School of Information Technology and Engineering at the ADA University

School of Engineering and Applied Science at the George Washington University

TOPOLOGY OPTIMIZATION USING GENERATIVE MODELS

A Thesis
Presented to the Graduate Program of Computer Science and Data Analytics
of the School of Information Technology and Engineering
ADA University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Computer Science and Data Analytics
ADA University

By
Ilham Mirzayev

Supervisor Dr. Samir Rustamov

April 2023

THESIS ACCEPTANCE

This Thesis by: Ilham Mirzayev
Entitled: *Topology optimization using Generative Models*

has been approved as meeting the requirement for the Degree of Master of Science in Computer Science and Data Analytics of the School of Information Technology and Engineering, ADA University.

Approved:

(Adviser)	(Date)
(Program Director)	(Date)
(Dean)	(Date)

ACADEMIC INTEGRITY STATEMENT

“I affirm that this is my own work, I attributed where I used the work of others, I did not facilitate academic dishonesty for myself or others, and I used only authorized resources for my Thesis, per the ADA University Academic Integrity requirements. If I failed to comply with this statement, I understand consequences will follow my actions. Consequences may range from failing the course to expulsion from the program/university and may include a transcript notation.”

Ilham Mirzayev		24.04.1023
(Full Name)	(Signature)	(Date:DD.MM.YY)

ABSTRACT

Manufacturing has always sought to decrease production costs and increase efficiency. Moreover, it also needs to consider the environmental impact. Topology Optimization is a way to achieve both by reducing material used during production, leading to less waste and better long-term environmental benefits. The process involves using iterative algorithms to achieve optimal structural layout while considering different constraints, conditions, and loads. The aim is to maintain the strength of the final product while removing unnecessary material loads. There are several approaches to achieving topology optimization, including density-based methods such as SIMP and BESO. Although effective, these methods can be computationally costly and time-consuming. Deep learning models, such as CNN and GAN networks, have shown promise in decreasing the time and cost required for the optimization process. Research in this area is ongoing, and new techniques are continually being developed. In this research, the potential of different Generative Adversarial Networks in Topology Optimization has been analyzed. The training process of the GAN models has been tried to improve while keeping them highly accurate. This was achieved by the proposal of a new hybrid generator architecture. To compare the results of proposed network, [1] model and results has been referred as a benchmark. The proposed model could achieve nearly 1.6 times improved training time. A slight decrease of 0.02 MAE and MSE scores have been observed.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Explanation
GAN	Generative Adversarial Networks
SIMP	Solid Isotropic Material with Penalization
BESO	Bi-directional Evolutionary Structural Optimization
CNN	Convolutional Neural Networks

1 INTRODUCTION

As the manufacturing industry evolved, there have always been ways to decrease the costs of production and make production faster. Along with making the manufacturing process efficient, the ideal production process should also need to consider the environmental effects. One of the approaches to decrease the cost of production but at the same time have environmental benefits is achieved through decreasing the material used. Compared to traditional additive manufacturing, the usage of less material during production causes less waste. The long-term environmental benefit of optimized manufacturing continues even after the production is complete and the products are in use. For example, lightweight parts of the airplane decrease fuel consumption resulting in less environmental effect on nature. The process of trying to use less material while production in the manufacturing industry is called Topology Optimization. It is the method where the iterative algorithmic models try to achieve optimal structural layout by considering different sets of constraints, conditions and loads. The optimization process achieves better performance and efficiency by removing unnecessary material loads from the structure that does not have sufficient effect on the overall strength of the final product.

1.1 Definition of the Problem

The main point to consider while decreasing the used material is about keeping the structural strength of the product as strong as it was before. To achieve the optimal structural design the solver applies different sets of holes and cuts to the different parts of the structure resulting in the reduction of the material to manufacture the structure. Several approaches have been developed and implemented for effective optimization of topology during manufacturing. Currently, the optimizers used widely in the industry include density-based approaches and utilizes either Simplified Isotropic Material with Penalization (SIMP) or Bi-Directional Evolutionary Structural Optimization (BESO). To converge to a final binary solution, the SIMP approach suggests using penalization of the material's intermediary density values. BESO method takes advantage of the idea where it adds materials to the areas where the stress is higher while removing the material form the low stress area. These methods apply different sets of equations on a given conditions iteratively to achieve the optimal layout. Although, those approaches are very effective in achieving the desirable results for the optimization, their efficiency during the process is questionable. The iterative approach of improvement of the material distribution is computationally costly and time-consuming process. Considering the performance of deep learning models in many areas, they can be a source of solution for decreasing the cost and time required for the topology optimization process. The proposing future of machine learning especially deep learning models in optimization of materials in manufacturing has inspired several novel works in this area. The research in the area ranges from the use of CNN models to the utilization of the GAN networks for the same problem. I will be reviewing several state-of-the-art novel research in the area in the following paragraphs.

1.2 Objective of the study

The objective of this study is to investigate the effectiveness of new hybrid generator architectures in topology optimization using a GAN model. The study will focus on designing and testing various hybrid generator architectures with the goal of improving the efficiency of training while maintaining high accuracy in topology optimization tasks. The study will compare the performance of the new hybrid generator architectures against the TopologyGAN [1]model, which will serve as the baseline model for comparison. The study will evaluate the performance of the different architectures by analyzing various metrics such as convergence speed, solution quality, and computational efficiency. The overall

objective of the study is to develop new hybrid generator architectures that can improve the efficiency and effectiveness of topology optimization using GAN models, leading to better design outcomes in various engineering and design applications. To evaluate the performance of the proposed method, the study will utilize two main datasets. The first dataset is provided by the authors of [1], which includes a range of examples of topology optimization problems. The second dataset is generated by the TopoDiff [2] project, which uses diffusion models to achieve faster and more accurate topology optimization results. Total 3 experiments have been realized which are:

a) InceptionUNET architecture with TopologyGAN [1] dataset with 16, 32, 64, 128, 256, 512 increasing channel size
b) InceptionUNET architecture with TopologyGAN [1] dataset with all 512 channel size
c) InceptionUNET architecture with TopoDiff [2] dataset.

1.3 Significance of the Problem

Several engineering and design applications place a high value on the topology optimization challenge. In order to accomplish a particular performance objective, such as minimum weight or maximum stiffness, topology optimization seeks to identify the best material distribution within a given design space, subject to specific design restrictions. Topology optimization can help structures become lighter and use less material, while also enhancing their structural performance and lowering their environmental effect. However traditional topology optimization techniques, including gradient-based or level set methods, can have drawbacks like sluggish convergence, sensitivity to initial conditions, and difficulties dealing with complex geometries. As a result, there is growing interest in applying deep learning methods for topology optimization, such as GAN models. There are several potential advantages to using GAN models for topology optimization, including quicker convergence, better handling of difficult geometries, and the capacity to produce more varied and creative design solutions. The use of GAN models for topology optimization is not without its difficulties, however, including stability issues, the requirement for huge quantities of training data, and the difficulty in creating efficient generator architectures. The significance of this research question therefore resides in its ability to overcome these issues and contribute to the creation of more effective and efficient design tools for topology optimization using GAN models. This study aims to advance the state-of-the-art in topology optimization research and offer useful solutions for various engineering and design applications by designing new hybrid generator architectures and investigating the impact of various hyperparameters and training strategies on the performance of GAN-based topology optimization. The results of this study may result in engineering designs that are more inventive, efficient, and environmentally and economically beneficial.

1.4 Review of Significant Research

The first interesting approach is by Sosnovik and Oseledets in their paper named “Neural networks for topology optimization” [3]. The authors of the paper approached the problem of speeding up the topology optimization process as an image segmentation problem and tried a neural network for solution. Their idea for speed up process is about using SIMP solver for some iteration and then applying neural networks. They used an encoder decoder architecture where they take the input as two 40x40 binary image or an image with two channels. The first image is the density distribution obtained after the application of last performed iteration by the optimization solver. The second image is last performed update of the densities. It can be obtained by the subtraction of the result in $(n-1)^{th}$ iteration from n^{th} iteration. The model outputs a grayscale image in the same resolution representing final

predicted structure. The proposed solution by the authors [3] have achieved the maximum of 99.6 percent binary accuracy even after the application of SIMP solver for 80 iterations. The model showed results more than 92 percent in different data distributions of the data even after the 5th iteration of the SIMP solver. The final insights gotten from the paper is about the successful transferability of the proposed solution from dataset of small resolution images to the problems defined on grids with better resolution.

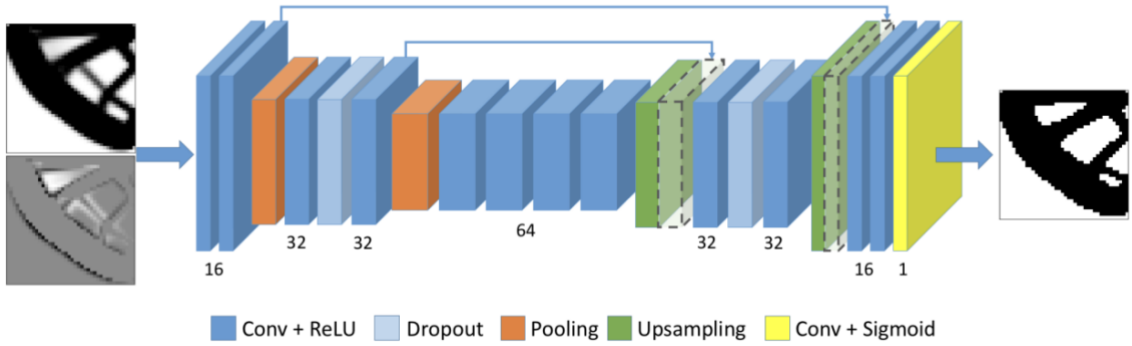

Figure 1. CNN for Topology Optimization

Another novel machine learning for speeding up the TO process is through the usage of Generative Adversarial Networks (GAN). The GAN is special type of machine learning architecture where two network competes for final generation of the output. The first network is called the generator and based on the features learnt from the labelled training dataset, it tries to generate an image that is indistinguishable from the ground truth image in the ideal case. However, on the other hand the discriminator network tries to detect whether the image fed into it is the image from the set of real images or the one generated by the generator network. The discriminator is a binary classifier predicting whether the image is real or not. The networks are playing adversarial zero-sum game against each other while both trying to decrease their losses resulting in the decrease of the total loss function. The special type of GAN networks is called Conditional GAN where a conditional setup is implemented. Both generator and discriminator are dependent on the extra data originating from information of other modalities. Thus, the architecture is capable of learning multi-modal mapping from input to output using various contextual data.

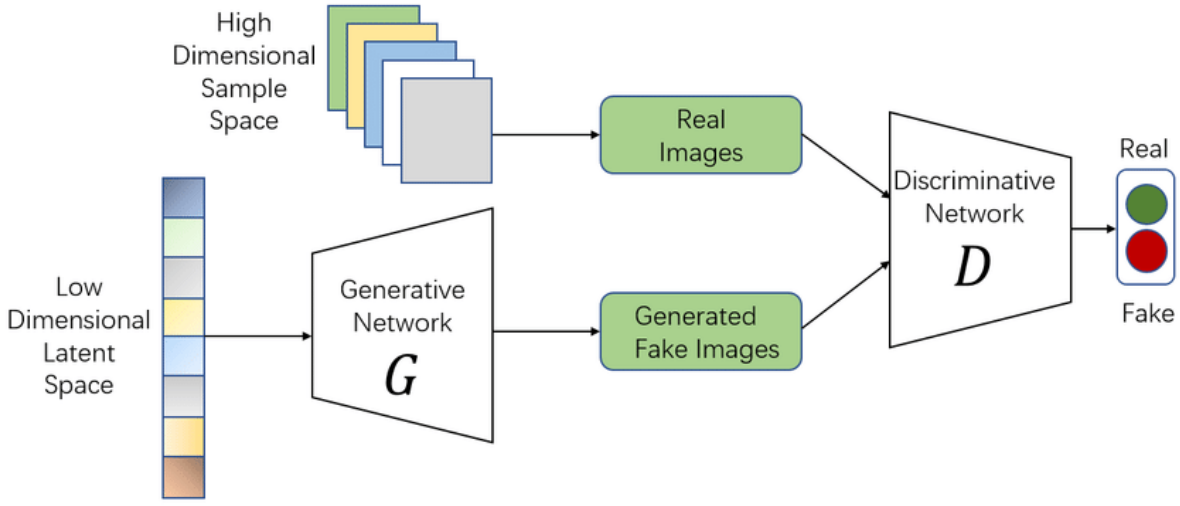

Figure 2. GAN Architecture [4]

Zhenguo Nie and et al proposed the application of modified Conditional GAN for the solution of speed improvement in TO [1]. The authors came up with idea that, along with the boundary conditions, the information of physical fields computed on initial and unoptimized domain subject may also have necessary features that the network can learn. Those physical fields include von mises stress, strain energy and displacement fields. The research also encompasses the development of the new conditional GAN where the authors have developed a new hybrid generator architecture called SE-Res-Unet. The architecture combines the strength of Squeeze and Excitation, Residual, and UNet networks in a single network. The new hybrid generator helps the model to perform dynamic channel-wise feature recalibration. The overall architecture of the network consists of 2 CNN and 1 SE networks. The authors used a SIMP based optimization framework called ToPy [5] to generate their dataset. They have created totally 49078 optimized structures with 42 different boundary conditions. The random conditions fed into the ToPy framework for generation of the dataset is as follows:

• volume fraction: [0.3: 0.02: 0.5]

• displacement BCs: 42 scenarios

• load position: any point on the boundary of the domain

• load direction: [0: π /6: π]

• SIMP penalty: 2

• SIMP filter radius: 1.5

The authors further process the data before feeding it into training process. The final processed data consists of 7 matrices with shape of 64x128, which are the information about volume fraction, VM stress, strain energy, load along x and y axis, boundary condition and output ground truth image. The research has achieved quite fascinating results with 0.001808, 0.019133 0.070128, values for MAE scores of training, validation and test data correspondingly. The MSE scores in the same order is 0.001340, 0.018022 and 0.059943. However, the authors have stated several limitations related their work which are the following. There is no mechanism that ensures the generated architecture will be a single connected component. Secondly, while the volume fraction should be treated as an upper boundary by the optimizer, their solution tries to match it as closely as possible. And the final limitation is about the required extra modification of the solver from 2D to 3D structures.

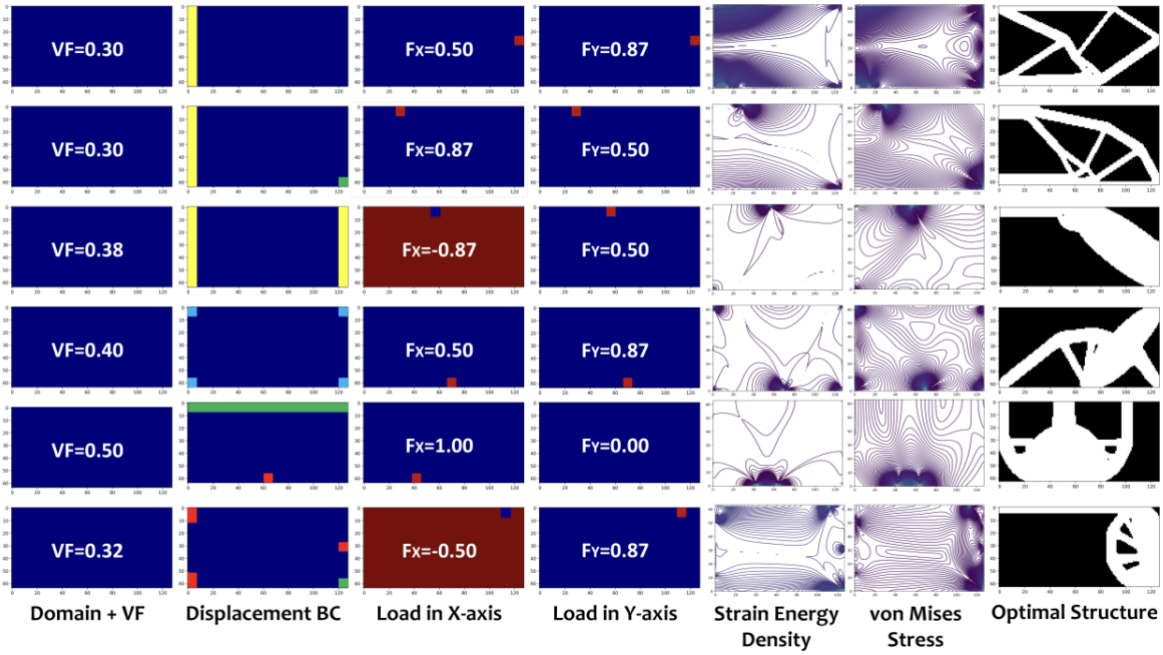

Figure 3. Dataset for TopologyGAN

One of the most recent research projects in the topology optimization domain using generative models is by Faez Ahmed [2]. The authors have used diffusion model for training generalized topo0logy optimization solver model. The authors have argued that GAN models have drawbacks, including the inability to be easily trained, the lack of generalizability, and the disregard for manufacturability and performance goals. They suggest TopoDiff [2], a conditional diffusion-model-based architecture for performance and manufacturability-aware topology optimization, to overcome these drawbacks. The surrogate model-based guidance technique that their model offers actively favors structures with low compliance and strong manufacturability. They claim that their method greatly outperforms a state-of-the-art conditional GAN by producing eleven times fewer infeasible samples and reducing the average error on physical performance by a factor of eight. The authors demonstrate that conditional diffusion models can outperform GANs in engineering design synthesis applications by integrating diffusion models into topology optimization. The research of Ahmet and et. al. also offers a comprehensive framework for engineering optimization issues that incorporates constraint-aware guidance, diffusion models, and external performance. TopoDiff [2] offers a method for topology optimization that is both performance- and manufacturability-aware for a variety of engineering design issues.

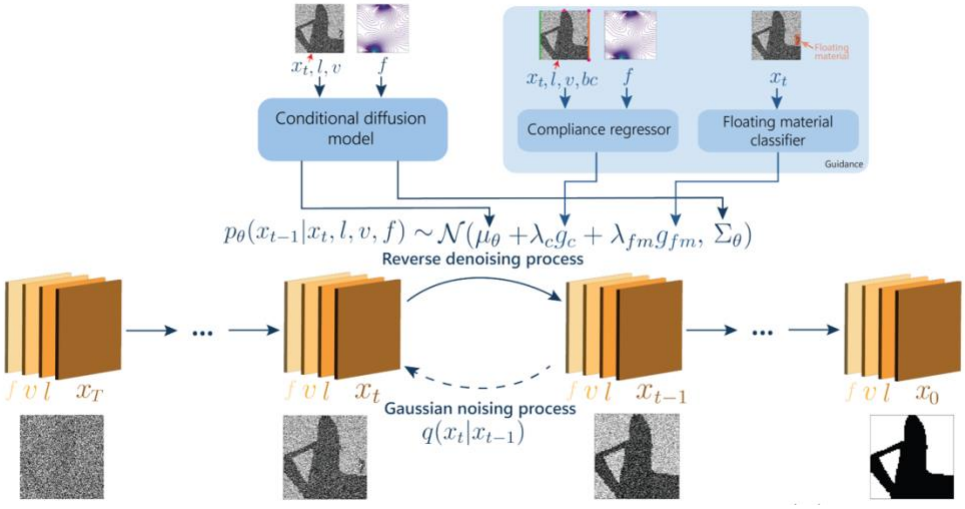

Figure 4. Topology Optimization using Diffusion Models [2]

All the previously mentioned research is about the optimization of the TO process during the generation of the structure meeting the certain conditions. To develop and implement a generalized solution that is able to generate a structure given the conditions requires huge amount of the dataset. The generation of the dataset for the training process of the machine learning algorithms for TO task itself is very resource consuming and costly process.

Tan and et. al proposed the optimization in the process of TO during the generation of the dataset for the training of the optimization solve [6]. In their paper the authors proposed the usage of two types of architectures to achieve the speed improvement in the TO process. The first network is called mapping network that is responsible of mapping coarse field to a fine field to generate fine-scale training data. By doing this, they achieve generation of a portion of the data using simulations performed on coarse mesh instead of the generation of all data through full-scale calculation. A certain portion of the whole data required for the full training process is generated at both coarse and large scales using Finite Element Analysis (FEM) calculations. Afterwards, the mapping network is trained using the portion of the data which has strain energy field data at both scales. The remaining part of the full-scale dataset is generated using the application of the mapping network on data which has strain energy filed in coarse scale. Since it is doing a field-to-field mapping, the application of the mapping network also handles the poor transferability of ANN based surrogate models. The second part of the improvement is achieved through the utilization of a surrogate model called Property Network that is responsible for mapping structure designs directly to their strain energy field. As the property network the authors have taken and modified a network called StressNet from the work of Nie et al [7]. Authors have concluded that the efficiency got through the application of the mapping network does not compromise the performance of surrogate models. To prove their hypothesis the authors have done experiments in designs of cantilever and L-shaped beams. The efficiency achieved through the application of the mapping network gets even higher when the scale difference between the coarse and fine mesh is large.

1.5 Assumptions and Limitations

The authors of TopologyGAN [1] paper have mentioned the problem of having a single connected structure at the end of the optimization as the problem. The problem of deep learning approaches in the mentioned area could not guarantee that the generated optimized structure will be a single connected component. The same problem will also be present even with the different generator architecture. The research study is also subject to some challenges and limitations related to the high volume of the

dataset. To be able to train a generalized solution that is able to perform accurately in different conditions, samples from different boundary conditions has to be considered together. The challenge is about the requirement of large CSV files utilization during the training process, with each file being approximately 1-2 GB in size. The total volume of dataset goes up to 80-90 GB adding up the examples from 42 different boundary conditions. The ability of the machine to handle such amount of data all loaded into memory at once to enable the trained model to be more general is among the most challenging limitations of the project. Despite the availability of batch processing approaches that could partially address the issue of high RAM requirements by ensuring that only the relevant data for the current training batch is loaded into memory, the lack of high-performance computing resources results in significantly long training hours. Consequently, the extensive training time poses a significant challenge for the study to try and experiment with different methods in an efficient manner. This limitation presents an obstacle to the research study as it requires ample resources to manage the dataset and conduct the necessary experiments. Therefore, it is important to carefully consider these limitations when designing and planning future research in this area.

2 LITERATURE REVIEW

Widely used solvers for topology optimization in industry are SIMP and BESO approaches which applies set of equations to the given structure to achieve optimal structure. The SIMP approach operates by giving each element in a finite element model of the structure being optimized a density value. The density value, where a value of 0 indicates no material and a value of 1 indicates the maximum quantity of material, indicates the percentage of the element that is occupied by material. The density values are iteratively adjusted during the optimization process to identify the best configuration that satisfies the design requirements, such as reducing the structure's weight while retaining structural integrity.

$$\begin{aligned} \min_{\chi} \quad & \int_D f \cdot u(\chi)\, \mathrm{dx} + \int_{\Gamma_N} T \cdot u(\chi)\, \mathrm{dS} \\ \text{s.t.} \quad & \int_D \chi(x)\, \mathrm{dx} = \eta \mathrm{Vol}(D) \\ & \chi(x) \in \{0, 1\} \end{aligned}$$

Mathematical methods, such as gradient-based optimization algorithms, evolutionary algorithms, or other heuristic optimization techniques, are frequently used to carry out this optimization. A penalization term is another component of the SIMP method that aids in preventing unrealistic solutions, such as those involving parts with wildly disparate densities. Each element with a density value that is significantly different from the structure's average density is penalized by the penalization term, which is added to the objective function being optimized. By encouraging a smoother transition between high-density and low-density areas, the penalization term also ensures that the resulting designs are both physically plausible and attainable. It is frequently essential to transform the ideal density distribution into a more usable format, such a CAD model or a set of production instructions, after it has been identified. Due to the complexity and difficulty of manufacturing the designs produced by topology optimization using SIMP, this conversion procedure might be tough.

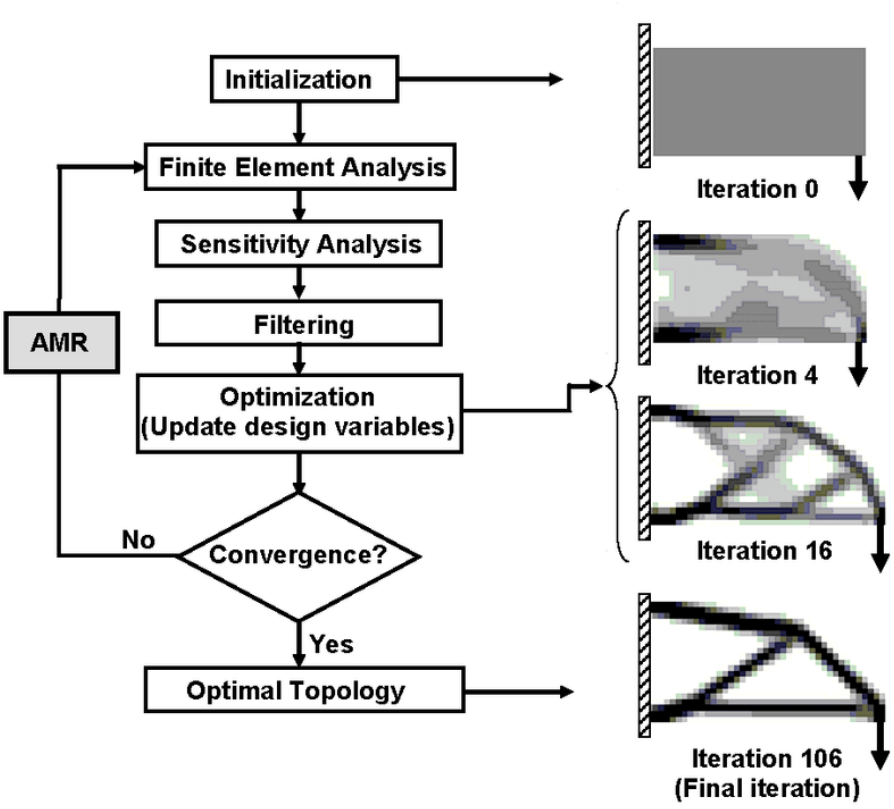

Figure 5. Iterative Topology Optimization Approach [8]

The Bi-directional Evolutionary Structural Optimization (BESO) method is one strategy for topology optimization. To determine the ideal material distribution within a design space, BESO is an iterative approach that combines topology optimization and genetic algorithms. The method's foundation is the idea that material should be added where it is most needed and removed from the design space where it is least important. The density of each element is considered a design variable in BESO, which divides the design space into a finite number of elements. The first step of the method is to give each element a density value and determine the corresponding structural performance. Then, in areas where performance is most compromised, new pieces are introduced and the lowest density elements are eliminated. Until the target level of performance is attained or convergence is obtained, the process is repeated. The BESO method's ability to eliminate any initial assumptions or guesses about the ideal topology of the structure is one of its main benefits. Instead, it removes and adds material in an iterative process to change the topology. Because manual optimization might be difficult or even impossible for complicated design problems, the approach is very helpful in certain situations. The simultaneous handling of several performance criteria by BESO is another benefit. While still achieving the weight loss target, the approach can optimize for additional goals including stiffness, strength, and natural frequency.

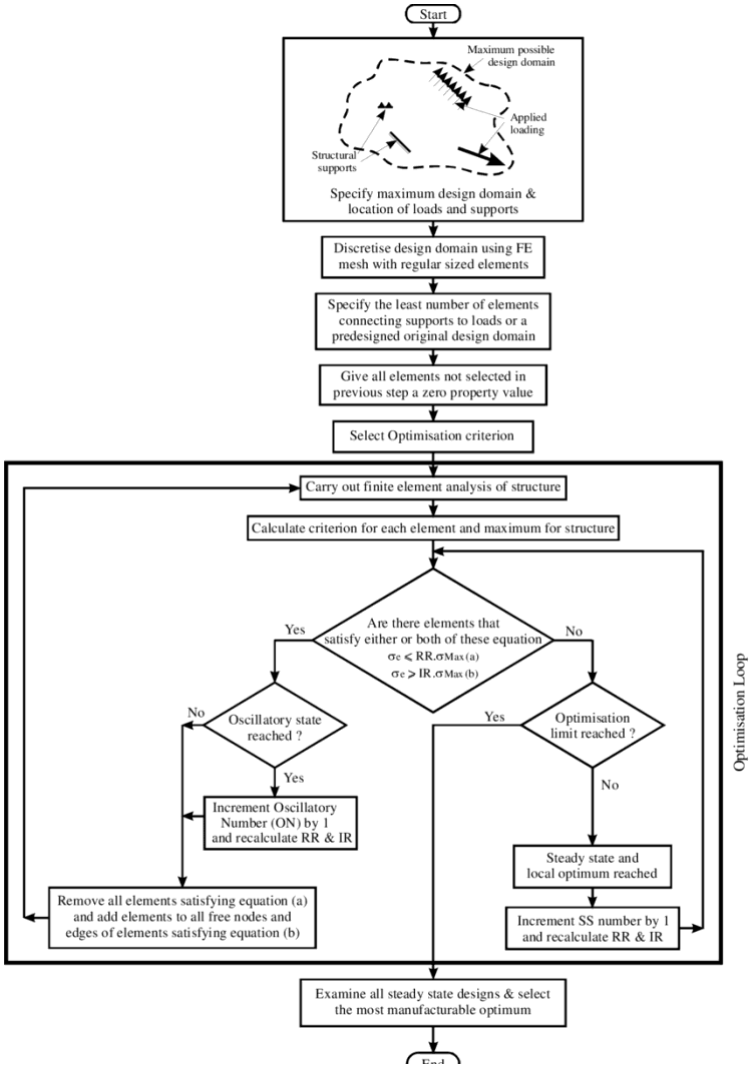

Figure 6. Topology Optimization using BESO [9]

The mentioned methods have been widely used in the industry to find the optimal structures in production of objects in different industries. Since those methods are iteratively apply different sets of equations within given constraints, the process of finding optimal structure becomes costly and time-consuming process. To speed up the process different acceleration approaches have been researched. Topology optimization using acceleration approaches are categorized into 3 groups [10].

i) CPU accelerated – multi-core CPUs are being utilized in this category of accelerated topology optimization. The application of parallel computing to science and engineering problems started decades ago to obtain numerical results in a shorter time. Borrvall and Petersson were among the first to computationally study topology optimization using a high-performance computer, decomposing the numerical domain into subdomains to allow parallel processing [10]. They achieved significant speedup values using multiple processors compared to a single processor sequential algorithm. Kim et al [11]. also used parallel computing to solve large-scale structural eigenvalue-related design problems, using a parallel subspace iteration method based on PCG solver and Message Passing Interface (MPI) for communication between subdomains. The authors used a successive estimation scheme for eigenvalue analysis and applied the optimal criteria (OC) method for optimization, achieving improved computational time through parallelization.

ii) GPU accelerated – in this category topology optimization is parallelized using multi-core graphics cards. Modern high-performance systems now use graphics processing units

(GPU) for scientific computing because of the GPU's many-core processor design, which can manage numerous concurrent threads. Although topology optimization (TO), an iterative method, precludes the possibility of parallelizing the entire procedure, research has been done to speed up TO issues utilizing GPUs. Studies comparing CPU and GPU acceleration methods have demonstrated that as the number of element sizes rises, GPU-based solutions outperform CPU-based ones.

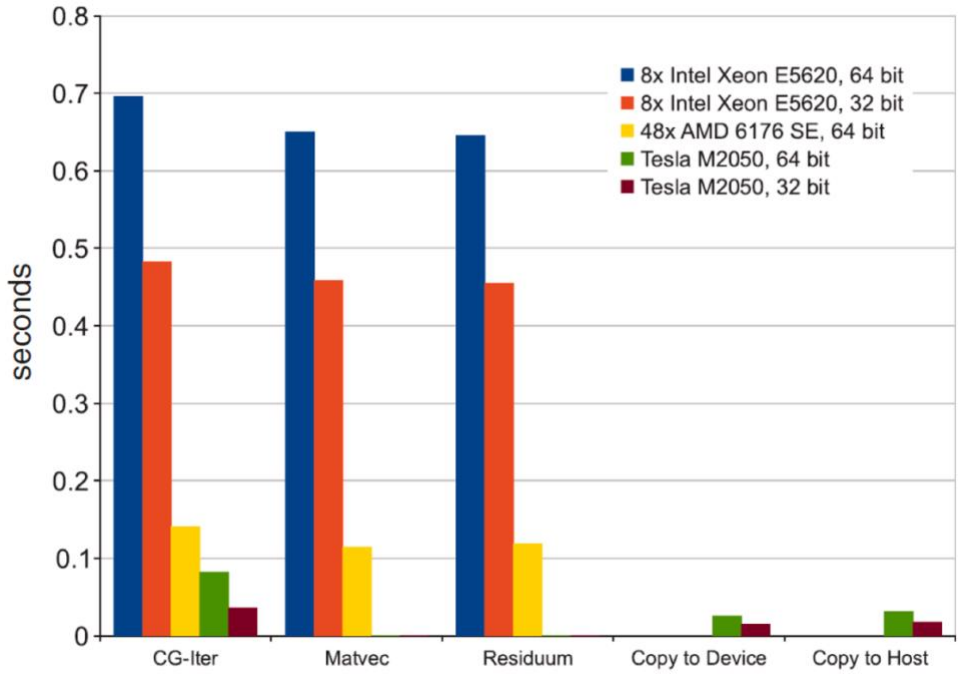

Figure 7.CPU and GPU time needed for different subroutines [11]

iii) ML-accelerated: TO problems are addressed using machine learning techniques, such as artificial neural networks and image processing techniques, with the goal of lowering the computation cost. The AI methods used for topology optimization varies from CNNs to generative models like GAN and Diffusion models. The research of GAN models in topology optimization domain has yielded promising results. Yu et al. [12] proposed a method to replicate optimal designs for material stress using low-resolution information by coupling SRGAN with a convolutional encoder-decoder network. This approach differs from the one used by Li et al. [13], who used an additional GAN to integrate with SRGAN. The dataset used in this method was composed of an image channel created from a set of force and geometric boundary conditions, passive zone, and mass fraction. As some optimization parameters are scalar, they are separated from the input domain and directly connected to the latent space achieved after completing the encoding operation in the first network. The SRGAN takes the low-resolution structures generated by the convolutional encoder-decoder network and refines them to high resolution. The mean absolute error was added to the final loss function of cGAN, resulting in mean absolute percentage errors of 0.99% and 2.20% for the train and validation sets, respectively. The authors claimed that the proposed method can generate a near-optimal structure in terms of pixel values. However, some predicted structures at high resolution showed disconnections for some of the design spaces.

3 RESEARCH METHODOLOGY

3.1 Development of GAN models

GAN (Generative Adversarial Networks) is a machine learning network consisting of two neural networks. The parts of GAN are called generator and discriminator and those networks play adversarial game with each other trying to beat each other which results in the improvement of both models. While the generator plays the role of art forger by trying to generate samples that should mimic the real data as much as possible, the responsibility of discriminator is to detect whether the sample is real or generated by a generator. The generator network is typically trained using a random noise vector as input, which it then transforms into a generated sample. In case of multiple classes generated by the GAN network, the input of the generator will also have the label. The role of the generator is to take the noise vector from latent space and/or label and generate features for the given label.

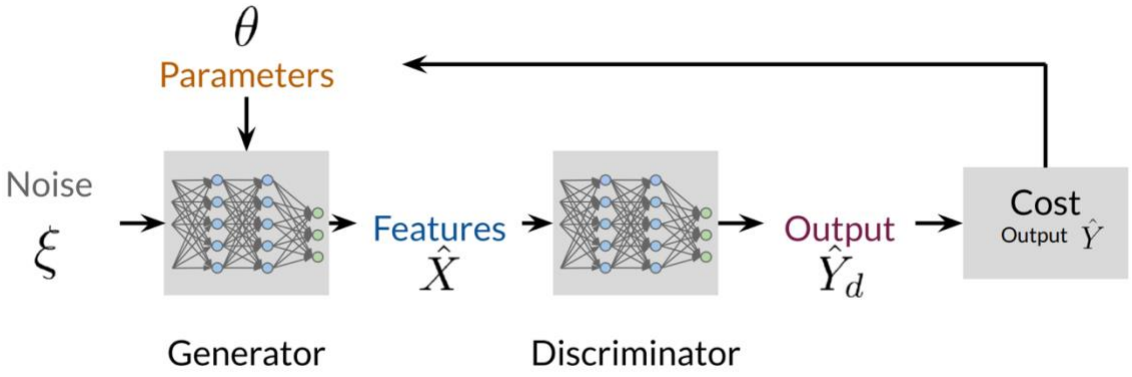

Figure 8. Generator Training (deeplearning.ai)

The discriminator network is trained on both the real data and the generated data, learning to distinguish between the two. While the generator model is not allowed to look at the real images, the discriminator can see the batch of images consisting of both real and fake images. Generator model gets better only by the score that the discriminator model assigns to its generated image.

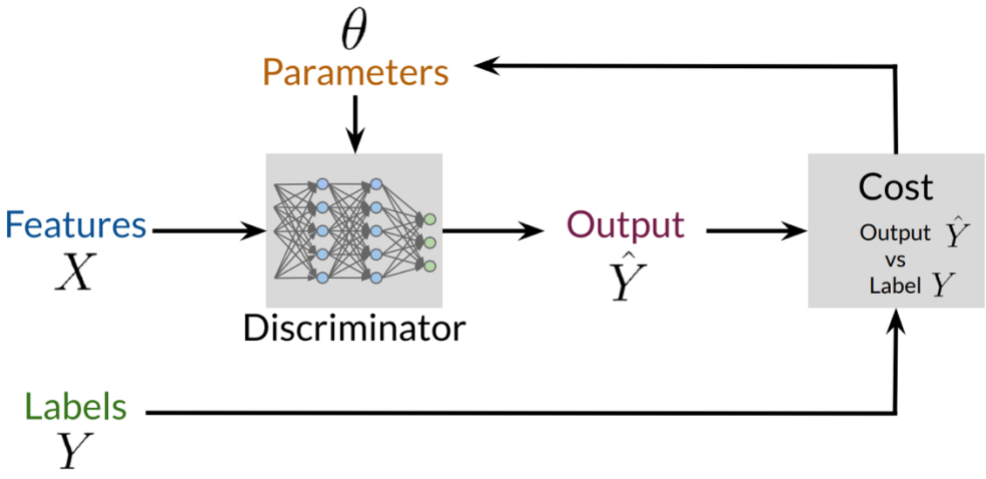

Figure 9. Discriminator Training (deeplearning.ai)

The two networks are trained simultaneously, with the generator attempting to produce more realistic data and the discriminator attempting to correctly identify whether a given sample is real or generated. This back-and-forth process continues until the generator produces data that is indistinguishable from the real data. One of the key features of GANs is their ability to generate data that is not simply a direct copy of the training data, but rather represents new, realistic variations on that data. This makes them useful for a variety of creative and practical applications, such as generating art, synthesizing new medical images, and creating new synthetic data for training other machine learning models. However, GAN training can be challenging, as the generator and discriminator networks must be carefully balanced in order to achieve good results. One common problem is that the generator may produce samples that are too similar to each other, or that fail to capture the full range of variability in the training data. Another issue is that the discriminator may become too good at distinguishing between real and generated data, leading to a "mode collapse" where the generator produces only a few types of samples that fool the discriminator, rather than a wide range of realistic samples. In case when the models are not improving in balanced way the training process becomes useless. If the discriminator is too much powerful than the generator, it will simply output 100 percent fake result for the images generated by the generator. 100 percent feedback by the discriminator is not helpful for the generator since the model will not be able to distinguish which direction to make improvement. Otherwise, if the generator is much powerful than the discriminator, the 100 percent confidence assigned to the generated images will make the training process suffer again. 100% real label assigned by the discriminator will disable the discriminator’s ability to get better and distinguish the real images from the fake ones. These problems result in common problems called mode collapse and vanishing gradients with GAN training process. Mode collapse occurs when a GAN generates a limited set of output examples, even though the input data may have a more diverse range. In other words, the generator produces similar-looking samples, rather than capturing the full range of variation present in the input data. This can happen when the discriminator network becomes too powerful and is able to distinguish between real and fake samples too easily. As a result, the generator may find it easier to generate a limited set of samples that are difficult for the discriminator to distinguish, rather than trying to capture the full diversity of the input data.

Vanishing gradients, on the other hand, occur when the gradients of the loss function with respect to the parameters of the GAN become too small. This can happen when the discriminator network becomes too good at distinguishing real and fake samples, and as a result, the gradients become too small to be useful for updating the generator's parameters. This can cause the generator to get stuck in a local minimum, preventing it from improving further. Both mode collapse and vanishing gradients can be challenging to address in GANs, and researchers have proposed several approaches to mitigate these issues. Some methods for addressing mode collapse include adding noise to the input data or the generator's parameters, using different loss functions, or adding regularization terms to encourage diversity in the generated samples. To address vanishing gradients, techniques such as weight initialization, batch normalization, and alternative loss functions can be used.

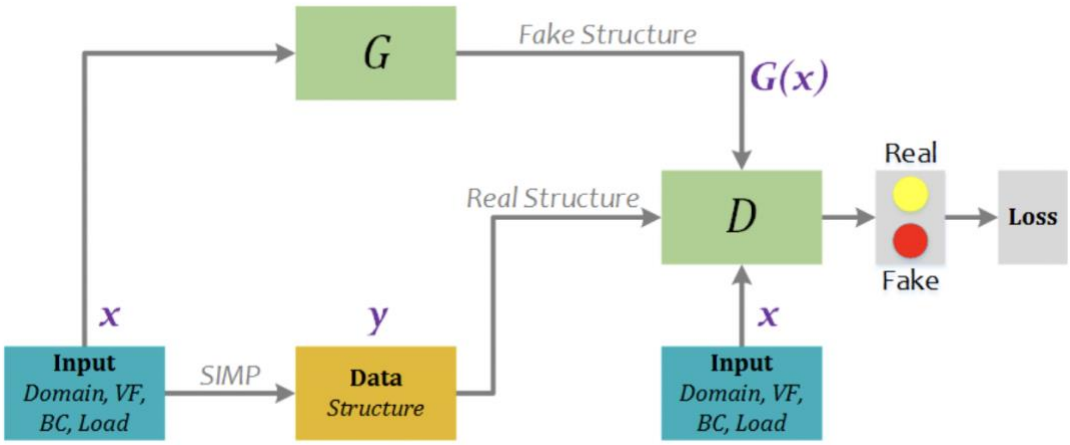

Figure 10. Conditional GAN architecture in Topology GAN

To address these challenges, researchers have proposed many variations and extensions of the basic GAN architecture. For example, conditional GANs allow the generator to be conditioned on additional input, such as class labels or attribute values, in order to generate samples that meet specific criteria. Topology Optimization tasks have been treated as an image segmentation tasks by many machine learning approaches. UNet [14] is a deep learning architecture that was developed for image segmentation tasks, particularly in biomedical image analysis. The architecture of UNet [14] consists of a contracting path, which is responsible for capturing context, and an expansive path, which is responsible for precise localization. The contracting path is a traditional convolutional neural network that learns to encode the input image into a lower-dimensional representation. It consists of several layers of convolution, followed by max pooling operations. These operations gradually reduce the spatial dimensions of the input image while increasing the number of learned features. The expansive path, on the other hand, consists of several up-sampling layers and convolutional layers that gradually increase the spatial resolution of the learned features. This path is designed to recover the spatial information lost during the contracting path and provide precise segmentation maps. Additionally, skip connections are added between corresponding layers of the contracting and expansive paths to preserve spatial information at different scales.

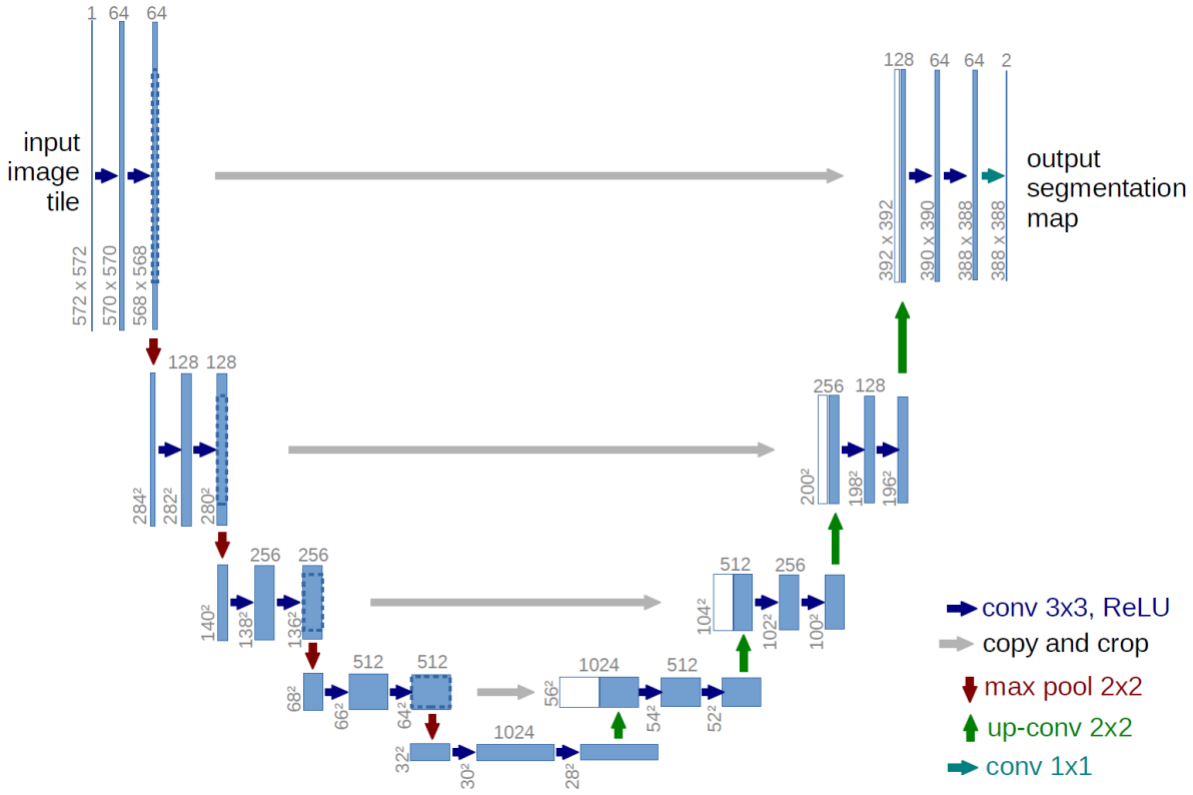

Figure 11. U-Net architecture

The UNet architecture has been widely used for various segmentation tasks, such as segmenting cells in microscopy images, segmenting organs in medical images, and segmenting roads and buildings in satellite images. It has also been modified and extended for various other tasks, such as image denoising, super-resolution, and object detection. UNet has shown excellent performance on various benchmark datasets and has become one of the most popular architectures for image segmentation tasks. Its success is due to its ability to capture both global context and local details, making it suitable for a wide range of segmentation tasks.

3.2 Model training

The first experiment was done using the SE-RES-UNET generator architecture in TopolgyGAN [7] as base model and its dataset. To test the performance of the other hybrid generator architectures, a new Inception-UNET architecture has been proposed. Overall, the 32 SE-RES blocks within the base model have been replaced with 6 inception blocks. The inception block consisted of concatenation of 1x1 followed by 5x5, 1x1 followed by 3x3, pooling followed by 1x1 and 1x1 filters. The residual blocks use shortcut connections running parallel to the convolutional layer that allows the network to skip over certain layers. This helps to avoid the vanishing gradient problem that can occur in very deep networks by allowing gradients to bypass shallow layers and flow more directly to deeper layers. However, inception blocks use a combination of multiple filters of different sizes in parallel to capture information at different scales. This helps to make the network more efficient and allows for better performance on a wider range of input sizes. Both blocks have been shown to improve performance on a variety of tasks, but they have different strengths and weaknesses. Residual blocks are particularly effective in very deep networks with hundreds of layers, while Inception blocks are best for networks with varying input sizes and are more computationally efficient. Since the training of GAN architecture is very costly

process, the replacement of the residual blocks with inception blocks tests the possibility of efficiency improvement while keeping the performance of the model high enough.

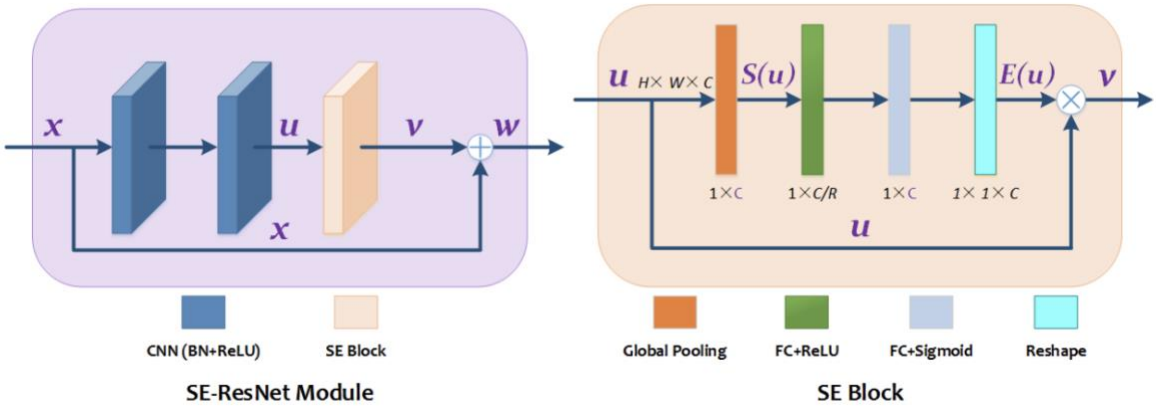

Figure 12. SE-ResNet Block and SE Block Architecture

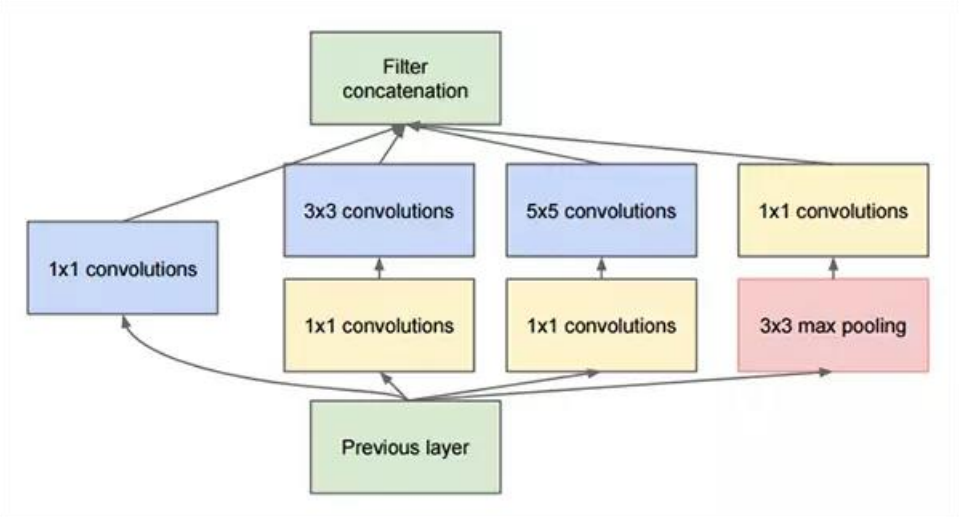

Figure 13. Inception Block

TopologyGAN [7] dataset has been used for the training of the proposed architecture. The dataset consisted of total 42 CSV files each around 1~2 GB. Each csv file describes a different boundary condition. To be able to train a generalized model for the solution, examples from different boundary conditions should be considered together. However, loading up all data to the memory at once and training using different boundary conditions was not an option due to memory limitations. To overcome this issue, several steps have been done.

A) The files in the dataset were direct results of the ToPy [5] framework. The authors in TopologyGAN have applied data processing on this dataset and uses only specific information within them. To remove unnecessary parts and decrease the overall size of the dataset, the files were saved after preprocessing is applied on them. This decreased the size of each file almost 2 or 3 times.

B) Even though the size of the total dataset is decreased 2 to 3 times, loading all the data to the memory at once was not still an option. A different approach which is about dynamic loading the data to memory in batches was applied. Using the generators in python,

development of a data generator pipeline that would load the necessary files into memory only when they are needed was achieved. This way, the requirement for all files to be in memory during the training was removed.

C) Since the data batches are generated dynamically based on the call by the training model, the data structure of the files should also be considered to speed up the loading process which will result in improvement of the total training time. The initial version of the files was in CSV format. However, since NumPy library is the main framework for data handling and manipulation of the Tensorflow framework, the files were converted and saved in npy format. It is the file extension of the data saved as NumPy arrays.

After the dataset is processed and pipeline for the training have been established, two experiments using this dataset have been realized.

3.2.1 Datasets

1. TopolgyGAN [1] dataset – The dataset consists of 42 different CSV files each representing a different boundary condition. After preprocessing of the initial dataset, the final training data becomes a 128x64 sized tensor with 6 channels. For the generation of the image volume fraction, Von-mises stress and strain energy fields were used for the creation of latent space. The other fields which are boundary condition, loadX and loadY were used for the evolution of the generated image.
2. TopoDiff [2] dataset – The dataset consists of 3 type of different files which are for constraints, loads, and the ground truth optimal image of the structure. To prepare a single training example same numbered file from each different file type were brought together and a 5-channel tensor with the shape of 64x64 was generated. There are overall 33000 files within the training dataset. The inputs of the dataset represent the prescribed volume fraction, the von Mises stress, the strain energy density and the loads applied to the boundary of the domain.

3.2.2 Experiments

1. **First Experiment** - The first experiment has been realized with the Inception-UNET architecture as generator of GAN. The channel count of added inception blocks starts with 16 and increases up to 512 by factor of two in each following layer.
2. **Second Experiment** – The second experiment have been realized in the same hybrid model with channels' sizes all being 512 in 6 sequential inception blocks.
3. **Third Experiment** – The TopoDiff [2] dataset was also used during training process to assess the performance of the model in different environments. However, different from TopologyGAN [1] dataset, the image size in this dataset is 64x64.

To provide accurate comparison of the proposed model with base model, the base model has also been trained with the mentioned datasets.

3.2.3 Model Evaluation

Generative Adversarial Networks (GANs) and other machine learning methods must go through model evaluation as part of their training process. GAN models' distinct structure and objective call for particular evaluation methods. Here are several methods for GAN model evaluation that can enhance their effectiveness and dependability:

1. Visual inspection: The accuracy and value of the generated samples must be carefully examined. Although visual examination is a typical evaluation method, it has several shortcomings and should be combined with more objective measurements. Visual examination entails comparing the generated samples in terms of appearance, structure, and diversity to the original dataset. This method is sensitive to the evaluator's personal prejudices and preferences, which might induce inconsistencies in the evaluation process even though it can offer some useful insights on the overall quality of the generated samples.

2. Quantitative evaluation: The quality and diversity of generated samples may be evaluated more objectively thanks to quantitative measurements. The metrics Inception Score and Fréchet Inception Distance (FID) are frequently employed to quantify variety. Using a pre-trained classifier, Inception Score compares the predicted class distribution of the generated samples to that of the original data to determine how diverse the generated samples are. FID, on the other hand, use the same pre-trained classifier to calculate the distance between the feature distributions of produced and real data. Structural Similarity Index (SSIM) and Pixel Distance are two examples of common quantitative metrics for evaluating GANs. While SSIM analyzes the structural similarity between the created and real data, accounting for variations in contrast, brightness, and structure, Pixel Distance measures the difference between the pixel values of the generated and actual data.

3. Discriminator loss: The generator is taught to minimize the loss function while the discriminator is trained to maximize it, and the discriminator loss is computed during training. If the discriminator loss is significant during training, it means that the generator is producing samples that are difficult for the discriminator to separate from the original data because they are similar to it. It is crucial to remember that the discriminator loss should not be the only parameter utilized for evaluation. It is also important to evaluate the loss function of the generator, including the reconstruction loss and adversarial loss. In addition, as was already indicated, quantitative measurements like Inception Score, FID, Pixel Distance, and SSIM offer a more thorough assessment of the GAN.

4. Generator loss: The goal of the GAN generator is to generate samples that are identical to the input data, and the loss function is a key component in attaining this goal. The generator loss quantifies the difference between the distributions of the real and created data, which the generator seeks to minimize. A low generator loss suggests that the generator is producing high-quality samples, and the generated samples are similar to the original data. The choice of an appropriate loss function is essential for producing produced samples of high quality because the generator's loss function differs with different GAN models. For instance, the loss function may incorporate an additional term that calculates the difference between the generated samples and the desired output in conditional GANs, where the generator is conditioned on a particular input.

5. Human evaluation: One method that can offer insightful information about the model's performance is human review. Asking human evaluators to grade the generated samples' quality using several criteria, such as visual quality, realism, and coherence, is known as

human evaluation. Since the goal of GAN models is to provide samples that are identical to real data, human review can offer a trustworthy evaluation of the model's performance. Aspects like image quality, consistency, and overall impression can be evaluated by humans who can offer comments that may not be fully represented by quantitative measures or automated review methods.

6. Cross-validation: A popular method in machine learning for assessing model performance is cross-validation, which can be utilized with GAN models. The dataset is divided into training and testing sets for cross-validation, with the model being trained on the training set and assessed on the testing set. Given that GAN models are extremely sensitive to the data distribution, it is crucial to make sure that the testing set is representative of the data distribution when applying cross-validation to GAN models. To accurately assess the model's performance, the testing set should include a wide range of samples that cover the entire data distribution. The generalization capacity of the model can also be evaluated via cross-validation. We may determine the model's capacity to generalize to fresh and unexplored data by evaluating the model's performance on a testing set that is different from the training set. Cross-validation can, however, be computationally expensive, particularly for large datasets and intricate models, thus it is vital to keep this in mind. Additionally, because GAN models generate samples rather than classify them, the evaluation metrics employed in cross-validation, such as accuracy or F1 score, may not be appropriate for assessing their performance.

By using a combination of these techniques, GAN model evaluation can provide a comprehensive understanding of the model's strengths and weaknesses and help improve the quality of the generated data.

4 RESEARCH RESULTS AND ANALYSIS OF RESULTS

4.1 Experiment I

During the first experiment the Inception-UNET generator was trained using the TopologyGAN [1] dataset. Before training the modified architecture, the original base model has been trained to establish the benchmark of the comparison. The training of the original SE-RES-UNET generator took 41.55 hours on GTX1060 graphic card. During the training, 38 of the csv files were used as in original paper of TopologyGAN [1] and the remaining 4 files were used as test dataset. The training data has also been split into 80/20 ratio of training and validation data. After the performance of the base model is recorded as a benchmark, the modified architecture has also been trained. The training of modified architecture took 28.85 hours. Both models have been trained for 100 epochs. The result of the training is visualized in Table 1.

Table 1 Comparison of Results of RE-RES-UNET and Inception-UNET (16, 32, 64, 128, 256, 512)

	Original Paper	Proposed Method
Total Training Time	41.55 Hours	28.85 Hours
Train MSE	0.006889	0.009297
Train MAE	0.014022	0.018684
Validation MSE	0.118662	0.191486
Validation MAE	0.141975	0.213091
Test MSE	0.05178334	0.117094
Test MAE	0.06482150	0.137205

As it can be seen from the results, the training time of the modified architecture is 1.46 times less than the original architecture. Despite achieving improvement in computational load of training using modified architecture, the performance of the model got degraded. After analyzing the loss curves of modified architecture's training process, it seems that the model is having hard times to learn higher features of the samples. It may originate from the number of channels used in inception blocks.

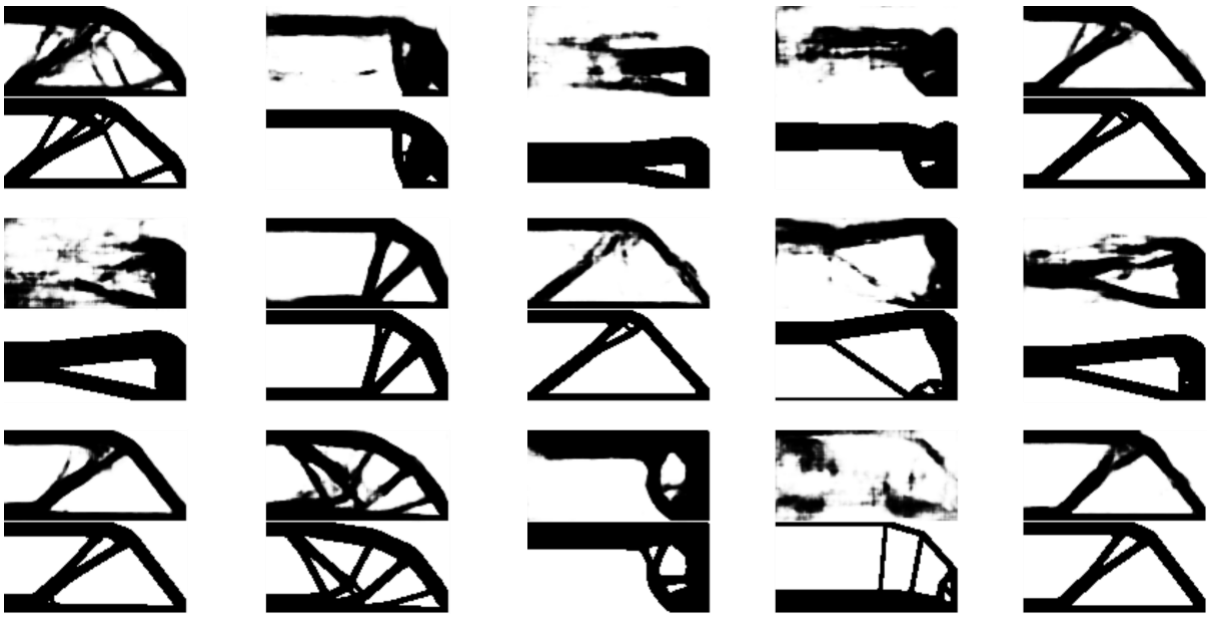

Figure 14. Structures Generated by Original RE-RES-UNET Architecture

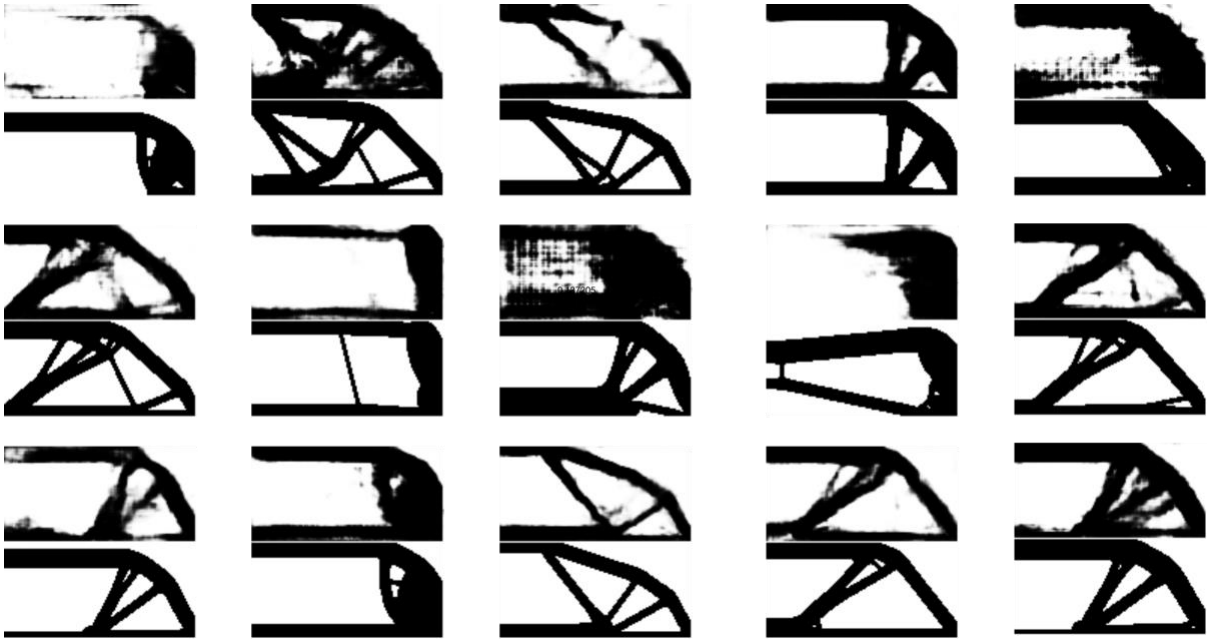

Figure 15. Structures Generated by Inception-UNET (16, 32, 64, 128, 256, 512)

4.2 Experiment II

To overcome the model's inability to learn the high order features in the second experiment, the proposed architecture kept some, while the number of filters used in inception blocks were changed 512 in all the blocks. The number of epochs both model trained has also been increased to 200. The model showed similar performance where its training time decreased twice while the performance of it got degraded.

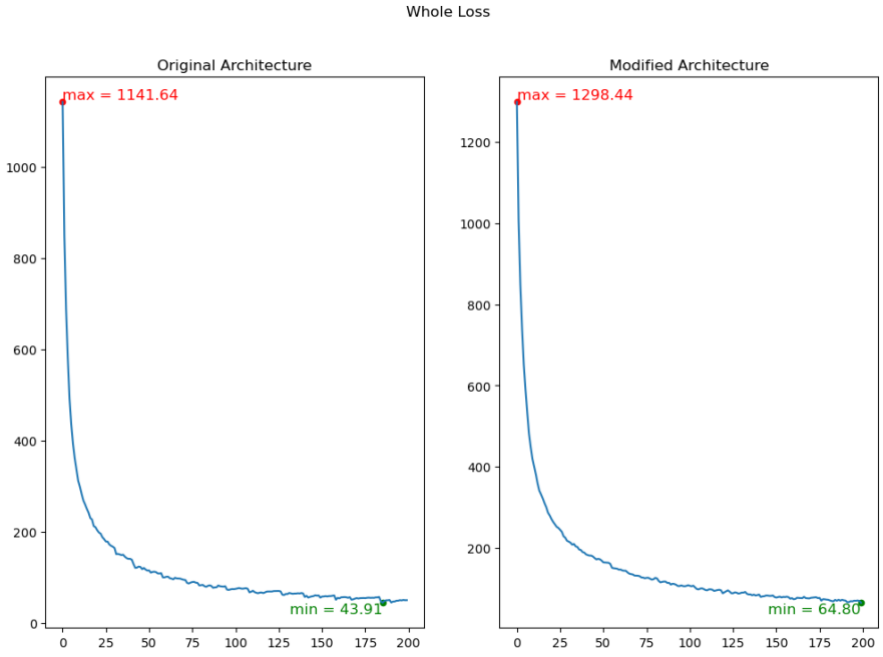

Figure 16. Whole Loss Graph of GAN training

The loss curve of the model training visualizes the improvement of the model over the time. While, the original model started with 1141 loss, the proposed model had initial 1298 loss. The original model decreased its loss till 43.91 while, the inception-unet model could only decrease up to 64.80. Even though it looks like model training may get better in the proposed model, its performance on the test

data gets worse as the number of the epoch increase. Since the whole loss is the cumulative loss of both generator and discriminator models, its decrease is not an exact sign of the generator model's improvement. Discriminator's condition to learn to differentiate fake and generated image is much simpler task compared to forcing generator to learn to produce real like and conditioned images. Thus, even though discriminator gets better, generator may still suffer from inability to learn.

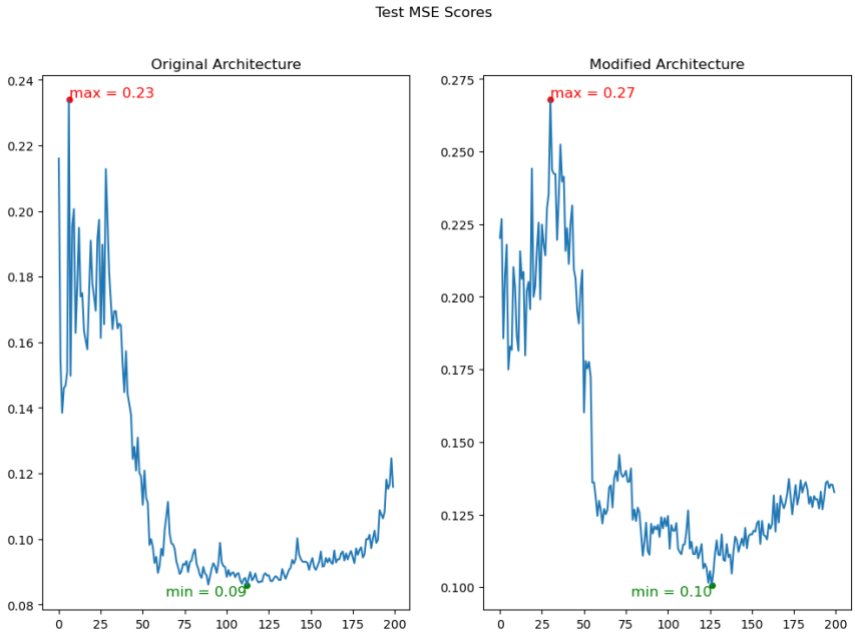

Figure 17. MSE Score comparison of Original and Modified Architecture

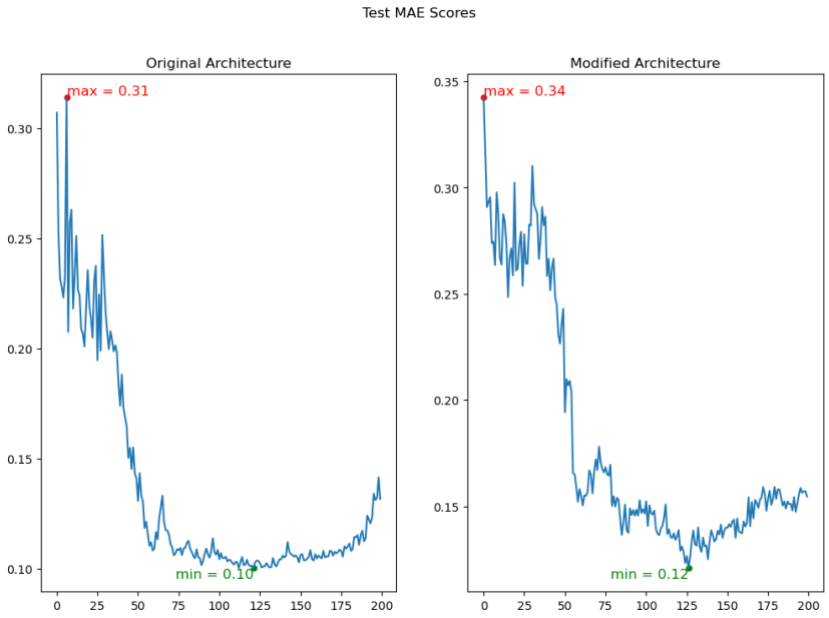

Figure 18. MAE Score Comparison of Original and Modified Architecture

As it can be seen from the accuracy metrics of the model on test data, modified architecture is more prone to overfitting. While the original architecture showed a smooth change of accuracy over epochs, the performance metrics modified architecture fluctuates a lot. However, the minimum value in both model is quite close.

Table 2. Comparison of Original and Modified Architecture (all 512) over 200 epochs

	Original Paper 200 epochs	Proposed Method all 512 Channel Size in 200 epochs
Total Training Time	84.29 Hours	51.94 Hours
Train MSE	0.003894	0.005747
Train MAE	0.008323	0.012064
Validation MSE	0.139144	0.167764
Validation MAE	0.162190	0.200245
Test MSE	0.115883	0.132732
Test MAE	0.131643	0.154603

The model training time of original model and modified architecture differs 1.6 times during training of 200 epochs. The performance of the modified architecture is poorer compared to original generator. Considering the gain of 1.6 times improved training time and approximately 0.02 unit more error in test MAE and MSE scores, I can conclude that the results are promising and may yield similar results in much shorter training time.

4.3 Experiment III

To compare the performance of the models in different examples, a different dataset [2] was also used for the training and evaluation of the models. Both original and modified architectures have been trained on the TopoDiff [2] data over 200 epochs.

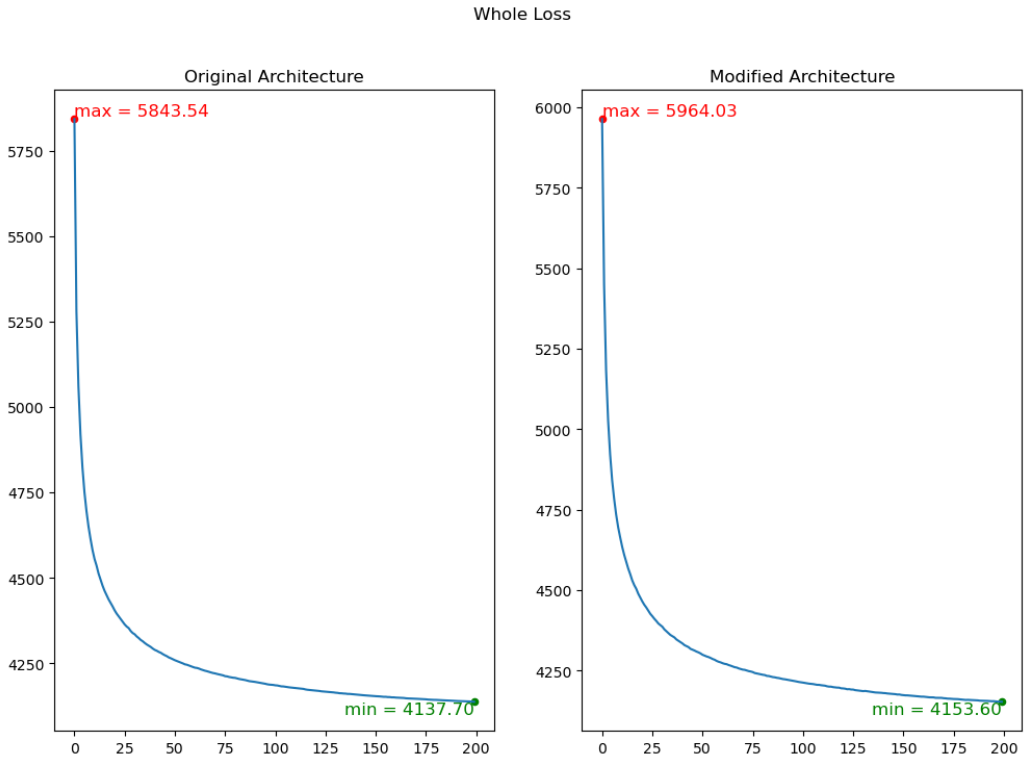

Figure 19. Whole training loss of Original and Modified architecture with TopoDiff data

The training of models with topodiff data was unsuccessful as it can be seen from the loss curves of models. The whole loss starts with 5843.54 and 5964.03 for original and modified architecture respectively. Even after 200 epochs the loss values is still quite high. The problem may originate from

two reasons. The data is not sufficient for the GAN models to learn, or there is code related issue that needs to be considered while changing the training process to train with Topodiff data. During future works, the origin of the problem will be researched and the training with the Topodiff data will be realized as well.

Table 3. Topodiff Data Training Results

	Original Paper 200 epochs with TopoDiff data	Proposed Method all 512 Channel Size in 200 epochs with TopoDiff data
Total Training Time	46.1 Hours	20.7 Hours
Train MSE	0.410695	0.412269
Train MAE	0.410757	0.412438
Validation MSE	0.494910	0.492673
Validation MAE	0.454198	0.452755

5 SUMMARY AND FUTURE WORK

5.1 Summary

I have carried out an experiment to assess the performance of GAN models in the field of topology optimization, taking into account their performance in various application areas. I referred to the TopologyGAN [1] paper and utilized their findings as a yardstick for comparison in order to assess the performance of the suggested approach. I investigated the data for the topology optimization task during the tests and looked at the function of various field information. I discovered a potential area for improvement after studying the TopologyGAN [1] model in the alteration of the generator architecture. I recommended the usage of inception blocks within the generator architecture to reduce the computational load of the training process while maintaining the high model performance. The competing results of different CNN blocks like Resnet, Inception etc. blocks in various problem domains gave me an inspiration to apply experimentation in the field of topology optimization. Two separate dataset sets were used to train the updated architecture. When compared to the original TopologyGAN [1] design, the changed architecture's training time was 1.5 times faster. However, the suggested model's performance suffered since it was unable to recognize higher order features in the training samples. I examined the impact of various hyperparameters on the model's performance in order to further enhance the performance of the redesigned architecture. I played with the epoch count, batch size, and learning rate. I discovered that increasing the number of epochs enhanced the model's performance, but there was a point beyond which the benefits diminished. The performance was initially enhanced by increasing the learning rate, but at a certain point, the performance began to deteriorate. I evaluated the model both quantitatively and qualitatively in order to validate the findings. A number of metrics, including the Pixel Distance, and Structural Similarity Index (SSIM), were calculated as part of the quantitative examination. The generated samples were visually inspected as part of the qualitative evaluation and contrasted with the original dataset. Overall, topology optimization objectives were successfully completed using the improved TopologyGAN [1] architecture. The suggested methodology addresses manufacturability and performance goals including mechanical compliance while offering a computationally effective substitute to conventional iterative topology optimization methods. The results of this experiment point to the possibility of enhancing the performance of GAN models across a range of domains while lowering computing burden by utilizing inception blocks within the generator design.

5.2 Future Work

Based on the results and observations from the experiments conducted in the domain of topology optimization using GANs, there are several potential areas for future work:

1. To boost performance, it is first possible to evaluate the effects of various hyperparameters on the redesigned TopologyGAN [1]architecture. The learning rate, batch size, and training epoch count are all included in this. The updated architecture might be able to learn higher order features and enhance its performance by adjusting these hyperparameters.
2. Investigating the application of alternative diffusion models for topology optimization challenges is another potential area of future research. Even though the proposed approach yields encouraging results, TopoDiff [2] might be examined and compared to GAN models.
3. Future research may expand on the transfer learning investigation by investigating the utilization of transfer learning from other GAN models that have been trained on comparable or related optimization problems. The training of the updated generator architecture can be sped up, which might result in better performance, by utilizing the pre-trained weights of a successful GAN model.
4. Future research may focus on extending the analysis of various input data types and how they affect the performance of the model. For the topology optimization task, for instance, the current research has used a single data format. However, it would be intriguing to investigate how the model's performance varies when various input formats, such as point clouds or 3D voxel grids, are used. Further enhancing the model's performance would be to examine the impact of various feature representation approaches, such as encoding boundary conditions or other physics-based characteristics. These investigations might increase performance on other engineering optimization challenges and help us better understand how input data types affect model performance.

6 Bibliography

[1] J. H. K. L. Nie Z, "Stress field prediction in cantilevered structures using convolutional neural networks," 2020.

[2] F. Mazé and F. Ahmed, "Diffusion Models Beat GANs on Topology Optimization," 2023.

[3] I. O. Ivan Sosnovik, "Neural networks for topology optimization," 2017.

[4] L. Cai, Y. Chen, N. Cai and W. Cheng, "Utilizing Amari-Alpha Divergence to Stabilize the Training of Generative Adversarial Networks," 2020.

[5] W. Hunter, "Topy-topology optimization with python," 2017. [Online]. Available: https://github.com/williamhunter/topy.

[6] C. Q. M. W. W. Y. Ren Kai Tan, "An efficient data generation method for ANN-based surrogate models," 2022.

[7] T. L. H. J. L. B. K. Zhenguo Nie, "TopologyGAN: Topology Optimization Using Generative Adversarial Networks Based on Physical Fields Over the Initial Domain," 2020.

[8] S. Almeida and J. Bae, "New Advances in Topology Optimization; Eric de Sturler".

[9] V. Young, O. Querin, G. P. Steven and Y. M. Xie, "3D and multiple load case bi-directional evolutionary structural optimization (BESO)," *Structural and Multidisciplinary Optimization*, vol. 18, no. 2, pp. 183-192.

[10] Y. Maksum, A. Amirli, A. Amangeldi, M. Inkarbekov, Y. Ding, A. Romagnoli, S. Rustamov and B. Akhmetov, "Computational Acceleration of Topology Optimization Using Parallel Computing and Machine Learning Methods – Analysis of Research Trends," *Journal of Industrial Information Integration*, 2022.

[11] S. Schmidt and V. Schulz, "A 2589 line topology optimization code written for the graphics card," *Comput. Vis. Sci*, 2011.

[12] Y. Yu, T. Hur, J. Jung and I. Jang, "Deep learning for determining a near-optimal topological design without any iteration," *Struct. Multidiscip. Optim.*, 2019.

[13] B. Li, C. Huang, X. Li, S. Zheng and J. Hong, "Non-iterative structural topology optimization using deep learning," *CAD Comput. Aided Des*, 2019.

[14] O. Ronneberger, P. Fischer and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," 2015.

[15] T. Borrvall and J. Petersson, " Large-scale topology optimization in 3D using parallel computing," *Comput. Methods Appl. Mech. Eng*, vol. 190, p. 6201–6229, 2001.

[16] T. Kim, J. Kim and Y. Kim, "Parallelized structural topology optimization for eigenvalue problems," *Int. J. Solids Struct*, pp. 2623-2641, 2004.

7 APPENDIX

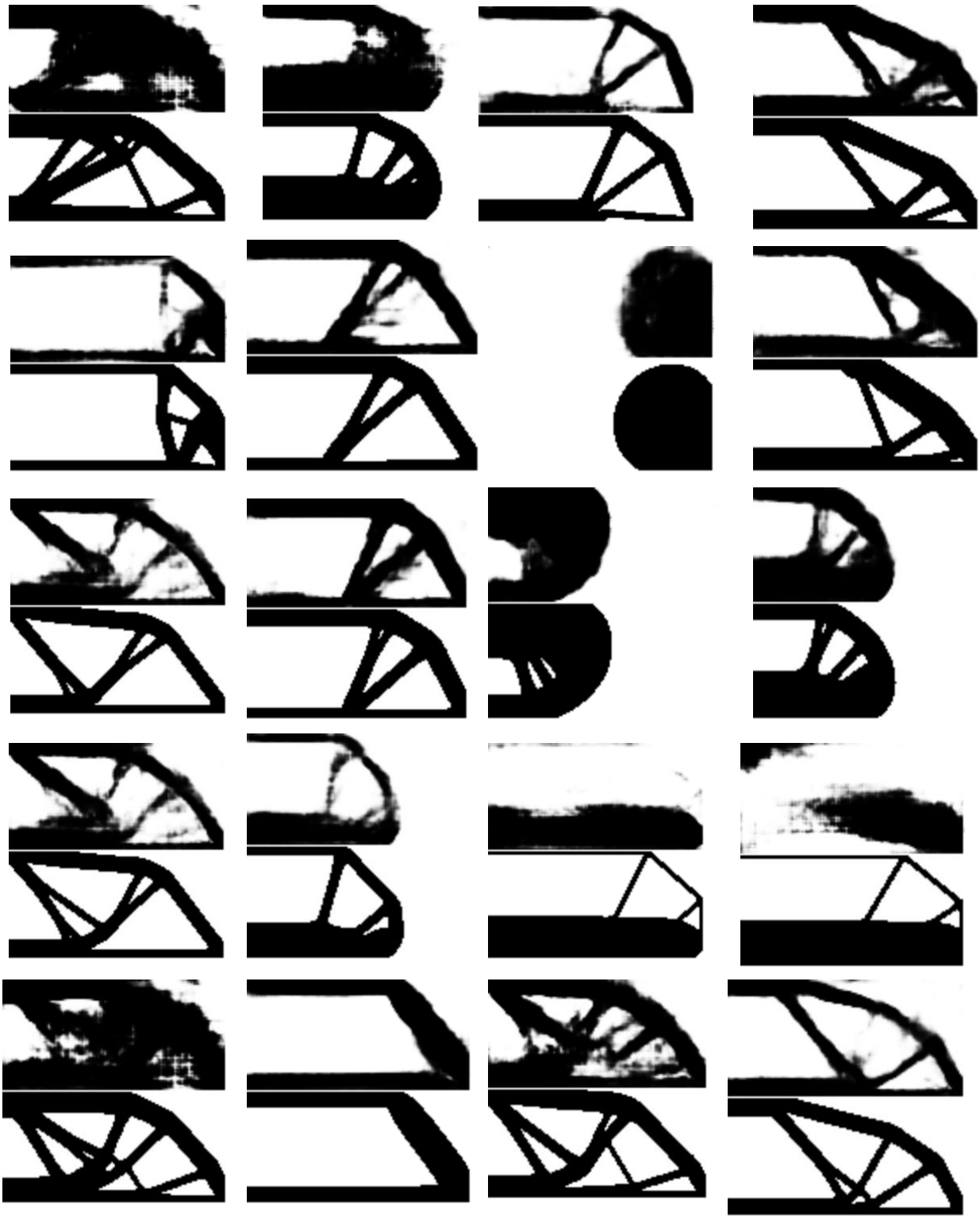

Figure 20. Other Generated Samples by new Hybrid Generator Architecture