predict sequences of representations given past actions (a_{t-1}), recurrent state (h_{t-1}) and latent representations (z_{t-1}).

$$h_t = f_\phi(h_{t-1}, z_{t-1}, a_{t-1}) \tag{2.28}$$

- Predictors: predict future rewards ($\hat{r}_t$), episode continuation flags ($\hat{c}_t$) and latent representations ($\hat{z}_t$).

$$\hat{z}_t \sim p_\phi(\hat{z}_t | h_t) \tag{2.29}$$

$$\hat{r}_t \sim p_\phi(\hat{r}_t | h_t, z_t) \tag{2.30}$$

$$\hat{c}_t \sim p_\phi(\hat{c}_t | h_t, z_t) \tag{2.31}$$

- Decoder: Reconstructs the original inputs from the latent representations to ensure that the latent states learned by the model are informative and retain essential information about the environment.

$$\hat{x}_t \sim p_\phi(\hat{x}_t | h_t, z_t) \tag{2.32}$$

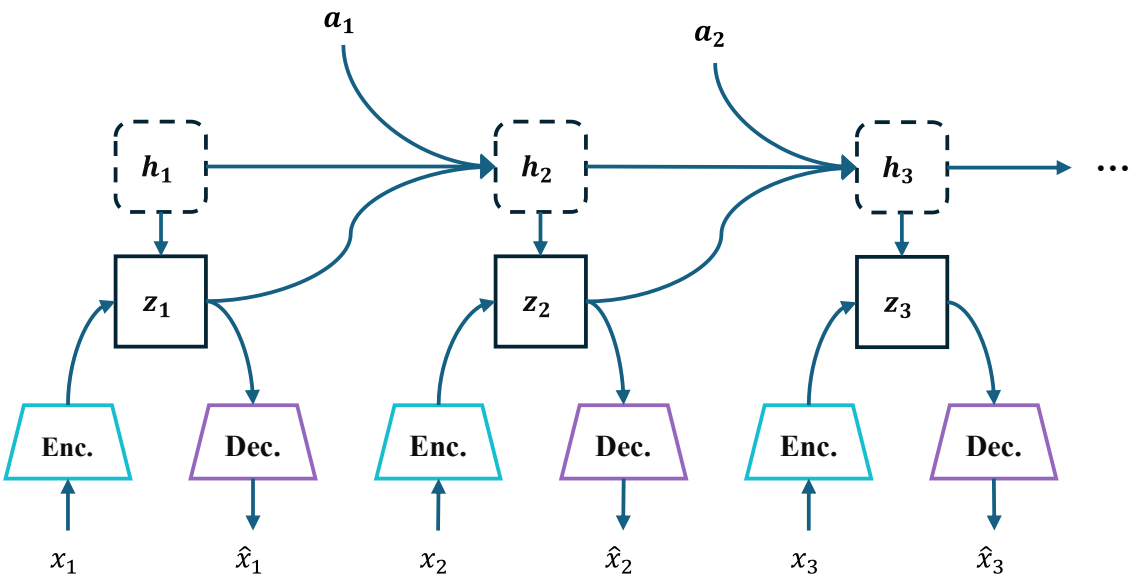


Figure 2.11: Recurrent State-Space Model (RSSM) of DreamerV3. The world model encodes sensory inputs into latent representations (z_t) using encoders (Enc.) and reconstructs inputs with decoders (Dec.). Recurrent states (h_t) predict future latent states using the past states and the actions (a_t), enabling the agent to imagine and plan future trajectories. Figure recreated from [95].

The actor-critic learning is done on 'imagined' trajectories only using the representations obtained from the world model, which itself is trained from the interaction with the environment. This allows for increased sample efficiency, especially for compute-intensive environments where direct interactions are costly or limited, enabling the agent to learn effectively from fewer real-world experiences by leveraging extensive simulated data. This process is illustrated in Figure 2.12.

The state of the environment used to train the actor-critic is the combination of the hidden state of the sequence model and the latent representation of the encoder: $s_t = \{h_t, z_t\}$. Similar to policy optimization methods, the actor model with parameters θ selects actions based on the current state to maximize the expected return:

$$a_t \sim \pi_\theta(a_t|s_t) \tag{2.33}$$

The critic model with parameters ψ evaluates the value of the 'imagined' state under the policy:

$$v_\psi(R_t|s_t) \tag{2.34}$$

Readers are referred to [95] for more details on the training and implementation details of DreamerV3.

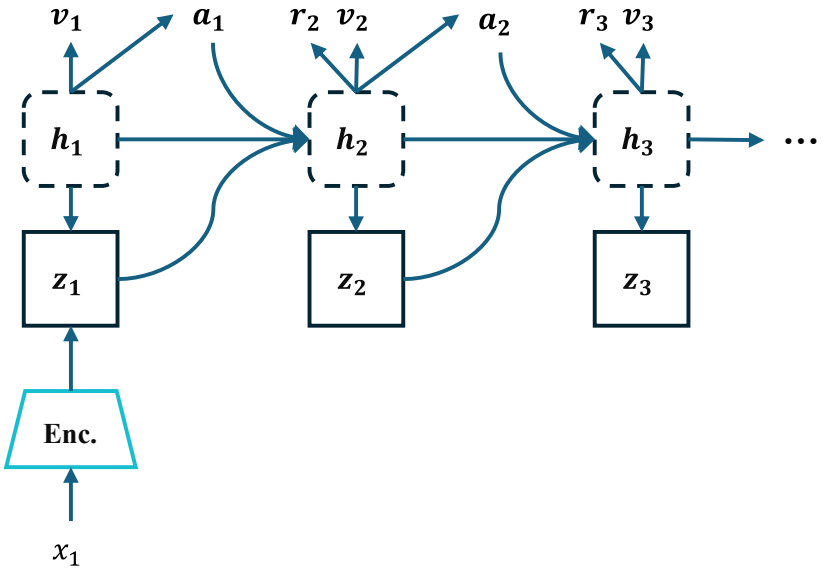


Figure 2.12: Actor-Critic Interaction in DreamerV3. The actor selects actions a_t to maximize returns (r_t), while the critic evaluates state values $v_\psi(s_t)$. Imagined trajectories from the world model enable efficient learning. Figure recreated from [95].

Reinforcement Learning for Structural Design and Optimization

While most of the literature on the application of RL for structural design consists of simplistic examples of shape optimization, one interesting example of real-world production use comes from the research team at Google, which trained a PPO agent to optimize the placement of macro components for new computer chip designs [96]. The training pipeline notably featured a pretraining supervised learning step to initialize the policy using a graph placement methodology for fast chip design. This enables the RL agent to efficiently generate chip floorplans that are comparable to or better than human experts' designs in terms of power, performance, and chip area, achieving this in under six hours compared to months of human effort.

A small baggage of literature is available on the applications of DRL to the task of shape optimization on single problems where a tiny number of design parameters are varied [97], [98], [99]. Although no clear performance advantages are presented in those papers, the flexibility provided by the reward signal shows an improvement over traditional optimization approaches like particle swarm [100] which is known to have difficulty handling design constraints.

The literature on DRL applied to TO is even more scarce. Two papers address the task of generating optimal designs via an iterative procedure involving the removal of discretized elements from the domain. One approach leverages Q-learning and a graph neural network to capture features of bar members, allowing an RL agent to sequentially eliminate unnecessary members from an initial ground structure [101]. The trained agent successfully generalized to boundary conditions and truss dimensions that are different than the training distribution. A similar methodology was applied to a 2D discretized mesh, where a Double Q-Learning Algorithm [102] was trained to minimize the compliance of the final structure [103]. Both of those papers work on an iterative procedure where material is removed until a ‘termination criterion’ is reached, like reaching a volume constraint, or an illegal action is performed by the agent, like trying to remove an already void element. They show the obvious drawback that the action space, the number of possible actions to take at each timestep, scales exponentially with the resolution of the ground structure or with the 2D FEA mesh discretization. Both papers tackle this issue by using model architectures which are invariant to the size of the discretization method, like a fully convolutional neural network or a graph neural network. While the graph-based truss TO method was tested successfully on a limited scale, generalizing from a 4x4 truss to a 6x6 truss for a specific boundary condition, the 2D FEA-based

method faced challenges when scaling from 6x6 to 24x24 grids, particularly under boundary conditions with closely packed loaded elements.

To provide a more scalable interface between the agent and the topology, exploratory work has been undertaken to use higher-level geometric functions like Bezier curves to interact with the density distribution in a mesh-independent way through a continuous actions space [104]. Like feature-mapping methods, a cubic Bezier curve can be parameterized with 4 point coordinates and a thickness value, allowing the projection of the geometry onto a fixed FEA mesh using a simple cell-intersection heuristic. The experiments were limited to uniaxial loading of a metamaterial beam but showed that the trained agent generated structures within 5-10% of conventional TO methods, demonstrating the benefits of leveraging a continuous actions space.

Outside of the direct design application, RL has been utilized to enhance the diversity of topology-optimized designs of automobile wheels [105]. The proposed pipeline features a supervised learning model which is trained to generate optimal topologies from a reference design and two design parameters. A PPO agent is then trained to maximize the diversity of the first model's generated structure by varying the two design parameters. The RL agent was shown to be competitive with an exhaustive search over the two design parameters while consuming a fraction of the computing time required to evaluate all the possible combinations of the design parameters. Again, a very small number of reference images were used for training (500) and testing (139), limiting the generalizability of the findings.

Overall, the literature on RL applied to TO is currently limited in the range of test cases used in experiments, which prevents drawing broad conclusions about the potential of RL for TO. Still, these limited test scenarios only allow for preliminary insights into the capabilities and limitations of RL in this domain.

2.3.6 Machine Learning for Post-Processing of Topology Optimization Outputs

The application of ML for PP-TO remains a largely unexplored area with minimal literature available. One notable exception focuses on constrained optimization using a cGAN in a direct design application with AM constraints [106]. The model is trained on optimal topologies generated using the SIMP method with an overhand constraint [107], supplemented by additional SIMP iterations to enhance performance.

While direct PP-TO applications are non-existent, a growing body of research addresses CAD reverse-engineering from 2D or 3D shape inputs using computer vision models, which could potentially be adapted for TO post-processing. CSGNet [108] combines supervised learning with reinforcement learning to generate a sequence of instructions for reconstructing shapes, employing a CNN encoder and a GRU decoder. UCSG-Net [109], an unsupervised approach that outperforms CSGNet, reconstructs 3D objects by predicting geometric primitives and their Constructive Solid Geometry (CSG) composition. It comprises an encoder, a primitive parameter prediction network, a converter from signed distance fields to indicator functions, and CSG layers that combine shapes using boolean operations, in order to produce an interpretable CSG parse tree. Point2CAD [110] focuses on reconstructing CAD models from 3D point clouds, utilizing a neural network autoencoder for point clustering, followed by separate steps for surface fitting and additional heuristic procedures to complete the CAD reconstruction. While these studies do not directly address TO outputs, they present promising approaches that could be extended and integrated into post-processing pipelines for topology optimization results. The adaptation of these methods to TO-specific challenges represents a significant opportunity for future research in the field.

2.4 Conclusion on the State-of-the-Art of Machine Learning for Topology Optimization

While the ML-TO literature has been influenced by the recent hype cycle in AI research, the emphasis on direct design approaches is well-founded. End-to-end methods, which process raw inputs directly to final outputs without intermediate steps, leverage the increasing computational power enabled by Moore's Law and have consistently outperformed other approaches in various fields where DL is applied [111]. However, the current literature on direct design ML-TO exhibits significant limitations in terms of scalability, generalization to different problem parameters, and the generation of disconnected structures. These limitations may explain the recent slowdown in publication growth for ML-TO observed in Figure 2.1. With the more straightforward research avenues now explored, the field is left with increasingly complex and difficult challenges that remain to be tackled.

Fully convolutional methods are straightforward to implement and are intrinsically compatible with density-based methods, but their computing requirements are directly coupled to the resolution of

the FEA mesh, which limits their scalability. Models built on feature-mapping methods could help address this problem, but the literature on this topic is limited.

The incompatibility between image similarity training and TO objectives has been demonstrated multiple times in the literature. This has pushed many researchers to explore 'physics-informed' approaches where the physics of the TO problem are included inside the optimization process of the model. This holds the potential to solve the problem of 'non-physical' generated structures, but brings additional software engineering challenges, as sparse linear solvers are not currently supported in leading DL frameworks like PyTorch and TensorFlow.

RL for TO is an under-researched field that has shown to be flexible enough to address TO problems using many representation methods, from ground structures, to density-based, to Bezier curves feature-mapping methods. By framing the TO objective as a reward function, RL eliminates the need to backpropagate through FEA. Instead, the expectations of rewards are utilized as the training objectives in policy optimization. As illustrated in Equation 2.19, a sampled mean of returns is used to approximate the policy gradient. This approach has not been scaled up to a large problem space and the literature has not explored more recent algorithms like policy gradient and model-based RL. An additional benefit of RL is the potential use of a stochastic policy which can be sampled at inference time to generate variations in the generated structures, similar to the generative modeling approaches detailed above.

Finally, the application of ML for post-processing TO outputs remains an underexplored area as well. While existing research on CAD reverse-engineering from 2D or 3D shapes offers promising techniques, their adaptation to TO-specific challenges presents opportunities for future work.

In summary, while ML approaches for TO have shown some interesting proof of concepts, there is still much work to be done before such methods help accelerate real industrial use cases. The current research is often limited in scope and quality, indicating that the potential of these methods is not yet fully realized. Future research should focus on addressing these limitations, exploring more complex and varied test cases, and improving the rigour and reporting of experimental designs to fully confirm and harness the potential of ML in topology optimization. The next chapter will discuss the scope of the articles presented in the thesis, addressing the research limitations identified in the literature review.

CHAPTER 3 New Approaches in Machine Learning for Direct Design and Post-Processing Topology Optimization Applications

The literature review highlighted the current focus on direct design methods for TO and its associated limitations of mesh-dependent model architecture and incompatibility between image similarity and the underlying physics. The overall lack of quality in the literature underscores the need for further research to develop "scalable" ML-TO solutions with the following characteristics:

1. Decoupling from FEA mesh: An ML-TO model should operate independently of the FEA mesh resolution to enable it to scale to arbitrary resolutions.
2. Boundary conditions generalization: The ability to generalize to a wide range of boundary conditions.
3. Domain shape independence: The model should perform well across various domain shapes.
4. Physics generalization: Similar to LLMs that work across multiple languages, an ideal ML-TO model should be capable of addressing various physical problems without the need for retraining.
5. A low breakeven threshold: An economic measure proposed in [18] that quantifies the number of new problems a model must successfully solve, different from those in the training dataset, for its reduced execution costs to offset the initial costs of training and data collection. This measure is only meaningful if the trained model generates structures that perform comparably to those obtained through conventional optimization methods.

These five points characterize the 'usefulness' of a trained ML-TO solution rather than focusing solely on the performance of the generated structures. In order to perform on par with conventional optimization methods, an ML-TO solution must exhibit strong generalization capabilities across these characteristics. Improving this aspect of direct design methods is the focus of the first two articles presented in this thesis. Furthermore, the lack of literature on leveraging state-of-the-art CV models for the post-processing of TO outputs presents a rich and open field for research, which is the focus of the third article.

From the research objectives presented in section 1.5, the following three unexplored research areas are the subject of this thesis:

- The use of feature-mapping methods for mesh-independent ML-TO approaches. Addressing SO1.
- Framing TO as a deep reinforcement learning problem with a physical reward function. Addressing SO2.
- Implementing a CV model for the automatic geometric reverse-engineering of optimal density distributions. Addressing SO3.

3.1 Feature-Mapping Methods for Machine Learning Topology Optimization

Feature-mapping methods are key enablers for all three research objectives. First, the use of higher-level geometric design variables provides a mesh-independent model architecture for supervised learning approaches and a scalable action space for RL environments. Secondly, it provides an obvious source of labelled training data to extract geometric information from black-and-white structures since each material pixel or voxel is known to belong to an individual component. The MMC framework was chosen as the basis on which all three articles of this thesis were built because of the good availability of code implementations. A Python implementation of the 188-line Matlab code [112] was used.

3.2 Summary of Articles

The following section provides a brief overview of the three key articles that form the foundation of this thesis. Each article addresses a distinct aspect of integrating machine learning and feature-mapping methods into TO, highlighting advancements in direct design and post-processing methods.

Article 1: Supervised Deep Learning for the Moving Morphable Components Topology Optimization Framework

The first article addresses the limitations of mesh-dependent models in TO by leveraging the MMC framework, directly addressing Secondary Objective 1 (SO1). A DL model is developed that generates geometric design variables independently of the FEA mesh, based on an input vector describing the parameters of the problem. Despite achieving state-of-the-art regression performance, with an average stiffness 11.48% lower than conventional MMC designs, the

direct design approach produces disconnected structures. To mitigate this, an acceleration strategy is introduced, using the model as an initial design generator for conventional optimization, resulting in a 36.84% reduction in computation time without sacrificing the final mechanical performance. This strategy also shows potential for efficiency gains on out-of-distribution (OOD) data. This work highlights both the potential and the current limitations of direct design approaches trained using image similarity, emphasizing the need to integrate models with the physical TO objective in order to achieve practical efficiency gains.

Article 2: Structural Design Through Reinforcement Learning

The second article addresses Secondary Objective 2 (SO2) by introducing the Structural Optimization gym (SOgym), a novel open-source RL environment designed to train agents to solve TO problems of compliance minimization under volume constraints. SOgym integrates the physics of linear elasticity into the reward function and employs feature-mapping methods to create a mesh-independent interface between the environment and the agent. Baseline results using PPO and DreamerV3 agents show that generated structures have no disconnections, although they exhibit compliance within 54% of that achieved by traditional MMC optimization. Additionally, the learning rate of the DreamerV3-100M agent is approximately four orders of magnitude lower than that of engineering students in an in-class experiment, illustrating the difference in learning speed between RL agents starting from a random policy and human learners. This study validates that RL can address continuous TO problems, offering the potential to explore and learn from diverse design solutions by exploiting the flexibility of the reward signal as feedback for learning.

Article 3: From Density to Geometry: Instance Segmentation for Reverse Engineering of Optimized Structures

The third article addresses Secondary Objective 3 (SO3) by developing an automatic geometric interpretation tool for elemental density distributions. YOLOv8-TO, a novel instance segmentation approach, is introduced to automatically reverse engineer topology-optimized structures into interpretable geometric parameters. This method leverages the YOLOv8 model and outperforms traditional skeletonization techniques, achieving an average improvement of 13.84% in the Dice coefficient, with peak enhancements of up to 20.78%. YOLOv8-TO is trained on a diverse dataset of optimized and random structures generated from the MMC

framework and demonstrates excellent generalization capabilities along with fast inference times, making it suitable for integration into design workflows. This approach effectively addresses the challenge of converting implicit density distributions into parametric representations, facilitating design exploration and integration with CAD tools. The research underscores the potential of computer vision models in enhancing the post-processing stage of TO, offering a more accurate and efficient means of generating geometrically meaningful representations from optimized structures.

In summary, the first two articles contribute to Secondary Objectives 1 (SO1) and 2 (SO2) by improving the generalization and scalability of direct design ML-TO approaches through the use of mesh-independent inputs and by embedding physical objectives into the model's training. The third article focuses on Secondary Objective 3 (SO3) by presenting a novel approach for the post-processing of optimal structures, framing the task as an instance segmentation problem that utilizes state-of-the-art computer vision models. Collectively, these articles address the Main Objective (MO) of developing scalable ML models for both the direct design generation and post-processing of optimal topologies.

CHAPTER 4 ARTICLE 1 : Supervised Deep Learning for the Moving Morphable Components Topology Optimization Framework

Thomas Rochefort-Beaudoin[a], Aurelian Vadean[a], Jean-François Gamache[b] and Sofiane Achiche[a]

[a]*Department of Mechanical Engineering, Polytechnique Montréal, 2500 Chemin de Polytechnique, Montréal, QC, Canada*

[b]*Kinova Robotics, 4333 Bd de la Grande-Allée, Boisbriand, Québec, Canada*

Published in *Engineering Applications of Artificial Intelligence,* Vol 123 Part C, May 6th 2023

Abstract

In the context of mechanical engineering design, the field of machine learning accelerated topology optimization is dominated by image-based models trained in a supervised manner on datasets with a limited diversity of boundary conditions. State-of-the-art methods show poor generalization capabilities and are strongly coupled to finite element mesh resolution, hindering scalability. In this paper, we leverage the explicit topology parameterization of the moving morphable components (MMC) framework to train a deep learning model that directly generates geometric design variables using a model architecture that is independent of the finite element mesh used for structural analysis. The developed model is trained on a large dataset of boundary conditions. Despite achieving state-of-the-art regression loss, evaluations reveal that direct design approaches generate topologies with poor mechanical performance. Specifically, the model-generated topologies have, on average, a stiffness 11.48% lower than conventional MMC designs, as evidenced by in-distribution and out-of-distribution test samples. We demonstrate that this is due to the incompatibility between the regression loss function typically used in literature and the topology optimization objective of compliance minimization. To address this issue, we propose a novel acceleration approach that leverages the trained model to generate improved initial designs for conventional optimization. Specifically, the deep learning model is used to generate an initial design, which is then refined by conventional optimization to arrive at a final, optimal design. This approach shows a computation time-saving of 36.84% without sacrificing the final mechanical performance of the optimal topology compared to conventional optimization starting from a uniform initial layout.

4.1 Introduction

Topology optimization (TO) is a numerical tool that aims to automatically solve the problem of efficiently distributing material inside a given domain to maximize specific mechanical properties. This structural optimization method, introduced by Bendsoe in 1988 [26], proposes to represent the material distribution using a set of design variables, and then move toward an optimum by computing the objective function derivative with respect to the design variables.

The overall TO research field has moved in various directions throughout the years, with multiple methods and optimization frameworks developed and applied to the problem. Readers are referred to the classical paper by Sigmund and Maute [113] for an overview of the field. One of those widely used methods in academia and industry is the density-based Solid Isotropic Material with Penalization (SIMP) which uses the individual element density of a Finite Element Analysis (FEA) mesh as design variables.

Even though the past twenty years of research has enabled the use of density-based TO for large-scale problems such as the minimum compliance optimization of an airplane wing [9] or a section of a bridge [10], the prohibitively large computing cost associated with TO has hindered its widespread adoption in industrial applications. The iterative nature of TO first implies the repetitive solving of high-fidelity FEA whose computing cost increases exponentially with the increase of the resolution of the mesh. Moreover, the higher resolution needed for accurate computation of the mechanical fields implies a higher number of design variables for density-based methods, which also has the effect of raising the computing cost of the sensitivity analysis involved in the optimization process.

This important barrier to adoption in TO has led to multiple research efforts in high-performance computing, multi-grid methods, and most recently, machine learning. As this paper will focus on machine learning approaches, readers can consult [15] for a review of the latest state-of-the-art methods for accelerating TO.

4.1.1 The Moving Morphable Components Topology Optimization Method

Results from density-based methods need to be interpreted [44] to transfer the obtained mechanical structure to CAD software, implicitly described by the discretized material distribution. This challenge, combined with the large number of design variables involved in density-based methods,

has led researchers to develop TO methods that seek to decouple the representation of the topology from the FEA mesh and use a topology parameterization that is more explicit [114].

In recent years, various feature-mapping methods [27] were developed in which the topology is parameterized by a set of higher-level geometric primitives which are mapped to a FEA mesh using a projection scheme. This allows for the parameterization of the topology to be decoupled from the discretization of the mechanical model. Therefore, an increase in the FEA mesh resolution for more accurate computation does not automatically lead to an increase in the number of optimization variables, providing better computing efficiency.

The method of Moving Morphable Components (MMC) has been proposed in 2014 to use geometric design variables that are easily interpretable [28]. This TO framework parameterizes the topology using a set of "structural components" which are defined using geometric design variables. For a linearly varying thickness component, the design variables involved in its definition are the center's coordinates (x_0, y_0), its orientation (θ), its half-length (L) and the two ends' half-thicknesses (t_1, t_2). The various design variables are illustrated in Figure 4.1.

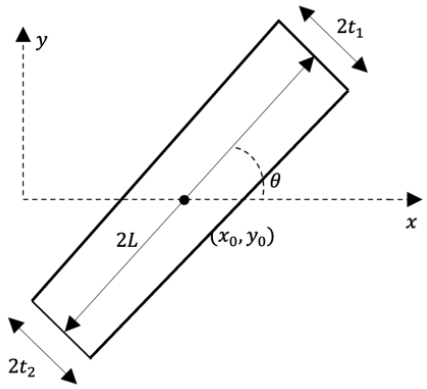


Figure 4.1: Design variables defining the shape of a linearly varying thickness component of the MMC framework

The topology of the structure is defined as the assembly of i components, each mathematically defined using superellipse functions which are called Topology Description Function (TDF) Φ_i, here described by Equations 4.1 through 4.3:

$$\Phi_i(x, y) = \left(\frac{x'}{L_i}\right)^q + \left(\frac{y'}{f(x')}\right)^q - 1 \tag{4.1}$$

$$f(x') = \frac{t_1 + t_2}{2} + \frac{t_2 - t_1}{2L}(x') \tag{4.2}$$

$$\begin{Bmatrix} x' \\ y' \end{Bmatrix} = \begin{bmatrix} cos\theta_i & sin\theta_i \\ -sin\theta_i & cos\theta_i \end{bmatrix} \begin{Bmatrix} x - x_{0i} \\ y - y_{0i} \end{Bmatrix} \tag{4.3}$$

Here, q is an integer (q =6 is used in this paper). The subset of the design domain D containing material is defined as Ω^s (Equation 4.4) and depends on the level of the obtained level-set function of the component's assembly.

$$\begin{cases} \Phi^s(x) > 0, if\ x \in \Omega^s \\ \Phi^s(x) = 0, if\ x \in\ \partial\Omega^s \\ \Phi^s(x) < 0, if\ x \notin \Omega^s \end{cases} \tag{4.4}$$

The density distribution on the discretized FEA mesh is obtained using a regularized Heaviside (H_e) projection of the TDF and the ersatz material model as in [112] and described in Equation 4.5:

$$H_e(\Phi_i^e) = \begin{cases} 1, & if\ \Phi_i^e > \epsilon \\ \dfrac{3(1-\alpha)}{4}\left(\dfrac{x}{\epsilon} - \dfrac{x^3}{3\epsilon^3}\right) + \left(\dfrac{1+\alpha}{2}\right), & if\ |\Phi_i^e| \le \epsilon \\ \alpha & otherwise \end{cases} \tag{4.5}$$

For each element e of the discretized mechanical model, Young's modulus E_e and density $\rho_e$are defined in Equations 4.6 and 4.7 respectively:

$$E_e = \rho_e E \tag{4.6}$$

$$\rho_e = \frac{1}{4}\sum_{i=1}^{4} H_e(\Phi_i^e) \tag{4.7}$$

Where E is the material's Young's modulus, and $H_e(\Phi_i^e)$ for i=1…,4 are the Heaviside projection values of the TDF function at the four nodes of the mesh element e. The general TO problem under design constraints for the MMC framework is expressed in Equation 4.8:

$$\begin{gathered} \text{Find } d = (d_1, \dots, d_n) \\ \text{to minimize } I = I(d) \\ s.t.\ \ g_i(d) \le 0, \quad i = 1, \dots, l \\ d \subset u_d \end{gathered} \tag{4.8}$$

Here $d = (d_1, \dots, d_n)$ represents the vector of design variables for n components, and for a linearly varying thickness component: $d_i = (x_{0i}, y_{0i}, L_i, \theta_i, t_{1i}, t_{2i})$. $I = I(d)$ can be any objective function of interest for the design problem at hands such as the mechanical compliance, the maximum stress, or the natural frequency of the system. $g_i(d), i = 1, \dots, l$ are the design constraints involved such as the desired volume fraction or the maximum local stress allowed. $u_d$is the admissible subset of the design variables that is defined through minimum and maximum values that are chosen by the designers.

Gradient-based non-linear optimizers such as the popular Method of Moving Asymptotes (MMA) [115] are used to update the design variables using first-order sensitivity analysis of the objective function.

The two main benefits of using the MMC framework over density-based methods are the reduced number of design variables as well as the increased ability to directly interpret the obtained topology. For example, a rectangular beam discretized using a 200x100 FEA mesh will imply 20000 design variables for density-based methods, while MMC can define the topology using 16 components, each defined using 6 variables, for a total of 96 design variables. These 96 variables all have an explicit geometric interpretation that can directly be used by engineers throughout the design process.

In addition to the high computing cost involved in the FEA analysis and non-linear optimization at each iteration, the MMC method is prone to convergence problems [14] which exacerbates the high computing cost of TO. In this paper, we evaluate if a supervised deep learning model can generate optimal topologies directly and if it can help "jumpstart" the direct optimization process by generating an initial layout for the conventional iterative optimization process.

4.1.2 Machine Learning for Topology Optimization

Biological evolution has generated numerous real-life examples of highly optimized biological structures, which has inspired the development of many nature-inspired optimization methods applied to mechanical design. Machine Learning (ML) algorithms such as genetic algorithms and swarm optimization have been successfully used in the automotive industry [116, 117].

ML applied to TO is a very young research field that aims to leverage large-scale data analysis to extract prior knowledge from already optimized topologies to reduce the computational cost of the iterative optimization process.

In recent years, Artificial Neural Networks (ANNs) have garnered considerable attention due to their capacity to learn intricate relationships between input and output variables. ANNs have proven successful in diverse engineering fields, ranging from biomedical [118] , energy management [119], to mechanical design [120]. The broad spectrum of ANNs applications underscores their versatility and potential advantages in tackling structural optimization problems that can vary greatly in terms of their shape [121], objectives [122], manufacturing constraints [123] and underlying physics [124].

As ANNs became more popular across all research fields, researchers started applying them to various parts of the TO process to accelerate the conventional iterative optimization scheme [18]. Convolutional Neural Networks (CNN) showed the potential to be trained to generate binary density distributions from intermediary iterations of the SIMP process, allowing to run 20 times fewer iterations to generate optimal topologies [54]. The same approach was successful on 3D problems, resulting in a 40% reduction in computational time compared to the conventional 3D SIMP optimization [55].

Direct design or “one-shot” approaches were also proposed where the model is directly trained to generate the optimal structure from a set of inputs that describe the problem [125]. By taking as inputs a description of the TO problem, the model aims to predict an optimal topology by a single forward pass through its parameters. By using a 2D matrix as input describing the various BCs, a CNN encoder-decoder architecture can be used to directly generate the optimal topology [60]. The predicted topologies showed an average pixel-value error of 2.72% while presenting a computing time equal to 0.06% of the time needed with conventional SIMP.

While the speedup obtained at inference is impressive, it is important to note that these results often misrepresent or oversell the potential of direct design approaches. Researchers often fail to present generalization studies to quantify the effectiveness of proposed models for TO problems that differ from their training dataset distribution. For a machine learning model to be useful for TO, the model needs to provide a speedup for multiple geometries and boundary conditions that will outweigh the computational cost that was needed to generate the training samples. To do so, model

architectures need to be as flexible as possible to not have any dependencies on mesh resolutions, domain size or shapes, and types or the number of boundary conditions.

Pixel-based models suffer from a strong coupling on the resolution used during training. Since these models output a 2D or 3D matrix of binary material distribution, their architectures are difficult to scale to different mesh resolutions and shapes, rendering their method extremely limited in application and generalization capacity to other problems. More explicit topology parameterization methods such as the MMC framework can be used to train models that output a fixed set of geometric design variables which can then be projected onto an FEA mesh for structural analysis.

The field of ML accelerated MMC has a very limited literature base that, while showing interesting results in terms of computational acceleration, shows published results on a very limited range of boundary conditions. An overview of the research field is presented in Table 4.1.

Table 4.1: Summary of the current ML accelerated MMC literature.

Paper	Year	Algorithm used	Size of training set (# samples)	Size of test set (# samples)	Average compliance difference with conventional optimization	FEA mesh resolution independent model architecture
[76]	2019	KNN & SVR	50	5	25.31%	Yes
[77]	2021	Attention-based Res-U-Net	537	67	< 3%	No
[126]	2022	MLP	2995	374	2.34%	Yes

Using the classical ML algorithms of K-Nearest Neighbors and Support Vector Regression, researchers have shown that they could produce topologies with a 25.31% error on the compliance [76]. The authors used a very limited set of boundary conditions for training where only the unit load was varied in a very small region of the design space. Even though the direct prediction performance was underwhelming, the authors proposed to use the predictions as the initial layout for a conventional optimization process. Results for a single load case show that starting from the model's prediction allows obtaining a converged topology in 23 iterations while starting from a

general initial layout required 298 iterations to obtain the same topology with the same final compliance. The potential of the speedup that an ML-generated initial topology could bring to the MMC framework is clouded by the very limited test run by the authors.

Modern CNN architectures, inspired by the impressive advances in the computer vision field, have also shown interesting potential for MMC. Using an attention-based Res-U-Net with an image-to-image framework, researchers trained the neural network on a very small dataset of 672 topologies of the classical 2x1 cantilever beam where only the vertical load position is varied in the last 7 of the right-most columns of the 192x96 mesh used [77]. Since the output of the model is a 2D matrix, the authors used per-pixel accuracy as a measure of predictive performance. Results show an average of 90% accuracy was achieved by the model on the test set with a less than 3% difference between the compliance of the predicted topology and the one obtained via direct MMC optimization. Although showing an impressive improvement over the classical methods, results in this paper were also limited by the very narrow variation in BCs. Also, using an image-to-image approach completely defeats the purpose of using geometrical design variables of the MMC framework. Obtaining a pixel density distribution as the model output removes the ability of the user to directly interpret the topology and easily transfer the results to CAD software. It also limits the scalability of the method in the same way that pixel-based methods are limited to a fixed grid resolution output. Finally, describing the BCs and the desired volume fractions of the TO problems using an image generates an input that is very sparse in terms of information content, as most of the pixel values carry redundant information.

This has led researchers to use a simple multilayer perceptron (MLP) to directly generate the design variables of a Moving Morphable Bars TO framework [126], a variant of MMC with a focus on imposing variable thicknesses of the topology members [127]. The output of the model is a single-column vector containing the geometric design variables defining each of the components in the topology assembly. The authors also used a single-column vector as input containing the variables defining the TO problems of a minimum compliance cantilever beam. To try to increase the diversity of the training topologies, the authors varied the desired volume fraction, the horizontal dimension of the domain and the vertical position of the unit load at the right-most boundary. Reported results show a Mean Absolute Error (MAE) of 2.34% on the final compliance of the predicted topologies, and an MAE value of 1.35% for the desired volume fractions. The authors claim they used the MSE between the ground-truth design variables and the predicted values as a

loss function, and they used an 80-10-10% split for the training-validation-test dataset split. However, a value of 0.0024 is reported for the loss function and a different value of 0.03 is reported for the MSE, without any mention of which dataset those performance metrics of the model are calculated. Even with this lack of robust ML methodology reporting, these results showed the potential of using geometric design variables as the output of a neural network model for TO as the authors compared their method to a density-based output model whose size increased dramatically with the resolution of the mesh used. For a 100x50 mesh size, the geometric-based model had a final output layer of 170 parameters while the pixel-based model had a final layer of 5000 parameters. The authors even showed that increasing the resolution of the mesh by 4x caused an out-of-memory error for the pixel-based model due to the large increase in the number of trainable parameters of the final output layer.

This paper aims to provide a more comprehensive view of the performance of a supervised deep learning model for the MMC framework by using a diverse set of boundary conditions and desired volume fractions for the minimum compliance TO of a 2x1 rectangular beam. The brute prediction power of the trained neural network model when used in a direct design strategy is presented. Also, a novel acceleration strategy inspired by [76] to reduce the computing time of TO is proposed where the deep learning model's predictions are used as an initial solution for the conventional MMC optimization.

4.2 Method

The model described in this section utilizes a simple architecture where the design variable vector d is directly generated as output, while a single description vector β containing the TO problem characteristics is used as input. The model is then trained in a supervised manner to learn the mapping function $d = f(\beta)$. The methodology used throughout this paper is illustrated in Figure 4.2 and is organized into three main stages, starting with the generation of the dataset using a hybrid MMA optimizer. Subsequently, the model is trained in a supervised manner, utilizing a regression loss function, and optimizing hyperparameters on a validation set. Finally, the model is tested using both direct design and model-accelerated approaches.

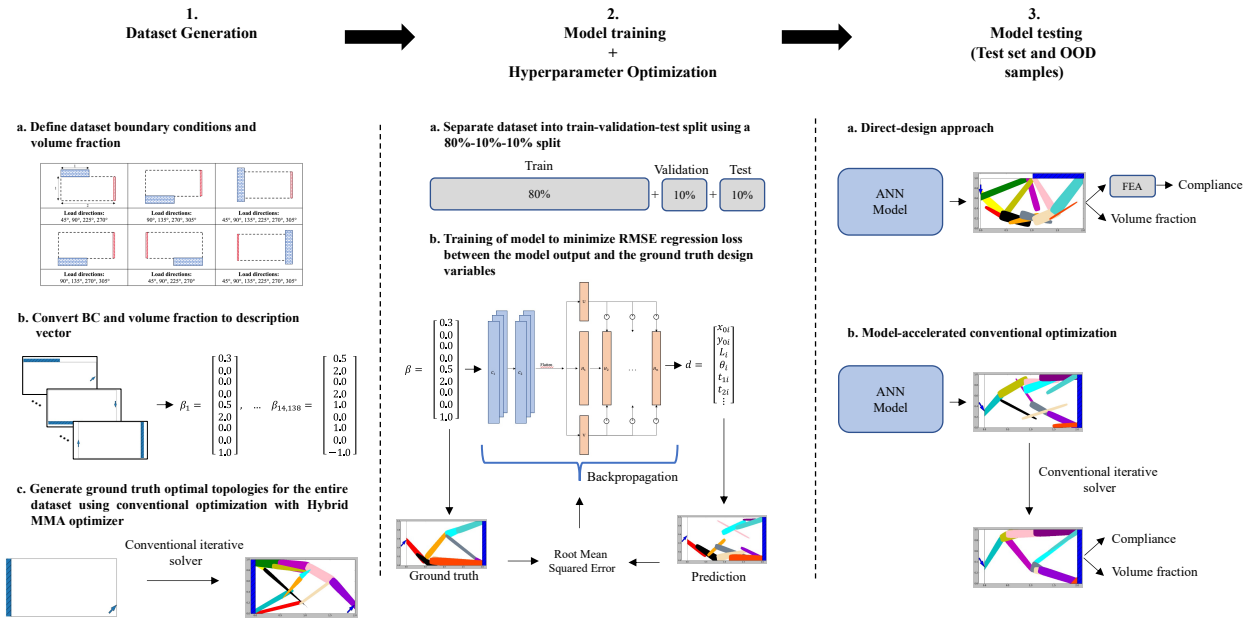


Figure 4.2: Flowchart illustrating the supervised learning pipeline used for the proposed method.

4.2.1 Benchmark Problems

For the training and the testing phases of the model, a set of different BCs was used to generate a dataset of over 14000 unique loading and support conditions for a fixed geometry of a 2x1 rectangular beam. The 6 various sets of BCs are illustrated in Figure 4.3. For the support conditions, a fully supported boundary was varied between the side boundaries and the upper and lower corners. A single load with a unit force magnitude was varied on each mesh node of the left and right-most boundaries. The load direction was varied with an increment of 45 degrees while removing the directions that would be aligned with the support boundary and would have solely generated tension or compression force. These loading conditions generate trivial column-like topologies that are not interesting for the test case. Volume fractions of 30%, 35%, 40%, 45% and 50% were used to further diversify the problems in the dataset.

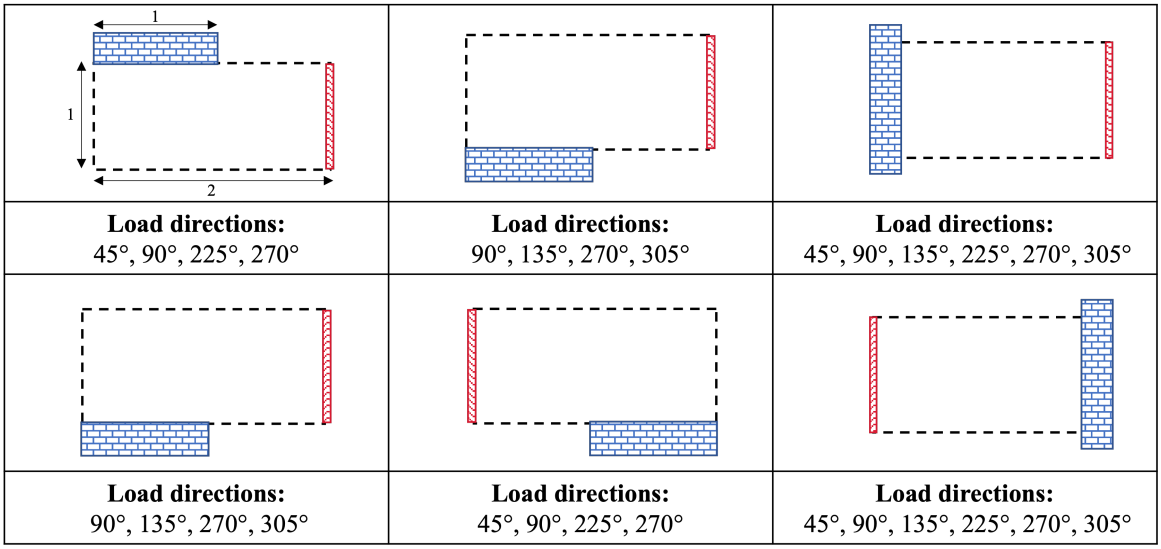


Figure 4.3: Loading and support conditions used. In red: the boundary where the load is placed. In blue: the fully supported boundary.

By considering every possible combination of support position, loading position and orientation, and volume fractions, a total set of 14138 unique TO problems was defined. The optimal topologies which will be used as "ground-truth" for training the neural network model were generated using the updated 216-line MMC code [40] because of its increased computational efficiency over the original 188-line MMC code [112]. A topology composed of 16 linearly varying components described using 6 geometric variables and a resolution of 200 x 100 for the FEA mesh was used. A hybrid optimizer combining MMA and the Globally Convergent Method of Moving Asymptotes (GCMMA) was used because of its good convergence behaviour near local optimums for diversified problems [128]. The hybrid optimizer started the optimization process using MMA, and switched to GCMMA when oscillations in the objective function were detected using the zero-crossing function in Equation 4.9:

$$-\delta < \frac{f_0\left(\boldsymbol{x}^{(k-2)}\right) - f_0\left(\boldsymbol{x}^{(k-1)}\right)}{\frac{|f_0(\boldsymbol{x}^{(k-2)})| + |f_0(\boldsymbol{x}^{(k-1)})|}{2}} * \frac{f_0\left(\boldsymbol{x}^{(k-1)}\right) - f_0\left(\boldsymbol{x}^{(k)}\right)}{\frac{|f_0(\boldsymbol{x}^{(k-1)})| + |f_0(\boldsymbol{x}^{(k)})|}{2}} < 0 \tag{4.9}$$

Where $f_0\left(\boldsymbol{x}^{(k)}\right)$ is the value of the objective function at iteration k and δ is a positive number that controls the sensitivity of the switch criterion. The parameters defining the hybrid optimization process are presented in Appendix A. An example of an optimal topology obtained for a specific set of conditions is illustrated in Figure 4.4. The Karush-Kuhn-Tucker (KKT) conditions [129, 130]

were used as the convergence criterion. The optimization process was stopped when the residual norm of the KKT conditions is smaller than 5e-2 or when the maximum number of outer iterations of 1000 is reached. These two criteria were selected to be stringent to maximize the probability of obtaining topologies of high quality in the training set. Experimental results showed that this was sufficient. Topologies in the dataset took an average of 891 iterations involving a FEA to complete. This amounts to more than 12 million total iterations of FEA to construct the training dataset.

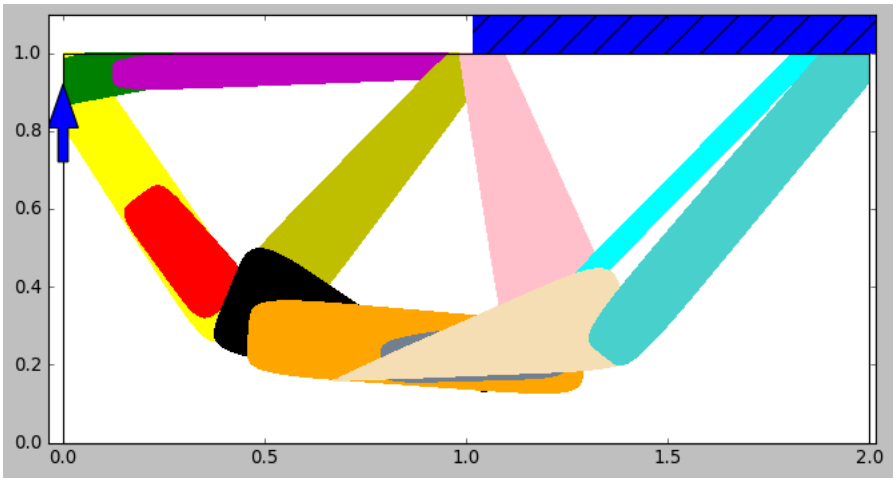


Figure 4.4: Example of a generated support/loading boundary condition and its corresponding optimal topology obtained via MMC. Each component's contour is represented by a unique colour.

The obtained dataset was separated randomly into a training and test set using an 80% - 20% split. The training set was further divided using a random 80% - 20% split to generate a validation set to provide a proxy for out-of-sample performance for the hyperparameter optimization process. This resulted in training, validation, and test sets of size 9048, 2262 and 2828 respectively. All the mechanical and computing performance metrics presented in the results section were computed on the test set, while the hyperparameter optimization was conducted using the validation set.

4.2.2 Network Architecture

Instead of using a sparse image as input to convey information on the BCs and desired volume fraction, we propose to use a dense description vector to convey the spatial attributes of the TO problem using coordinates. Since our benchmark problems for the dataset features the same 2 x 1

rectangular geometry, we only describe the position of the support (S_{A_x}, S_{A_y} , S_{B_x}, S_{B_y}) and the load (x_L , y_L), the direction of the load ($\overrightarrow{L_x}$, $\overrightarrow{L_y}$), and the desired volume fraction (V_0), which can all be expressed using floating-point numbers as described in Table 4.2.

Table 4.2: Description of the input description vector used to describe the topology optimization problem.

Input vector component ($\boldsymbol{\beta}$)	Description
V_0	Desired volume fraction
S_{A_x}	x-coordinate of the support boundary at the first fixed node
S_{A_y}	y-coordinate of the support boundary at the first fixed node
S_{B_x}	x-coordinate of the support boundary at the last fixed node
S_{B_y}	y-coordinate of the support boundary at the last fixed node
x_L	x-coordinate of the loading node
y_L	y-coordinate of the loading node
$\overrightarrow{L_x}$	Magnitude of the load in the x direction
$\overrightarrow{L_y}$	Magnitude of the load in the y direction

For a set of 16 components, the output layer of the model consists of a dense layer with 96 units and a sigmoid activation function which represents the normalized value of the design variables of all 16 components. The normalization of the values between 0 and 1 is made using the maximum and minimum values for the design variables that are presented in Appendix A.

By using a dense description vector as input instead of an image, and by using an output vector consisting of the MMC framework design variables, the proposed approach is entirely independent of the finite element mesh used for structural analysis. This allows for arbitrary mesh sizes to be used for dataset generation and structural analysis of the model's generated topologies.

The model was trained as a regression task where the loss function is the Root Mean Squared Error (RMSE) between the model's prediction and the ground-truth vector of design variables for the specific TO problem.

Two different neural network architectures were tested. The first one is a simple MultiLayer Perceptron (MLP) architecture consisting of only dense layers. The second one is an improved fully-connected architecture inspired by the state-of-the-art in the field of physics-informed neural networks (PINN), and consists of an improvement over generic MLP that results in 50-100x better

precision for PINN tasks [131]. Similar to the developments in the field of computer vision, this architecture adds residual connections [132] and multiplicative interactions between the various layers and the input. Both types of architectures are compared to see if performance gains observed for PINNs can also translate to ML accelerated TO. The use of two 1-D convolutional layers was also tested as trainable input embeddings to test if projecting the input space to a higher dimensional space improves accuracy.

Hyperparameter optimization was conducted for each of the two architectures tested. A tree-parsen estimator (TPE) sampling method [133] was used based on the Optuna library [134]. The PINN-inspired architecture with the use of 1-D convolutional input embeddings obtained the best performance on the validation set, with a MAE of 4.62%. The best-performing conventional fully connected network managed to score a validation MAE of 5.16%. The difference between the two architectures is too small to conclude on the potential synergies of using PINN SOTA architectures for topology optimization tasks. Further details on the network architecture and the hyperparameter optimization procedure can be found in Appendix A.

4.3 Numerical Results

4.3.1 Performance of the Model's Prediction in a Direct Design Strategy

The trained model was first evaluated by testing the out-of-sample prediction performance on the test set. TO problems in the test set resemble those in the training set, but the unique combinations of volume fractions and support/loading conditions have never been seen by the model during training. The mean absolute error (MAE) between the predicted design variables and the ground truth samples, as well as the difference in compliance and volume fraction, gives us three easily interpretable metrics to evaluate the brute prediction performance of the model.

The most obvious advantage of the model is its computing speed, as design variables are generated with a single forward pass through the network's parameters. On the test set, the model generated design variables for all 2828 TO problems in 178ms, which represents an average of 0.06 ms per problem. The relative mechanical performance of the model's predictions compared to the ground truths is presented in column (a) of Table 4.3. Over the entire test set, the model's predictions showed a median value for the difference in compliance with the ground truth samples of +11.48%. The model also generated topologies with a median gap between the observed volume and the desired volume fraction of +5.03%. Examples of ground truth topologies and their associated

model predictions for 4 test set samples are presented in Figure 4.5 along with their respective compliance and volume fraction value. The model regression error can be observed by visualizing the differences between the two columns.

Table 4.3: Mechanical performance evaluation of the direct MMC process, the model's predictions and the direct MMC process using the model's predictions as the initial layout.

Test set metrics	**(a) Model's predictions**	**(b) Conventional MMC with model's prediction as initial layout**
Median compliance difference with ground truth	+11.48%	+ 0.27%
Median volume difference to desired volume fraction	+ 5.03%	+ 0.1%

As the model is strictly trained to minimize the error between the predicted design variables and the ones from the ground truth, there is no hard constraint on the desired volume fraction and no mechanisms that lead the model to minimize compliance. This "inverse problem" approach can lead to a dramatic failure mechanism where a model-predicted structure with a small regression loss value can show a disconnected load, and therefore a very high compliance value. Examples of such failures are presented in Figure 4.6 where disconnected load-support paths cause extreme compliance values for the model predictions (right column). Figure 4.7 provides a visual demonstration of this key challenge in using regression loss to address minimum compliance optimization problems. Specifically, the top row shows a good-performing model-generated topology and its ground truth (top row). The bottom row shows an example of a bad-performing model prediction in terms of compliance. While the sample of the bottom row exhibits a lower regression loss value, it performs significantly worse in terms of compliance when compared to its ground truth. This indicates the limitations of relying solely on regression loss as a proxy for optimization success.

Results in terms of the difference in compliance and volume are higher than the current literature of deep learning accelerated MMC, but results presented in this paper are on a dataset that is more than 10 times larger and with more diverse boundary conditions and volume fractions. Results in this paper should therefore present a more accurate overview of the poor potential of direct design approaches.

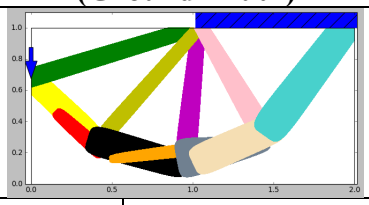
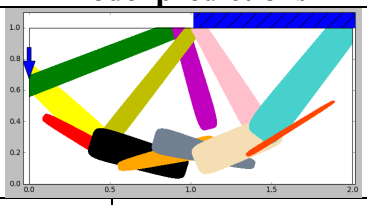
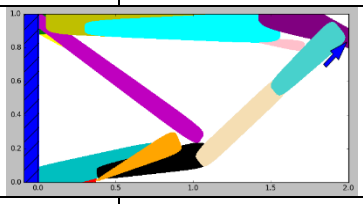
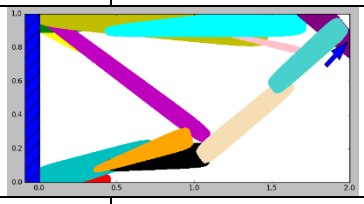
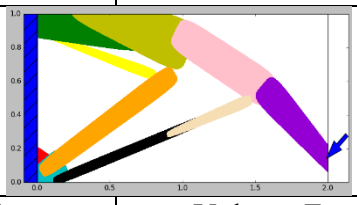
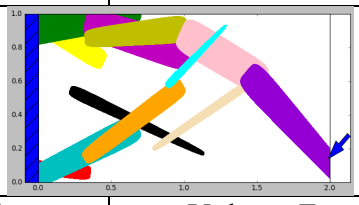
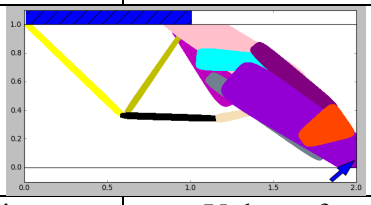
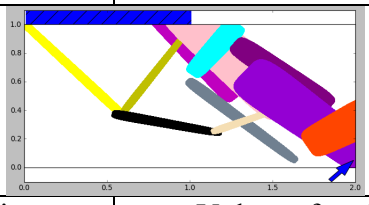
MMC Generated (Ground Truth)		**Neural Network Model predictions**	
			
Compliance: 43.58	Volume fraction: 40 %	Compliance: 45.63	Volume fraction: 42.99 %
			
Compliance: 64.99	Volume fraction: 35 %	Compliance: 67.05	Volume fraction: 35.59 %
			
Compliance: 31.12	Volume Fraction: 35 %	Compliance: 43.16	Volume Fraction: 36.61 %
			
Compliance: 8.94	Volume fraction: 30 %	Compliance: 9.25	Volume fraction: 33.34 %

Figure 4.5: Comparison of results obtained from the neural network model (right column) and the conventional MMC optimization (left column).

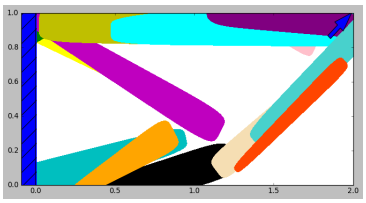
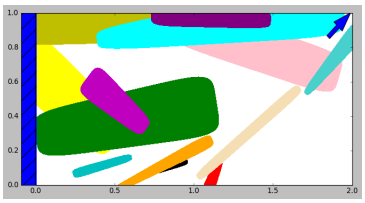
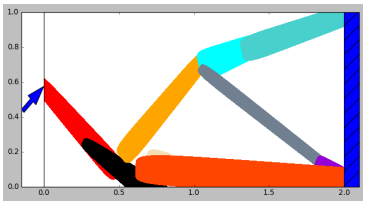
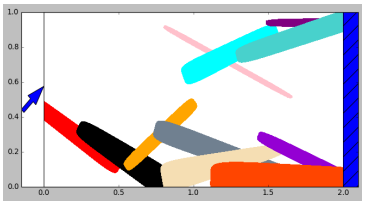
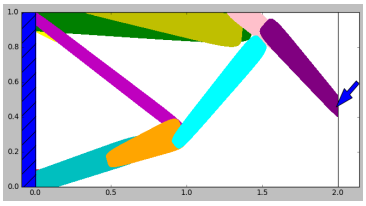
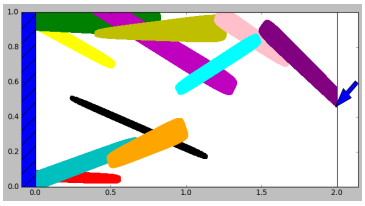
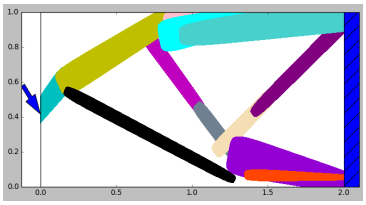
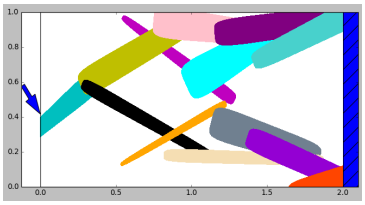
MMC Generated (Ground Truth)		Neural Network Model predictions	
			
Compliance: 67.36	Volume fraction: 50%	Compliance: 2151.51	Volume fraction: 55.4%
			
Compliance: 47.98	Volume fraction: 30%	Compliance: 2.70e9	Volume fraction: 30.50%
			
Compliance: 49.75	Volume Fraction: 30%	Compliance: 2583.36	Volume Fraction: 29.46%
			
Compliance: 56.48	Volume fraction: 35%	Compliance: 1.27e9	Volume fraction: 39.28%

Figure 4.6: Comparison of results obtained from the neural network model (right column) and the conventional MMC optimization (left column) for the 4 worst direct design prediction performances.

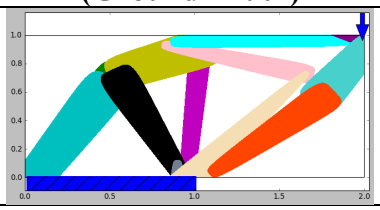
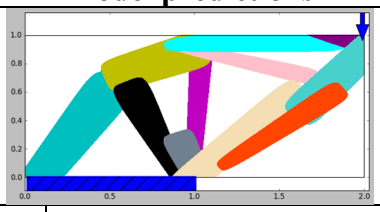
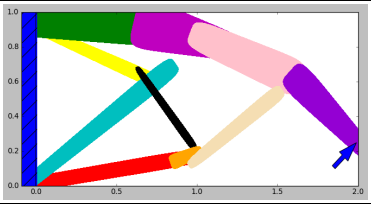
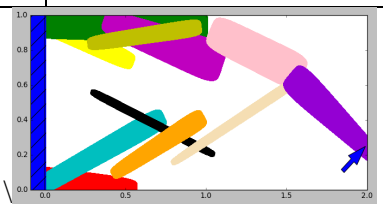
MMC Generated (Ground Truth)	Neural Network Model predictions	
		
Compliance: 39.77	Compliance: 39.79	RMSE: 0.79
		
Compliance: 34.14	Compliance: 97.35	RMSE: 0.71

Figure 4.7: Two samples from the test set showing how a lower RMSE does not translate to lower compliance. A good-performing model prediction (upper row) has a higher RMSE loss value than a bad-performing model prediction (lower row).

4.3.2 Performance of the Model as an Initial Layout Generator for MMC

As proposed in [76], the model was also tested as an initial layout generator to test if the model's predictions can provide a better starting point for conventional MMC optimization than a general initial solution where the components are uniformly dispersed in the domain as illustrated in Figure 4.8. The performance comparison between the uniform initial solution and the model's predicted initial solution was done by comparing the total number of iterations that are required to reach the convergence criterion and the final compliance of the final topology.

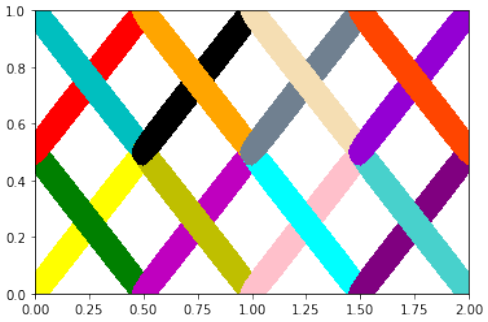


Figure 4.8: Uniform initial solution used for MMC.

The method of "performance profiles" [135] is used to provide a more global probabilistic measure of relative performance between the two methods. The core idea stands on the definition of a performance ratio ($r_{p,s}$) where, for a set of problems P and a set of solving methods S, the relative performance of a metric ($m_{p,s}$) is compared to the best-observed value across all methods as seen in Equation 4.10:

$$r_{p,s} = \frac{m_{p,s}}{min/max\left\{m_{p,s}: s \in S\right\}} \tag{4.10}$$

With the performance ratio defined for each problem p, a probability distribution of the relative performance of a method s can be defined by Equation 4.11:

$$\rho_s(\tau) = \frac{1}{n_p} size\left\{p \in P: r_{p,s} \leq \tau\right\} \tag{4.11}$$

This performance distribution can be interpreted as the probability of a performance ratio for a specific solving method s to be within a "distance factor" τ of the best possible ratio across all methods. A best-performing solving method will therefore have a high probability of being near values of $\tau = 1$.

To close the gap with the ground truth topologies in terms of mechanical performance, and for the topology to respect the desired volume constraint, the model's predictions are tested as initial solutions for the conventional MMC optimization process. The hypothesis behind the idea is that the model's predictions are closer to a local optimum in the design space than the general uniform distribution of the components generally used in MMC. In this section, this hypothesis is tested by looking at the savings in computing time based on the total number of iterations that the model's predictions can provide, as well as by comparing the compliance of the final topology with the ground truth sample on the same problem.

Figure 4.9 displays a series of side-by-side comparisons of test samples, with each column representing the ground truth, the model's prediction and the final topology generated by the optimization process when the model's prediction is used as the initial solution. While there are some differences between the ground-truth samples and the designs generated by starting from the model's predictions, these disparities are not unexpected given the significant impact of the initial

design on the local optima that an iterative optimizer can converge to. Notably, we observed that the magnitude of these differences tends to be greater for samples with larger errors between the ground-truth and models' predictions. This finding suggests that high-performing models can offer an advantage in guiding the iterative optimizer toward regions of the design space that are closer to the ground-truth local optima.

In terms of mechanical performance and respect of the volume constraint, column (b) of Table 4.3 shows that the obtained topologies with the accelerated MMC optimization process generated equally compliant topologies and respected the desired volume fraction when compared to the ground truth.

For comparison in terms of convergence speed, Table 4.4 presents the results of the per-problem comparison with conventional MMC optimization. On the entire test set, using the model's predictions as the initial solution resulted in a better convergence speed for 72% of the problems. The median iterations saving amounted to 36.84% while the average value was 14.68%.

Table 4.4: Computing time savings in terms of iteration of the conventional MMC optimization with the model-generated initial layout.

Computing Performance Metric	
% of problems with an acceleration	72.66%
Average % iterations saved	14.68%
Median % iterations saved	36.84%

The performance profiles for the final compliance and the total number of iterations until convergence are presented in Figure 4.10. One can see that the model accelerated MMC optimization showed a much higher chance of generating the least compliant structure compared to conventional MMC. For values of the distance factor τ larger than one, the two curves quickly overlap and trend toward 100% of the problems being within 5% of the best compliance observed, showing that the difference in final compliance is slim between the two methods.

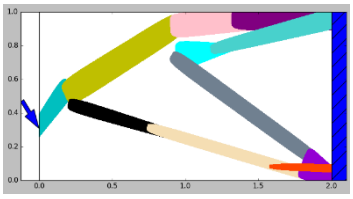
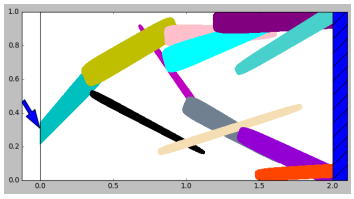
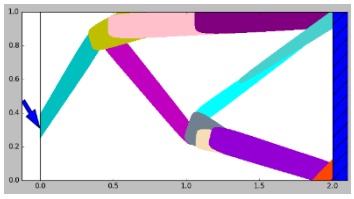
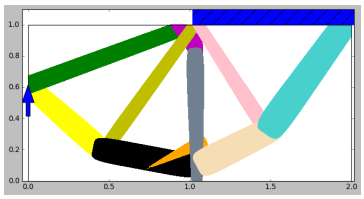
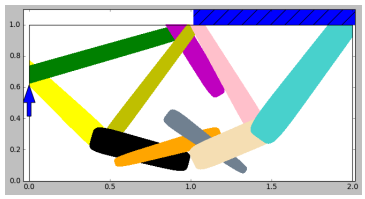
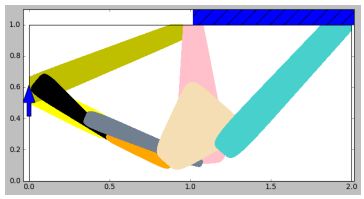
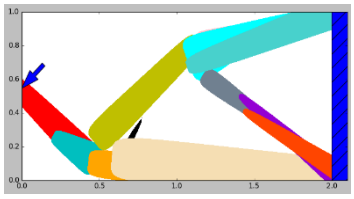
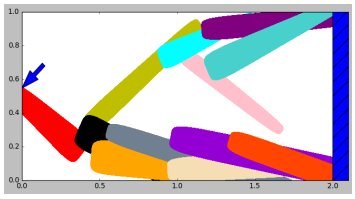
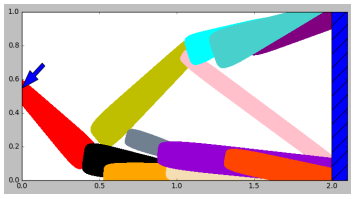
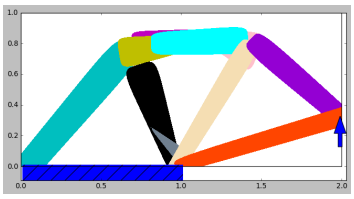
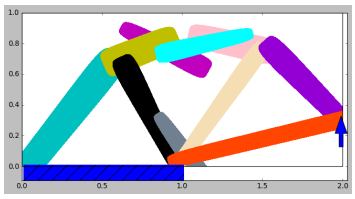
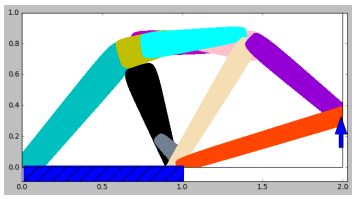
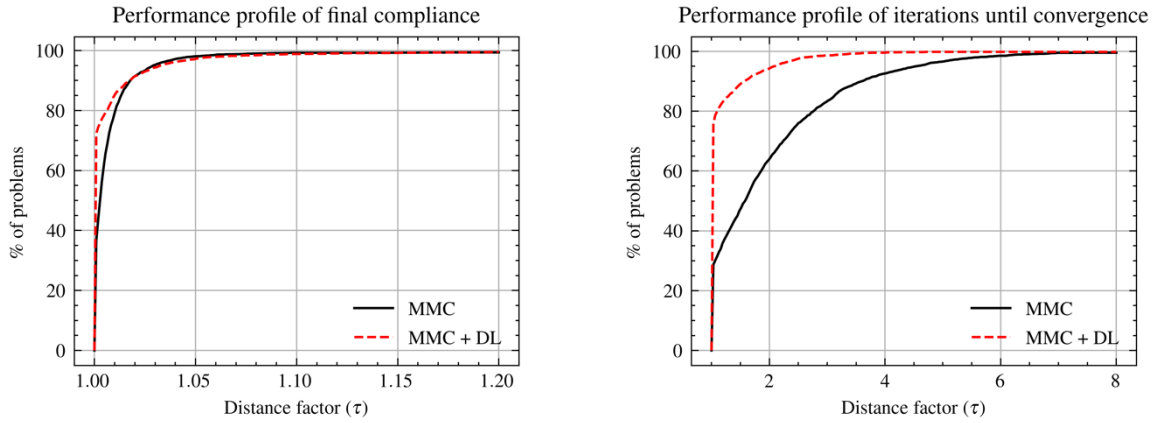
Topology obtained by MMC (Ground Truth)		**Initial layout predicted by the DL model**		**MMC optimization using model's prediction as initial layout**	
					
Compliance: 62.43	Total number of iterations: 1428	Compliance: 125.68	Total number of iterations: N/A	Compliance: 61.46	Total number of iterations: 1183
					
Compliance: 48.83	Total number of iterations: 1430	Compliance: 61.68	Total number of iterations: N/A	Compliance: 52.48	Total number of iterations: 959
					
Compliance: 38.68	Total number of iterations: 1181	Compliance: 72.26	Total number of iterations: N/A	Compliance: 38.63	Total number of iterations: 328
					
Compliance: 53.33	Total number of iterations: 527	Compliance: 56.29	Total number of iterations: N/A	Compliance: 52.97	Total number of iterations: 311

Figure 4.9: Model predictions as initial layout for conventional MMC optimization.

The large separation between the two lines in the performance profiles of the total number of iterations shows that the model-accelerated optimization process presents a much better convergence speed. Looking at the black curve of the conventional MMC optimization in Figure 4.10 b), we can see that about 65% of the test set problems reached convergence within a factor τ of less or equal to 2, implying that about 35% of the problems required over twice the number of iterations to reach convergence compared to the model-accelerated optimization process.

Overall, results show that using the model's predictions as the initial solution for a conventional optimization process can generate significant computing time savings by reducing the number of iterations required to reach convergence, without sacrificing mechanical performance.



(a) (b)

Figure 4.10: Performance profiles of the final compliance (a) and the total iterations until convergence (b) of the conventional MMC with uniform initial solution, and the model's initial solution accelerated MMC (MMC+DL). It can be observed that the model-accelerated method shows a clear reduction in total iterations by placing itself in the upper left corner of the graph (b), with a clear gap with the conventional MMC method.

4.3.3 Out-of-Distribution Boundary Conditions Performance

The dataset used in this work is very limited in terms of the possible BCs and geometries that can be encountered in 2D TO problems. While performance on our test set can be viewed as "in distribution performance", implying that the distribution of the BCs is the same as the one in the training set, it is interesting to investigate if the learned representation of the model can be leveraged on different BCs that are "Out-Of-Distribution" (OOD). Since the input description vector β does not describe the geometry, the model can't consider different geometries such as L-

beams or squared 2D domains and therefore the focus will be on testing the model on OOD BCs by placing the fully supported boundary and the load location in positions that were not part of the dataset distribution described in Figure 4.3.

In this section, the performance of the model on OOD boundary conditions is presented. A small sample of 5 randomly chosen test cases was used and is described in Table 4.5. Resulting topologies from the conventional MMC optimization process, the direct design model's predictions and the model-accelerated optimization process are shown in Figure 4.11.

Table 4.5: Randomly chosen OOD boundary conditions used.

Test Case	**Desired volume fraction**	**Location of fully supported boundary** ($\overline{x_1x_2}$, $\overline{y_1y_2}$)	**Location of the unit load** (x, y)	**Direction of the unit load**
A	40%	(0.5;1.5, 0;0)	(2.0, 1.0)	90 °
B	35%	(0.45;1.25, 1.0;1.0)	(1.5, 0)	235°
C	35%	(0;0, 0;0.5)	(2.0, 1.0)	305°
D	30%	(1.0;1.0, 0.6;1.0)	(0.45, 0)	270°
E	50%	(0.2;1.2, 0;0)	(1.5, 1.0)	0°

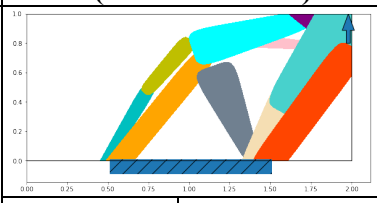
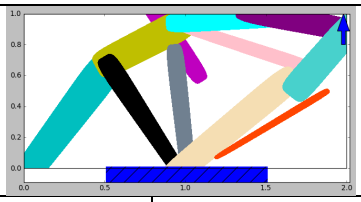
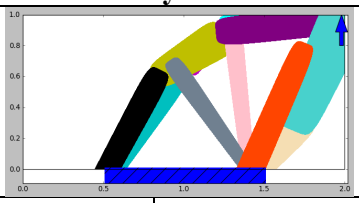
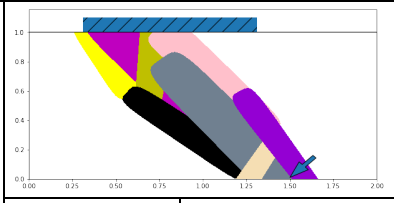
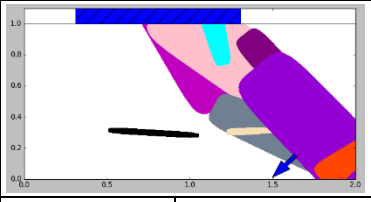
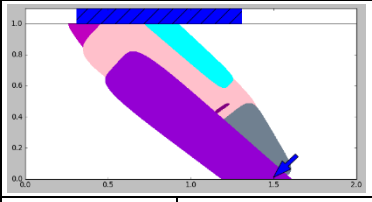
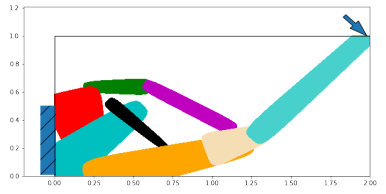
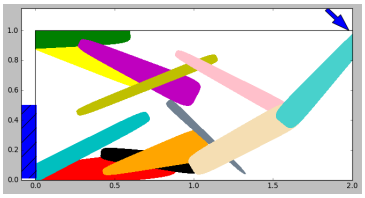
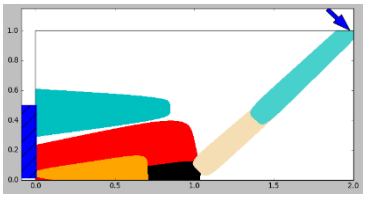
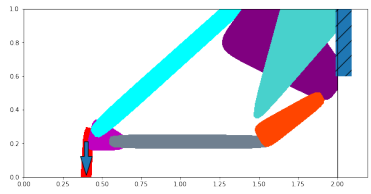
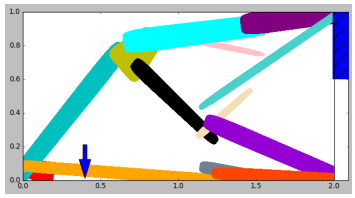
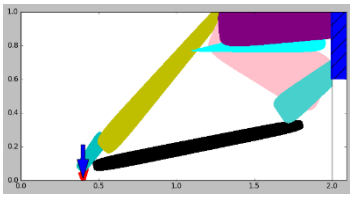
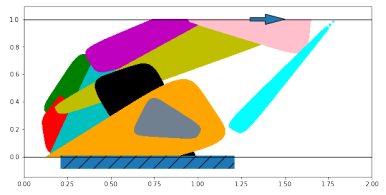
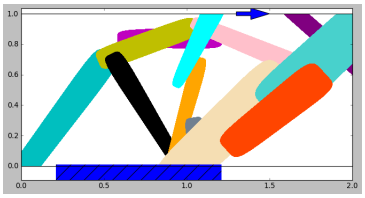
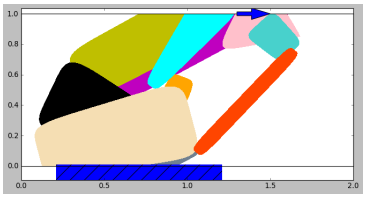
	Topology obtained by MMC (Ground Truth)		Initial layout predicted by the DL model		MMC optimization using model's prediction as initial layout	
A						
	Compliance: 14.75	Total number of iterations: 331	Compliance: 277.51	Total number of iterations: N/A	Compliance: 14.62	Total number of iterations: 270
B						
	Compliance: 5.19	Total number of iterations: 180	Compliance: 1.99e09	Total number of iterations: N/A	Compliance: 4.80	Total number of iterations: 724
C						
	Compliance: 38.87	Total number of iterations: 2918	Compliance: 1.27e09	Total number of iterations: N/A	Compliance: 40.98	Total number of iterations: 1431
D						
	Compliance: 162.10	Total number of iterations: 2912	Compliance: 7368.01	Total number of iterations: N/A	Compliance: 161.57	Total number of iterations: 1389
E						
	Compliance: 18.44	Total number of iterations: 828	Compliance: 2.48e09	Total number of iterations: N/A	Compliance: 18.33	Total number of iterations: 635

Figure 4.11: Topologies obtained for the OOD boundary conditions test.

As can be observed in Figure 4.11, the direct design model predictions performed poorly over the 5 OOD test cases in terms of compliance, resulting in structures disconnected from the load. It is also possible to conclude that the model averages the new boundary conditions to the closest support and loading conditions that it has observed in the training dataset, resulting in very similar generated topologies to those in the training and test set. Despite that, the small sample shows that using model predictions as the initial layout of the conventional MMC process resulted in faster convergence time in 4 out of the 5 cases. Test case B is the only one where using the model prediction as the initial starting point resulted in almost 3 times more iterations, but with a final compliance 7.5% lower than the ground truth. Finally, it is also interesting to note that the model-accelerated method converged to slightly different local optimums, resulting in different components placement.

4.4 Discussion

In this paper, a deep learning model is trained in a supervised manner to directly generate the design variables of the MMC framework. One advantage of this method over the popular convolutional neural networks (CNN) pixel-based methods of ML-accelerated TO literature is that the output of the model can be directly interpretable. The second advantage is that the neural network architecture used makes our model entirely independent of the resolution of the mesh used for the finite element analysis of the structure. A user can therefore decide to use an arbitrary mesh resolution without affecting the number of parameters of the model and the computing performance during training and inference of the model. This is not true for vision-based approaches whose computing requirements scale with the resolution of the mesh used to parameterize the topology.

While this paper aims to place a spotlight on the potential of DL and MMC topology optimization, the goal is also to express the important limitations of this very young research field. The supervised training is framed as a regression task between ground truth design variables and the model's predictions, and therefore lacks a feedback loop that grounds the model into the physics of the underlying problem. In other words, the model has no incentive to minimize the compliance of the generated structure which is the core objective of the task at hand. Trying to solve TO through this "inverse problem" shows impressive computing speed, but a lackluster performance for the generated topologies and poor generalization capability which currently makes it unviable for real-world use. By looking at other fields where deep learning was successful in practice, large-scale

datasets are probably the missing variable for direct design supervised learning approaches. Considering the high computing cost of generating such datasets for TO, it is improbable that such methods will have a real-world application in the near term.

Therefore, supervised learning models show greater potential when combined with a conventional optimization process. Numerical results have shown that model predictions can help jumpstart the optimization process by initiating the gradient-based optimizer from a solution that is closer to a local optimum. The local optimums where the model-accelerated optimization process converges are also marginally better or equal in terms of compliance than the ones obtained by starting from a general uniform initial layout. Convergence speed increases were also observed on the small-scale OOD generalization test, where the direct design approach performed poorly.

The proposed model architecture has certain drawbacks that require discussion. Notably, the fixed length of the output layer limits the complexity of the generated topologies, restricting the maximum number of structural components to 16. Consequently, topologies with a higher number of structural components would require retraining a new output layer with the possibility of transferring the other network parameters if the input layer does not need to change. However, the input layer fails to consider the domain's geometry, representing an additional limitation. Possible solutions to this issue include leveraging shape description methods such as chord-length functions [136] to enable a fixed dense representation of domains with varying shapes. This is still an ongoing research subject and will be implemented in future work. The proposed model's limitations regarding the input and output layers have implications for the computational efficiency of the method. Specifically, increasing the size of the input or output layer would increase the training and inference time in proportion to the increase in model parameters. Therefore, while expanding these layers could potentially overcome the representation limitations stated above, it would come at the cost of additional computational resources.

There is a long way to go for deep learning models to be useful in practice for TO. For models trained purely on design variable regression to perform "in the wild" on new, never-before-seen TO problems, the description vector used as an input needs to be able to describe a much larger range of domain representation and the distribution of the training dataset needs to encompass a much wider and diverse set of boundary conditions. In its essence, supervised learning for the

MMC framework is as much a challenge of training sample generation, as it is one of representation learning.

4.5 Conclusion

The research field of topology optimization aims to provide a numerical tool for engineers to design structures more efficiently in terms of time and performance. To lower the computing cost of large-scale topology optimization, machine-learning methods can leverage data representation learnt from past solutions to accelerate the optimization process. This research work presented a novel deep learning model trained to directly generate the design variables of the moving morphable components topology optimization framework.

The model was trained via a regression task on a dataset more than 10 times larger than the current state-of-the-art in deep learning-accelerated moving morphable components [77]. The dataset is composed of optimal topologies obtained via the method of moving morphable components and with a set of diverse volume fractions and boundary conditions. While presenting a state-of-the-art regression error on the test set, the compliance difference of +11.48% and the volume difference of +5.03% is 2 to 3 times larger than what is observed in the literature. It is the authors' belief that since the current literature is limited to a single test case of a cantilever beam with a very narrow variation in vertical load position, results on the more diverse dataset presented in this paper provide a much more realistic performance evaluation of such supervised learning models.

This paper also presented the first use of a deep learning model to generate an initial solution for a conventional moving morphable components optimization process. The model's prediction showed the potential to increase the convergence speed of a hybrid method of moving asymptotes optimizer, resulting in an average iteration saving of 14.68% and a median saving of 36.84%. Similarly, a small-scale test showed the potential to obtain such compute time savings on boundary conditions outside of the training dataset distribution.

The reduction of computational costs associated with topology optimization represents a fundamental advancement in design exploration capabilities for engineers and designers alike. By minimizing the economic cost barrier, topology optimization becomes more accessible to a broader range of entities, allowing them to explore a wider range of design options with increased speed and efficiency. As such, the reduction of the cost barrier for topology optimization can have a

significant impact on product performance, cost-effectiveness, and time-to-market, particularly in sectors such as the automotive and aerospace industries.

For future research, the flexibility of the proposed method will be exploited by widening the distribution of the training dataset with the addition of arbitrary 2D geometries and randomized boundary conditions and volume fractions. The goal of broadening the dataset is to increase the out-of-sample and out-of-distribution performance of the model. The proposed method can also be extended to three-dimensional problems, where the exponentially higher computing cost of the conventional optimization process could make the model much more attractive to leverage. Additionally, the model can also be used as a "pre-trained" reinforcement learning policy aiming to directly minimize the compliance of the structure, enabling the framework to be grounded in the physics of the topology optimization task.

This paper makes several key contributions:

- A large-scale study is presented that evaluates the performance of a deep learning model trained to solve the inverse TO problem under the MMC framework through a regression task. Results are presented on a dataset over 10 times larger than the existing ML-accelerated topology optimization literature.
- Major limitations of the supervised learning direct design approach for TO are highlighted. Specifically, the incompatibility of the regression loss function typically used and the TO objective is shown, revealing a fundamental challenge of the field.
- To address this incompatibility, a novel approach inspired by [76] is proposed. The supervised learning model is used as an initial design generator to jumpstart the conventional optimization process. This approach enables significant compute time savings of 36.84% without sacrificing the mechanical performance of the obtained structure.
- The performance of direct design approaches for the MMC framework is evaluated for out-of-distribution boundary conditions, which represents a new and challenging scenario. The study demonstrates the generalization potential of the novel model-accelerated optimization method.

CHAPTER 5 ARTICLE 2: Structural Design Through Reinforcement Learning

Thomas Rochefort-Beaudoin[a], Aurelian Vadean[a], Niels Aage,[b] Sofiane Achiche[a]

[a]*Department of Mechanical Engineering, Polytechnique Montréal, 2500 Chemin de Polytechnique, Montréal, QC, Canada*

[b]*Department of Civil and Mechanical Engineering, Solid Mechanics, Technical University of Denmark. Koppels Allé, B.404, 2800 Kgs. Lyngby, Denmark*

Submitted to *Engineering Applications of Artificial Intelligence* on July 12th, 2024

Abstract

This paper introduces the Structural Optimization gym (SOgym), a novel open-source Reinforcement Learning (RL) environment designed to advance machine learning in Topology Optimization (TO). SOgym enables RL agents to generate physically viable and structurally robust designs by integrating the physics of TO into the reward function. To enhance scalability, SOgym leverages feature-mapping methods as a mesh-independent interface between the environment and the agent, allowing efficient interaction with the design variables regardless of mesh resolution. Baseline results use a model-free Proximal Policy Optimization agent and a model-based DreamerV3 agent. Three observation space configurations were tested. The TopOpt game-inspired configuration, an interactive educational tool that improves students' intuition in designing structures to minimize compliance under volume constraints, performed best in terms of performance and sample efficiency. The 100M parameter version of DreamerV3 produced structures within 54% of the baseline compliance achieved by traditional optimization methods and a 0% disconnection rate, an improvement over supervised learning approaches that often struggle with disconnected load paths. When comparing the learning rates of the agents to those of engineering students from the TopOpt game experiment, the DreamerV3-100M model shows a learning rate approximately four orders of magnitude lower, an impressive feat for a policy trained from scratch through trial and error. These results suggest RL's potential to solve continuous TO problems and its capacity to explore and learn from diverse design solutions. SOgym provides a platform for developing RL agents for complex structural design challenges and is publicly available to support further research in the field.

5.1 Introduction

In structural design, the task of Topology Optimization (TO) seeks to distribute a limited amount of material within a given design space under specific design constraints and objectives. Density-based methods [26], and in particular the SIMP method [137], which parameterize the topology of a structure using penalized individual elemental densities in a discretized finite element mesh, became the standard TO framework thanks to their robustness, simplicity of implementation [59], and adaptability to various manufacturing constraints [138] and optimization objective [139]. TO is an iterative procedure where, most often, Finite Element Analysis (FEA) is used to compute the structural response of a given topology under specific boundary conditions. At each iteration, a sensitivity analysis is performed to obtain the gradient of the optimization objective as a function of the design variables.

The application of TO shows potential for reducing the material and energy consumption of large-scale structures such as suspension bridges [10] and entire airplane wings [9], presenting engineers with more sustainable designs in big energy-hungry industries such as transport and infrastructure. Although both FEA and adjoint sensitivity analysis can be performed highly efficiently on massively parallel computing systems, the sparse access to such facilities still curbs the widespread use of the TO method.

Optimizing for nonlinear multi-physics is especially important if the structures are to be manufactured through Additive Manufacturing (AM) which often generates lattice or cellular structures that are more prone to nonlinear deformation such as local buckling and inelastic yield [140]. These physics can be implemented in TO frameworks, but the additional computational cost associated with them escalates substantially, limiting their broader applicability [141].

Motivated to lower the cost barrier of large-scale and high-fidelity TO, researchers have proposed various methods to reduce the number of total iterations required or mitigate the computational expenditure associated with each iteration [15]. Over the past few years, the introduction of machine learning-based methodologies to expedite TO has been noticeably prevalent. Rapid advances in the subfield of deep learning, a branch of machine learning, stimulated a gold rush-like interest in the deployment of its methodologies across a wide range of applications. The parameterization of the TO density field as pixel/voxels has naturally brought many researchers to flock to image-based deep learning methods to try to accelerate the TO process as a low-hanging

fruit. Most of these approaches propose a standard supervised learning pipeline of generating optimal topologies, and then training a computer vision model such as a convolutional neural network via a direct regression loss, or in an adversarial context.

The training loss functions used in these supervised learning methods only optimize indirectly for the TO objective and do not consider the actual physics of the TO problem. In previous work, the incompatibility between a regression loss function on the design variables and the TO objective has been demonstrated, showing that a common failure mode of such models is to generate structures with a disconnected load path, and the inability to treat boundary conditions outside the training set [18, 142].

To tackle this issue, deep learning approaches for TO need to be enriched with the underlying physics of the problem [18]. Research has been conducted to reparametrize the density field in TO problems and to integrate the finite element analysis (FEA) operation directly into the model's loss function, enabling backpropagation through the entire structural analysis [82, 143]. This necessitates the use of sparse solvers in auto-differentiation software, which are currently only experimentally supported in popular deep learning libraries. Consequently, the cost of backpropagation scales not only with the model size but also with the size of the FEA mesh. Surprisingly, all these studies also omit the potential for the structure to be generated in a single forward pass through the model, instead concentrating on achieving an optimal structure by updating the parameters for each new problem through gradient descent on the model's parameters. Thus, this paradigm presents more as an alternative to typical optimizers, rather than offering a means to leverage past solutions for solving the current problem.

In this paper, the standard compliance minimization objective of TO is framed as a reinforcement learning (RL) task to train a deep learning agent to generate structures as efficiently as possible for various Boundary Conditions (BC). Unlike supervised learning, which is limited by the availability, diversity, and quality of training data, RL operates on the premise of trial-and-error, learning to make decisions based on the maximization of cumulative rewards over time.

Framing TO as a RL task draws inspiration from the interactive TopOpt game [144]. In this game, developed at the Technical University of Denmark, players use a user interface to design structures that maximize a score based on compliance minimization while adhering to volume constraints. The game's effectiveness was evaluated in a classroom setting with undergraduate design

engineering students. The study aimed to explore the potential of gamifying TO, and results demonstrated a positive correlation between the players' experience and the structural performance of their solutions, suggesting that students' intuition in TO improved with practice.

The literature on using RL for TO is sparse, with only two significant studies addressing this approach. The first study [101] introduced a method using a graph representation of a binary truss topology, where an RL agent was trained to remove unnecessary truss components from a fully connected graph, showing generalization capabilities to various boundary conditions and larger grid resolutions. However, it is worth noting that full FEA is still performed in each iteration and hence, that the speedup is most likely related to an improved design space representation. The second study [103] extended this approach to discretized 2D domains using a fixed grid mesh. Here, an RL agent iteratively removed individual pixels from a fully saturated 2D material mesh until reaching a sub-optimal solution. While these methods share similarities with evolutionary TO algorithms, they are constrained by the limited action space of manipulating individual elements, making it challenging to scale to higher mesh resolutions due to the exponential increase in computational complexity.

This paper introduces the Structural Optimization gym (SOgym), an open-source reinforcement learning (RL) environment. SOgym provides a platform for developing and training RL algorithms specifically for structural design. It recreates the TopOpt game's scoring system and user interface, allowing for direct comparisons between the learning rates of engineering students and trained RL agents. Baseline performance results for standard RL algorithms are also included, offering a foundation for future research.

5.2 Method

SOgym is based on the Gym [145] framework and defines the interactions between the agent and the environment. It specifically outlines three key components:

- The action space: a set of continuous actions that the agent can take, corresponding to the design variables of the topology.
- The observation space: a representation of the current state of the TO problem.
- The reward function: feedback provided to the agent based on the performance of its generated structure.

5.2.1 Action Space

This paper uses feature-mapping methods [27] as a more scalable and interpretable interface for RL agents to interact with the material distribution of the TO problem, in comparison to using an elemental density action space similar to density-based TO methods, as implemented in [101] and [103]. This approach offers two key advantages: (1) the number of design variables remains independent of the mesh resolution, enhancing scalability for the action space of the agent, and (2) the design variables exhibit geometric interpretability, which can streamline the post-processing phase of the optimal topology.

SOgym uses a modified implementation of the 188-line Method of Moving Morphable Components (MMC) code [112]. In this approach, the endpoints' coordinates and two thicknesses define a component with a linearly varying thickness, as illustrated in Figure 5.1 and detailed in Equations 5.1 through 5.7. The RL agent sequentially places MMC components in the design space to create the stiffest structure possible within a volume constraint. The maximum number of steps per episode, equivalent to the number of components, is a user-defined variable controlling the structure's complexity. For the baseline experiments in this paper, a maximum of 8 components was used. An episode involves iteratively placing these 8 components, with FEA performed 'per step' or 'per episode' for compliance evaluation, depending on the observation space configurations detailed later in this section. This framework can be scaled to larger models by increasing the number of components, enabling the exploration of more complex structures.

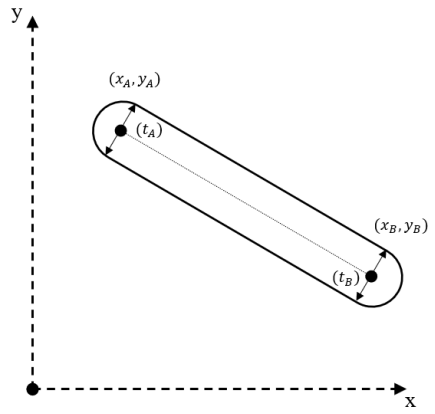


Figure 5.1: MMC component defined by endpoint coordinates and thicknesses.

$$x_0 = \frac{x_A + x_B}{2}, y_0 = \frac{y_A + y_B}{2} \tag{5.1}$$

$$L = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2} \tag{5.2}$$

$$\theta = arctan2\,(B_y - A_y\,, B_x - A_x) \tag{5.3}$$

$$x_1 = \cos(\theta)\,(X - x_0) + \sin\,(\theta)(Y - y_0) \tag{5.4}$$

$$y_1 = -\sin(\theta)\,(x - x_0) + \cos(y - y_0) \tag{5.5}$$

$$t(x_1) = \frac{(t_A + t_B)}{2} + \frac{(t_B - t_A)}{2L}(x_1) \tag{5.6}$$

$$\Phi(x_1, y_1) = 1 - \left(\left(\frac{x_1}{\frac{L}{2}}\right)^6 + \left(\frac{y_1}{t}\right)^6\right)^{1/6} \tag{5.7}$$

x_0, y_0, L and θ are the coordinates of the centroids, the length, and the orientation of the MMC component, defined by the endpoint coordinates x_A, y_A, x_B and y_B. A linearly varying thickness $t(x_1)$ is defined by interpolating between the two endpoint thicknesses t_A and t_B. These geometric variables define the shape of the hyperelliptic function $\Phi(x_1, y_1)$ named the Topology Description Function (TDF). The density distribution on the discretized FEA mesh is obtained using a regularized Heaviside (H_e) projection of the TDF, and the ersatz material model is used to define the elemental densities ρ_e and Young's modulus E_e. The projection scheme and the ersatz model are described in equations 5.8 through 5.10. A regularization parameter ϵ is used with a value of 1e-2, and a small positive value α of 1e-9 helps avoid numerical singularities with the stiffness value of void elements.

$$H_e(\Phi_i^e) = \begin{cases} 1, & if\ \Phi_i^e > \epsilon \\ \frac{3(1-\alpha)}{4}\left(\frac{x}{\epsilon} - \frac{x^3}{3\epsilon^3}\right) + \left(\frac{1+\alpha}{2}\right), & if\ |\Phi_i^e| \leq \epsilon \\ \alpha & otherwise \end{cases} \tag{5.8}$$

$$\rho_e = \frac{1}{4}\sum_{i=1}^{4} H_e(\Phi_i^e) \tag{5.9}$$

$$E_e = \rho_e E \tag{5.10}$$

The continuous action space of the agent is a vector of 6 elements with values normalized between [-1,1] corresponding to the two endpoint coordinates as well as the two endpoint thicknesses. The actions are scaled to the design space using min/max values defined in Table 5.1.

Table 5.1: Minimum and maximum boundaries of the design variables.

Design variable	Definition	Minimum value	Maximum value
x_a	X-coordinate of endpoint a	0.0	Width of domain
x_b	X-coordinate of endpoint b	0.0	Width of domain
y_a	y-coordinate of endpoint a	0.0	Height of domain
y_b	y-coordinate of endpoint a	0.0	Height of domain
t_a	Thickness of endpoint a	0.01	5% min(h,w)
t_b	Thickness of endpoint b	0.01	5% min(h,w)

5.2.2 Boundary Conditions

Instead of relying on the typical train-test set of Boundary Conditions (BC) common in machine learning TO literature, SOgym introduces a large, random distribution of BC at the beginning of each episode. This distribution is extensive enough to make encountering every possible combination impractical within a reasonable training timeframe. As a result, the agent consistently faces new and varied challenges, encouraging the development of generalized problem-solving skills instead of overfitting to a limited set of problems.

A 2D rectangular domain was used where support and load conditions, desired volume fractions, and domain dimensions were varied to generate multiple unique parameter combinations, bringing sufficient diversity during training. The distribution of parameters, which were all sampled from uniform distributions with ranges specified in the third column, is illustrated in Figure 5.2 and detailed in Table 5.2. The support condition is placed randomly on one of the four boundaries without spanning multiple boundaries, covering only a fraction of the chosen boundary at a time. The point load is then placed randomly on the opposite boundary, with a random orientation and a unit load magnitude. A coarse mesh resolution of 50 elements per unit of height and width was set for the FEA to limit the computational cost of running the environment.

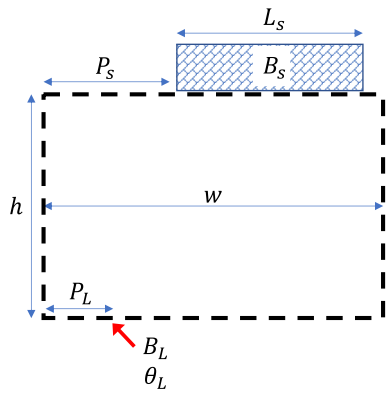


Figure 5.2: Boundary conditions parameters defining the random distribution

Table 5.2: SOgym environment parameters

Parameter	Description	Possible values
L_s	Length of support	[25%, 100%] of boundary
B_s	Boundary of support	Left, Right, Top, Bottom
P_s	Position of support	[0%, 100%-L_s]
B_L	Boundary of the load	Opposite of B_s
P_L	Position of the point load	[0% ,100%] of boundary
θ_L	Orientation of load	[0°, 360°]
h, w	Height, Width of domain	[1.0, 2.0]
V	Desired volume fraction	[20%, 40%]

5.2.3 Observation Space

SOgym offers three observation space configurations: Vector, Image, and a simplified imitation of the TopOpt game interface, as detailed in Table 5.3.

- Vector configuration: It includes a boundary conditions description vector ($\boldsymbol{\beta}$), based on [142] and detailed in Table 5.4. This vector encapsulates all the parameters defining the current TO problem. Although our baseline experiments use a single point load and a fully

supported support type, the β vector was designed to handle up to five-point loads and multiple support types. Additionally, this configuration includes:

  - A fraction indicating the number of remaining steps in the episode,
  - A vector containing the values of all the current design variables,
  - A fraction representing the current volume.

- Image configuration: It adds a low-resolution image of the current design, illustrating support and load arrows.
- TopOpt game configuration: It adds the current score and an image of the normalized strain energy in a jet color scheme, only shown if the load path is connected. Figure 5.3 illustrates this configuration at the end of an episode with random actions and a connected final structure.

The *Vector* configuration is more compact, while the *Image* and *TopOpt* game configurations offer richer observations at the expense of higher computational costs. The *TopOpt* game configuration demands solving the displacement field at every iteration, unlike the others, which only do so at the episode's end. This complexity enriches the RL agent's training data but slows down training due to extra computations and solver calls. The trade-off between the richness of the observation space and the computational efficiency of the environment is an important consideration that will be further explored and analyzed in the results section of this paper.

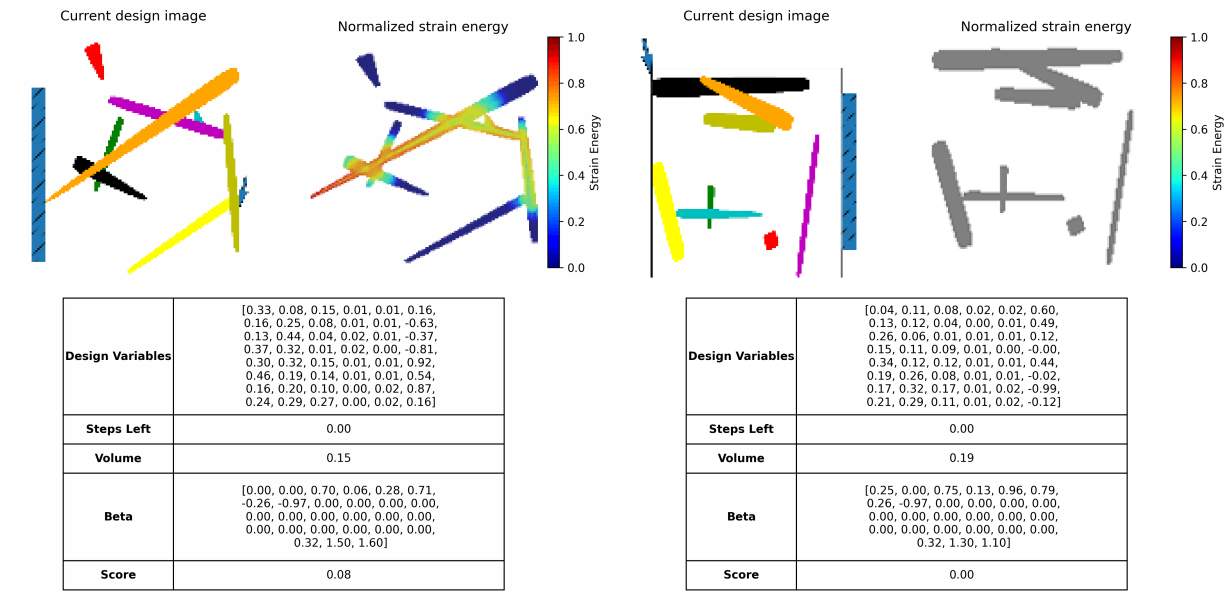
Table 5.3: Supported observation space configurations of the SOgym environment.

			Observation Space Configurations		
Name	**Description**	**Shape**	***Vector***	***Image***	***TopOpt Game***
β	Boundary conditions description vector	(27,1)	✓	✓	✓
# steps left	Number of steps (components) left for the problem	(1,1)	✓	✓	✓
Design variables	Vector with all the design variables of the current design	$(6 \cdot t_{max},1)$	✓	✓	✓
Volume	Current volume at iteration t	(1,1)	✓	✓	✓
Image	Low-resolution image of the current design with distinct color MMC components	(3,64,64)		✓	✓

Strain energy	The logarithm of the current strain energy plotted using the jet color scheme	(3,64,64)	✓
Score	Current score at iteration t	(1,1)	✓

Table 5.4: Composition of the β description vector.

Component	Sub-Component	Details	Index Range
Support conditions	Selected Boundary	Boundary selected for support	1
	Support Type	Type of support applied	2
	Boundary Length	Length of the boundary	3
	Boundary Position	Position of the boundary	4
Load conditions	Load Positions	Position of loads	5-9
	Load Orientations	Orientation of loads	10-14
	Magnitudes x	x-component of point load magnitudes	15-19
	Magnitudes y	y-component of point loads magnitudes	20-24
Volume fraction	Volume fraction	Volume fraction constraint	25
Domain shape	Width	Domain width (x-dimension)	26
	Height	Domain height (y-dimension)	27



Design Variables	[0.33, 0.08, 0.15, 0.01, 0.01, 0.16, 0.16, 0.25, 0.08, 0.01, 0.01, -0.63, 0.13, 0.44, 0.04, 0.02, 0.01, -0.37, 0.37, 0.32, 0.01, 0.02, 0.00, -0.81, 0.30, 0.32, 0.15, 0.01, 0.01, 0.92, 0.46, 0.19, 0.14, 0.01, 0.01, 0.54, 0.16, 0.20, 0.10, 0.00, 0.02, 0.87, 0.24, 0.29, 0.27, 0.00, 0.02, 0.16]
Steps Left	0.00
Volume	0.15
Beta	[0.00, 0.00, 0.70, 0.06, 0.28, 0.71, -0.26, -0.97, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.32, 1.50, 1.60]
Score	0.08

Design Variables	[0.04, 0.11, 0.08, 0.02, 0.02, 0.60, 0.13, 0.12, 0.04, 0.00, 0.01, 0.49, 0.26, 0.06, 0.01, 0.01, 0.01, 0.12, 0.15, 0.11, 0.09, 0.01, 0.00, -0.00, 0.34, 0.12, 0.12, 0.01, 0.01, 0.44, 0.19, 0.26, 0.08, 0.01, 0.01, -0.02, 0.17, 0.32, 0.17, 0.01, 0.02, -0.99, 0.21, 0.29, 0.11, 0.01, 0.02, -0.12]
Steps Left	0.00
Volume	0.19
Beta	[0.25, 0.00, 0.75, 0.13, 0.96, 0.79, 0.26, -0.97, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.32, 1.30, 1.10]
Score	0.00

a) b)

Figure 5.3: Illustration of the TopOpt observation space. a) Example of a random episode leading to a connected final structure. b) Example of a random episode leading to a disconnected structure where the strain energy is not shown.

5.2.4 Reward Function

Like the TopOpt game, a sparse reward is given only at the final timestep (t_{max}) and corresponds to the inverse of the compliance with a hard constraint on volume and load-path connectivity. The function sets the reward to 0 if the structure's final volume exceeds the defined constraint or if the load and support are disconnected. Since the magnitude of the compliance can vary widely depending on the boundary conditions and the generated structure, SOgym uses the natural logarithm of the compliance to squash the reward signal. This reward-squashing strategy, shown in Figure 5.4, has been demonstrated to enable RL algorithms to generalize better across diverse environments [95]. Equation 5.11 shows the SOgym reward given at t_{max} where V^* is the volume constraint, $V_{t_{max}}$ is the volume at the final timestep and $K_{t_{max}}$ is a boolean variable that is equal to 1 if the load-path of the final structure is connected.

$$r_{t_{max}} = \begin{cases} \dfrac{1}{\log_e(compliance_{t_{max}})}, & \Leftrightarrow \left(V_{t_{max}} \leq V^* \; and \; K_{t_{max}} = 1\right) \\ 0, & elsewhere \end{cases} \tag{5.11}$$

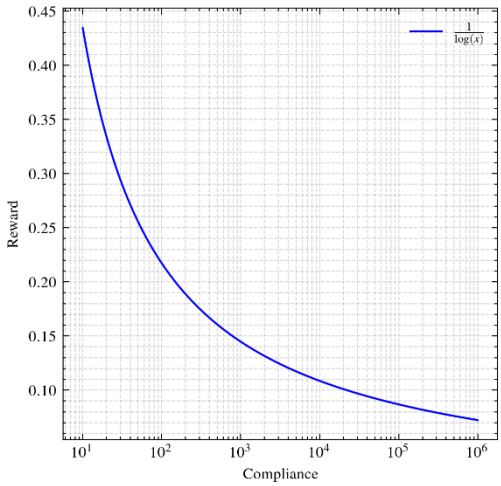


Figure 5.4: Squashed SOgym reward using the inverse of the natural logarithm of the compliance.

5.2.5 RL Algorithms

Two RL algorithms, one model-free and one model-based, are trained and tested to establish an initial performance baseline for SOgym. Proximal Policy Optimization (PPO) [94] is a popular model-free, on-policy algorithm known for its simplicity, robustness, and successful applications in various domains, including chip design [96], defeating world champions in complex esports [88], and human alignment of large language models [146]. DreamerV3 [95], a model-based RL algorithm, is chosen for its innovative blend of policy optimization and world model learning. It utilizes a learned world model to create imagined trajectories, predicting environmental responses to specific actions. The world model consists of three key components: a representation model that compresses observations into latent states, a transition model that predicts the next latent state based on the current state and action, and a reward model that estimates rewards from these latent states. This approach enables DreamerV3 to simulate future outcomes, plan strategically, and explore the environment more efficiently, resulting in improved sample efficiency in most RL benchmarks.

5.2.6 Model Architecture

Two versions of DreamerV3 were trained: the smallest model, with 12 million parameters, and a larger model with 100 million parameters. The 12M parameter model was chosen because it was the smallest tested in [95], providing a baseline for comparison. The 100M parameter model was

selected as it represents a middle-ground model, balancing complexity and performance. Details about the model architectures and default hyperparameters used for the baseline results of this paper are provided in [95]. For the PPO algorithm, the IMPALA CNN network architecture [147] was chosen due to its effective balance of performance and computational cost [148]. The CNNs used as a feature extractor for the image inputs are detailed in Figure 5.5. Vector inputs are processed through a single fully connected layer to extract features. These feature embeddings are concatenated and passed to an LSTM layer, which then feeds into both the policy and value networks. The architecture illustrated in Figure 5.5 is repeated for the policy and the value networks, as they do not share feature extractors. The PPO hyperparameters used in [95] are used as they were selected based on recommendations for on-policy RL algorithms [149].

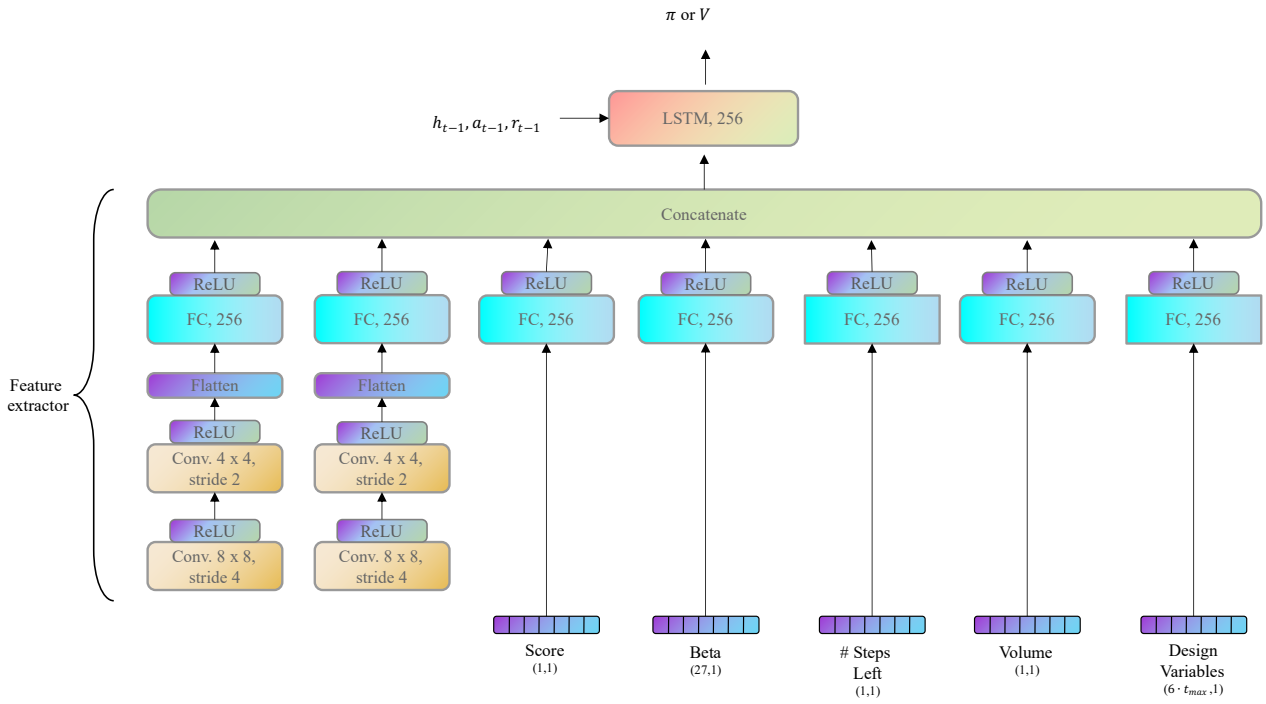


Figure 5.5: Actor-critic IMPALA-inspired architecture of the PPO model.

5.3 Results

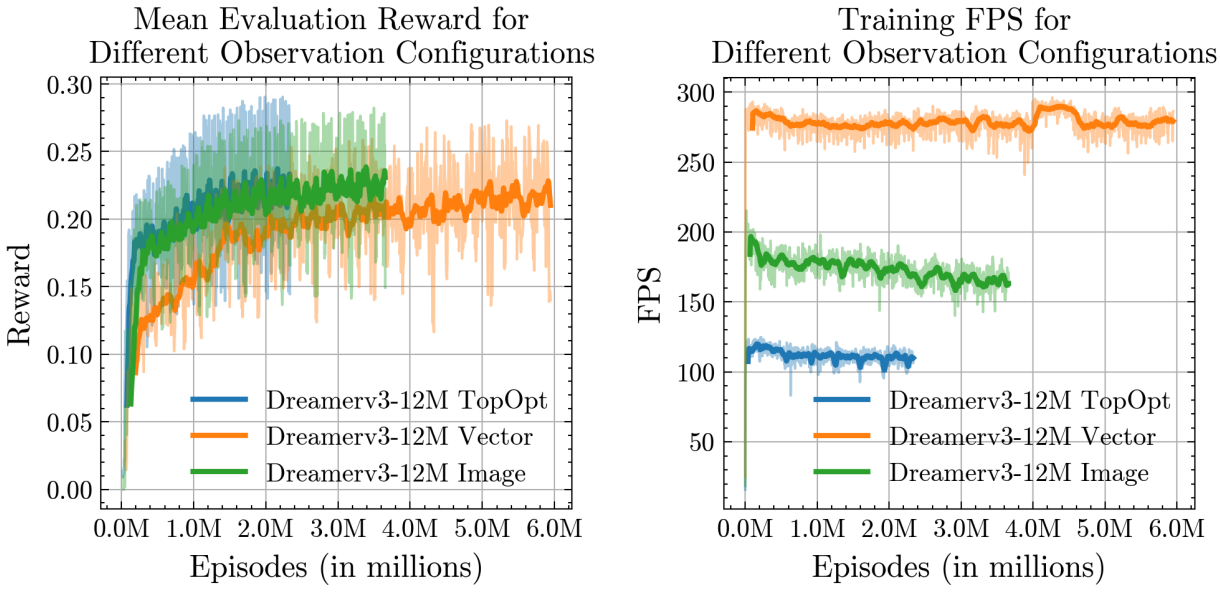
To monitor the models' performance during training, a fixed set of 10 problems was randomly sampled from the BC distribution to serve as evaluation benchmarks. This number was kept small to limit the impact on the training duration. At regular intervals during training, the agent evaluates its performance by greedily sampling actions across these 10 problems. The mean evaluation

reward from these runs is used to assess the performance across the three observation space configurations and the two baseline algorithms. This evaluation reward was also used to calculate a learning rate on the same basis as the in-class experiment from the TopOpt game.

All training runs were conducted on a single compute node equipped with an Nvidia A100 GPU and a 48-core CPU. The requirement for FEA to evaluate rewards makes the environment CPU-bound. To mitigate this bottleneck 48 parallel copies of the environment and policy are run simultaneously to aggregate experiences between gradient updates efficiently.

5.3.1 Evaluation of Performance Across Different Observation Spaces

The impact of the observation space was first tested on the Dreamerv3-12m model since it was the smallest model tested in terms of the number of trainable parameters. Due to computational constraints, the first training runs with DreamerV3-12M are limited to 48 hours. As illustrated in Figure 5.6a) and Figure 5.6b), the Vector observation space configuration completed just under 6M training episodes and reached a best evaluation reward of 0.27 for an average of 279 Frames Per Second (FPS), while the Image configuration completed around 3.75M episodes with a best evaluation reward of 0.28 with an average FPS of 173. The TopOpt game configuration showed a much slower FPS of 112, but it managed to reach a higher maximum evaluation reward of 0.29 in just under 2.5M episodes.



a) b)

Figure 5.6: Performance comparison of different observation spaces for DreamerV3-12M model. a) Mean evaluation reward: TopOpt (blue) achieves the highest reward of 0.29 in around 2.3M episodes. Image (green) achieves a maximum reward of 0.28 in around 3.75M episodes, while Vector (orange) reaches 0.27 after nearly 6M episodes. b) Training Frames Per Second (FPS): Vector (orange) maintains the highest FPS at 279, followed by Image (green) at 173 FPS. TopOpt (blue) has the lowest FPS at 112.

5.3.2 DreamerV3 vs PPO

The TopOpt observation space was used as a benchmark to compare the performance of the DreamerV3-12M, DreamerV3-100M, and PPO agents. To investigate whether increasing the training budget could enhance performance, as the DreamerV3-12m evaluation reward in Figure 5.6a) had not plateaued after the initial 48-hour time budget, an additional 96 hours of training time was allocated, bringing the total training time for each algorithm to 6 days. Figure 5.7 illustrates the evaluation rewards throughout the training period for the three algorithms. Both versions of DreamerV3 performed better than the PPO agent, with maximum evaluation rewards reaching 0.30 for the 100M parameter version, and 0.29 for the 12M parameter version. They showed a rapid increase in evaluation reward and a slower rate of learning after about 1M episodes. The PPO agent showed a slower rate of increase in evaluation rewards, reaching a maximum value of 0.19 before suffering from a significant episode of instability, causing the evaluation reward to drop by more than 25%. However, this performance drop was mitigated by saving the best-performing version of the model during training. This ensured that even if the model's performance declined later, the best result achieved up to that point was preserved. This instability coincided with a spike in Kullback-Leibler (KL) divergence, as can be observed in Figure 5.8, which measures the divergence between the new and old policy distributions, providing insight into the algorithm’s learning stability. An increase in KL divergence indicates potential training instability.

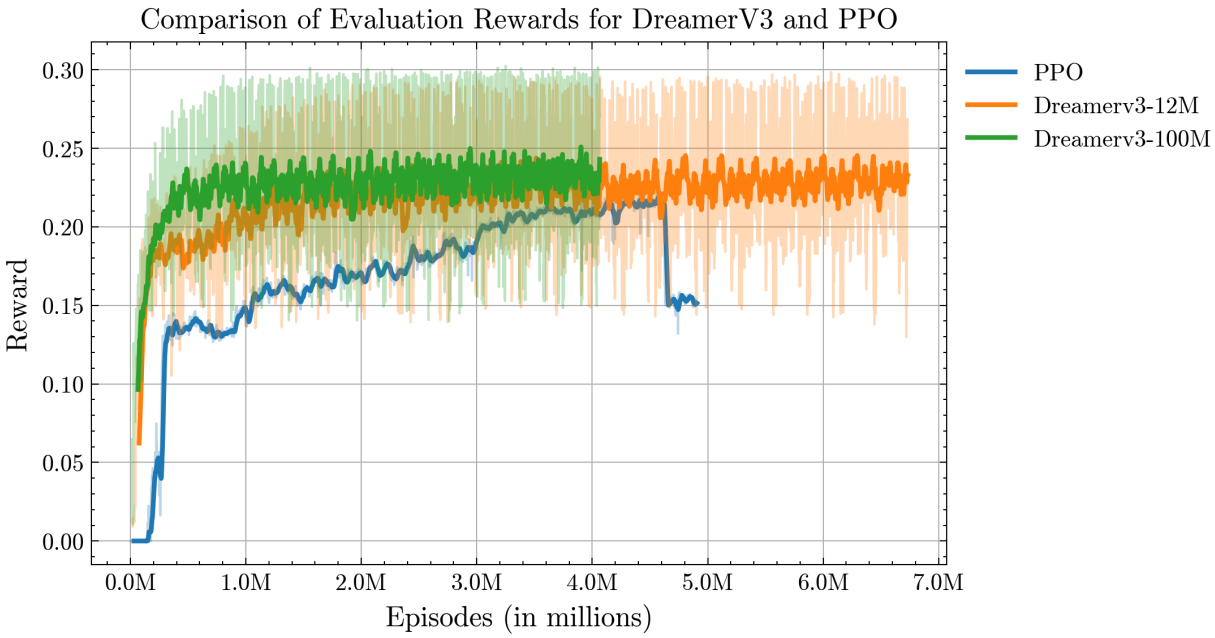


Figure 5.7: Comparison of evaluation rewards for DreamerV3-12M (orange), DreamerV3-100M (green), and PPO (blue). PPO reaches the highest evaluation performance while the two DreamerV3 show the fastest pace of improvement early in training and then rapidly top off without improving much.

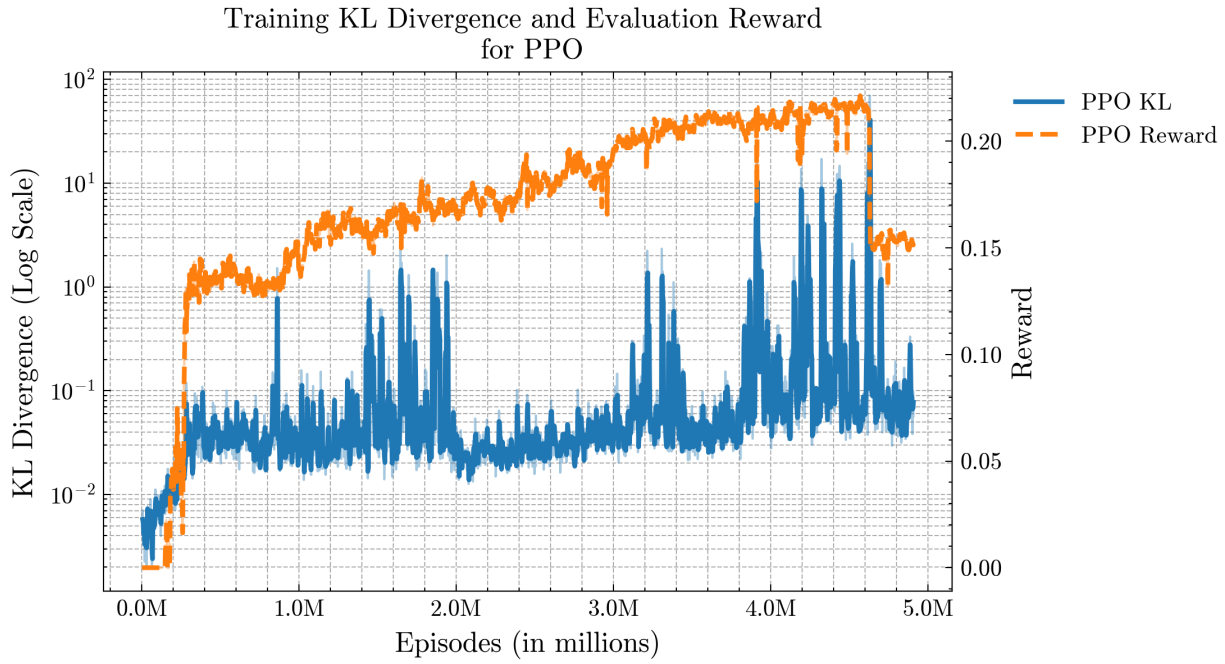


Figure 5.8: PPO Training Dynamics: KL Divergence and Evaluation Reward. This figure shows the Kullback-Leibler (KL) divergence in logarithmic scale (in blue) and evaluation reward (in orange) during PPO training. The KL divergence generally rises, indicating learning instability, with a sharp spike around the 4.6 million episodes that aligns with a significant drop in evaluation performance, highlighting instability in the training process.

5.3.3 Learning Rate Comparison

A linear regression was fitted to the points on the training curve up to the moment of best evaluation performance for each algorithm. To match the scoring method of the TopOpt game paper [144], the evaluation reward was converted to inverse compliance and normalized using the maximum observed evaluation reward across the three RL training runs. The origin of the linear regression was forced to zero since the three agents started from scratch. Figure 5.9 illustrates the best regression lines for all three agents alongside the linear regression derived from the in-class experiment. The DreamerV3-100M model demonstrated the fastest learning rate among the three RL agents, with a value of 1.78e-07 per episode. This rate is nearly four orders of magnitude lower than the 2.40e-03 learning rate observed among the engineering students in the in-class experiment. The DreamerV3-12M agent followed with a learning rate of 9.13e-08, and PPO had the lowest learning rate of 5.46e-08.

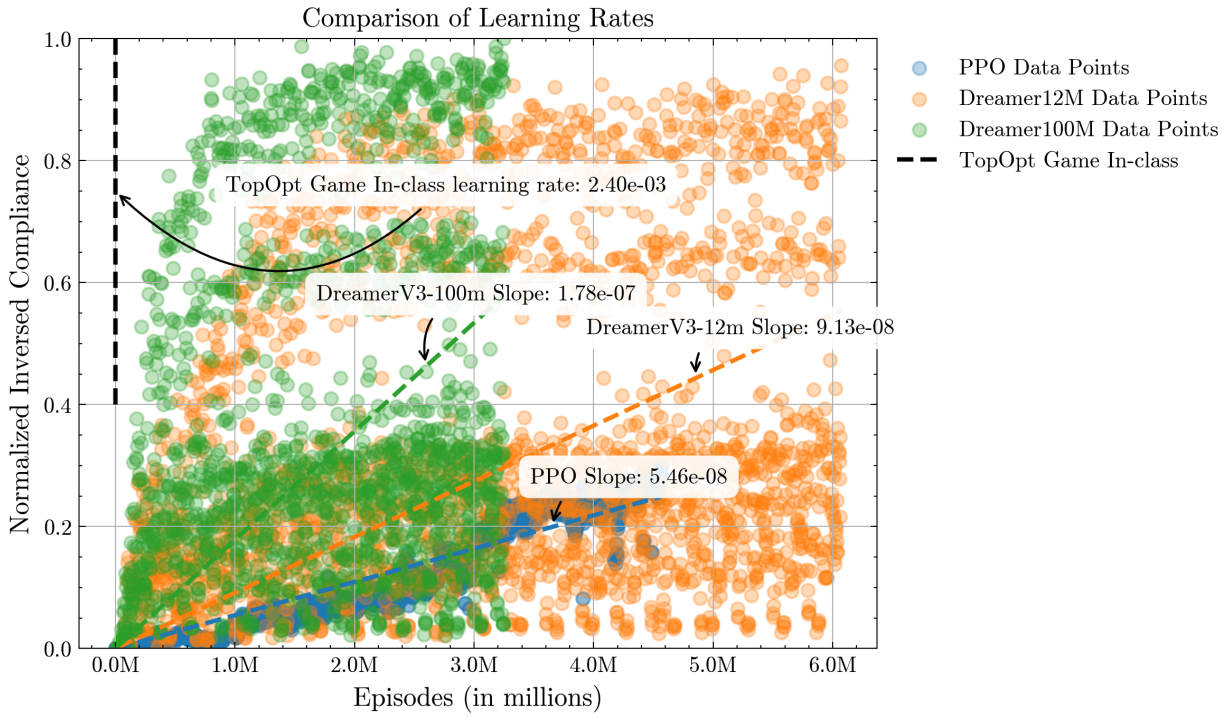


Figure 5.9: Normalized inverse compliance over episodes for PPO (blue), DreamerV3-12M (orange), and DreamerV3-100M (green). The DreamerV3-100M demonstrated the highest learning rate of 1.78 -07, just under 4 orders of magnitude smaller than then the learning rate of the TopOpt game in-class experiment (2.40e-03, black vertical line).

5.3.4 Structural Performance Comparison with Baseline

The performance of the generated structures from the 3 RL agents was tested using 1000 random boundary conditions problem sampled uniformly from the SOgym distribution. A baseline structure was established with an optimal design obtained via conventional optimization. A hybrid optimizer, integrating both MMA and GCMMA [115], was utilized because of its superior convergence speed on similar problems within the MMC framework [128]. The 188-line MMC implementation [112] in Python was used and the parameters defining the optimization process are presented in Appendix B. The RL agents' generated test structures, presented in Figure 5.10, were evaluated using the following metrics:

- Median compliance deviation: The median difference in compliance compared to the optimal designs. The median value is used due to the possibility of having very large outliers in structures with a disconnected load path.
- Disconnection rate: The proportion of generated structures with disconnected load paths.
- Mean volume deviation: The mean deviation in volume between the generated structures and the desired volume fraction constraint.

Table 5.5: Performance of the RL-generated structures compared to the corresponding baseline design obtained via conventional MMC optimization.

Algorithm	Median Compliance Δ (%)	Disconnection Rate (%)	Mean Volume Δ (%)
PPO	+219.47%	0.1%	-29.30%
DreamerV3 12M	+56.19%	0.0%	-20.14%
DreamerV3 100M	+53.96%	0.0%	-19.45%

In evaluating the 1,000 test samples, the PPO agent's performance significantly lagged the TO baseline, with its generated structures exhibiting a median compliance over 219% higher than the optimal designs, indicating substantially lower stiffness. Additionally, the PPO agent's designs averaged 29.30% below the constraint, suggesting an inefficient use of the allowed volume. Despite these shortcomings, only one structure from PPO displayed a disconnected load path. In contrast, the DreamerV3 agents demonstrated considerably better performance. The 12M parameter DreamerV3 model produced structures with a median compliance of 56.16% higher than the baseline, while the 100M parameter model showed a slightly lower median compliance difference of 53.96%. Furthermore, the 100M model approached the volume constraint more closely, with an average deviation of -19.45%, compared to -20.14% for the 12M model. Critically, both DreamerV3 agents achieved a 0% disconnection rate, ensuring fully connected load paths in all generated structures. Figure 5.10 displays 5 randomly selected samples from the 1000 test problems, showcasing the structures generated by the three RL agents compared to the baseline. These examples illustrate the general trends observed in Figure 5.7 and Table 5.5, with the DreamerV3-100M agent typically producing the stiffest structures.

Table 5.6 compares the total processing time required to generate structures for the 1000 test samples using the baseline conventional optimization and the trained RL algorithms. To account for parallel processing, the compute time across the different methods is normalized by reporting the minutes used per CPU worker. The inference times for the trained RL models were 18 minutes for PPO and 28 minutes for both DreamerV3-12M and DreamerV3-100M to process all 1000 problems. In contrast, the conventional iterative optimizer required a total of 17,792 CPU-worker minutes. While the RL models demonstrate impressive inference speeds, it is important to factor in the substantial computational resources consumed during their training phases. By simulating early stopping and considering only the training episodes up to the maximum evaluation reward (the point where the best model was saved), it was observed that the fastest RL algorithm, DreamerV3-100M, consumed 331,526 CPU-worker minutes over 6 days of training, nearly 19 times more than the conventional optimization process.

Table 5.6: Processing time comparison between conventional optimization and RL algorithms.

	Processing time (CPU-Worker minutes)		
Method	**Inference only (1000 test samples)**	**Training (total)**	**Training (with simulated early stopping)**
Baseline optimizer	17 792	-	-
PPO	18	414 720	386 021
DreamerV3-12M	28	414 720	373 911
DreamerV3-100M	28	414 720	331 526

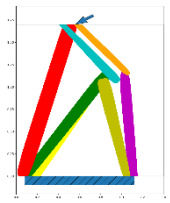
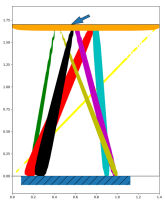
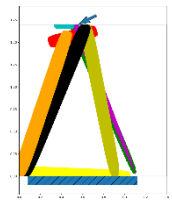
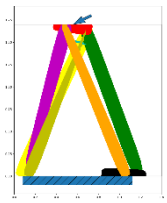
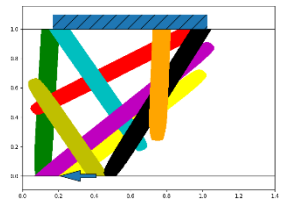
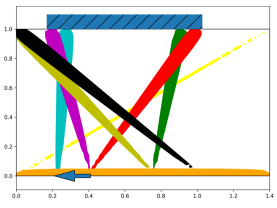
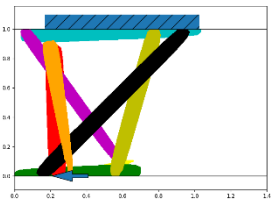
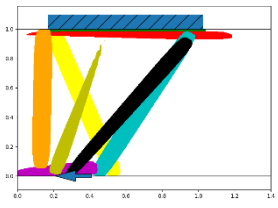
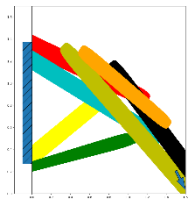
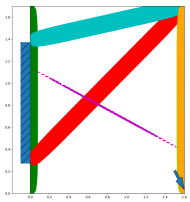
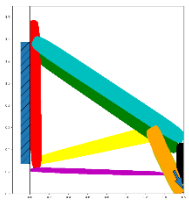
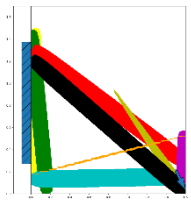
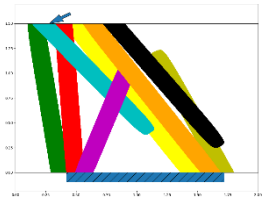
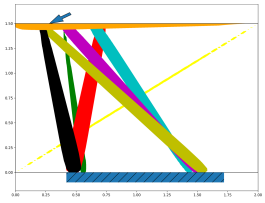
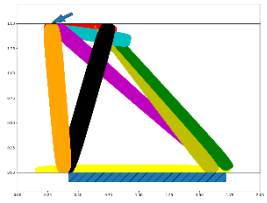
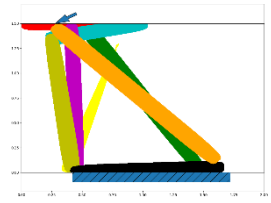
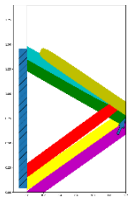
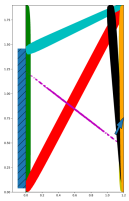
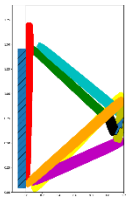
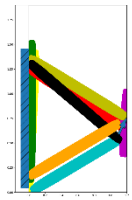
	Baseline	PPO	DreamerV3-12M	DreamerV3-100M
A				
	Compliance: 39.49 Volume: 0.300	Compliance: 88.05 Volume: 0.263	Compliance: 51.93 Volume: 0.305	Compliance: 42.28 Volume: 0.318
B				
	Compliance: 17.87 Volume: 0.410	Compliance: 41.09 Volume: 0.281	Compliance: 37.65 Volume: 0.287	Compliance: 27.98 Volume: 0.313
C				
	Compliance: 15.17 Volume: 0.360	Compliance: 340.59 Volume: 0.294	Compliance: 23.94 Volume: 0.304	Compliance: 20.79 Volume: 0.367
D				
	Compliance: 19.63 Volume: 0.490	Compliance: 51.77 Volume: 0.259	Compliance: 32.20 Volume: 0.328	Compliance: 38.53 Volume: 0.308
E				
	Compliance: 9.77 Volume: 0.340	Compliance: 95.35 Volume: 0.296	Compliance: 15.56 Volume: 0.297	Compliance: 12.14 Volume: 0.327

Figure 5.10: Comparison of generated structures from the RL agents and the corresponding baseline. The samples were randomly chosen from the 1000 test samples.

5.4 Discussion

The baseline results have shown that the SOgym environment is effective for the development of RL agents for the task of TO. Although the trained baseline agents showed worse structural performance when compared to the baseline, it is noticeable that they have achieved this level of performance from a random policy, discovering stiffening patterns and load-path connectivity through pure trial-and-error exploration.

5.4.1 Algorithm Performance and Structural Quality

A common pattern across the RL-generated designs that can be observed in Figure 5.10 is the strategic placement of MMC components along the boundary where the support and load are positioned. This behavior is likely influenced by the load-path connectivity requirement in the reward function. During the early stages of training, RL agents seem to adopt a straightforward approach to ensure connectivity by spanning the entire support and load boundaries with components. This can be seen prominently in the PPO-generated samples of Figure 5.10, where an MMC component consistently covers the boundary area where the load is applied. The DreamerV3 agents develop a more refined approach for the load boundary as they are more accurate in the placement of the components to connect the load, resulting in more efficient designs. They still exhibit components spanning the support boundary.

Although the implementation of the environment in this paper is made to stick as close as possible to the parameters of the TopOpt game experiment, soft design constraints like the one described in Equation 5.12 could provide a denser reward scheme and provide more information to the agent which could help address the behavior of covering the entire support and load boundary with material. Also, the usefulness of using the load path connectivity boolean should be investigated as this also has the impact of sparsifying the feedback from the environment early on during training.

$$r_{t_{max}} = \frac{1}{\log_e(compliance_{t_{max}})} \cdot \left(1 - \left|V_{t_{max}} - V^*\right|\right)^2 \tag{5.12}$$

5.4.2 Breakeven Threshold

When looking at the breakeven threshold, an economic measure proposed in [150] which evaluates the number of problems that the model needs to generalize to compensate for its training cost, the

DreamerV3-100M agent exhibits a breakeven threshold of about 19 000. This is derived from the fact that the total processing time for the RL agent, including training, is approximately 19 times longer than that of the baseline optimization method for generating 1000 structures. However, the method shows good potential to be useful in practice if the trained agent can get to a structural performance level that is more competitive with the conventional optimization process.

5.4.3 Richer Observation Space Brings Training Efficiency and Performance

The evaluation across different observation space configurations revealed the trade-offs between computational cost and the richness of the observation space. The TopOpt game configuration, which provides the most detailed observations including the load path and strain energy visuals, facilitated the highest reward among the tested configurations at the expense of slower training speeds due to the increased computational demands of computing the displacement field at each iteration and processing the additional image observations. However, the higher evaluation reward was reached in a smaller number of episodes, showing that the higher training efficiency per environment interaction offsets the slower FPS.

5.4.4 Model-Based DreamerV3 Outperforms Model-Free PPO

The sample efficiency and performance demonstrated by DreamerV3 highlight the benefits of using a world model for planning. In compute-intensive environments like SOgym, where interacting with the environment is inherently slow, this efficiency is particularly crucial. The ability to predict the future state of strain energy based on material placement is intuitively fundamental for designing efficient structures. By understanding how material changes impact strain energy distribution, the agent can make better decisions. Accurately predicting the outcome of placing components and thus altering the strain field is essential for optimizing performance in complex design tasks. This highlights the importance of world models in applying RL to topology optimization.

Increasing the training budget from 48 hours to 6 days did not improve significantly the performance of the DreamerV3-12M model. Both DreamerV3-12M and DreamerV3-100M exhibited plateaus in their evaluation curves, which might be due to insufficient exploration caused by a lower entropy coefficient (3e-4) compared to PPO's higher entropy coefficient (1e-2). This lower exploration could limit the models' ability to discover more optimal design solutions. Further

optimization of hyperparameters could perhaps enhance both sample efficiency and structural performance. Deep reinforcement learning methods are susceptible to these parameters, and careful tuning has been shown to impact performance in various applications dramatically [151].

5.4.5 A "Physical" RL Training Objective Solves the Disconnection Problem

A noteworthy result is the 0% disconnection rate of the structures generated by the RL agents. This outcome suggests the effectiveness of embedding the physics of the TO problem directly into the training objective. This helps address the common limitation in direct design TO approaches, where disconnection rates can soar as high as 60% in out-of-distribution test cases [142]. Framing the TO objective as an RL problem also has the added benefit of simplifying the implementation by avoiding the need for backpropagation through sparse solvers. The RL approach relies solely on the forward pass through the FEA and estimates gradients from the rewards gathered over trajectories. This avoids integrating sparse solvers with automatic differentiation software, though it does result in a more sample-inefficient learning process due to its reliance on the less direct form of feedback of trial and error.

5.4.6 Comparing Learning Rates With Engineering Students

The comparison of learning rates between the RL agents and the engineering students in the TopOpt game experiment reveals a notable difference. The DreamerV3-100M model, the best-performing RL agent, showed a learning rate about four orders of magnitude lower than that of the students. This significant difference highlights the advantage humans have in applying their intuition and prior knowledge to new and complex tasks. Even though the RL agent's learning rate is much slower, it is still impressive considering that, for an equal comparison, one would need to compensate for the inherent visual and cognitive abilities that the students bring to the task.

During the in-class experiment, the engineering students received a brief 5-minute introduction explaining the basic theory of TO for compliance minimization. This session provided them with crucial insights into the importance of material distribution, explaining that areas depicted in red on the strain-energy image indicated regions needing more material and that an optimal design should aim for a uniform distribution of strain energy. To mirror this added context, future research could study the integration of a dual-objective reward function that incorporates the notion of uniform strain energy distribution into the agent's objective. This concept, as described in Equation

5.13, suggests scaling the minimum compliance objective by the normalized standard deviation of the strain energy. This reward function could encourage RL agents to produce designs that are both stiff and demonstrate a balanced strain energy distribution across the design domain, thereby providing richer feedback to the agent.

$$r_t = \frac{1}{\log_e(compliance)} \cdot \left(1 - \frac{\sigma(W)}{\sigma(W) + \mu(W)}\right) \tag{5.13}$$

In addition to refining the reward function, another proposed idea to enhance the RL agents' learning efficiency is to leverage optimized topologies for pretraining. This approach involves a two-step training pipeline: starting with an imitation learning phase where the policy network is pretrained using supervised behavioral cloning [152] or adversarial imitation learning [153] from optimal topologies obtained via conventional optimization, followed by a pure policy training phase through RL. Initializing the policy with expert data can provide the RL agents with a strong starting point, rather than beginning from a random policy, similar to providing students with basic TO theory. This method has proven effective in complex environments such as StarCraft II [87] and the game of Go [25], leading to significant breakthroughs in those challenging domains.

5.5 Conclusion

This paper introduces the Structural Optimization gym (SOgym), a novel RL environment designed for advancing the application of machine learning in TO. By framing the TO objective as an RL task, SOgym enables the development and training of RL agents to explore and exploit design spaces for optimal structural solutions.

The baseline results illustrate progress in training RL agents to solve continuous TO problems. This suggests the potential of RL, especially model-based algorithms like DreamerV3, in producing structurally robust designs. Despite beginning from a random policy, these agents successfully learned to produce connected topologies that were within 54% of the compliance achieved through conventional optimization. The comparison between model-free (PPO) and model-based (DreamerV3) approaches highlights the significant benefits of incorporating planning and world models, especially in enhancing sample efficiency and performance in computationally intensive environments like SOgym.

However, it is important to note that the primary goal of this paper is to introduce the SOgym environment and present initial baseline results, providing a preliminary understanding of what can

be achieved with this new tool. These results are intended as a starting point rather than a definitive solution, and further research is needed to fully explore and validate the capabilities of RL in structural optimization.

Additionally, RL's framework presents opportunities for incorporating qualitative criteria into the optimization process. Inspired by the success of Reinforcement Learning from Human Feedback (RLHF) in fine-tuning large language models based on human preferences [154], future research could explore similar methods in structural optimization. By integrating qualitative aspects such as aesthetics into the reward function, RL agents could potentially be trained to produce designs that balance both functional and visual appeal. This approach could lead to innovative solutions that not only meet technical specifications but also enhance architectural beauty and user satisfaction.

Beyond exploring different environment objectives, the SOgym environment can be extended to support three-dimensional structures. This expansion would enable the exploration of more complex and realistic design scenarios, broadening the applicability of RL in structural engineering.

In summary, RL presents an interesting take on building up intuition for minimum compliance TO problems using ML. Moreover, the SOgym provides a valuable foundation for ongoing and future research and paves the way for future human feedback interaction which could prove a valuable design tool.

5.6 Data & Code Availability

The SOgym environment is open-source and available at https://github.com/ThomasRochefortB/sogym

5.7 Usage of Generative AI

During the preparation of this work, the authors utilized GPT-4o (version 2024-05-13) to enhance the clarity and conciseness of the language of their own text. After using this tool/service, the authors also reviewed and edited the AI-generated results as needed and they take full responsibility for the content of the paper.

5.8 Acknowledgements

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC) [funding reference number 569251]. This research was enabled in part by support provided by Calcul Québec (https://www.calculquebec.ca) and the Digital Research Alliance of Canada (DRAC) (alliancecan.ca).

CHAPTER 6 ARTICLE 3: From Density to Geometry: Instance Segmentation for Reverse Engineering of Optimized Structures

Thomas Rochefort-Beaudoin[a], Aurelian Vadean[a], Sofiane Achiche[a], Niels Aage[b]

[a]*Department of Mechanical Engineering, Polytechnique Montréal, 2500 Chemin de Polytechnique, Montréal, QC, Canada*

[b]*Department of Civil and Mechanical Engineering, Solid Mechanics, Technical University of Denmark. Koppels Allé, B.404, 2800 Kgs. Lyngby, Denmark*

Submitted to *Engineering Applications of Artificial Intelligence* on April 30th, 2024

Abstract

This paper introduces You Only Look Once v8 for Topology Optimization (YOLOv8-TO), a novel approach for reverse engineering topology-optimized structures into interpretable geometric parameters using the YOLOv8 instance segmentation model. Density-based topology optimization methods require post-processing to convert the optimal density distribution into a parametric representation for design exploration and integration with computer-aided design tools. Traditional methods such as skeletonization struggle with complex geometries and require manual intervention. YOLOv8-TO addresses these challenges by training a custom YOLOv8 model to automatically detect and reconstruct structural components from binary density distributions. The model is trained on a diverse dataset of both optimized and random structures generated using the Moving Morphable Components method. A custom reconstruction loss function based on the dice coefficient of the predicted geometry is used to train the new regression head of the model via self-supervised learning. The method is evaluated on test sets generated from different topology optimization methods, including out-of-distribution samples, and compared against a skeletonization approach. Results show that YOLOv8-TO significantly outperforms skeletonization in reconstructing visually and structurally similar designs. The method showcases an average improvement of 13.84% in the Dice coefficient, with peak enhancements reaching 20.78%. The method demonstrates good generalization to complex geometries and fast inference times, making it suitable for integration into design workflows using regular workstations. Limitations include the sensitivity to non-max suppression thresholds. YOLOv8-TO represents a significant advancement in topology optimization post-processing, enabling efficient and accurate reverse engineering of optimized structures for design exploration and manufacturing.

6.1 Introduction

Topology Optimization (TO) is used by engineers to improve the design process, by leveraging numerical methods to seek the optimal material distribution within a given domain, while maximizing key properties such as stiffness or heat transfer in the resulting structure.

Density-based TO [26], such as the Solid Isotropic Material with Penalization (SIMP) method [137], employs a discretization approach where the structural domain, outlining the structure's maximum bounds, is divided into a grid of pixels (or voxels for 3D). Each pixel assumes a continuous density value, ranging from 0 (void) to 1 (presence of material). The SIMP method has become the de-facto framework in the industry and academia due to its simplicity of implementation [32], scalability for large-scale problems [9], [10], support for a wide range of manufacturing constraints [138], and optimization objectives [139].

The density distribution matrix encodes the spatial distribution of material inside the given domain. Therefore, the optimal structure’s geometry is implicitly represented by the assembly of pixel values close to 1, representing the presence of material. TO results often contain several millions of density variables and lead to highly complex material layouts with a large number of structural features. This means that human interpretation and feature extraction is extremely time-consuming, if possible at all. Therefore, automatic post-processing steps are required to interface the resulting structure with other steps in the mechanical design process [140].

The literature on the geometric post-processing of topology-optimized structures can be separated into three distinct downstream applications: prototyping, design validation, and design exploration [12]. While the first two applications are focused on generating a high-quality surface mesh for 3D printing or finite-element analysis (FEA), design exploration applications require a parametric output that can be manipulated by the engineer to iterate toward the optimal design. Ideally, the density field is converted to a set of interpretable geometrical parameters which is compatible with CAD software to streamline the design process and facilitate design iterations.

To address this, feature-mapping methods [27] for TO were developed to directly optimize geometric variables defining high-level geometric components which are then mapped onto the density field via some projection methods. The Method of Moving Morphable Components (MMC) [28] uses hyperelliptic functions defined by center coordinates, length, orientation, and thicknesses to build a density distribution out of an assembly of the high-level geometric primitives

which are then projected onto a discretized domain. These methods allow direct interpretability of the design into simple geometric variables but currently show worse convergence rates [14] than the well-established density-based methods due to being more prone to getting stuck in poor local minima [27].

Ideally, the geometric post-processing for design exploration should be done independently of the optimization process used and can be done as a reverse-engineering task where a structure is reconstructed from a set of high-level geometries predefined by the engineer. By performing the geometric post-processing as a separate step, well-established and efficient density-based methods can be used for optimization, ensuring faster convergence and better performance. The resulting optimized density distribution can then be reverse engineered into a set of high-level geometric primitives, which are more intuitive and easier to manipulate by the engineer. This two-step approach combines the strengths of both density-based methods and feature-mapping methods, allowing for efficient optimization and interpretable geometric representation of the optimized structure, ultimately streamlining the design process, and facilitating design iterations.

The field of reverse-engineering of 3D point clouds into Constructive Solid Geometry (CSG) trees is well-documented in the literature [155], [156], [157]. However, the specific application of these methods to TO outputs within a design exploration framework has been limited to skeletonization techniques. These techniques essentially involve finding a one-dimensional representation of the optimal density distribution. Despite their relevance, these methods underperform for structures that do not resemble trusses [44], [158], [159]. Even advanced methods using curved skeleton members, which are better suited for TO outputs with complex shapes, struggle with accurately locating the intersections of skeleton branches [160]. Furthermore, the most effective skeletonization methods often require manual inputs and are reliant on heuristics specific to each case [161]. Even so, skeletonization-based methods used to convert density-based outputs to MMC design variables have demonstrated the added potential of accelerating convergence of conventional MMC optimization by offering the possibility to set the initial solution from an interpreted intermediate SIMP topology, resulting in the reduction of computation time from 16% to 23% [162].

The development of a reverse engineering method for TO can be compared to creating a geometrical compression algorithm for encoding complex 2D or 3D density distributions into a set

of one-dimensional geometric parameters as illustrated in Figure 6.1. The key challenge is to ensure that this 1D representation allows for almost lossless decoding back to the original 2D or 3D form, hence preserving essential features of the optimized design. The effectiveness of this process should be evaluated both at the image level (per-pixel accuracy) and the mechanical level (structural response under load).

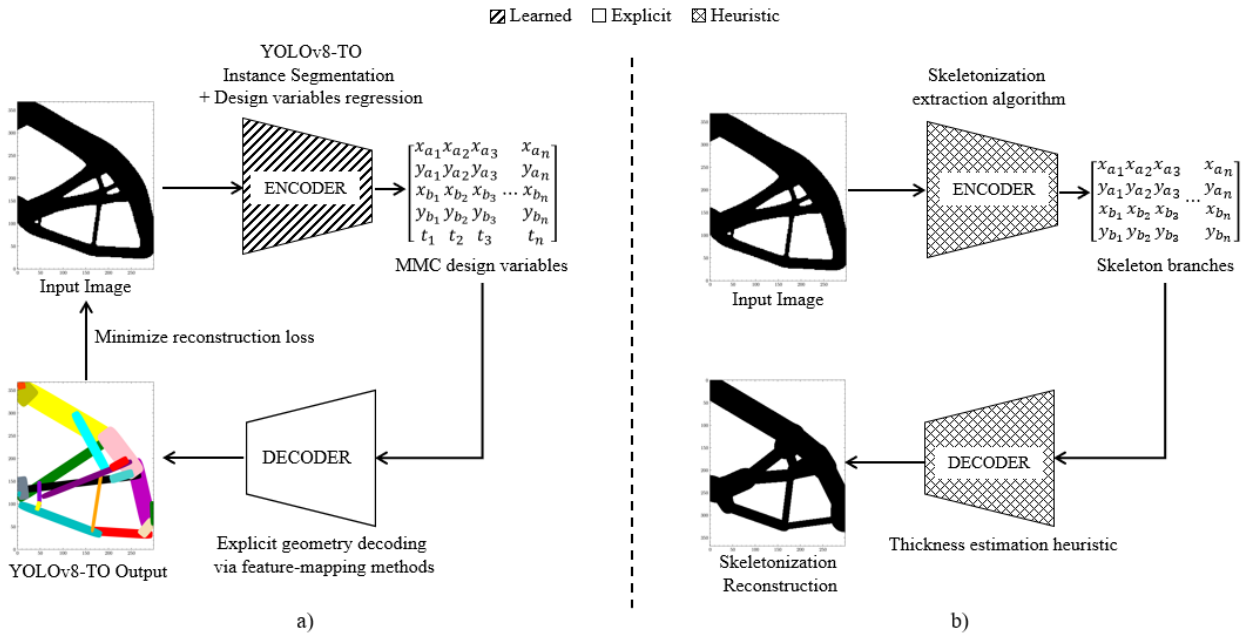


Figure 6.1: Compression algorithm analogy. a) YOLOv8-TO as a learnable geometrical encoder with an explicit geometry decoding defined by the MMC hyperelliptic functions. b) Skeleton extraction heuristic as a geometrical encoder with a heuristic-based thickness estimation decoder.

This article proposes a novel approach for reverse-engineering TO results by framing the problem as an instance segmentation task. Instance segmentation is a computer vision technique that involves detecting and delineating distinct objects of interest within an image, while simultaneously classifying each object. By leveraging the YOLOv8 algorithm, our method integrates image segmentation with design variable regression, focusing on accurately identifying and reconstructing component shapes within monochrome images. This approach represents an efficient geometric encoding method with an explicit decoding algorithm, offering a more straightforward and robust alternative to heuristics-based skeletonization approaches. The novelty of this work lies in the tailored application of state-of-the-art instance segmentation techniques to

the field of TO, enabling the automatic extraction of interpretable geometric parameters from TO results.

6.2 Method

The task of identifying critical structural members in a 2D density distribution can also be understood as an instance segmentation task with the specific requirement to output, for each detected instance, geometric variables defining its position and shape. Figure 6.2 illustrates the segmentation process that enables the reconstruction of each structural sub-structure in a TO output.

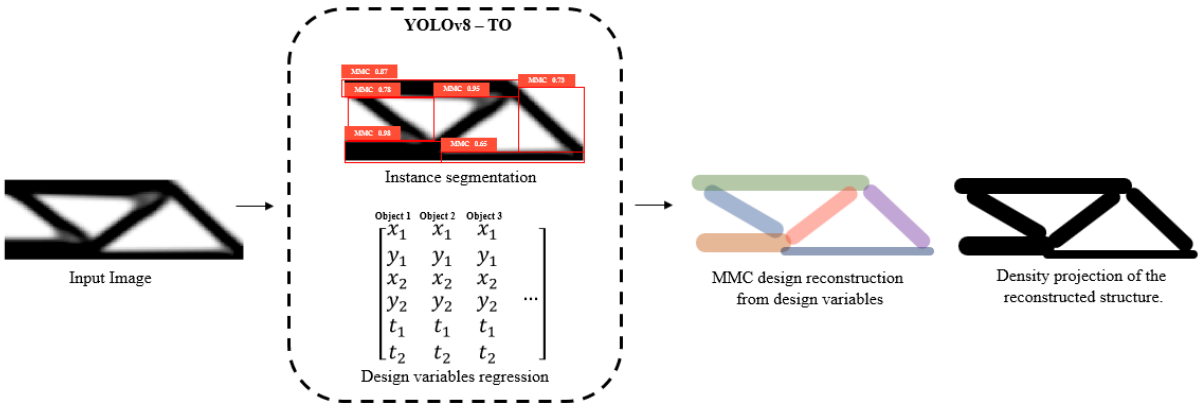


Figure 6.2: Reverse-engineering of an optimal density distribution as an instance segmentation approach.

The presented instance segmentation approach is based on YOLOv8 [163], a cutting-edge computer vision model focused on having low inference latency and presenting an open-source and well-maintained codebase. In addition to the object classification, bounding box prediction, and pixel segmentation heads, an additional regression head was added to the model for the task of design variables regression. The custom model is detailed in section 6.2.2.

6.2.1 Dataset

The model was trained and tested on a diverse set of datasets to evaluate its performance and generalization capabilities. The datasets include optimal topologies obtained from the MMC and SIMP methods, random assemblies, and out-of-distribution (OOD) samples from the literature. The MMC dataset was split into training, validation, and testing subsets, while the random

assembly dataset was used for training only. The SIMP, low volume fraction SIMP (SIMP5%), and OOD datasets were used for testing.

MMC Dataset

The MMC method was used as the basis of the training dataset because its output can be directly utilized as a segmentation label for the model training. The MMC implementation from [40] was used and a hybrid optimizer combining both the Method of Moving Asymptotes (MMA) and its globally convergent version (GCMMA) was chosen because of its better convergence behavior for similar problems in 2D [128]. To generate many samples, random boundary conditions, inspired by [142], were sampled from predefined distributions as presented in Table 6.1 and Figure 6.3. The black-and-white density projections obtained via a Heaviside projection were used as input for the instance segmentation model as illustrated in Figure 6.4. The dataset was split into training (80%), validation (10%) and testing (10%) subsets to evaluate the model's performance on optimal topologies.

Table 6.1: Parameters defining the boundary conditions distribution.

Parameter	Name	Distribution
h	Height	[1.0, 2.0]
w	Width	[1.0, 2.0]
L_s	Support length	50% to 75%
P_s	Support position	0 to (100% of L_s)
P_L	Load position	0% to 100% of boundary opposite from support
θ_L	Load orientation	[0°,360°] *

* The selected angle is filtered to ensure there is at least 45 degrees of difference with the support normal

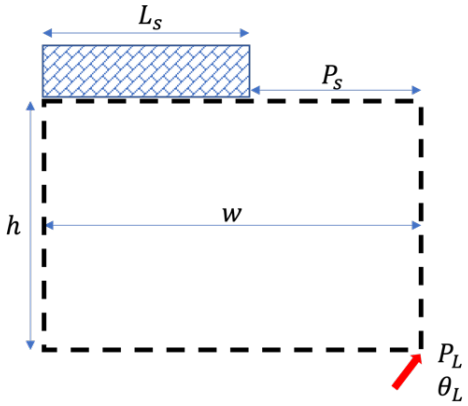
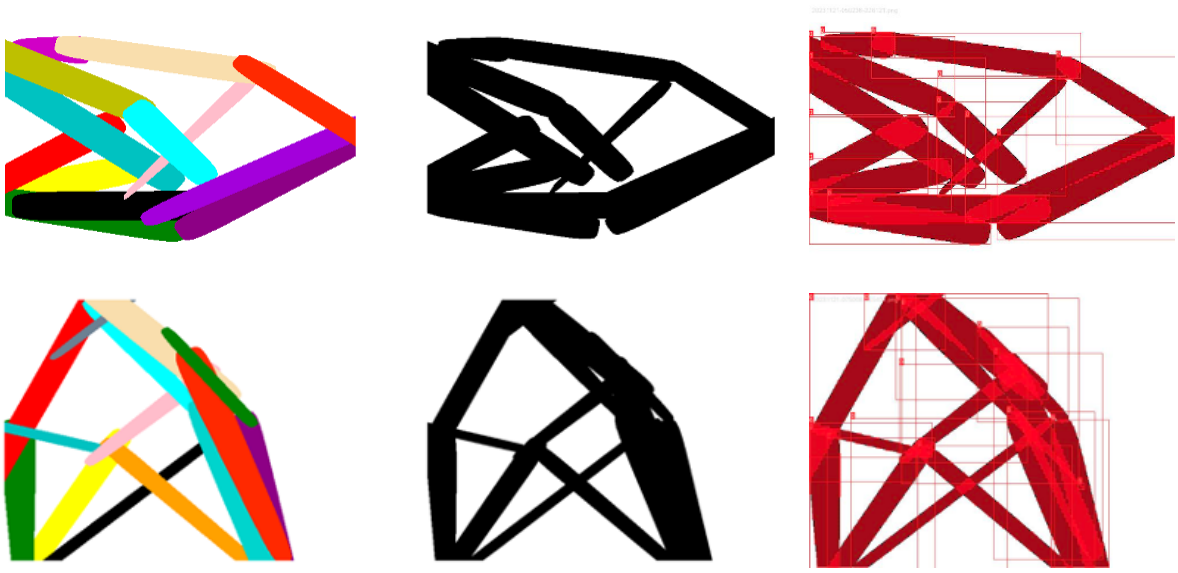


Figure 6.3: Illustration of the boundary conditions distribution



a) b) c)

Figure 6.4 : MMC dataset training sample. a) Output from MMC optimization. b) Black and white input for the instance segmentation model. c) Instance segmentation labels.

Random Assembly Dataset

The generation of training data for machine learning models using iterative gradient-based TO methods is computationally expensive. This computational burden has constrained much of the existing research on machine learning applications in TO to rely on small, homogeneous datasets with low-resolution samples for training and testing [18], [142]. YOLOv8-TO's ability to learn from non-optimized structures is a key property as it can be trained on simple, binary images of structural components. Randomly sampling the space of design variables allows for the cost-effective generation of assemblies composed of randomly distributed components. The geometry function defining the components implicitly defines the segmentation labels required by the detection head of YOLOv8 while the random design variables are used as labels by the regression head. Samples of the random assembly dataset are shown in Figure 6.5.

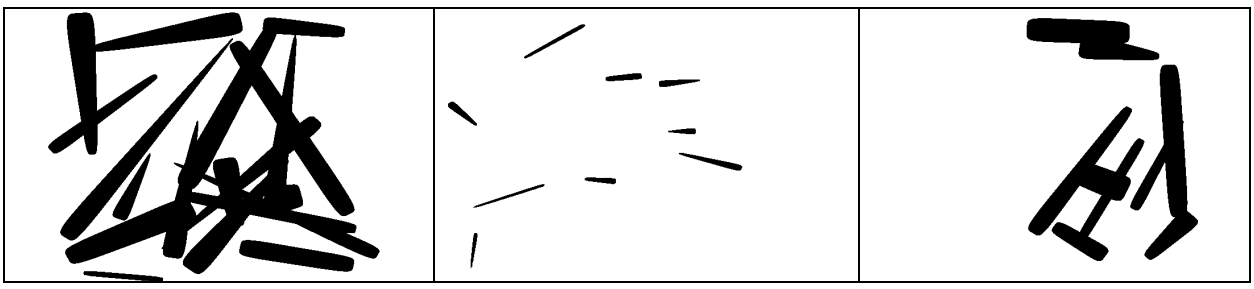


Figure 6.5: Random MMC training samples.

SIMP Dataset

Testing the model on a dataset of optimal structure generated from a different TO method allows to test the capability of the method as a general post-processing tool, independent of the optimization technique used. A dataset of 2000 TO structures was generated using the boundary conditions sampled from the same distribution as the MMC dataset. Samples are presented in Figure 6.6. A mesh resolution of 150 elements per unit of height/width and the Python implementation from the 88-line MATLAB code for TO were used [59].

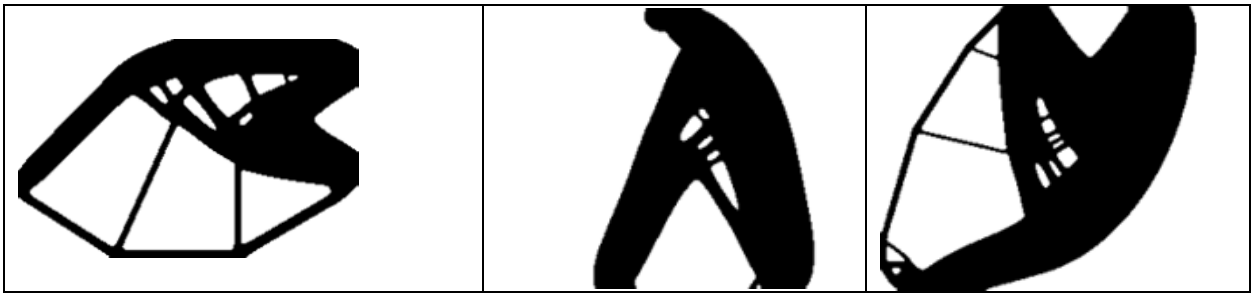


Figure 6.6 : SIMP test dataset samples.

Low Volume Fraction SIMP Dataset ($SIMP_{5\%}$)

To compare the performance of YOLOv8-TO against a skeletonization approach, 2000 random SIMP samples with the same boundary conditions were generated with a low volume fraction of 5%. This generated thin structures that better fit the "truss-like" property which is better suited for skeletonization approaches. To obtain a binary density distribution, as shown in Figure 6.7, a threshold value of 0.1 was applied to the generated continuous density values. All density values above 0.1 were set to 1.0, while values equal to or below 0.1 were set to 0.0. This dataset will be referred to as '$SIMP_{5\%}$'.

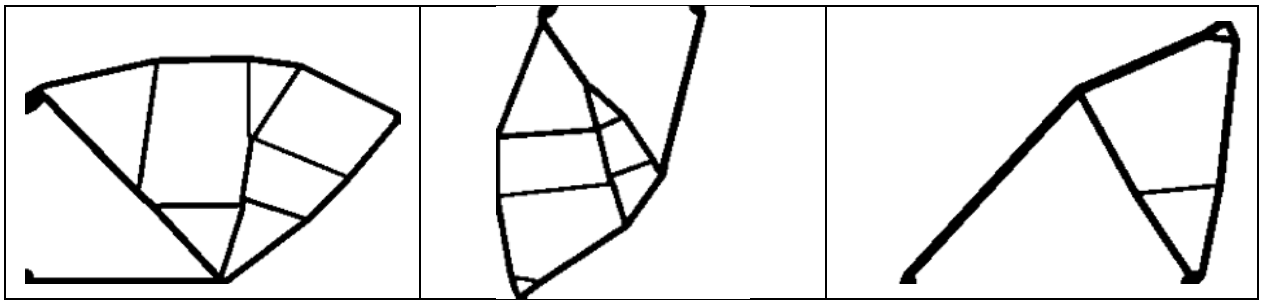


Figure 6.7: Low volume fraction SIMP samples with volume constraint of 5%.

Out-of-Distribution Dataset

To test the generalization capability of the proposed approach, 4 images of TO structures taken from the literature were selected to test against some complex structure types that were not encountered in the training phase. Figure 6.8a) and b) depict a 2D femur structure that has been optimized using a density-based SIMP method, subject to total volume constraints and local volume constraints respectively [164]. These two images enable the evaluation of the model's ability to interpret curved and porous sub-structures. Figure 6.8c) represents the classic 2D cantilever beam problem partially optimized via the level set method [165]. The image was generated using the 88-line level set educational code [30] and was limited to 20 optimization iterations to obtain thick and curved structural members which are different from the straight MMC components used in training. Figure 6.8d) corresponds to a compliant mechanism force inverter obtained with the 88-lines SIMP code [59].

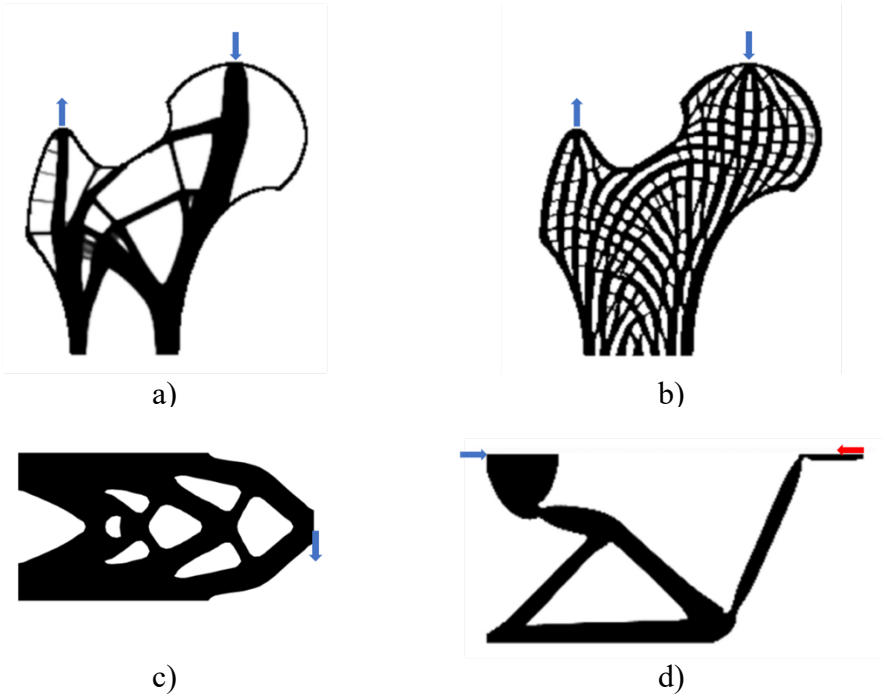


Figure 6.8: Out-of-distribution test samples. a): 2D femur with total volume constraint. b) 2D femur with local volume constraints, reproduced with permission from [164]. c) Cantilever beam partially optimized with the level set method, obtained with [30]. d) Compliant mechanism force inverter, obtained with [59]. The red arrow shows the optimization objective to maximize.

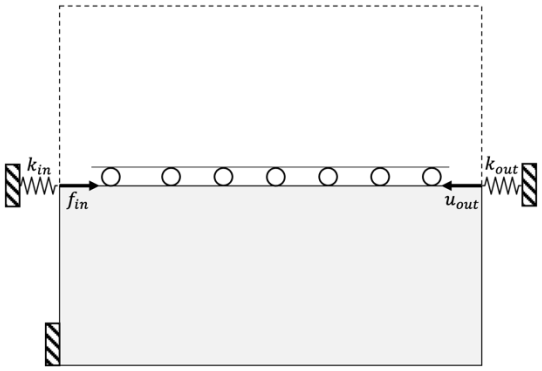


Figure 6.9: Compliant mechanism force inverter TO problem definition with symmetry conditions.

As illustrated in Figure 6.9, this benchmark TO problem [166] is defined by applying a force in the x-direction at the center of the left boundary (f_{in}) with the optimization objective of maximizing the negative x-displacement at the center of the right boundary (u_{out}). This structure is difficult to interpret because the model must accurately replicate the hinges of the mechanism to preserve its compliant force inversion structural response.

6.2.2 The YOLOv8-TO Model

YOLOv8, an anchor-free CNN computer vision model [167], consists of a CSP-Darknet53 backbone and neck architecture [168]. As shown in Figure 6.10, split heads generate instance segmentation outputs, including bounding box prediction, object classification, and pixel classification. A custom YOLOv8 implementation was developed to predict design variables for each detected structural component. The new regression head uses convolutional layers with ReLU activation, followed by a final convolutional layer to produce a 5D vector of normalized design variables, which are then scaled using minimum and maximum bounds (Appendix C). The regression predictions are made at multiple scales of the feature maps, allowing the model to utilize different feature scales. Nano, medium, and extra-large (x-large) models with 4.85M, 29.18M, and 73.38M parameters, respectively, were trained to assess the impact of model size on performance. Training continued until the fitness metric (Equation 6.1), a weighted sum of the mean Average Precision (mAP) at different Intersection over Union (IoU) thresholds, did not improve for 100 epochs. The $mAP_{@0.5-0.95}$ represents the average mAP over IoU thresholds from 0.5 to 0.95 with a step size of 0.05, comprehensively assessing the model's performance across various levels of overlap between the predicted and ground truth bounding boxes or masks. Data augmentation from

the Albumentations library [169] was used during training, with parameters and hyperparameters listed in Appendix C.

$$fitness = \left(0.1\, mAP_{@0.5\ (Bbox)} + 0.9\, mAP_{@0.5-0.95\ (Bbox)}\right) + \left(0.1\, mAP_{@0.5\ (Mask)} + 0.9\, mAP_{@0.5-0.95\ (Mask)}\right) \quad (6.1)$$

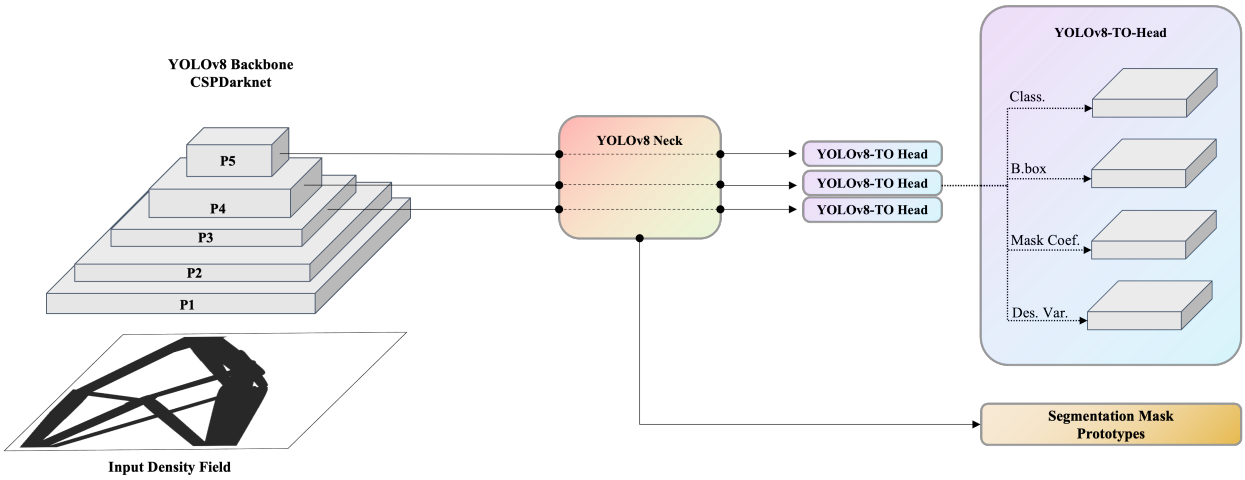


Figure 6.10: Architecture of the YOLOv8-TO model. The input density field is processed by a CNN CSPDarknet feature extractor (left). The neck architecture (middle) passes three different layers from the backbone to separate YOLOv8-TO heads and a segmentation mask prototype network. Each YOLOv8-TO head (right) generates predictions for classification, bounding box, mask coefficients, and design variables. The segmentation mask prototype network is a fully convolutional network that generates prototype segmentation masks.

During training with the original MMC formulation from [40], very large gradient values occurred due to the use of non-smooth projection methods. The implementation of the level set function and its smoothed Heaviside projection resulted in abrupt changes in the gradient flow, causing the model training to become unstable. To address this, a simplified MMC approach was adopted, utilizing endpoint coordinates and a single thickness value. This method circumvents the issue of angle prediction with a sigmoid function, which can misrepresent similar angles. The two approaches are shown in Figure 6.11and Figure 6.12. A normalized sigmoid projection was employed to convert the level set function of each component into a binary density field, as detailed in Equations 6.2 to 6.9. Figure 6.13 illustrates the sigmoid projection scheme for a single component.

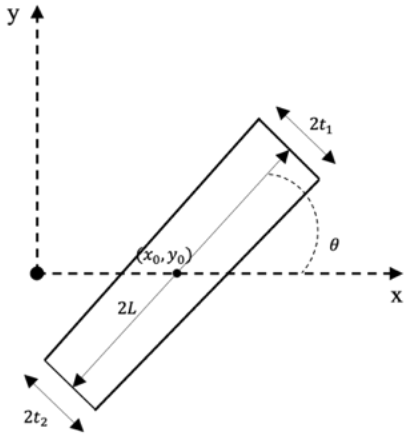


Figure 6.11: MMC Component formulation used in [40]

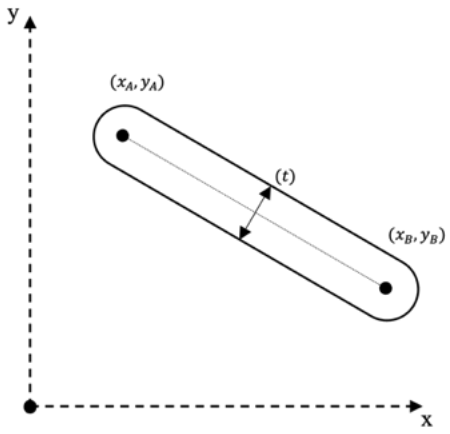
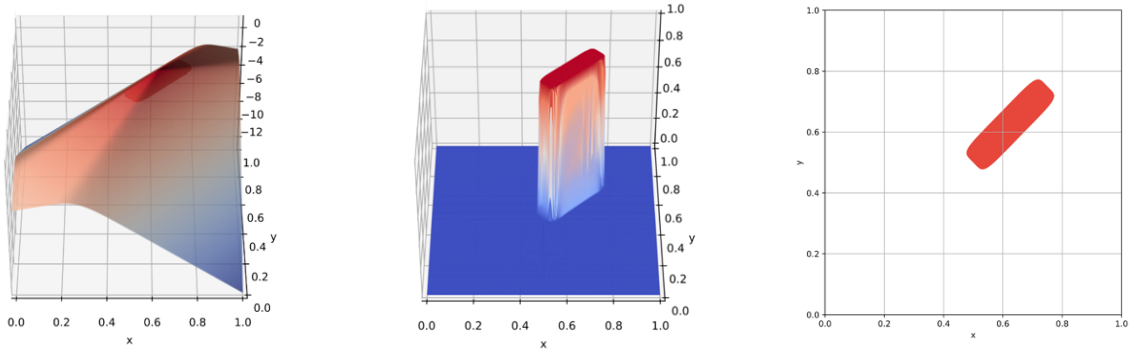


Figure 6.12: Endpoint MMC formulation used in this paper.



a) b) c)

Figure 6.13: Illustration of the MMC component. a) The contour surface of the level set function. Everything above the '0th' level is highlighted in red. b) The corresponding sigmoid projection. c) 2D view of the sigmoid projection corresponding to the density field of the single MMC component.

$$x_0 = \frac{A_x + B_x}{2}, y_0 = \frac{A_y + B_y}{2} \tag{6.2}$$

Pixel-Based Bounded Reconstruction Loss

The presence of overlapping components in the MMC-generated training dataset poses a challenge for the model's training process. In the MMC method, overlapping components can combine into a larger structural component with a single visible boundary. Using a classic regression loss such as the mean-squared error (MSE) on the design variables would force the model to predict the design variables of these overlapping components, which is an undesirable behavior. To mitigate this the new design variables regression head is trained using a dice loss bounded within the

predicted bounding box. The pseudocode for the loss function is presented in Figure 6.14. One important advantage of using such reconstruction loss is that it is inherently self-supervised, requiring no ground-truth design variables for training.

Pixel-based bounded reconstruction loss:

for img in batch:

$\hat{y} = YOLOv8(img)$	(Normalized design variables prediction in the local bounding box reference system)
$\hat{y} \rightarrow \tilde{y}$	(Scale the design variables using the bounding box coordinates)
$\Phi = MMC(\tilde{y})$	(Generating MMC level set functions for each component)
$\rho = sigmoid(\Phi)$	(Sigmoid projection)
$img \rightarrow mask_{cropped}$	(Crop corresponding pixel mask using the predicted bounding box)

$$intersection = \sum(\rho \cdot mask_{cropped})$$

$$dice = \frac{2 \cdot intersection}{\sum \rho(\Phi) + \sum mask_{cropped}}$$

return $(1 - dice)/batchsize$

Figure 6.14: Pixel-based reconstruction loss algorithm.

Segmentation Mask Non-Max Suppression

In the original YOLOv8 implementation, non-max suppression (NMS) is performed using bounding boxes during inference. However, this poses a challenge for detecting structural patterns, as symmetrical structures might share identical bounding boxes yet consist of two separate elements. An example of this is shown in Figure 6.15 with an equal-length orthogonally intersecting pair of components, a configuration that is commonly found in topology-optimized structures due to its high stiffness. To address this, a NMS algorithm based on the dice coefficient of the predicted mask of the model was implemented. Each mask pair is compared and overlapping pairs with a dice coefficient above a certain threshold are marked as too similar, with only the mask with the highest confidence score kept in the final prediction. This operation is, however, computationally expensive, as demonstrated by the example in Figure 6.15, where the bounding box NMS requires 2 milliseconds, while the dice coefficient NMS takes 82 milliseconds, highlighting the trade-off between improved accuracy and computational cost.



a) b) c)

Figure 6.15: a) Orthogonally intersecting pair of equal length MMC component. b) Model detection results with NMS on the bounding box. c) Model detection results with NMS based on the Dice coefficient of masks.

6.2.3 Comparing YOLOv8-TO to Skeletonization

Results from the YOLOv8 approach will be compared to the skeletonization approach proposed in [44]. A thickness estimation algorithm was developed to obtain a similar parametric reconstruction as the MMC approach of YOLOv8-TO. For each detected skeleton branch, a constant thickness is estimated with the proposed algorithm described in Figure 6.16.

Skeleton thickness estimation:

$dist = dist.transform(input\ image)$ (Distance transform matrix of the input binary image using Euclidean distance and kernel size of 5)

for branch in skeleton: (Loop through each branch)

  for point in numPoints: (Determine a set of equally spaced points along the branch)

    $diameter[point] = 2 \cdot \max\left(kernel_{3x3}(dist[point])\right)$ (Calculate the maximum distance of a 3x3 kernel centered around the point)

  $thickness[branch] = mean(diameter)$ (Assign the branch thickness as the mean of the calculated diameters)

$thickness_{5\%} = 5th\ percentile\ of\ thicknesses$ (Determine the 5th percentile thickness as minimum threshold)

$thickness = max(thickness, thickness_{5\%})$ (Apply the minimum thickness)

return *thickness*

Figure 6.16: Skeleton thickness estimation algorithm.

6.2.4 Disconnected Load Path in Predicted Structures

A common issue with image-based prediction models is their tendency to generate structures with disconnected load paths. Image similarity loss functions alone do not ensure that the generated structure maintains contact with the support and loaded regions [142]. To address this problem and enable a fair comparison of structural performance between the original image and the predicted structures from YOLOv8-TO and skeletonization algorithms, a simple connection algorithm was developed.

The connection algorithm treats the support and load conditions independently, verifying that at least one pixel of material is in contact with each boundary condition to ensure the structure's connectivity. In cases where disconnection is detected, the algorithm identifies the nearest skeleton or MMC node to the initial pixel adjacent to the boundary condition. The nearest node is determined based on its row and column order in the image. The algorithm then repositions the node to establish a connection with the boundary condition.

It is important to note that the algorithm does not perform any post-processing to check for internal disconnections within the structure. Addressing internal disconnections would require the development of a more complex heuristic to identify and correct such issues. The current implementation focuses solely on ensuring connectivity at the support and load boundaries, which is sufficient for comparing the structural performance of the original and predicted structures.

6.3 Results

YOLOv8-TO and the skeletonization method were evaluated based on their ability to reconstruct the original image accurately. Three metrics were used to assess the reconstruction quality:

- Dice coefficient, which measures the image similarity between the reconstructed and input images.
- Relative volume difference, which quantifies the similarity in volume between the reconstructed and input images.
- Relative compliance difference, which assesses the similarity in structural response between the reconstructed and input images.

The Dice coefficient and volume similarity were calculated using mean values across the test datasets. In contrast, the median compliance was used due to the presence of a few internally

disconnected structures that can cause a highly skewed structural response distribution. Furthermore, when the reconstructed structures' volumes are comparable between the two methods, analyzing the product of compliance and volume offers valuable insights into their performance from a structural optimization perspective. However, it is important to note that this additional metric is only valid for comparing structures with similar volumes. As a result, this metric will only be used when analyzing individual results within this section, where the volume similarity between the input and reconstructed structures is ensured.

The aggregated results, presented in Table 6.2, were obtained using the x-large YOLOv8-TO model trained on a combined dataset of MMC-optimized structures and random assemblies. To determine the Intersection over Union (IoU) and confidence thresholds for the NMS post-processing step, a subset of 10 randomly selected samples was used. Model predictions were generated for various IoU and confidence threshold combinations, and the combination that yielded the lowest mean dice loss across all 10 samples was selected. This optimal combination was then applied to run inference on the entire corresponding test dataset. The specific confidence and IoU combinations used for each dataset are indicated under the dataset name in Table 6.2. On the three MMC, SIMP & $SIMP_{5\%}$ test datasets, YOLOv8-TO yielded an average Dice coefficient improvement of 13.84%, with a maximum improvement of 20.78%.

Table 6.2: Aggregated results of YOLOv8-TO vs skeletonization on all test datasets.

		Evaluation Datasets						
Method	**Metrics**	**MMC @ (0.01, 0.5)**	**SIMP @ (0.01, 0.45)**	**SIMP5% @ (0.01, 0.45)**	**OOD Femur total vol.**	**OOD Femur local vol.**	**OOD Level set 20 iter.**	**OOD Comp. Mech.**
YOLOv8-TO	Mean Dice Coefficient [1.]	**0.94**	**0.94**	**0.90**	**0.94**	0.89	**0.94**	**0.95**
	Mean Volume Δ (%) [2.]	**-6.18**	+6.83	+11.06	+5.61	+5.44	+**1.04**	-5.64
	Median Compliance Δ (%) [3.]	**+24.91**	**+12.99**	**+31.12**	**+0.98**	**-6.09**	**-**22.76	**-4.36**[*]
Skeletonization	Mean Dice Coefficient [1.]	0.83	0.88	0.74	0.85	0.92	0.88	0.70
	Mean Volume Δ (%) [2.]	-15.67	**-2.70**	**-4.12**	**0.00**	**-4.24**	+6.12	**+3.37**
	Median Compliance Δ (%) [3.]	+30.41	+18.99	+100.89	+4.45	+12.07	+**11.08**	-99.50[*]

[*]**The compliant mechanism force inverter's structural performance is evaluated based on the x-displacement at the right corner.**

[1.] **Higher is better.**

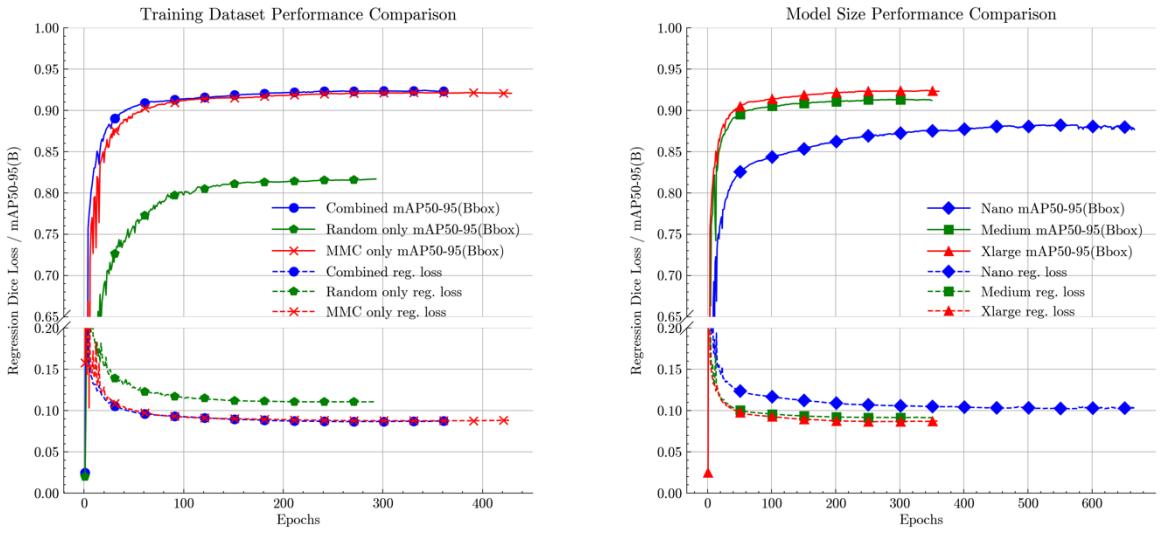
[2.] **Lower is better.**

[3.] **Closer to 0 is better.**

The training curves for the x-large YOLOv8-TO model trained on three different datasets are presented in Figure 6.17a). The datasets include optimized MMC structures only, random assemblies only, and a combination of both. Validation was performed on a separate subset of MMC-optimized samples for each training run. The model trained solely on random samples achieved a final bounding box mean average precision at IoU thresholds in the range of 50% to 95% ($mAP_{@0.5-0.95}$) of 0.82 and a validation dice loss of 0.11. The bounding box mAP was chosen as the preferred metric over the mask mAP due to its direct involvement in the design variables prediction. The model trained on MMC-optimized structures only attained a $mAP_{@0.5-0.95}$ of 0.921 and a validation dice loss of 0.0876. The combination of random and MMC-optimal images resulted in a slightly better performance, with a $mAP_{@0.5-0.95}$ of 0.924 and a validation dice loss of 0.0866.

Figure 6.17b) shows the training curves for three different YOLOv8 model sizes (nano, medium, and extra-large) trained on the combined dataset. The extra-large model achieved the highest $mAP_{@0.5-0.95}$of 0.924 for the ground truth bounding box and the smallest validation dice loss of

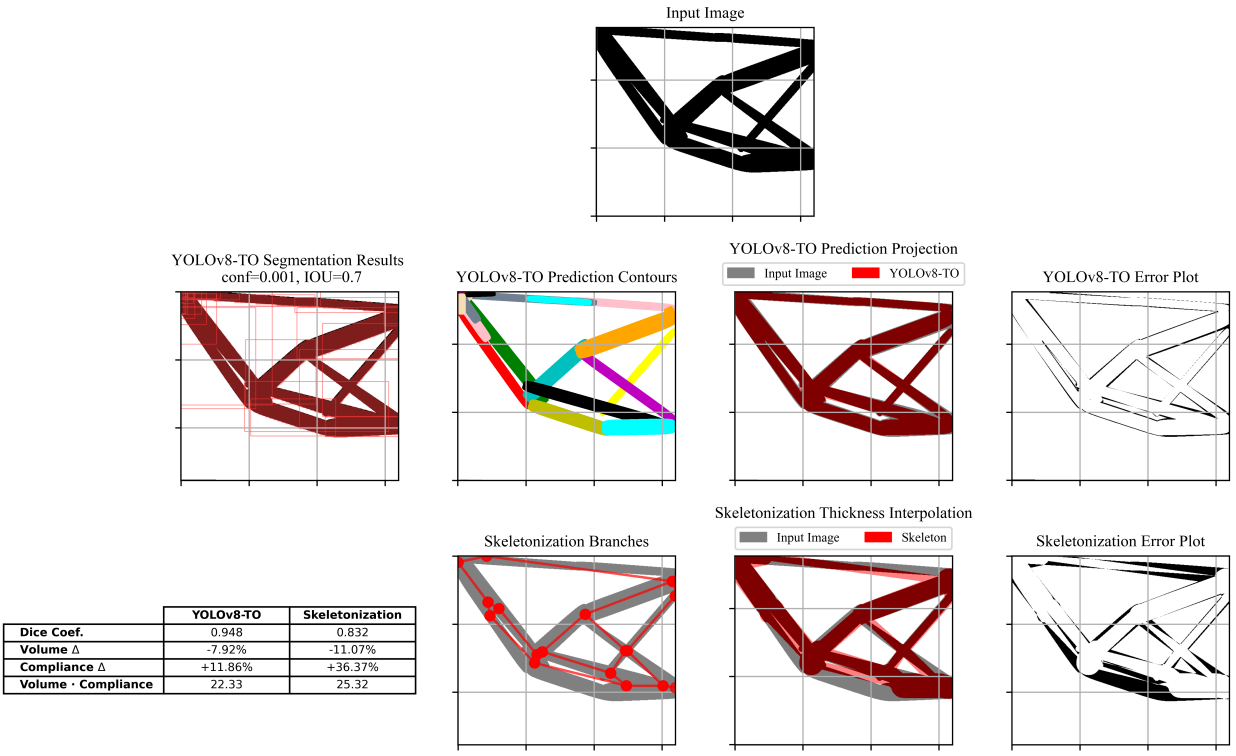
0.086. This indicates that increasing the model capacity allows for better learning of the complex structural patterns present in the topology-optimized designs.



a) b)

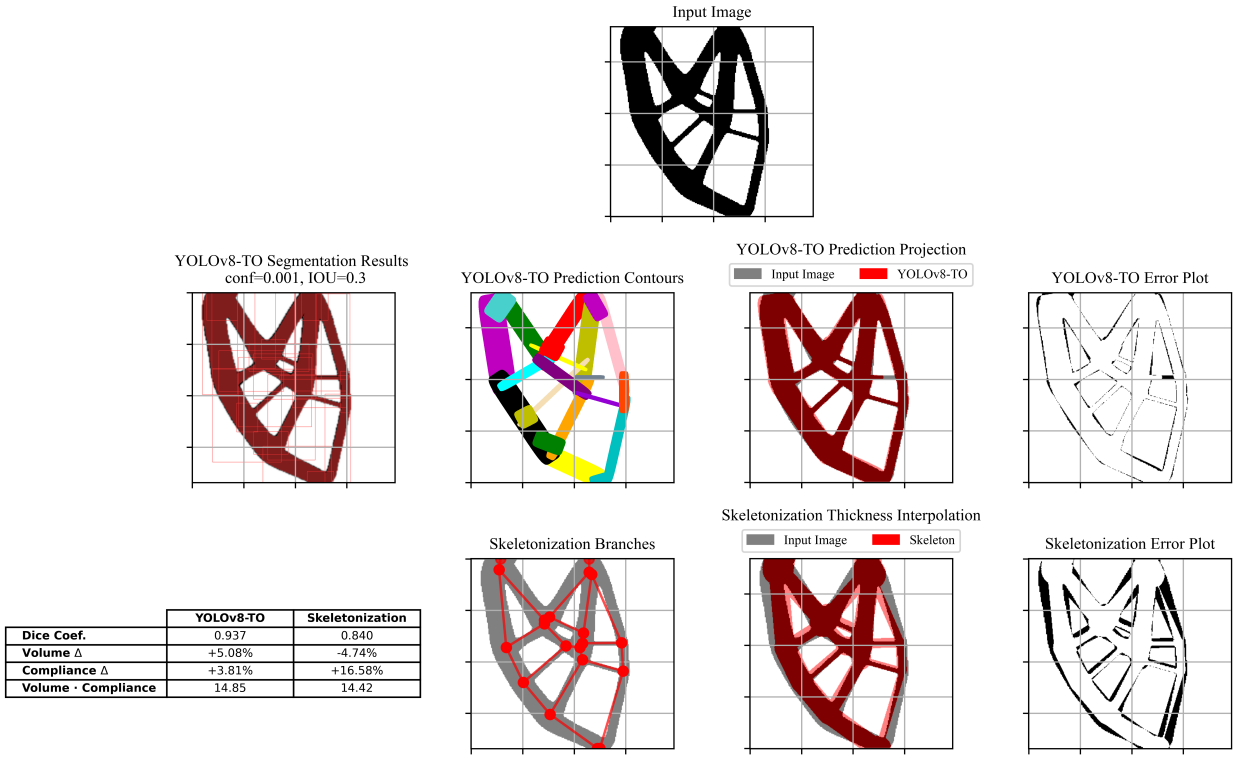
Figure 6.17: Comparative training curves for bounding box mean Average Precision (mAP) with a range of 50-95 and regression head loss across a) datasets incorporating MMC, random, and a combination of both, and b), various model scales.

Figure 6.18, Figure 6.19 & Figure 6.20 present sample results from the MMC, SIMP, and $SIMP_{5\%}$ test datasets, respectively. YOLOv8-TO, despite being trained only on MMC-optimized data, demonstrates its effectiveness in handling topologies obtained via the MMC algorithm, but also on topologies obtained with SIMP. It outperforms the skeletonization method even on truss-like structures in the $SIMP_{5\%}$ dataset.



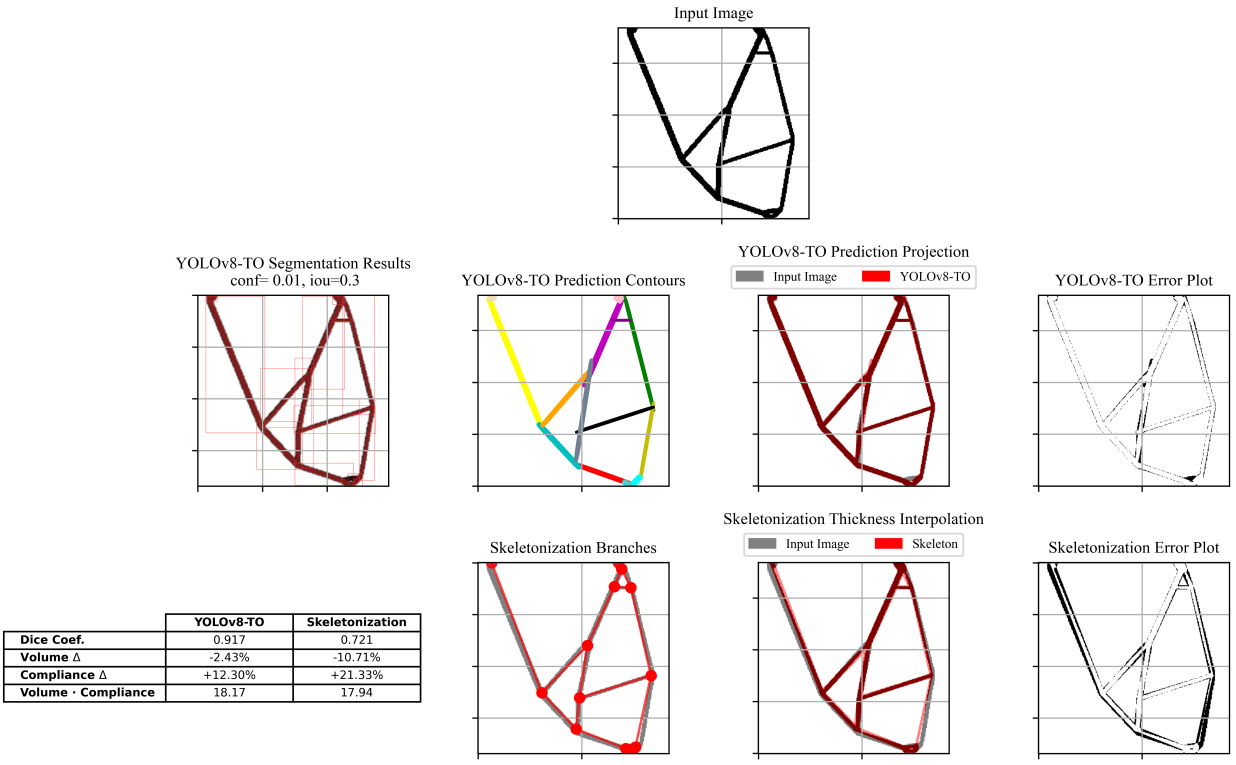
	YOLOv8-TO	Skeletonization
Dice Coef.	0.948	0.832
Volume Δ	-7.92%	-11.07%
Compliance Δ	+11.86%	+36.37%
Volume · Compliance	22.33	25.32

Figure 6.18: Results sample from the MMC test dataset.



	YOLOv8-TO	Skeletonization
Dice Coef.	0.937	0.840
Volume Δ	+5.08%	-4.74%
Compliance Δ	+3.81%	+16.58%
Volume · Compliance	14.85	14.42

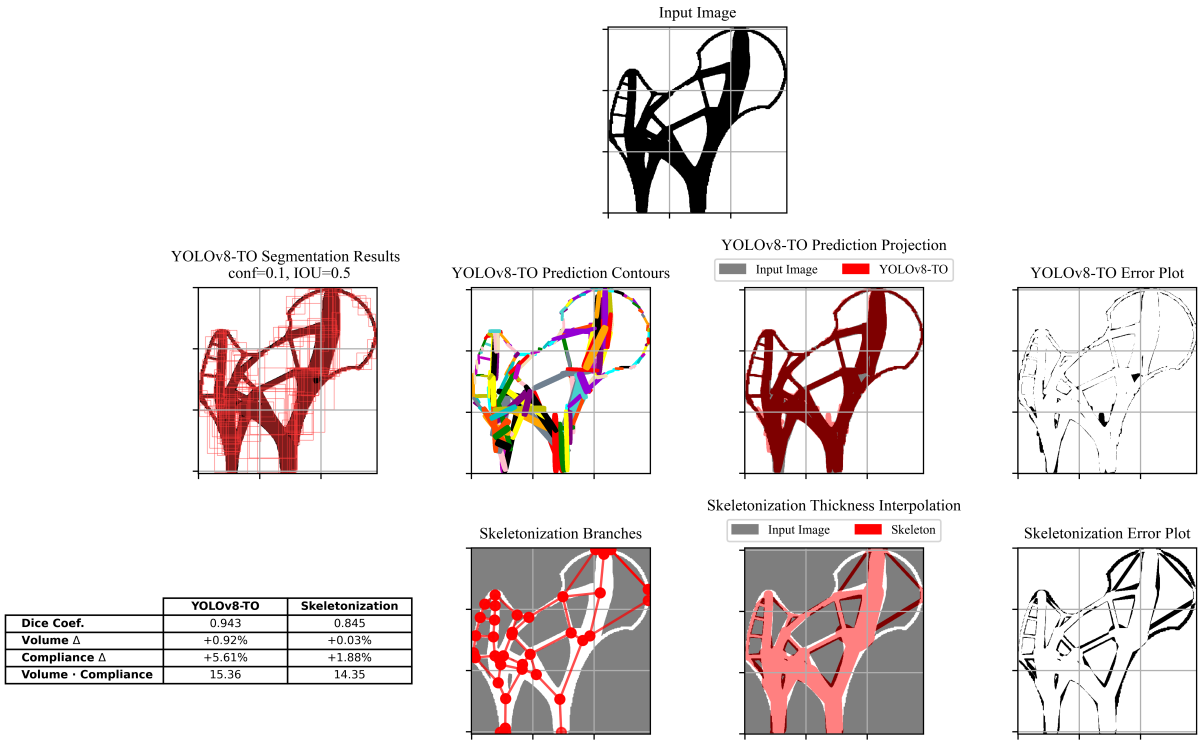
Figure 6.19: Results sample from the SIMP dataset.



	YOLOv8-TO	Skeletonization
Dice Coef.	0.917	0.721
Volume Δ	-2.43%	-10.71%
Compliance Δ	+12.30%	+21.33%
Volume · Compliance	18.17	17.94

Figure 6.20: Results sample from the $SIMP_{5\%}$ dataset.

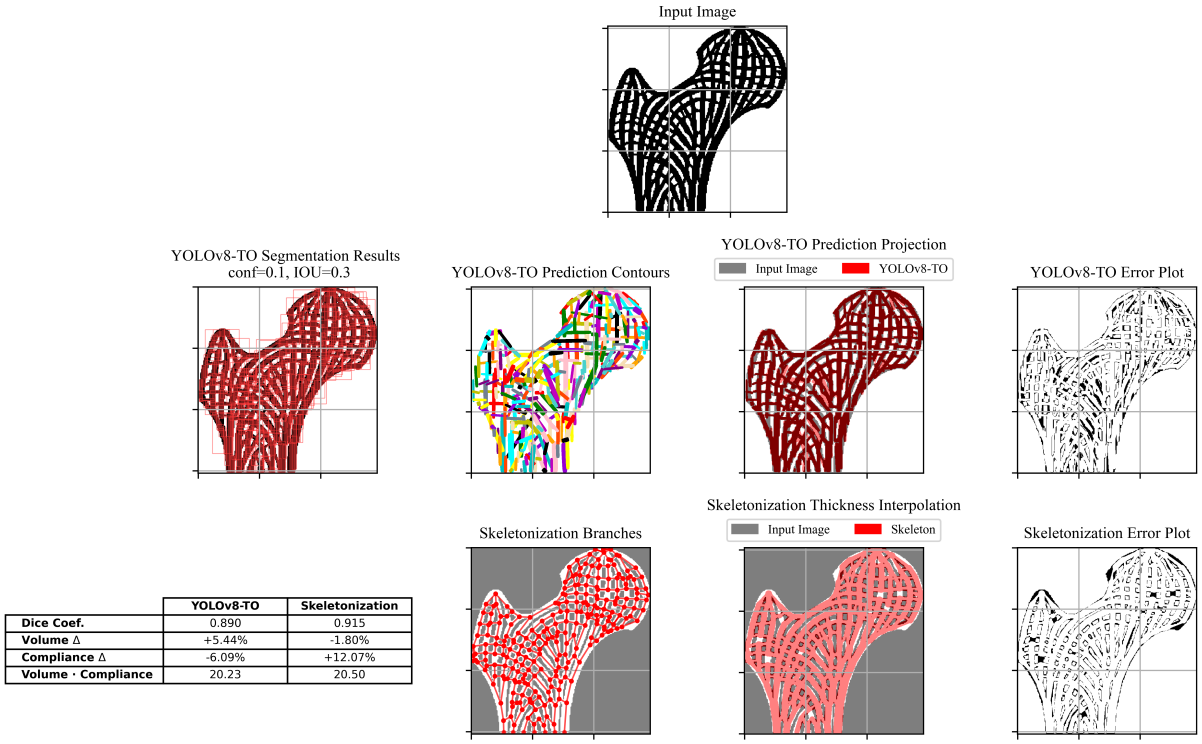
Figure 6.21 through 6.24 showcase the results of the OOD test samples. YOLOv8-TO achieves better image similarity compared to the skeletonization method on all OOD samples, except for the 2D femur with local volume constraint, where the skeletonization method yields a structure with a slightly higher dice coefficient.



	YOLOv8-TO	Skeletonization
Dice Coef.	0.943	0.845
Volume Δ	+0.92%	+0.03%
Compliance Δ	+5.61%	+1.88%
Volume · Compliance	15.36	14.35

Figure 6.21: Results for the 2D femur case with total volume constraint.

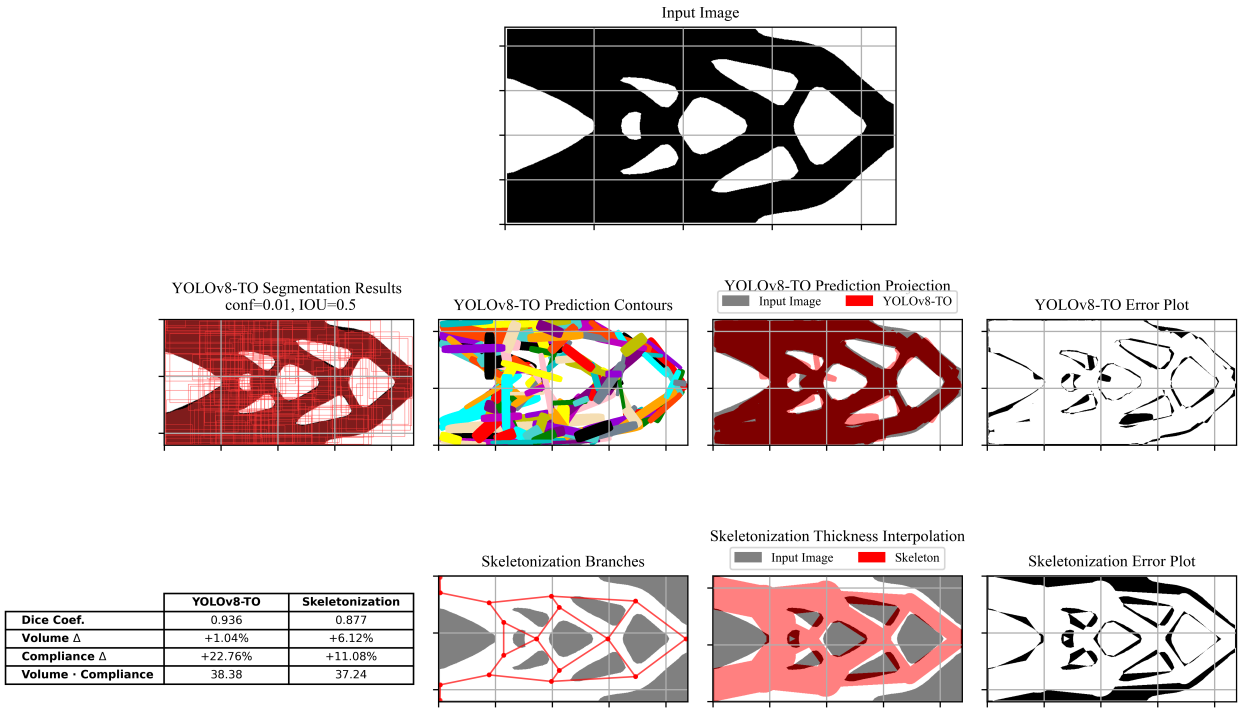
The 2D femur example with total volume constraint, illustrated in Figure 21, demonstrates YOLOv8-TO's ability to accurately reconstruct curved structural members by assembling multiple straight MMC components, as evident in the circular top right femur head. Despite YOLOv8-TO producing a more visually similar reconstruction (Dice coefficient: 0.94 for YOLOv8-TO vs 0.85 for skeletonization), the skeletonization method outperforms YOLOv8-TO in terms of compliance (relative difference: +1.88% for skeletonization vs +5.61% for YOLOv8-TO) and volume similarity (relative difference: +0.03% for skeletonization vs +0.92% for YOLOv8-TO).



	YOLOv8-TO	Skeletonization
Dice Coef.	0.890	0.915
Volume Δ	+5.44%	-1.80%
Compliance Δ	-6.09%	+12.07%
Volume · Compliance	20.23	20.50

Figure 6.22: Results for the 2D femur case with local volume constraint.

The 2D femur with local volume constraint, shown in Figure 6.22, is the OOD sample where the skeletonization method surpasses the YOLOv8-TO method on image similarity (Dice coefficient: 0.92 for skeletonization vs 0.89 for YOLOv8-TO). The accurate detection of skeleton nodes, facilitated by the presence of short structural members and numerous junctions, enables the skeletonization approach to fit curves precisely. However, when considering the structural efficiency, the YOLOv8-TO reconstruction still exhibits a lower product of volume and compliance (20.23 for YOLOv8-TO vs 20.50 for skeletonization), suggesting a more optimal utilization of the reconstructed material.



	YOLOv8-TO	Skeletonization
Dice Coef.	0.936	0.877
Volume Δ	+1.04%	+6.12%
Compliance Δ	+22.76%	+11.08%
Volume · Compliance	38.38	37.24

Figure 6.23: Results for the level set cantilever beam after 20 iterations.

The results of the partially optimized level set cantilever beam in Figure 6.23 reveal that the YOLOv8-TO model struggles to reconstruct structural members thicker than those encountered during its training phase. This limitation also arises from the use of an upper bound on the predicted components' thicknesses, resulting in an unnecessarily large number of components being predicted to achieve a reconstruction with a high Dice coefficient (0.94 for YOLOv8-TO vs 0.88 for skeletonization). Consequently, the YOLOv8-TO reconstruction exhibits a higher relative difference in compliance (+22.76% for YOLOv8-TO vs +11.08% for skeletonization) and volume (+1.04% for YOLOv8-TO vs +6.12%for skeletonization) compared to the skeletonization method.

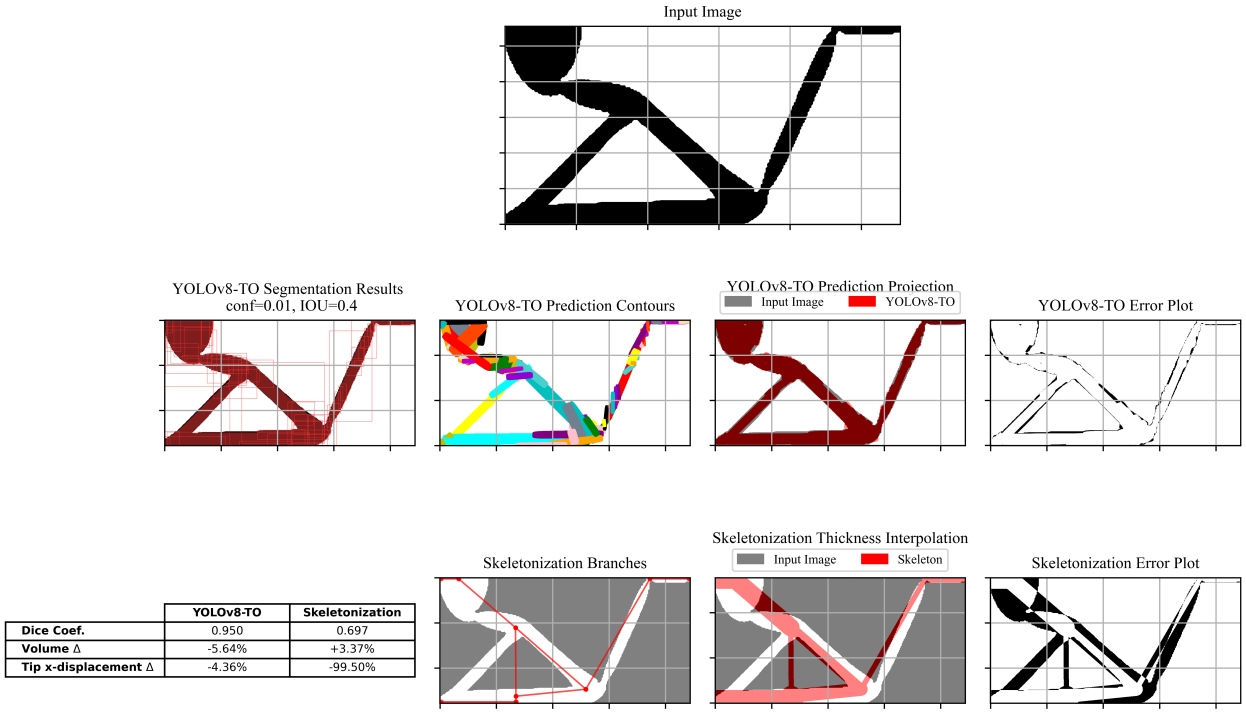


Figure 6.24: Results for the compliant mechanism force inverter.

The compliant mechanism results in Figure 6.24 demonstrate YOLOv8-TO's ability to accurately reconstruct crucial structural features, such as the mechanism's hinges, enabling the reconstructed structure to exhibit a tip displacement similar to the original design (relative difference: -4.36% for YOLOv8-TO vs -99.50% for skeletonization). In contrast, the skeletonization method fails to detect skeleton nodes at the hinge locations, resulting in straight bars where a thinning of the structure would normally be required to allow bending. Consequently, the skeletonization approach produces a stiff reconstructed structure that does not capture the essential characteristics of the input design. Additionally, the YOLOv8-TO reconstruction achieves a higher Dice coefficient (0.95 for YOLOv8-TO vs 0.70 for skeletonization), further highlighting its superior performance in reconstructing the compliant mechanism.

6.4 Discussion

In comparative analyses, YOLOv8-TO exhibits a marked superior performance over the skeletonization method, with an average Dice coefficient improvement of 13.84% in generating reverse-engineered designs on the three large test datasets. This notable advancement provides a

significant increase in visual similarity to the original structures, with maximum observed enhancements reaching as high as 20.78%.

6.4.1 Improvement in Precision

It was observed that the creation of skeletons typically encounters issues with the misplacement of nodes during the extraction process, primarily due to the reliance on geodesic distances for the skeleton node detection. Such an approach tends to place nodes at the centers of structural "centroids" rather than at the center of the actual junctions, leading to inaccuracies. An example from the MMC test dataset, illustrated in Figure 6.18, demonstrates a significant misplacement of the lower skeleton members by the skeletonization algorithm, resulting in a notable deviation between the reconstructed and the original structure. In contrast, the YOLOv8-TO reconstruction is remarkably precise, with only minor discrepancies in the thickness of the members that marginally affect structural compliance. The results from Table 6.2 validate the effectiveness of the thickness estimation algorithm described in Figure 6.16, as the skeletonization method produced structures with more similar volumes due to the use of mean diameter calculations to predict the final diameter of each skeleton branch. When considering only the compliance-volume product and disregarding the visual similarity to the original structure, the skeletonization method often produces structures with lower or similar values compared to YOLOv8-TO on the individual samples of the results section, suggesting its effectiveness in generating more structurally efficient designs.

6.4.2 Small Impact From Using Random Structures

The combination of randomly generated MMC assemblies, such as those illustrated in Figure 6.5, with optimal structures resulted in a 1.14% decrease in validation dice loss and a 0.33% increase in $mAP_{@0.5-0.95}$. Although using only random structures severely underperforms compared to using MMC-only or combined datasets, the marginal performance improvement observed with the combined dataset suggests that the inclusion of random samples can be beneficial. While the higher training cost associated with processing a larger number of training samples at each epoch for the combined dataset is a consideration, it can be argued that the one-time training cost is less important than the potential long-term benefits of a better-performing model. The improved performance, even if small, could lead to more accurate and efficient results when the model is deployed and

used extensively in real-world applications. Future work will explore the potential of using more structured noise to improve performance while minimizing the impact on training costs.

6.4.3 Scaling Yields a Significant Performance Boost

Scaling the model size upwards led to notable improvements in performance metrics and training efficiency. Transitioning from the nano-sized model to the medium-sized one yielded a 3.50% enhancement in the validation bounding box $mAP_{@0.5-0.95}$, coupled with a 10.41% reduction in reconstruction dice loss. As an additional scaling benefit, the larger models converged much faster, requiring almost half the number of epochs (350 vs 650) to reach the same level of performance as the nano model. Furthermore, upgrading from a medium-sized model to an extra-large variant resulted in a 1.20% increase in $mAP_{@0.5-0.95}$ and a 5.29% diminution in reconstruction dice loss.

6.4.4 Faster Inference Speed on Common Workspace Hardware

A comparison of inference cost was made by deploying the three tested YOLOv8-TO model sizes and the skeletonization method on a MacBook Air M2 CPU. Table 6.3 presents the processing times on the two 2D femur OOD test samples with total and local volume constraints, which were processed at resolutions of 736x640 and 736x672 pixels respectively. The inference time of the skeletonization method does not include the thickness estimation algorithm. The skeletonization method's inference cost exhibits a substantial increase (5.50s. vs 53.58s.) when moving from total volume constraint to local volume constraints, indicating that its processing time scales with the number of detected skeleton members. In contrast, the relatively constant inference processing time between the two images for the YOLOv8-TO models demonstrates that the forward pass through the model scales with the image resolution. The data also highlights the significant processing time required for mask-based NMS operations, emphasizing the costly nature of pixel-wise operations for each detected component. The paper's implementation uses a basic batched loop over the tensor of detected components, which lacks performance optimization. This approach results in scalability issues as the number of components increases, a problem highlighted by the increasing processing times observed across the two test samples. Nevertheless, once trained, YOLOv8-TO achieves faster inference speed on the CPU than the skeletonization method and can offer a balance between processing speed and accuracy depending on whether the bounding box or the mask-based NMS operation is selected at the post-processing step.

Table 6.3: Comparison of inference cost of YOLOv8-TO vs Skeletonization method.

Test sample	Model / Method	Inference processing time (s.)	Inference + Box NMS (s.)	Inference + Mask NMS (s.)
2D femur with total volume constraint	YOLOv8-TO Nano	0.56	0.62	20.12
	YOLOv8-TO Medium	1.35	1.39	20.49
	YOLOv8-TO X-large	2.76	2.80	17.49
	Skeletonization	5.50	-	-
2D femur with local volume constraint	YOLOv8-TO Nano	0.57	0.76	153.55
	YOLOv8-TO Medium	1.36	1.54	172.42
	YOLOv8-TO X-large	2.83	2.98	186.48
	Skeletonization	53.58	-	-

6.4.5 Sensitivity Issues

One significant challenge with YOLOv8-TO is its strong reliance on the confidence and IoU thresholds during the NMS post-processing step, which affects the number of reconstructed structural elements. The confidence threshold determines the minimum predicted probability required for a detected component to be considered valid, while the IoU threshold is used to suppress overlapping components. Higher confidence thresholds lead to fewer detected components, as only the most confident predictions are retained, reducing false positives but potentially increasing false negatives. Similarly, higher IoU thresholds result in more aggressive suppression of overlapping components, reducing false positives but possibly eliminating true positive detections. Balancing these thresholds is crucial to minimize both false positives and false negatives in the reconstructed structure. Figure 6.25 illustrates the sensitivity of the interpreted structure to different confidence and IoU threshold pairs for the level set OOD example, highlighting the need to test various values when reconstructing an image to find the optimal balance. The number of interpreted components is also higher than the skeletonization output, as shown in Figure 6.23, which is a recurrent weakness when applying YOLOv8-TO to "thick" structures. To improve the performance of the YOLOV8-TO model, one will need to address the following three challenges:

- The first challenge is the model's reliance on predefined design variable min/max boundaries, which limits the maximum thickness of individual predicted elements. This limitation often requires the aggregation of multiple components to mimic the desired thickness, suggesting that expanding these boundaries could be a promising direction for future exploration.
- The second challenge stems from the constraints of only using straight MMC components during training, forcing the model to combine multiple straight components to reconstruct curved segments. Figure 6.21 exemplifies this issue, where the model assembles numerous tiny MMC components to reconstruct the curved femoral head in the upper right corner. Incorporating curved MMC components [170], which are supported by the YOLOv8 framework as a separate class of object, will be investigated in future work. Nevertheless, the model's ability to achieve this level of reconstruction using only MMC-optimized and random MMC data is remarkable.
- The third challenge arises from the presence of overlapping components in the MMC-optimized training dataset, which forces the model to predict the presence of hidden components during training. This is a consequence of the MMC method's need to "hide" or move components out of the domain to remove their contribution to the overall design. Implementing a data processing step to remove hidden MMC components or employing a feature-mapping method that handles intermediate densities [171] could prevent the model from being forced to detect redundant overlapping structures.

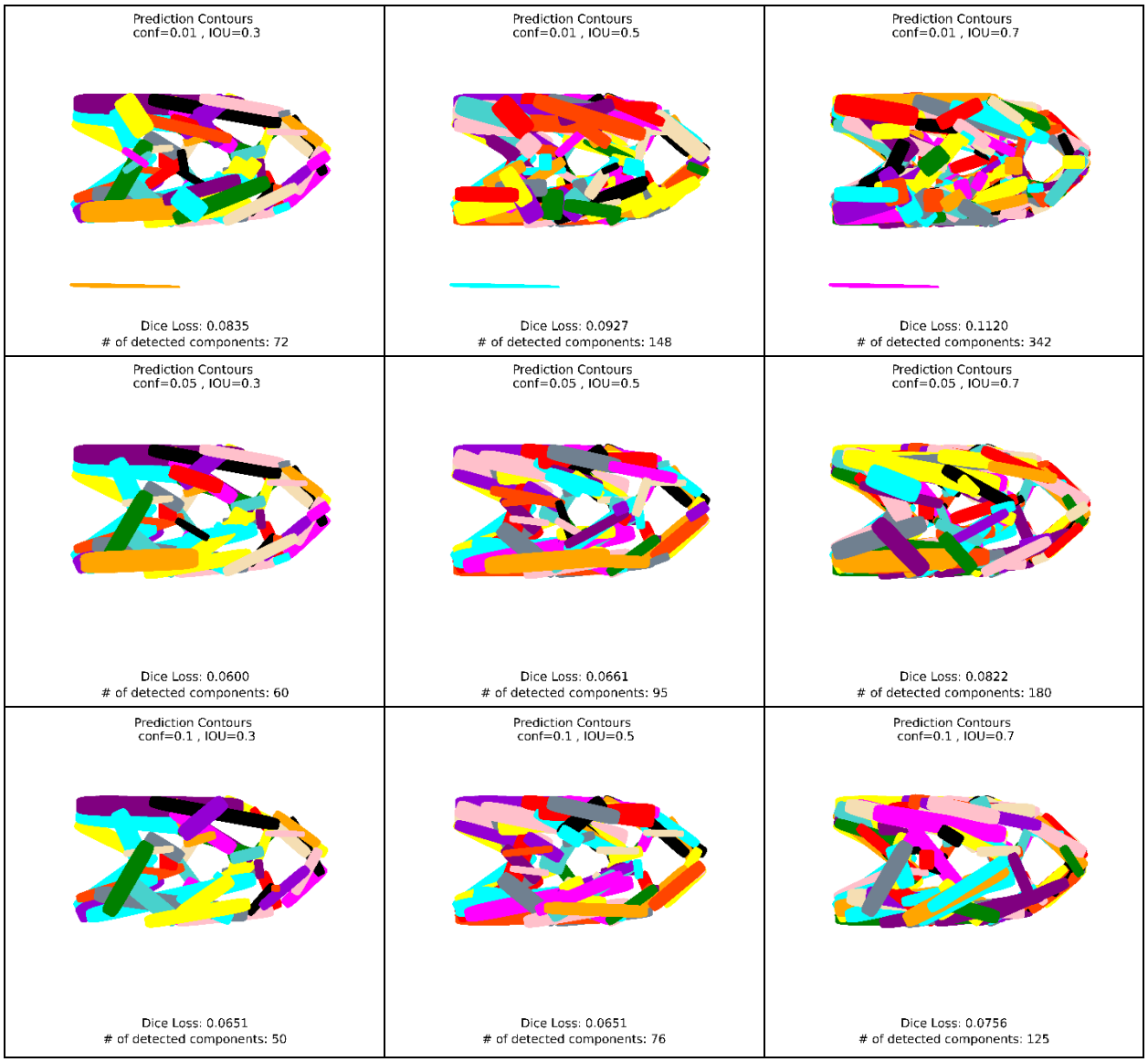


Figure 6.25: Effect of confidence and IoU threshold on detected components for the OOD level set example with YOLOv80TO x-large.

6.5 Conclusion

This paper presents YOLOv8-TO, a novel approach that harnesses the power of the YOLOv8 computer vision neural network model to bridge the gap between density-based TO outputs and interpretable geometric parameters. As the first application of an instance segmentation model in the field of TO, this approach significantly enhances the post-processing stage of structural optimization, enabling a more seamless integration with subsequent design validation and exploration processes.

Our findings demonstrate that YOLOv8-TO can accurately identify and reconstruct the shapes of structural components within monochrome images of optimized structures generated from various TO methods, providing a geometrically meaningful representation that is directly usable within CAD tools. The trained model was shown to be a substantial advancement over traditional skeletonization methods, particularly in its ability to generate visually similar reverse-engineered structures. One significant advantage of this approach is that the size of the design variable regression head is independent of the input image resolution, thanks to the use of feature-mapping methods. This means that the model can predict the same number of design variables regardless of the resolution of the input image, making it a notable improvement over other machine learning approaches to TO that are often bound to coarse density resolutions. The superiority of YOLOv8-TO in handling a diverse range of structural geometries, combined with its low inference latency, underscores its potential to significantly streamline the design process in engineering applications.

The high performance and ease of use of YOLOv8-TO open the possibilities for innovative workflows in the structural design process. One such interesting application is the ability to convert hand-drawn sketches of structures into parameterized geometry by applying the model to images of such sketches. Appendix C showcases examples of doodles created using a laptop trackpad and a pen on paper, along with their corresponding reverse-engineered structures generated by the segmentation model. With YOLOv8's focus on efficiency, the trained model can be easily deployed on common workspace hardware, such as laptops or even mobile devices, enabling a seamless transition from conceptual sketches to parameterized initial designs. This ease of deployment significantly enhances the accessibility and practicality of the proposed approach, making it a valuable tool for engineers and designers in their everyday work environments.

The performance of YOLOv8-TO is constrained by the specifics of its training settings and datasets, as it currently only supports straight MMC components and has been trained exclusively on optimized and random MMC structures. Consequently, its performance may be limited when dealing with sub-structures that fall outside the dimensional bounds of the components encountered during training. Furthermore, the proposed segmentation mask non-maximum suppression (NMS) post-processing step is computationally expensive compared to the default bounding box NMS algorithm, necessitating improvements in computational efficiency before it can be applied to larger images or extended to 3D inputs.

Future research should focus on improving the generalizability of YOLOv8-TO across different structural complexities. This could involve the incorporation of more complex geometric primitives and the development of advanced data augmentation strategies to enhance OOD generalization. Enhancing the algorithm's scalability to efficiently handle 3D structures and integrating it more seamlessly with CAD and FEA tools are also critical areas for development. Moreover, exploring adaptive non-max suppression thresholds and investigating alternative loss functions could yield further improvements in the method's accuracy and reliability.

YOLOv8-TO represents a significant step forward in topology optimization post-processing. By providing an efficient and accurate means of converting density-based optimization outputs into geometrically interpretable parameters, this method holds the promise of streamlining the design process, fostering innovation, and enhancing the manufacturability of optimized structures.

Data & Code Availability

The code and data used for training and testing the YOLOv8-TO model will be made publicly available on GitHub at https://github.com/COSIM-Lab/YOLOv8-TO. This repository contains the necessary scripts, datasets, and instructions to reproduce the results presented in this article.

Acknowledgements

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC) [funding reference number 569251]. This research was enabled in part by support provided by Calcul Québec (https://www.calculquebec.ca) and the Digital Research Alliance of Canada (DRAC) (alliancecan.ca).

CHAPTER 7 DISCUSSION

This thesis presents an exploration of ML methodologies applied to TO, addressing key challenges in computational efficiency and physics-aligned model training. The findings from Chapters 4, 5, and 6 collectively illustrate a multifaceted approach to enhancing the TO process for direct design and post-processing applications, each contributing unique insights and advancements. Figure 7.1 positions these contributions in the TO process as presented in Chapter 1.

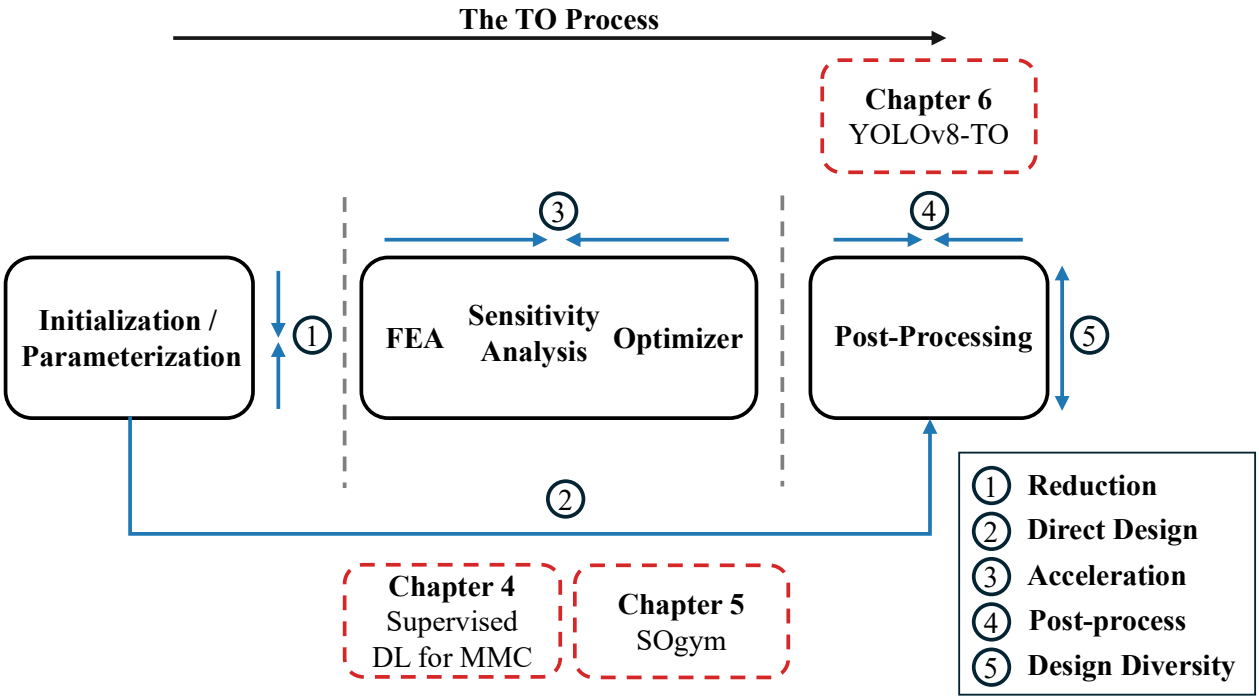


Figure 7.1: Positioning of the three articles presented in this thesis in relation to the ML-TO applications presented in Chapter 1. Chapter 4 and 6 focus on addressing key limitations of direct-design applications. Chapter 5 present a novel framework to leverage robust computer vision models for the post-processing of TO outputs.

At the beginning of this PhD, the limitations of supervised learning methods relying on image similarity loss functions were identified. SO1 was established to test the hypothesis that mesh-independent approaches could offer a solution to 'scale-out' of the problem of non-physical model-generated structures. In other words, by increasing the diversity and size of the training dataset, the problem could potentially be resolved. Chapter 4 describes a supervised learning model based on MMC design variables outputs and a problem description vector as input, enabling a DL model architecture without any notion of FEA mesh or discretized elements. Experimental results on the

largest and most diverse training dataset did not manage to solve the structural disconnection issue, and the only computational efficiency improvement that such a model could provide was through the generation of initial designs for conventional optimization.

Nevertheless, these results do not allow for the complete dismissal of image regression methodologies for generating optimal structures from ML models. In the separate field of language modeling, LLMs have demonstrated that coherent language can be generated without explicit grammatical or syntactic rules during training, focusing instead on predicting the next word token. Drawing on that analogy, one might envision an image-based TO approach without any physical grounding that, at a sufficient scale, could produce structures with a very low probability of being non-physical, similar to how modern LLMs generate semantically correct sentences with very high probabilities. However, in contrast to language modelling, the internet is not filled with petabytes of optimal structures on which such large-scale training can be envisaged. Until such datasets exist, physics-informed approaches are the only methodologies showing the potential to generate useful solutions for direct design TO applications.

Framing TO as a RL task addresses SO2 by incorporating the physics of the problem directly into the reward function, thus eliminating the need for backpropagation through the FEA operation. This approach offers the advantage of flexibility in designing a wide range of reward functions but comes with the drawback of reduced sample efficiency compared to the direct feedback provided in supervised learning. SOgym was developed with this inherent flexibility, serving as a proof of concept for SO2 and demonstrating that RL can effectively integrate TO physics into the model's training objective. However, baseline results indicated that the performance of RL-generated structures was inferior to those produced by conventional optimization methods. As a result, SO2 can be considered partially achieved. Future research should focus on bridging this performance gap, with specific directions proposed in the next chapter.

During the experimentations of SO1 and SO2, the limitations of the MMC method became increasingly obvious, particularly when compared to the more robust and efficient SIMP method for generating optimal topologies under the problem conditions explored in all three articles. The MMC method proved inefficient, requiring excessive computation time and a high number of iterations to reach a final design. This inefficiency prompted the pursuit of SO3, which aimed to develop a universal interface between density-based TO outputs and parameterized feature-

mapping methods. This interface facilitates the automatic mapping of implicit density distributions to explicit geometric models, streamlining the post-processing of density fields into precise geometric structures. Chapter 6 introduces YOLOv8-TO, a tool designed to efficiently interpret and generate parametric geometries from black-and-white density distributions. This CV model demonstrated strong performance on complex OOD images, leveraging the YOLOv8 architecture, which is compact enough, both in terms of memory size and processing power requirements, to run on conventional workstations and even mobile devices. However, similar to the challenges encountered in image-based direct design approaches, the issue of disconnected structures persisted when testing the mechanical response of the reconstructed geometries. This reiterates the fundamental challenge of using image-similarity training objectives for structures required to meet specific boundary conditions.

The primary goal of this thesis, as defined in the MO, is to develop scalable ML-TO approaches that do not impose hard limitations on their applicability across various structural design challenges. This includes adapting to different boundary conditions, physics objectives, nonlinear structural responses, and a wide range of domain shapes and dimensions. Achieving this goal involves overcoming two key challenges. The first is the challenge of problem representation, where model inputs must be flexible enough to accommodate any parameters defining the TO problem. The second challenge concerns the model outputs, which require an architecture capable of generating structures of varying complexity.

Feature-mapping methods play a crucial role in addressing both challenges. By decoupling the FEA resolution from the design variables, these methods enable the model to interact explicitly with the topology. This flexibility in model inputs eliminates the need for fully convolutional architectures, allowing the use of vector-space models to represent problem parameters, as demonstrated with the description vector introduced in Chapters 4 and 5. Similar to how LLMs use token vectors to represent positions in a vocabulary, the description vector approach can be scaled to represent a 'vocabulary' encompassing diverse design problems and parameters. Chapter 4 also introduced the first mesh-independent model output, capable of scaling to any domain, though with some limitations on the number of generated components. In Chapter 6, the MMC method demonstrated how an instance segmentation computer vision model can reconstruct complex structures with hundreds of detected components, without being constrained by the size of input images.

Additionally, framing TO as a RL task in Chapter 6 provides a second key strategy for overcoming the scaling limitations of direct design approaches. In the supervised learning approach of Chapter 4, the fixed output layer size limited the number of design variables generated in each forward pass. However, by incorporating a time dimension into an RL episode, this limitation is mitigated, enabling the model to generate the structure over multiple forward passes and thereby allowing for a variable number of design variables to be produced. The use of the expected sum of rewards as a training objective further enhances flexibility, enabling the incorporation of various physical phenomena during training and paving the way for a multi-task RL agent trained on a broad range of TO problems.

Although this thesis has made substantial progress in enhancing the scalability of ML-TO approaches, the limited range of experiments conducted has restricted the ability to fully demonstrate their potential applications. To fully validate the advantages discussed, additional experimentations, as outlined in the next chapter, will be essential. Consequently, the primary objective of this research can be considered partially achieved. Nevertheless, this thesis has made valuable contributions to the ML-TO field by introducing important new research directions and addressing critical challenges that must be overcome to transition from the current state of over-hyped research to practical industrial applications.

7.1 Contributions and Importance of Research

This thesis has made several key contributions to the direct design and post-processing subfields of ML-TO. First, it introduced the first mesh-independent direct design DL model, trained with a design variable regression loss function. While this model demonstrated promise, it also revealed significant performance limitations that warrant further investigation.

Additionally, the thesis introduced SOgym, the first DRL environment for TO, which is open-sourced for the community. SOgym demonstrated that using a physics-based reward function can address the issue of 'non-physical' structures commonly produced by regression-trained models.

Lastly, YOLOv8-TO was developed, significantly outperforming current SOTA skeletonization approaches. This novel CV technique represents the first DL approach for post-processing TO showing robust performance in reconstructing complex structures well outside its training distribution. A summary of the specific contributions to the ML-TO subfields of direct design and post-processing is provided in Table 7.1.

Table 7.1: Summary of major contributions to the direct design and post-processing fields of the machine learning for topology optimization community.

ML-TO subfield	**Contributions**
Direct design	• Developed a mesh-independent model architecture by leveraging the MMC design variables. • Designed and published SOgym, an open-source Python RL environment for the development of direct design ML-TO approaches with physical reward functions. • Benchmarked PPO and DreamerV3 DRL agents on SOgym and compared their performance with conventional MMC optimization. • Compared the learning rates of the baseline RL agents with the in-class experiment from [144].
Post-processing	• Developed YOLOv8-TO, a novel instance segmentation methodology to extract the geometric definition of sub-components of black-and-white structures. • Implemented a segmentation mask-based NMS algorithm for the filtering of detected structural components in YOLOv8-TO.

In addition to the three core articles presented in this thesis, this PhD also led to presentations at international conferences and fostered collaborations with colleagues from the COSIM (*Conception de Systèmes Intelligents et Mécatronique*) research group at Polytechnique Montréal, as well as with the *TopOpt* group from the Technical University of Denmark (DTU), on papers that further advanced the field of TO.

Table 7.2 provides a list of all academic contributions stemming from this PhD.

Table 7.2 : Academic contributions completed during this PhD.

1	**Rochefort-Beaudoin, T**., Vadean, A., Aage, N., Achiche, S. (2024). Structural Design Through Reinforcement Learning. *arXiv preprint arXiv:2407.07288*. Submitted to *Engineering Applications of Artificial Intelligence* on July 12th , 2024.
2	**Rochefort-Beaudoin, T.,** Vadean, A., Achiche, S. (2024). Learning Structures Through Reinforcement Learning. Presented at the *9th European Congress on Computational Methods in Applied Sciences and Engineering* (ECCOMAS 2024).
3	**Rochefort-Beaudoin, T**., Vadean, A., Achiche, S., Aage, N. (2024). From Density to Geometry: Instance Segmentation for Reverse Engineering of Optimized Structures. *arXiv preprint arXiv:2404.18763*. Submitted to *Engineering Applications of Artificial Intelligence* on May 13th, 2024.
4	**Rochefort-Beaudoin, T**., Vadean, A., Gamache, J. F., Achiche, S. (2023). Supervised deep learning for the moving morphable components topology optimization framework. *Engineering Applications of Artificial Intelligence*, *123*, 106436.
5	Gamache, J. F., Vadean, A., Capo, M., **Rochefort-Beaudoin, T.**, Dodane, N., Achiche, S. (2023). Complexity-driven layout exploration for aircraft structures. *Design Science*, 9, e13.
6	**Rochefort-Beaudoin, T.**, Vadean, A., Gamache, J. F., Achiche, S. (2022). Comparative study of first-order moving asymptotes optimizers for the moving morphable components topology optimization framework. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* (Vol. 86212, p. V002T02A028). American Society of Mechanical Engineers. [Paper presented at IDETC 2022].
7	Capo, M., Gamache, J.-F., **Rochefort-Beaudoin, T.,** Vadean, A., Achiche, S. (2021). A Novel use of the Ground Structure Topology Optimization for the Design of Pressurized Stiffened Panels. Presented at the *International Congress in Theoretical and Applied Mechanics* (ICTAM) Milano 2020+. [Accepted: Poster, Video Presentation and Extended Abstract].

7.2 Limitations

The three articles presented in this thesis act as initial exploratory studies, introducing innovative research directions aimed at addressing key limitations in the existing ML-TO literature. To manage the computational demands, the scope of the TO problems used for training, validation, and testing was intentionally constrained. All three chapters utilized MMC components with either

constant or linearly varying thickness. The direct design approaches discussed in Chapters 4 and 5 focused exclusively on the classical compliance minimization under a volume constraint, assuming linear elasticity conditions. The experiments were further restricted to a simple rectangular 2D domain with full supports and point loads, which are typical setups for exploratory research in TO. For post-processing in Chapter 6, only 2D black-and-white structures were used for training and testing.

While these approaches have demonstrated potential, significant developments are still needed before they can be considered viable for industrial applications. Since industrial applications predominantly involve 3D problems, extending SOgym and YOLOv8-TO to 3D scenarios will be essential to validate the robustness and practicality of the proposed methods.

CHAPTER 8 CONCLUSION AND RECOMMENDATIONS

As ML-TO is still an emerging research field with still a limited number of publications supporting its potential for industrial applications, this project set out to explore various strategies to address critical limitations highlighted in the existing literature. Early in this research and development activity, it became evident that the current approaches lacked essential characteristics necessary for a truly scalable ML-TO direct design methodology.

Consequently, the focus of this thesis shifted towards addressing these foundational challenges by utilizing feature-mapping methods and reinforcement learning. The contributions detailed in this work have successfully enhanced each of the scaling properties outlined in Chapter 3, which are summarized below:

Decoupling from FEA mesh: This thesis introduces innovative ML-TO model architectures and training paradigms based on the MMC feature-mapping method, which enables the model to explicitly generate structures without relying on a discretized elemental representation of the topology.

Boundary conditions & domain shape generalization: Mesh-independent model architectures remove the need for fully convolutional architectures which make use of sparse and inefficient model inputs. The direct design approaches in this thesis make use of a novel description input vector which can be extended to encode details about many types of boundary conditions and domain shapes.

Physics generalization: SOgym allows to include any physics which can be evaluated via FEA to be used as a reward function without the need to implement the backpropagation operation through the FEA. This greatly simplifies the development and implementation steps and opens the way for multi-task agents to be developed in the future.

Low breakeven threshold: A novel training paradigm was developed for the SOgym environment where a training distribution was defined instead of a training dataset. This allows the model to experience a wide range of problems during training, without the need to generate a potentially repetitive and exhaustive training dataset which is costly to generate via conventional optimization.

It is important to note that the true innovation in the proposed approaches lies in the methodologies, which effectively remove several of the limits on the scalability properties discussed above. However, much of this scalability potential has not yet been demonstrated experimentally. These methods primarily serve as initial demonstrations, and their full potential as scalable solutions still requires validation through further empirical studies.

In summary, this thesis proposes to frame direct design approaches using RL in order to tackle the limitations currently exhibited in the ML-TO literature. While feature-mapping methods are not widely used in conventional TO due to their inefficient convergence behavior, it is proposed as a key enabler in ML-TO approaches, unlocking key scaling properties like mesh-independence and low-dimensional action spaces. Through its mapping operation, it also provides a source of labelled data to develop a novel instance segmentation approach, YOLOv8-TO, for the automatic post-processing of structures.

As the work presented here serves primarily as an initial exploration, future efforts should focus on refining and enhancing these methods to increase their relevance and applicability in industrial contexts. Below is a summary of potential future development for both direct design and post-processing ML-TO approaches:

Direct design:

- The current description input vector used in both supervised and RL direct design approaches is limited to rectangular domains and point loads. To provide a more generalizable and compact representation of various domain shapes, future research can explore the use of shape encoding functions, such as Fourier descriptors.
- The representation of boundary conditions should be expanded to accommodate multiple types of loads and physics. A potential direction proposed in this PhD is to describe the problem using a textual prompt and feed it to an LLM encoder, with its fixed-size encoding output serving as a universal description vector for a wide range of TO problems.
- The ability of RL to generate a variable number of design variables should also be validated. This can be achieved by introducing a ‘stop’ action, allowing the agent to terminate an episode early if no additional components are necessary. Additionally, implementing a

reward function that incentivizes simpler structures, characterized by fewer components, would likely be beneficial.

- Beyond the obvious extension of SOgym to 3D structures, future research should study the integration of diverse physics into the SOgym environment as conditional rewards can enable the training of multi-task agents capable of addressing a broad spectrum of physical TO objectives. Furthermore, incorporating qualitative objectives, like human aesthetics, through reinforcement learning from human feedback (RL-HF) can enhance the system's ability to produce designs aligned with subjective preferences.

Post-processing:

- Given that YOLOv8 already supports multiple classes of detected objects, YOLOv8-TO can be expanded to handle various high-level geometries as reconstruction targets. The primary challenge here is generating a training dataset of segmented geometries, which can be tackled by creating optimal topologies using non-linear MMC components.
- The segmentation mask NMS plays a critical role in accurately reconstructing sub-structures with overlapping bounding boxes. Future work should aim to improve its computational efficiency by vectorizing operations and reducing output sensitivity to IOU and confidence thresholds through adaptive NMS techniques.
- To further validate the industrial relevance of YOLOv8-TO, extending its application to 3D structures and improving its integration with CAD software is essential. Future work should focus on refining the interface between YOLOv8-TO and CAD systems, ensuring seamless data transfer and enhancing the accuracy of 3D reconstruction. This can involve developing specialized modules for CAD integration and testing the system's performance in real-world industrial scenarios.

DECLARATION OF GENERATIVE AI AND AI-ASSISTED TECHNOLOGIES IN THE WRITING PROCESS

During the preparation of this work, the author used GPT-4o (version 2024-05-13) to enhance the clarity and conciseness of the language of their own text. After using this tool/service, the author reviewed and edited the content as needed and takes full responsibility for the content of the thesis.

REFERENCES

[1] P. W. Christensen and A. Klarbring, *An Introduction to Structural Optimization*. Springer Netherlands, 2008.

[2] G. Eiffel, "Tour de 300 metres, Société des Imprimeries Lemercier, 57, rue de Seine, Paris, 47 plates," *Lemoine B,* 2015.

[3] J.-H. Zhu, W.-H. Zhang, and L. Xia, "Topology Optimization in Aircraft and Aerospace Structures Design," *Archives of Computational Methods in Engineering,* vol. 23, no. 4, pp. 595-622, 2016/12/01 2016, doi: 10.1007/s11831-015-9151-2.

[4] K. Svanberg, "The method of moving asymptotes—a new method for structural optimization," *International Journal for Numerical Methods in Engineering,* vol. 24, no. 2, pp. 359-373, 1987, doi: https://doi.org/10.1002/nme.1620240207.

[5] K. Svanberg, "MMA and GCMMA-two methods for nonlinear optimization," *vol,* vol. 1, pp. 1-15, 2007.

[6] K. Mo, H. Wang, Z. Cheng, and Y. Li, "IRA assisted MMC-based topology optimization method," p. arXiv:1810.07021. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2018arXiv181007021M

[7] O. Sigmund, "EML webinar overview: Topology Optimization — Status and Perspectives," *Extreme Mechanics Letters,* vol. 39, p. 100855, 2020/09/01/ 2020, doi: https://doi.org/10.1016/j.eml.2020.100855.

[8] S. Balay *et al.*, "PETSc," *See http://www. mcs. anl. gov/petsc,* 2001.

[9] N. Aage, E. Andreassen, B. S. Lazarov, and O. Sigmund, "Giga-voxel computational morphogenesis for structural design," *Nature,* vol. 550, no. 7674, pp. 84-86, 2017/10/01 2017, doi: 10.1038/nature23911.

[10] M. Baandrup, O. Sigmund, H. Polk, and N. Aage, "Closing the gap towards super-long suspension bridges using computational morphogenesis," *Nature Communications,* vol. 11, no. 1, p. 2735, 2020/06/01 2020, doi: 10.1038/s41467-020-16599-6.

[11] N. Aage, E. Andreassen, and B. S. Lazarov, "Topology optimization using PETSc: An easy-to-use, fully parallel, open source topology optimization framework," *Structural and Multidisciplinary Optimization,* vol. 51, no. 3, pp. 565-572, 2015/03/01 2015, doi: 10.1007/s00158-014-1157-0.

[12] S. C. Subedi, C. S. Verma, and K. Suresh, "A Review of Methods for the Geometric Post-Processing of Topology Optimized Models," *Journal of Computing and Information Science in Engineering,* vol. 20, no. 6, 2020, doi: 10.1115/1.4047429.

[13] T. Shannon, T. T. Robinson, A. Murphy, and C. G. Armstrong, "Post-processing feature-mapping topology optimisation designs towards feature-based CAD processing," *Structural and Multidisciplinary Optimization,* vol. 66, no. 11, p. 238, 2023/11/06 2023, doi: 10.1007/s00158-023-03650-5.

[14] R. Lian, S. Jing, Z. He, Z. Shi, and G. Song, "An Accelerating Convergence Rate Method for Moving Morphable Components," *Mathematical Problems in Engineering,* vol. 2020, p. 2478292, 2020/01/30 2020, doi: 10.1155/2020/2478292.

[15] S. Mukherjee *et al.*, "Accelerating Large-scale Topology Optimization: State-of-the-Art and Challenges," *Archives of Computational Methods in Engineering,* vol. 28, no. 7, pp. 4549-4571, 2021/12/01 2021, doi: 10.1007/s11831-021-09544-3.

[16] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature,* vol. 521, no. 7553, pp. 436-444, 2015.

[17] A. Vaswani *et al.*, "Attention is all you need," *Advances in neural information processing systems,* vol. 30, 2017.

[18] R. V. Woldseth, N. Aage, J. A. Bærentzen, and O. Sigmund, "On the use of Artificial Neural Networks in Topology Optimisation," p. arXiv:2208.02563. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2022arXiv220802563W

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

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

[21] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[22] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep Learning for Computer Vision: A Brief Review," *Computational Intelligence and Neuroscience,* vol. 2018, no. 1, p. 7068349, 2018, doi: https://doi.org/10.1155/2018/7068349.

[23] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.

[24] K. Samel *et al.*, "End-to-End Query Term Weighting," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023, pp. 4778-4786.

[25] D. Silver *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature,* vol. 529, no. 7587, pp. 484-489, 2016/01/01 2016, doi: 10.1038/nature16961.

[26] M. P. Bendsøe and N. Kikuchi, "Generating optimal topologies in structural design using a homogenization method," *Computer methods in applied mechanics and engineering,* vol. 71, no. 2, pp. 197-224, 1988.

[27] F. Wein, P. D. Dunning, and J. A. Norato, "A review on feature-mapping methods for structural optimization," *Structural and Multidisciplinary Optimization,* vol. 62, no. 4, pp. 1597-1638, 2020/10/01 2020, doi: 10.1007/s00158-020-02649-6.

[28] X. Guo, W. Zhang, and W. Zhong, "Doing topology optimization explicitly and geometrically—a new moving morphable components based framework," *Journal of Applied Mechanics,* vol. 81, no. 8, 2014.

[29] M. P. Bendsøe, "Optimal shape design as a material distribution problem," *Structural optimization,* vol. 1, no. 4, pp. 193-202, 1989/12/01 1989, doi: 10.1007/BF01650949.

[30] P. Wei, Z. Li, X. Li, and M. Y. Wang, "An 88-line MATLAB code for the parameterized level set method based topology optimization using radial basis functions," *Structural and*

Multidisciplinary Optimization, vol. 58, no. 2, pp. 831-849, 2018/08/01 2018, doi: 10.1007/s00158-018-1904-8.

[31] O. Sigmund, "A 99 line topology optimization code written in Matlab," *Structural and Multidisciplinary Optimization,* vol. 21, no. 2, pp. 120-127, 2001/04/01 2001, doi: 10.1007/s001580050176.

[32] F. Ferrari and O. Sigmund, "A new generation 99 line Matlab code for compliance topology optimization and its extension to 3D," *Structural and Multidisciplinary Optimization,* vol. 62, no. 4, pp. 2211-2228, 2020/10/01 2020, doi: 10.1007/s00158-020-02629-w.

[33] S. Singamneni, L. Yifan, A. Hewitt, R. Chalk, W. Thomas, and D. Jordison, "Additive manufacturing for the aircraft industry: a review," *J. Aeronaut. Aerosp. Eng,* vol. 8, no. 1, pp. 351-371, 2019.

[34] M. Mani, P. Witherell, and H. Jee, "Design rules for additive manufacturing: A categorization," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2017, vol. 58110: American Society of Mechanical Engineers, p. V001T02A035.

[35] A. T. Gaynor and J. K. Guest, "Topology optimization considering overhang constraints: Eliminating sacrificial support material in additive manufacturing through design," *Structural and Multidisciplinary Optimization,* vol. 54, no. 5, pp. 1157-1172, 2016/11/01 2016, doi: 10.1007/s00158-016-1551-x.

[36] F. Wang, B. S. Lazarov, and O. Sigmund, "On projection methods, convergence and robust formulations in topology optimization," *Structural and multidisciplinary optimization,* vol. 43, pp. 767-784, 2011.

[37] T. Dbouk, "A review about the engineering design of optimal heat transfer systems using topology optimization," *Applied Thermal Engineering,* vol. 112, pp. 841-854, 2017/02/05/ 2017, doi: https://doi.org/10.1016/j.applthermaleng.2016.10.134.

[38] J. Alexandersen and C. S. Andreasen, "A review of topology optimisation for fluid-based problems," *Fluids,* vol. 5, no. 1, p. 29, 2020.

[39] F. Lucchini, R. Torchio, V. Cirimele, P. Alotto, and P. Bettini, "Topology optimization for electromagnetics: A survey," *IEEE Access,* vol. 10, pp. 98593-98611, 2022.

[40] Z. Du, T. Cui, C. Liu, W. Zhang, Y. Guo, and X. Guo, "An efficient and easy-to-extend Matlab code of the Moving Morphable Component (MMC) method for three-dimensional topology optimization," *Structural and Multidisciplinary Optimization,* vol. 65, no. 5, p. 158, 2022/05/09 2022, doi: 10.1007/s00158-022-03239-4.

[41] J. Lim, K. Jung, Y. Jung, and D.-N. Kim, "Accelerating topology optimization using deep learning-based image super-resolution," *Engineering Applications of Artificial Intelligence,* vol. 133, p. 108370, 2024/07/01/ 2024, doi: https://doi.org/10.1016/j.engappai.2024.108370.

[42] Z. Li, H. Xu, and S. Zhang, "A Comprehensive Review of Explicit Topology Optimization Based on Moving Morphable Components (MMC) Method," *Archives of Computational Methods in Engineering,* 2024/01/29 2024, doi: 10.1007/s11831-023-10053-8.

[43] A. Amroune, J.-C. Cuillière, and V. François, "Automated Lofting-Based Reconstruction of CAD Models from 3D Topology Optimization Results," *Computer-Aided Design,* vol. 145, p. 103183, 2022/04/01/ 2022, doi: https://doi.org/10.1016/j.cad.2021.103183.

[44] J.-F. Gamache, A. Vadean, É. Noirot-Nérin, D. Beaini, and S. Achiche, "Image-based truss recognition for density-based topology optimization approach," *Structural and Multidisciplinary Optimization,* vol. 58, no. 6, pp. 2697-2709, 2018/12/01 2018, doi: 10.1007/s00158-018-2028-x.

[45] P. K. Saha, G. Borgefors, and G. Sanniti di Baja, "A survey on skeletonization algorithms and their applications," *Pattern Recognition Letters,* vol. 76, pp. 3-12, 2016/06/01/ 2016, doi: https://doi.org/10.1016/j.patrec.2015.04.006.

[46] D. Khan *et al.*, "Surface Remeshing: A Systematic Literature Review of Methods and Research Directions," *IEEE Transactions on Visualization and Computer Graphics,* vol. 28, no. 3, pp. 1680-1713, 2022, doi: 10.1109/TVCG.2020.3016645.

[47] J. Chacón, J. Bellido, and A. Donoso, "Integration of topology optimized designs into CAD/CAM via an IGES translator," *Structural and Multidisciplinary Optimization,* vol. 50, pp. 1115-1125, 2014.

[48] I. Khan, B. Ullah, I. Siraj ul, Z. Ullah, and W. Khan, "An adaptive B-spline representation of topology optimization design for Additive Manufacturing," *Advances in Engineering Software,* vol. 186, p. 103552, 2023/12/01/ 2023, doi: https://doi.org/10.1016/j.advengsoft.2023.103552.

[49] M. Marinov *et al.*, "Generative Design Conversion to Editable and Watertight Boundary Representation," *Computer-Aided Design,* vol. 115, pp. 194-205, 2019/10/01/ 2019, doi: https://doi.org/10.1016/j.cad.2019.05.016.

[50] M. Denk, K. Rother, and K. Paetzold, "Subdivision Surface mid-surface reconstruction of topology optimization results and thin-walled shapes using surface skeletons," *Proceedings of the Design Society,* vol. 1, pp. 2771-2780, 2021.

[51] G. Yi and N. H. Kim, "Identifying boundaries of topology optimization results using basic parametric features," *Structural and Multidisciplinary Optimization,* vol. 55, no. 5, pp. 1641-1654, 2017/05/01 2017, doi: 10.1007/s00158-016-1597-9.

[52] J.-C. Cuillière, V. Francois, and J.-M. Drouet, "Towards the Integration of Topology Optimization into the CAD Process," *Computer-Aided Design and Applications,* vol. 11, no. 2, pp. 120-140, 2014/03/04 2014, doi: 10.1080/16864360.2014.846067.

[53] S. Larsen and C. G. Jensen, "Converting Topology Optimization Results into Parametric CAD Models," *Computer-Aided Design and Applications,* vol. 6, no. 3, pp. 407-418, 2009/01/01 2009, doi: 10.3722/cadaps.2009.407-418.

[54] I. Sosnovik and I. Oseledets, "Neural networks for topology optimization," *Russian Journal of Numerical Analysis and Mathematical Modelling,* vol. 34, no. 4, pp. 215-223, 2019, doi: doi:10.1515/rnam-2019-0018.

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

[56] N. Napier, S.-A. Sriraman, H. T. Tran, and K. A. James, "An Artificial Neural Network Approach for Generating High-Resolution Designs From Low-Resolution Input in Topology Optimization," *Journal of Mechanical Design,* vol. 142, no. 1, 2019, doi: 10.1115/1.4044332.

[57] N. A. Kallioras, G. Kazakis, and N. D. Lagaros, "Accelerated topology optimization by means of deep learning," *Structural and Multidisciplinary Optimization,* vol. 62, no. 3, pp. 1185-1212, 2020/09/01 2020, doi: 10.1007/s00158-020-02545-z.

[58] C. Wang, S. Yao, Z. Wang, and J. Hu, "Deep super-resolution neural network for structural topology optimization," *Engineering Optimization,* vol. 53, no. 12, pp. 2108-2121, 2021/12/02 2021, doi: 10.1080/0305215X.2020.1846031.

[59] E. Andreassen, A. Clausen, M. Schevenels, B. S. Lazarov, and O. Sigmund, "Efficient topology optimization in MATLAB using 88 lines of code," *Structural and Multidisciplinary Optimization,* vol. 43, no. 1, pp. 1-16, 2011/01/01 2011, doi: 10.1007/s00158-010-0594-7.

[60] Y. Yu, T. Hur, J. Jung, and I. G. Jang, "Deep learning for determining a near-optimal topological design without any iteration," *Structural and Multidisciplinary Optimization,* vol. 59, no. 3, pp. 787-799, 2019/03/01 2019, doi: 10.1007/s00158-018-2101-5.

[61] S. Zheng, Z. He, and H. Liu, "Generating three-dimensional structural topologies via a U-Net convolutional neural network," *Thin-Walled Structures,* vol. 159, p. 107263, 2021/02/01/ 2021, doi: https://doi.org/10.1016/j.tws.2020.107263.

[62] C. Xiang, D. Wang, Y. Pan, A. Chen, X. Zhou, and Y. Zhang, "Accelerated topology optimization design of 3D structures based on deep learning," *Structural and Multidisciplinary Optimization,* vol. 65, no. 3, p. 99, 2022/02/28 2022, doi: 10.1007/s00158-022-03194-0.

[63] Y. Zhang, B. Peng, X. Zhou, C. Xiang, and D. Wang, "A deep convolutional neural network for topology optimization with strong generalization ability," *arXiv preprint arXiv:1901.07761,* 2019.

[64] J. Seo and R. K. Kapania, "Topology optimization with advanced CNN using mapped physics-based data," *Structural and Multidisciplinary Optimization,* vol. 66, no. 1, p. 21, 2023/01/06 2023, doi: 10.1007/s00158-022-03461-0.

[65] J. Luo, Y. Li, W. Zhou, Z. Gong, Z. Zhang, and W. Yao, "An improved data-driven topology optimization method using feature pyramid networks with physical constraints," *CMES-Computer Modeling in Engineering & Sciences,* vol. 128, no. 3, pp. 823-848, 2021.

[66] W. Zhang, L. Su, and X. Wang, "Real-time topology optimization based on multi-scale convolutional attention mechanism," *Engineering Optimization,* pp. 1-20, 2024.

[67] H. Chi *et al.*, "Universal machine learning for topology optimization," *Computer Methods in Applied Mechanics and Engineering,* vol. 375, p. 112739, 2021/03/01/ 2021, doi: https://doi.org/10.1016/j.cma.2019.112739.

[68] G. C. Ates and R. M. Gorguluarslan, "Two-stage convolutional encoder-decoder network to improve the performance and reliability of deep learning models for topology

optimization," *Structural and Multidisciplinary Optimization,* vol. 63, no. 4, pp. 1927-1950, 2021/04/01 2021, doi: 10.1007/s00158-020-02788-w.

[69] I. Goodfellow *et al.*, "Generative adversarial nets," *Advances in neural information processing systems,* vol. 27, 2014.

[70] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems,* vol. 33, pp. 6840-6851, 2020.

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

[72] S. Herath and U. Haputhanthri, "Topologically optimal design and failure prediction using conditional generative adversarial networks," *International Journal for Numerical Methods in Engineering,* vol. 122, no. 23, pp. 6867-6887, 2021, doi: https://doi.org/10.1002/nme.6814.

[73] Z. Nie, T. Lin, H. Jiang, and L. B. Kara, "TopologyGAN: Topology Optimization Using Generative Adversarial Networks Based on Physical Fields Over the Initial Domain," *Journal of Mechanical Design,* vol. 143, no. 3, 2021, doi: 10.1115/1.4049533.

[74] F. Mazé and F. Ahmed, "Diffusion Models Beat GANs on Topology Optimization," *Proceedings of the AAAI Conference on Artificial Intelligence,* vol. 37, no. 8, pp. 9108-9116, 06/26 2023, doi: 10.1609/aaai.v37i8.26093.

[75] M. M. Behzadi and H. T. Ilieş, "GANTL: Toward Practical and Real-Time Topology Optimization With Conditional Generative Adversarial Networks and Transfer Learning," *Journal of Mechanical Design,* vol. 144, no. 2, 2021, doi: 10.1115/1.4052757.

[76] X. Lei, C. Liu, Z. Du, W. Zhang, and X. Guo, "Machine Learning-Driven Real-Time Topology Optimization Under Moving Morphable Component-Based Framework," *Journal of Applied Mechanics,* vol. 86, no. 1, 2018, doi: 10.1115/1.4041319.

[77] S. Zheng, H. Fan, Z. Zhang, Z. Tian, and K. Jia, "Accurate and real-time structural topology prediction driven by deep learning under moving morphable component-based framework," *Applied Mathematical Modelling,* vol. 97, pp. 522-535, 2021/09/01/ 2021, doi: https://doi.org/10.1016/j.apm.2021.04.009.

[78] L. Wang, D. Shi, B. Zhang, G. Li, and P. Liu, "Real-time topology optimization based on deep learning for moving morphable components," *Automation in Construction,* vol. 142, p. 104492, 2022/10/01/ 2022, doi: https://doi.org/10.1016/j.autcon.2022.104492.

[79] D. Geng *et al.*, "Real-Time structure topology optimization using CNN driven Moving Morphable component method," *Engineering Structures,* vol. 290, p. 116376, 2023/09/01/ 2023, doi: https://doi.org/10.1016/j.engstruct.2023.116376.

[80] Z. Dua *et al.*, "Real-time generative design of diverse," truly" optimized structures with controllable structural complexities," *arXiv preprint arXiv:2401.08981,* 2024.

[81] D. Bielecki, D. Patel, R. Rai, and G. F. Dargush, "Multi-stage deep neural network accelerated topology optimization," *Structural and Multidisciplinary Optimization,* vol. 64, no. 6, pp. 3473-3487, 2021/12/01 2021, doi: 10.1007/s00158-021-03028-5.

[82] A. Chandrasekhar and K. Suresh, "TOuNN: Topology Optimization using Neural Networks," *Structural and Multidisciplinary Optimization,* vol. 63, no. 3, pp. 1135-1149, 2021/03/01 2021, doi: 10.1007/s00158-020-02748-4.

[83] A. Chandrasekhar, S. Sridhara, and K. Suresh, "AuTO: a framework for Automatic differentiation in Topology Optimization," *Structural and Multidisciplinary Optimization,* vol. 64, no. 6, pp. 4355-4365, 2021, doi: 10.1007/s00158-021-03025-8.

[84] G. A. Hazelrigg, "A Framework for Decision-Based Engineering Design," *Journal of Mechanical Design,* vol. 120, no. 4, pp. 653-658, 1998, doi: 10.1115/1.2829328.

[85] R. S. Sutton and A. G. Barto, *Reinforcement Learning, second edition: An Introduction*. MIT Press, 2018.

[86] D. Silver *et al.*, "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play," *Science,* vol. 362, no. 6419, pp. 1140-1144, 2018.

[87] O. Vinyals *et al.*, "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature,* vol. 575, no. 7782, pp. 350-354, 2019/11/01 2019, doi: 10.1038/s41586-019-1724-z.

[88] C. Berner *et al.*, "Dota 2 with large scale deep reinforcement learning," *arXiv preprint arXiv:1912.06680,* 2019.

[89] V. Mnih *et al.*, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602,* 2013.

[90] L. Kocsis and C. Szepesvári, "Bandit Based Monte-Carlo Planning," Berlin, Heidelberg, 2006: Springer Berlin Heidelberg, in Machine Learning: ECML 2006, pp. 282-293.

[91] D. Ha and J. Schmidhuber, "World models," *arXiv preprint arXiv:1803.10122,* 2018.

[92] I. Clavera, J. Rothfuss, J. Schulman, Y. Fujita, T. Asfour, and P. Abbeel, "Model-based reinforcement learning via meta-policy optimization," in *Conference on Robot Learning*, 2018: PMLR, pp. 617-629.

[93] M. P. Deisenroth and C. E. Rasmussen, "PILCO: a model-based and data-efficient approach to policy search," presented at the Proceedings of the 28th International Conference on International Conference on Machine Learning, Bellevue, Washington, USA, 2011.

[94] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347,* 2017.

[95] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap, "Mastering diverse domains through world models," *arXiv preprint arXiv:2301.04104,* 2023.

[96] A. Mirhoseini *et al.*, "A graph placement methodology for fast chip design," *Nature,* vol. 594, no. 7862, pp. 207-212, 2021/06/01 2021, doi: 10.1038/s41586-021-03544-w.

[97] P. Garnier, J. Viquerat, J. Rabault, A. Larcher, A. Kuhnle, and E. Hachem, "A review on deep reinforcement learning for fluid mechanics," *Computers & Fluids,* vol. 225, p. 104973, 2021/07/15/ 2021, doi: https://doi.org/10.1016/j.compfluid.2021.104973.

[98] J. Viquerat, J. Rabault, A. Kuhnle, H. Ghraieb, A. Larcher, and E. Hachem, "Direct shape optimization through deep reinforcement learning," *Journal of Computational Physics,* vol. 428, p. 110080, 2021/03/01/ 2021, doi: https://doi.org/10.1016/j.jcp.2020.110080.

[99] F. Dworschak, S. Dietze, M. Wittmann, B. Schleich, and S. Wartzack, "Reinforcement Learning for Engineering Design Automation," *Advanced Engineering Informatics,* vol. 52, p. 101612, 2022/04/01/ 2022, doi: https://doi.org/10.1016/j.aei.2022.101612.

[100] Y.-y. Wang, B.-q. Zhang, and Y.-c. Chen, "Robust airfoil optimization based on improved particle swarm optimization method," *Applied Mathematics and Mechanics,* vol. 32, pp. 1245-1254, 2011.

[101] K. Hayashi and M. Ohsaki, "Reinforcement Learning and Graph Embedding for Binary Truss Topology Optimization Under Stress and Displacement Constraints," (in English), *Frontiers in Built Environment,* Original Research vol. 6, 2020-April-30 2020, doi: 10.3389/fbuil.2020.00059.

[102] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI conference on artificial intelligence*, 2016, vol. 30, no. 1.

[103] N. K. Brown, A. P. Garland, G. M. Fadel, and G. Li, "Deep reinforcement learning for engineering design through topology optimization of elementally discretized design domains," *Materials & Design,* vol. 218, p. 110672, 2022.

[104] N. K. Brown, A. P. Garland, G. M. Fadel, and G. Li, "Deep reinforcement learning for the rapid on-demand design of mechanical metamaterials with targeted nonlinear deformation responses," *Engineering Applications of Artificial Intelligence,* vol. 126, p. 106998, 2023/11/01/ 2023, doi: https://doi.org/10.1016/j.engappai.2023.106998.

[105] S. Jang, S. Yoo, and N. Kang, "Generative Design by Reinforcement Learning: Enhancing the Diversity of Topology Optimization Designs," *Computer-Aided Design,* vol. 146, p. 103225, 2022/05/01/ 2022, doi: https://doi.org/10.1016/j.cad.2022.103225.

[106] N. Hertlein, P. R. Buskohl, A. Gillman, K. Vemaganti, and S. Anand, "Generative adversarial network for early-stage design flexibility in topology optimization for additive manufacturing," *Journal of Manufacturing Systems,* vol. 59, pp. 675-685, 2021/04/01/ 2021, doi: https://doi.org/10.1016/j.jmsy.2021.04.007.

[107] M. Langelaar, "An additive manufacturing filter for topology optimization of print-ready designs," *Structural and Multidisciplinary Optimization,* vol. 55, no. 3, pp. 871-883, 2017/03/01 2017, doi: 10.1007/s00158-016-1522-2.

[108] G. Sharma, R. Goyal, D. Liu, E. Kalogerakis, and S. Maji, "Csgnet: Neural shape parser for constructive solid geometry," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5515-5523.

[109] K. Kania, M. Zieba, and T. Kajdanowicz, "UCSG-NET-unsupervised discovering of constructive solid geometry tree," *Advances in neural information processing systems,* vol. 33, pp. 8776-8786, 2020.

[110] Y. Liu, A. Obukhov, J. D. Wegner, and K. Schindler, "Point2CAD: Reverse Engineering CAD Models from 3D Point Clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 3763-3772.

[111] R. Sutton, "The bitter lesson," *Incomplete Ideas (blog),* vol. 13, no. 1, p. 38, 2019.

[112] W. Zhang, J. Yuan, J. Zhang, and X. Guo, "A new topology optimization approach based on Moving Morphable Components (MMC) and the ersatz material model," *Structural and Multidisciplinary Optimization,* vol. 53, no. 6, pp. 1243-1260, 2016/06/01 2016, doi: 10.1007/s00158-015-1372-3.

[113] O. Sigmund and K. Maute, "Topology optimization approaches," *Structural and Multidisciplinary Optimization,* vol. 48, no. 6, pp. 1031-1055, 2013/12/01 2013, doi: 10.1007/s00158-013-0978-6.

[114] J.-F. Gamache, "On Topology Optimization in the Generative Design Framework : use on Aircraft Structure design," Ph.D. thesis, Department of Mechanical Engineering, Polytechnique Montréal, 2021. [Online]. Available: https://publications.polymtl.ca/9133/

[115] K. Svanberg, "MMA and GCMMA: two methods for nonlinear optimization," 2014.

[116] N. Panagant, N. Pholdee, S. Bureerat, K. Kaen, A. R. Yıldız, and S. M. Sait, "Seagull optimization algorithm for solving real-world design optimization problems," *Materials Testing,* vol. 62, no. 6, pp. 640-644, 2020, doi: doi:10.3139/120.111529.

[117] B. S. Yildiz, "Robust design of electric vehicle components using a new hybrid salp swarm algorithm and radial basis function-based approach," *International Journal of Vehicle Design,* 2020.

[118] C. Leblond-Menard and S. Achiche, "Non-Intrusive Real Time Eye Tracking Using Facial Alignment for Assistive Technologies," *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* vol. 31, pp. 954-961, 2023, doi: 10.1109/TNSRE.2023.3236886.

[119] W. Muhammad Ashraf *et al.*, "Optimization of a 660 MWe Supercritical Power Plant Performance—A Case of Industry 4.0 in the Data-Driven Operational Management Part 1. Thermal Efficiency," *Energies*, vol. 13, no. 21*,* doi: 10.3390/en13215592.

[120] N. Yüksel, H. R. Börklü, H. K. Sezer, and O. E. Canyurt, "Review of artificial intelligence applications in engineering design perspective," *Engineering Applications of Artificial Intelligence,* vol. 118, p. 105697, 2023/02/01/ 2023, doi: https://doi.org/10.1016/j.engappai.2022.105697.

[121] J. Wu, O. Sigmund, and J. P. Groen, "Topology optimization of multi-scale structures: a review," *Structural and Multidisciplinary Optimization,* vol. 63, no. 3, pp. 1455-1480, 2021/03/01 2021, doi: 10.1007/s00158-021-02881-8.

[122] J. D. Deaton and R. V. Grandhi, "A survey of structural and multidisciplinary continuum topology optimization: post 2000," *Structural and Multidisciplinary Optimization,* vol. 49, no. 1, pp. 1-38, 2014/01/01 2014, doi: 10.1007/s00158-013-0956-z.

[123] J. Zhu, H. Zhou, C. Wang, L. Zhou, S. Yuan, and W. Zhang, "A review of topology optimization for additive manufacturing: Status and challenges," *Chinese Journal of Aeronautics,* vol. 34, no. 1, pp. 91-110, 2021/01/01/ 2021, doi: https://doi.org/10.1016/j.cja.2020.09.020.

[124] S. B. Dilgen, C. B. Dilgen, D. R. Fuhrman, O. Sigmund, and B. S. Lazarov, "Density based topology optimization of turbulent flow heat transfer systems," *Structural and*

Multidisciplinary Optimization, vol. 57, no. 5, pp. 1905-1918, 2018/05/01 2018, doi: 10.1007/s00158-018-1967-6.

[125] J. Rade *et al.*, "Algorithmically-consistent deep learning frameworks for structural topology optimization," *Engineering Applications of Artificial Intelligence,* vol. 106, p. 104483, 2021/11/01/ 2021, doi: https://doi.org/10.1016/j.engappai.2021.104483.

[126] V.-N. Hoang, N.-L. Nguyen, D. Q. Tran, Q.-V. Vu, and H. Nguyen-Xuan, "Data-driven geometry-based topology optimization," *Structural and Multidisciplinary Optimization,* vol. 65, no. 2, p. 69, 2022/01/31 2022, doi: 10.1007/s00158-022-03170-8.

[127] V.-N. Hoang and G.-W. Jang, "Topology optimization using moving morphable bars for versatile thickness control," *Computer Methods in Applied Mechanics and Engineering,* vol. 317, pp. 153-173, 2017/04/15/ 2017, doi: https://doi.org/10.1016/j.cma.2016.12.004.

[128] T. Rochefort-Beaudoin, A. Vadean, J.-F. Gamache, and S. Achiche, "Comparative Study of First-Order Moving Asymptotes Optimizers for the Moving Morphable Components Topology Optimization Framework," in *ASME 2022 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2022, vol. Volume 2: 42nd Computers and Information in Engineering Conference (CIE), V002T02A028, doi: 10.1115/detc2022-88722. [Online]. Available: https://doi.org/10.1115/DETC2022-88722

[129] W. Karush, "Minima of functions of several variables with inequalities as side constraints," *M. Sc. Dissertation. Dept. of Mathematics, Univ. of Chicago,* 1939.

[130] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," in *Traces and emergence of nonlinear programming*: Springer, 2014, pp. 247-258.

[131] S. Wang, Y. Teng, and P. Perdikaris, "Understanding and mitigating gradient pathologies in physics-informed neural networks," p. arXiv:2001.04536. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2020arXiv200104536W

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

[133] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," *Advances in neural information processing systems,* vol. 24, 2011.

[134] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A Next-generation Hyperparameter Optimization Framework," p. arXiv:1907.10902. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2019arXiv190710902A

[135] E. D. Dolan and J. J. Moré, "Benchmarking optimization software with performance profiles," *Mathematical Programming,* vol. 91, no. 2, pp. 201-213, 2002/01/01 2002, doi: 10.1007/s101070100263.

[136] B. Wang and C. Shi, "Shape Matching Using Chord-Length Function," in *Intelligent Data Engineering and Automated Learning – IDEAL 2006*, Berlin, Heidelberg, E. Corchado, H. Yin, V. Botti, and C. Fyfe, Eds., 2006// 2006: Springer Berlin Heidelberg, pp. 746-753.

[137] M. P. Bendsøe and O. Sigmund, "Material interpolation schemes in topology optimization," *Archive of Applied Mechanics,* vol. 69, no. 9, pp. 635-654, 1999/11/01 1999, doi: 10.1007/s004190050248.

[138] L. Meng *et al.*, "From Topology Optimization Design to Additive Manufacturing: Today's Success and Tomorrow's Roadmap," *Archives of Computational Methods in Engineering,* vol. 27, no. 3, pp. 805-830, 2020/07/01 2020, doi: 10.1007/s11831-019-09331-1.

[139] J.-F. Gamache, A. Vadean, M. Capo, T. Rochefort-Beaudoin, N. Dodane, and S. Achiche, "Complexity-driven layout exploration for aircraft structures," *Design Science,* vol. 9, p. e13, 2023, Art no. e13, doi: 10.1017/dsj.2023.12.

[140] J. Liu *et al.*, "Current and future trends in topology optimization for additive manufacturing," *Structural and Multidisciplinary Optimization,* vol. 57, no. 6, pp. 2457-2483, 2018/06/01 2018, doi: 10.1007/s00158-018-1994-3.

[141] L. Xia and P. Breitkopf, "Recent Advances on Topology Optimization of Multiscale Nonlinear Structures," *Archives of Computational Methods in Engineering,* vol. 24, no. 2, pp. 227-249, 2017/04/01 2017, doi: 10.1007/s11831-016-9170-7.

[142] T. Rochefort-Beaudoin, A. Vadean, J.-F. Gamache, and S. Achiche, "Supervised deep learning for the moving morphable components topology optimization framework," *Engineering Applications of Artificial Intelligence,* vol. 123, p. 106436, 2023/08/01/ 2023, doi: https://doi.org/10.1016/j.engappai.2023.106436.

[143] S. Hoyer, J. Sohl-Dickstein, and S. Greydanus, "Neural reparameterization improves structural optimization," *arXiv preprint arXiv:1909.04240,* 2019.

[144] M. Nobel-Jørgensen, D. Malmgren-Hansen, J. A. Bærentzen, O. Sigmund, and N. Aage, "Improving topology optimization intuition through games," *Structural and Multidisciplinary Optimization,* vol. 54, no. 4, pp. 775-781, 2016/10/01 2016, doi: 10.1007/s00158-016-1443-0.

[145] G. Brockman *et al.*, "Openai gym," *arXiv preprint arXiv:1606.01540,* 2016.

[146] L. Ouyang *et al.*, "Training language models to follow instructions with human feedback," *Advances in Neural Information Processing Systems,* vol. 35, pp. 27730-27744, 2022.

[147] L. Espeholt *et al.*, "Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures," in *International conference on machine learning*, 2018: PMLR, pp. 1407-1416.

[148] K. Cobbe, C. Hesse, J. Hilton, and J. Schulman, "Leveraging procedural generation to benchmark reinforcement learning," in *International conference on machine learning*, 2020: PMLR, pp. 2048-2056.

[149] M. Andrychowicz *et al.*, "What matters in on-policy reinforcement learning? a large-scale empirical study," *arXiv preprint arXiv:2006.05990,* 2020.

[150] V. Keshavarzzadeh, M. Alirezaei, T. Tasdizen, and R. M. Kirby, "Image-Based Multiresolution Topology Optimization Using Deep Disjunctive Normal Shape Model," *Computer-Aided Design,* vol. 130, p. 102947, 2021/01/01/ 2021, doi: https://doi.org/10.1016/j.cad.2020.102947.

[151] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," in *Proceedings of the AAAI conference on artificial intelligence*, 2018, vol. 32, no. 1.

[152] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 1.

[153] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Advances in neural information processing systems,* vol. 29, 2016.

[154] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," *Advances in neural information processing systems,* vol. 30, 2017.

[155] F. Buonamici, M. Carfagni, R. Furferi, L. Governi, A. Lapini, and Y. Volpe, "Reverse engineering modeling methods and tools: a survey," *Computer-Aided Design and Applications,* vol. 15, no. 3, pp. 443-464, 2018.

[156] T. Du *et al.*, "InverseCSG: automatic conversion of 3D models to CSG trees," *ACM Trans. Graph.,* vol. 37, no. 6, p. Article 213, 2018, doi: 10.1145/3272127.3275006.

[157] Y. Li, X. Wu, Y. Chrysathou, A. Sharf, D. Cohen-Or, and N. J. Mitra, "Globfit: Consistently fitting primitives by discovering global relations," in *ACM SIGGRAPH 2011 papers*, 2011, pp. 1-12.

[158] J. Mayer and S. Wartzack, "Computational Geometry Reconstruction from 3D Topology Optimization Results: A New Parametric Approach by the Medial Axis," *Comput. Des. Appl,* vol. 20, pp. 960-975, 2023.

[159] J. Polak and M. Nowak, "From Structural Optimization Results to Parametric CAD Modeling—Automated, Skeletonization-Based Truss Recognition," *Applied Sciences,* vol. 13, no. 9, p. 5670, 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/9/5670.

[160] J.-C. Cuillière, V. François, and A. Nana, "Automatic construction of structural CAD models from 3D topology optimization," *Computer-Aided Design and Applications,* vol. 15, no. 1, pp. 107-121, 2018.

[161] J. Mayer, H. Völkl, and S. Wartzack, "Feature-Based Reconstruction of Non-Beam-Like Topology Optimization Design Proposals in Boundary-Representation," in *DS 119: Proceedings of the 33rd Symposium Design for X (DFX2022)*, 2022, pp. 1-10.

[162] R. Ling, G. Xu, X. Zhang, J. Xu, and X. Guo, "Converting pixel-type topology optimization results to MMC-representation based on sparse optimization and its applications," *International Journal for Numerical Methods in Engineering,* vol. 125, no. 8, p. e7437, 2024, doi: https://doi.org/10.1002/nme.7437.

[163] G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by Ultralytics," *URL: https://github. com/ultralytics/ultralytics,* 2023.

[164] J. Wu, N. Aage, R. Westermann, and O. Sigmund, "Infill Optimization for Additive Manufacturing—Approaching Bone-Like Porous Structures," *IEEE Transactions on Visualization and Computer Graphics,* vol. 24, no. 2, pp. 1127-1140, 2018, doi: 10.1109/TVCG.2017.2655523.

[165] M. Y. Wang, X. Wang, and D. Guo, "A level set method for structural topology optimization," *Computer methods in applied mechanics and engineering,* vol. 192, no. 1-2, pp. 227-246, 2003.

[166] O. Sigmund, "On benchmarking and good scientific practise in topology optimization," *Structural and Multidisciplinary Optimization,* vol. 65, no. 11, p. 315, 2022/10/25 2022, doi: 10.1007/s00158-022-03427-2.

[167] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779-788.

[168] C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, "CSPNet: A new backbone that can enhance learning capability of CNN," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020, pp. 390-391.

[169] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and Flexible Image Augmentations," *Information,* vol. 11, no. 2, p. 125, 2020. [Online]. Available: https://www.mdpi.com/2078-2489/11/2/125.

[170] X. Guo, W. Zhang, J. Zhang, and J. Yuan, "Explicit structural topology optimization based on moving morphable components (MMC) with curved skeletons," *Computer Methods in Applied Mechanics and Engineering,* vol. 310, pp. 711-748, 2016/10/01/ 2016, doi: https://doi.org/10.1016/j.cma.2016.07.018.

[171] J. A. Norato, B. K. Bell, and D. A. Tortorelli, "A geometry projection method for continuum-based topology optimization with discrete elements," *Computer Methods in Applied Mechanics and Engineering,* vol. 293, pp. 306-327, 2015/08/15/ 2015, doi: https://doi.org/10.1016/j.cma.2015.05.005.

APPENDIX A SUPPORTING MATERIAL FOR CHAPTER 4

Table A.1: Parameters used for the hybrid method of moving asymptotes optimizer.

Solver	Parameter	Description	Value
MMA / GCMMA	epsimin	Termination value of ε	10^{-10}
	raa0	Coefficient in the update of $p_{ij}^{(k,v)}$ and $q_{ij}^{(k,v)}$	0.01
	albefa	Coefficient for the update of the design variables bounds α_j^k and β_j^k.	0.4
	asyinit	Initial step size of lower and upper asymptote update	0.05
	asyincr	Increasing lower and upper asymptote step size	0.8
	asydecr	Decreasing lower and upper asymptote step size.	0.6
	c	Column vector with the constants c_i in the terms $c_i y_i$	1000
	d	Column vector with the constants d_i in the terms $\frac{1}{2} d_i y_i^2$	1
	a_0	Constant in the term $a_0 z$	1
	a	Column vector with the constants a_i in the terms $a_i z$	0
GCMMA	maxinnerit	Maximum number of inner iterations	2
	move	Coefficient for the update of the design variables bounds α_j^k and β_j^k.	1.0
Hybrid	switch	Criterion to switch between MMA and GCMMA	$2e^{-5}$

Table A.2**:** Minimum and maximum values for the design variables defining each component.

Design variable	Definition	Minimum	Maximum
x_0	x-coordinate of the components center	0	2
y_0	y-coordinate of the components center	0	1
L	Half-length of the component	0	1.12
θ	The orientation of the component in	$-\pi$	π
t_1	Half thickness of endpoint A	0	0.3
t_2	Half thickness of endpoint B	0	0.3

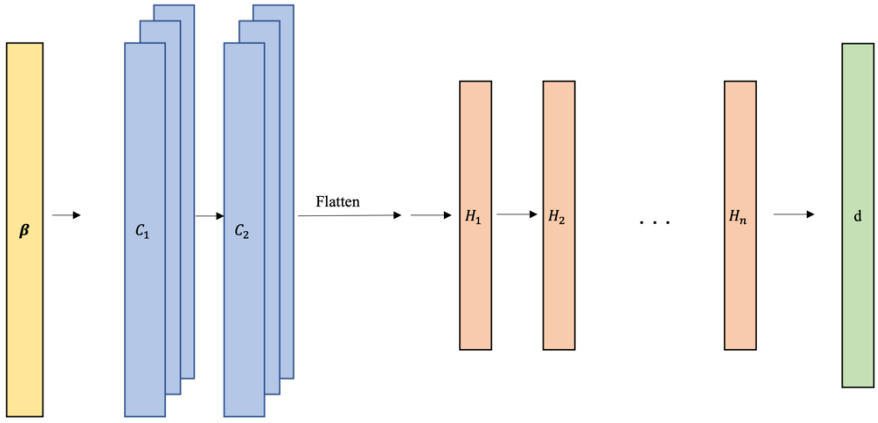


Figure A.1: Architecture diagram of the fully connected neural network. In yellow (β): the problem description vector. In blue (C_i): 1D convolutional layers. In orange (H_i): fully connected hidden layers. In green (d) the output vector of design variables.

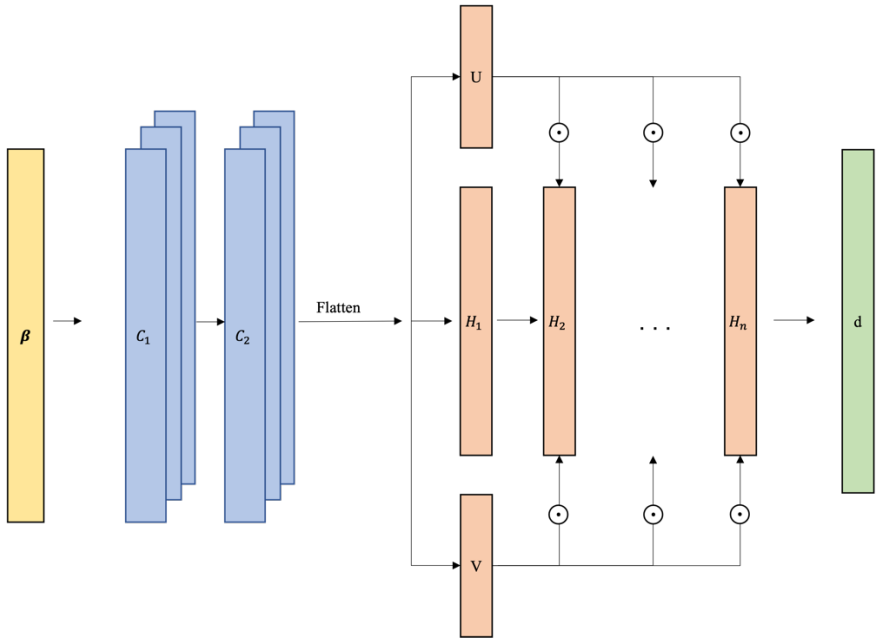


Figure A.2: Architecture diagram of the PINN-inspired neural network architecture [131]. In yellow (β): the problem description vector. In blue (C_i): 1D convolutional layers. In orange (H_i): fully connected hidden layers. In green (d) the output vector of design variables. U and V are fully connected input layers with residual dot-product connections to each hidden layer.

Table A.3: Details and values of the hyperparameters tested.

Hyperparameter	Description	Range tested
conv1d	Whether to use two 1-D	True / False
#filters	Number of convolutional kernels	[16,32,64,128,256,512]
#layers	Number of fully connected layers	[1 to 20]
activation	Type of activation functions to	[relu, sigmoid, tanh, elu, selu,
drop_rate	Dropout rate to use between	5-90% with 5% increments
hidden_units	Number of units to use in the fully	[32, 64, 128, 256, 512, 1024, 2048]
batch_size	Batch size for the ADAM	[32, 64, 128, 256, 512, 1024, 2048]

Table A.4: Results from the hyperparameter optimization

	Best value	
Hyperparameter	**Conventional MLP**	**PINN-inspired**
conv1d	True	True
#filters	32	32
#layers	14	5
activation	‘elu’	‘sigmoid’
drop_rate	5%	55%
hidden_units	64	2048
batch_size	128	64

Table A.5: Regression error metrics from hyperparameter optimization.

Regression metrics	**Fully connected**	**PINN-inspired**
Train RMSE	0.0876	0.0761
Train MSE	0.0077	0.0058
Train MAE	0.0543	0.0455
Validation RMSE	0.0859	0.0833
Validation MSE	0.0074	0.0070
Validation MAE	0.0516	0.0465
Test RMSE	0.0861	0.0834
Test MSE	0.0075	0.0070
Test MAE	0.0518	0.0469

APPENDIX B SUPPORTING MATERIAL FOR CHAPTER 5

Conventional optimization was run using a hybrid MMA-GCMMA optimizer which switched from MMA to GCMMA after detecting oscillations in the objective function according to a switch criterion. A maximum of 1000 outer iterations were allowed during the iterative process. A low resolution of 50 elements per dimension unit was used to fit the resolution of SOgym. Table B.1 describes the parameters defining the MMA and GCMMA optimizer.

Table B.1: Parameters used for the hybrid MMA-GCMMA optimizer.

Solver	Parameter	Description	Value
MMA / GCMMA	*epsimin*	Termination value of ε	10^{-10}
	raa0	Coefficient in the update of $p_{ij}^{(k,v)}$ and $q_{ij}^{(k,v)}$	0.01
	albefa	Coefficient for the update of the design variables bounds α_j^k and β_j^k.	0.4
	asyinit	Initial step size of lower and upper asymptote update.	0.05
	asyincr	Increasing lower and upper asymptote step size.	0.8
	asydecr	Decreasing lower and upper asymptote step size.	0.6
	c	Column vector with the constants c_i in the terms $c_i y_i$	1000
	d	Column vector with the constants d_i in the terms $\frac{1}{2} d_i y_i^2$	1
	a_0	Constant in the term $a_0 z$	1
	a	Column vector with the constants a_i in the terms $a_i z$	0
GCMMA	*maxinnerit*	Maximum number of inner iterations for the GCMMA optimizer	2
	move	Coefficient for the update of the design variables bounds α_j^k and β_j^k.	1.0
Hybrid	*switch*	Criterion to switch between MMA and GCMMA in the	$2e^{-5}$

APPENDIX C SUPPORTING MATERIAL FOR CHAPTER 6

C.1 “Drawings” to geometry test of YOLOv8-TO

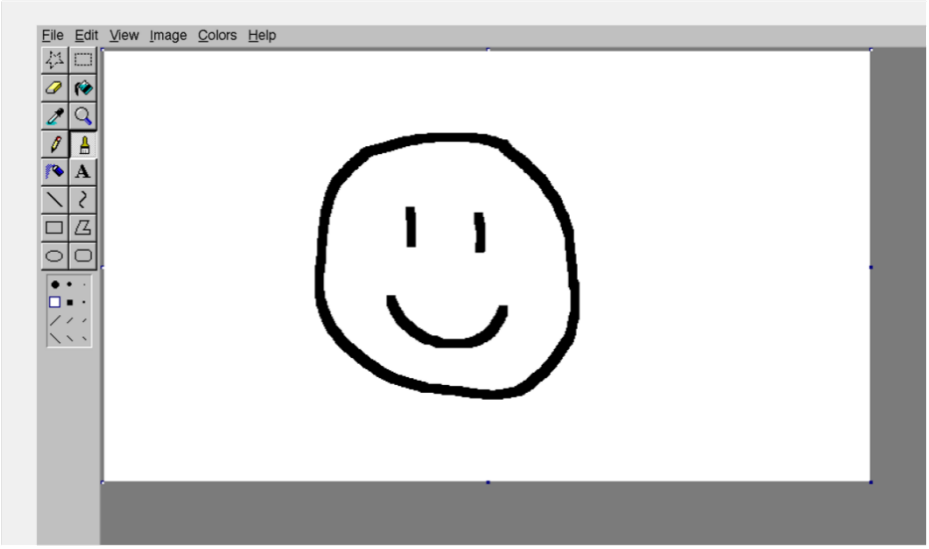


Figure C.1: Smiley face computer doodle made via a laptop trackpad.

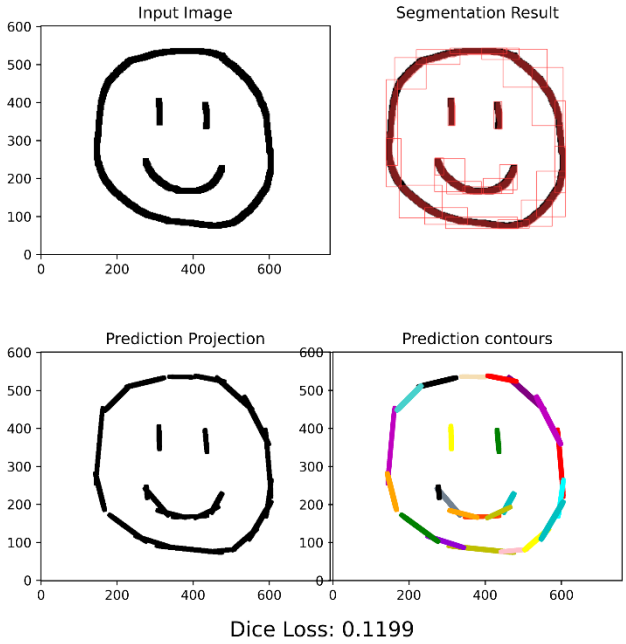


Figure C.2: YOLOv8-TO reconstruction of the smiley face doodle.

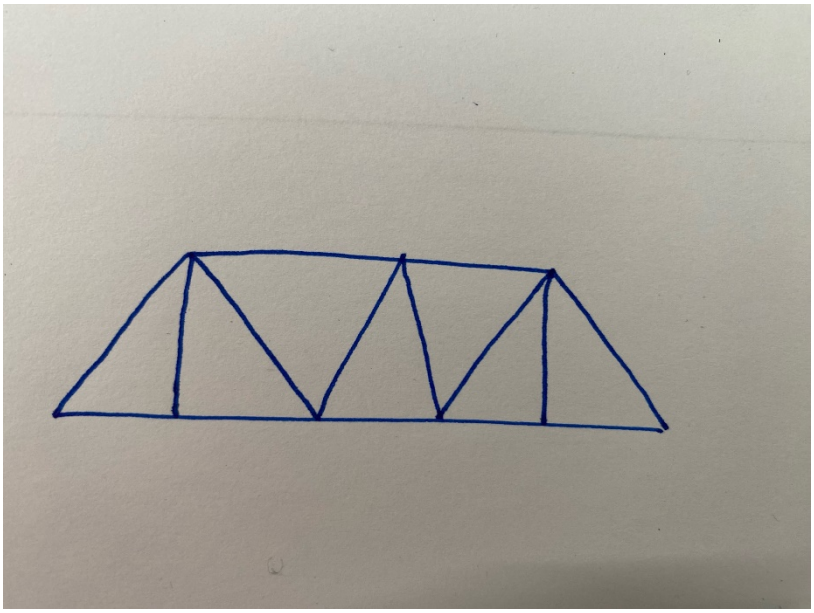


Figure C.3: Cellphone image of a hand-drawn doodle of a bridge.

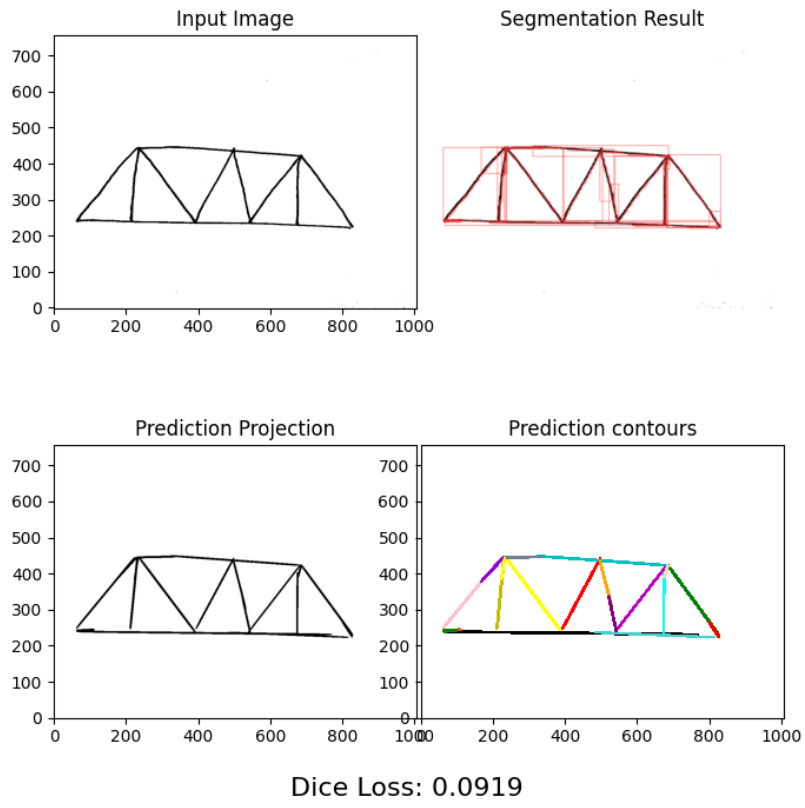


Figure C.4: YOLOv8-TO reconstruction of the hand-drawn bridge image.

C.2 YOLOv8-TO training hyperparameters

The final training of all the models was done on a single compute node of the Narval compute cluster of Digital Research Alliance of Canada (DRAC), using 48 CPU cores and 4 Nvidia A100 for data parallelism distributed YOLOv8 training. Preliminary training was done on a single Nvidia 3080 GPU and 32 cores CPU on a single compute server. A cosine learning rate scheduler was used with a final learning rate factor of 1% of the initial learning rate. Table C.1 details the hyperparameters used to configure the training phase of YOLOv8 that are different from their default values. Table C.2 lists the minimum and maximum bounds used to normalize the design variables between 0 and 1. Table C.3 lists the parameters of the data augmentation used during training.

Table C.1: YOLOv8 Hyperparameters

Hyperparameter name	Value
Per device batch size	16
Maximum number of epochs	1000
Automatic Mixed Precision (AMP)	True
Training image size	640
Initial learning rate	3e-4
Optimizer	AdamW
L2 regularization	0.01
Warmup epochs	0

Table C.2: Design Variables maximum and minimum boundaries.

Design variable	Definition	Minimum value	Maximum value
xa	X coordinate of	Minimum x	Maximum x
ya	Y coordinate of	Minimum y	Maximum y
xb	X coordinate of	Minimum x	Maximum x
yb	Y coordinate of	Minimum y	Maximum y
t	MMC thickness	0.001	0.2

Table C.3: Data augmentation parameters used during training of YOLOv8-TO

Augmentation type:	**Probability:**	**Parameters:**
Mosaic	1.0	2 x 2 grid
Horizontal flip	0.5	-
Vertical flip	0.5	-
Image rotation	1.0	$U(-45°, +45°)$
Image translation	1.0	$U(-40\%, +40\%)$
Image shear	1.0	$U(-10°, +10°)$
Image scaling	1.0	$U(50\%, +150\%)$
Image perspective distortion	1.0	x/y-distortion:
JPEG Image Compression	0.1	$U(75\%, 100\%)$
Image blur	0.01	Kernel size : $U(3,7)$
Median blur	0.01	Aperture size : $U(3,7)$