Multi-Material Topology Optimization using Neural Networks

Aaditya Chandrasekhar^a, Krishnan Suresh^a

^aUniversity of Wisconsin-Madison

Abstract

The objective of this paper is present a neural network (NN) based multi-material topology optimization (MMTO) method. The primary concept is to use the NN's activation functions to span the unknown material volume fractions, while the weights and bias associated with the NN serve as the design variables, i.e., the representation is thus independent of the finite element mesh. The sensitivities are computed analytically using NN's back-propagation. Then, by relying on the NN's built-in optimization routines, and a conventional finite element solver, the MMTO problem is solved.

The salient features of the proposed framework are: (1) the number of design variables is only weakly dependent on the number of materials considered, (2) it inherently guarantees that material do not get mixed, and (3) it leads to a crisp and differentiable material interface. The proposed framework is simple to implement, and is illustrated through several examples.

Keywords: Multi-material, Topology optimization, Backward propagation, Neural networks, Finite element

1. Introduction

Topology optimization (TO) is a rapidly evolving field, encompassing a rich set of methods including Solid Isotropic Material with Penalization (SIMP) [5], [35], level-set methods [51], evolutionary methods [56] and topological sensitivity methods [41] [10], [24], [25]. The reader is referred to [37] for a critical review.

The main focus of this paper is on multi-material TO (MMTO), where the objective is to not only compute the optimal topology, but also the distribution of two or more materials within the topology. MMTO is partly motivated by the advent of additive manufacturing (AM) [22], and in particular, multi-material AM [13] [49] [3]. Various single-material TO methods, most notably SIMP-based methods, have been extended to multi-materials (see Section 2). Along similar lines, the current work is an extension of a single-material neural-network based TO framework recently proposed in [9] to multi-materials. The proposed framework is discussed in Section 3, followed by the algorithm in Section 3.7. In Section 4, several numerical experiments are carried out to establish the validity, robustness and other characteristics of the method. Open research challenges, opportunities and conclusions are summarized in Section 5.

2. Literature Review

Solid Isotropic Material with Penalization (SIMP) based TO was first extended to MMTO in 1992 [45]. This was followed by [39] where a single variable interpolation scheme for three-phase TO, with extreme thermal

Email addresses: achandrasek3@wisc.edu (Aaditya Chandrasekhar), ksuresh@wisc.edu (Krishnan Suresh)

expansion, was proposed. The concept was extended in [40] to incorporate numerous candidate materials, and has been used to design multi-physics actuators [36], piezo-composites [47], and functionally graded structures with optimal Eigen-frequencies [43]. A common challenge in multi-material TO is that the number of design variables increases with the number of materials. To address this challenge, [58] introduced a peak function concept where multiple materials were represented by one smooth function with several peaks (means). However, the method posed numerical challenges for more than two phases.

The level-set based TO was extended to solve MMTO in [53] where m level-set functions were used to obtain designs with 2^{*m*} materials. This was exploited in [50] towards designing compliant mechanisms, and multi-phase microstructures in [54]. The shape derivatives proposed in [53] were found to be inaccurate unless certain assumptions were made [1]. More accurate shape derivatives were derived in[1], while also retaining a constant interface zone thickness. The concept was extended in [48] to consider discontinuities by sharp interfaces by using multiple intermediate interfaces. A typical challenge in level-sets based MMTO is numerical diffusion across material boundaries.

Independently, the MMTO problem was converted in [52] into a phase transition problem represented by a set of nonlinear PDE, to be solved using Cahn-Hillard phase field method, producing smooth designs without checkerboards. The proposed method was however computational expensive, requiring several iterations to achieve a steady-state solution.

The BESO method was extended to MMTO in [15], where the optimal topology ws determined according to the relative ranking of sensitivities. The optimal solution may require several trials and mesh refinements. The topological sensitivity based TO method was extended to MMTO in [25], where the total mass and compliance were treated as conflicting objectives, thereby leading to the concept of Pareto-optimal multi-material topologies. MMTO in the context of composite fiber optimization was explored in [16]. The method does not safeguard against material mixing, requiring large number of constraints [17]. More recently, the authors of [44] proposed a MMTO method within the framework of isogeometric analysis. Checkerboard free topologies with desired level of continuity was obtained. The authors of [34] observed that the volume constraints from discrete material optimization are linearly separable, and proposed a fast 'ZPR' update scheme.

In conclusion, we have the following observations:

- Almost all single-material TO methods have been extended to MMTO; SIMP based MMTO being the most common.
- Some of the MMTO methods impose a total mass constraint, while others impose constraints on individual material fractions; in general, the former leads to better designs [12], [30].
- In all MMTO methods, the number of design variables scales linearly with the number of materials (exception being [58] mentioned above).
- All MMTO methods use the underlying finite element space to represent the material design variables; thus, the quality of obtained solution depends on the finite element discretization.

In this paper, we propose a SIMP based MMTO method that extends the single-material TO method proposed recently in [9] to multiple material. The main characteristic of this method is that instead of using the finite element space to represent the material variables, we use the functional space of a neural net. The salient features of the proposed MMTO method are:

- The number of design variables is weakly dependent on the number of materials considered, requiring only two additional design variables for every additional material.
- The material interface is smooth and differentiable, and the interface resolution is not limited by the underlying mesh.
- Sensitivities are computed analytically by exploiting the back-propagation capabilities of neural nets.
- Finally, the method inherently guarantees that there is no mixing of materials, i.e., additional constraints are not needed.

3. Proposed Method

Prior to describing the proposed method, the mathematical nomenclature used in the remainder of the paper is summarized below for convenience.

- Ω_0 Design domain
- X Point (x,y) in Ω_0
- *S* Number of non-void materials
- E^i Young's modulus of the *i*th material i = 0, 1, ..., S; $E^0 \approx 0$
- \overline{E} Effective Young's modulus (defined later)
- ρ^i Physical density (unit of kg/m^3 in SI) of the *i*th material; $\rho^0 = 0$
- $v^i(X)$ Volume fraction (unit-less) of i^{th} material at location X
- $v(\mathbf{X})$ Volume fraction vector: { $v^0(\mathbf{X}), v^1(\mathbf{X}), \dots, v^S(\mathbf{X})$ }
- v_e^i Volume fraction of i^{th} material evaluated at the center the e^{th} element
- *J* Compliance at any instance
- m^* Upper limit on allowable mass
- A_e Area of the e^{th} finite element
- *K* Finite element stiffness matrix
- *u* Finite element displacement vector
- *f* Finite element force vector
- *α* Constraint penalty parameter
- $L(\cdot)$ Loss function
- *w* Set of bias and weights associated with the neural network
- ϵ_m Absolute value of mass constraint deviation
- ϵ_g Fraction of gray elements; an element is gray if any of its volume fractions is in the range [0.2, 0.8]

A few points worth noting about the nomenclature are:

- *S* is the number of *non-void* materials; since void is a possible material choice, we have a total of S + 1 materials to choose from.
- The volume fraction of a material is denoted by vⁱ; this is often referred to as pseudo-density in SIMP.
 We have avoided the term pseudo-density in this paper since it can be easily confused with the physical density (mass/volume) that will be needed.
- Quantities such as effective modulus and loss function are defined later in the paper.

3.1. Brief Review of TOuNN

As mentioned earlier, this paper extends the single-material topology optimization using neural networks (TOuNN) framework proposed in [9]. We therefore provide here a brief summary of TOuNN. Consider a typical single-material TO problem illustrated in Figure 1. In SIMP, a volume fraction field v(X) (commonly referred to pseudo-density) is defined over the entire domain Ω_0 . Further, in a typical finite element based SIMP formulation, the (unknown) volume fraction is typically a constant over each element. Then, by relying on standard SIMP penalization technique, the volume fraction is optimized using optimality criteria [6] or MMA [42], say to minimize compliance, resulting in the desired topology. This is illustrated schematically in Figure 3.



Figure 1: Single material TO using SIMP.

On the other hand, in TOuNN, the volume fraction field is captured externally using a neural-network's (NN) activation functions and weights, as described in [9]. In other words, the volume fraction field is independent of the underlying mesh. Then, by relying on SIMP penalization, the volume fraction is optimized using the NN's built-in optimizer, to result in the desired topology; see Figure 2. The benefits of the TOuNN framework include [9]: (1) analytical sensitivities can be computed using NN's back-propagation, and (2) the boundary of the topology is crisp and differentiable since the field v(X) is analytically defined over the entire domain. Other characteristics of TOuNN are discussed in [9].



Figure 2: Single material TO using TOuNN.

3.2. MMTO Problem Definition

Now consider the multi-material problem in Figure 3 where we will assume that a set of S + 1 materials (including void) are specified. For simplicity, we will assume the compliance must be minimized subject to a total mass constraint.



Figure 3: Multi-material topology optimization involves assigning material from (S+1) candidates at each point within the domain.

Let $v^i(X) \in (0,1]$ be the relative volume fraction of material i (i = 0, 1, ..., S) at a point $X \in \Omega_0$. Thus the material composition at any point X is described by a vector $v(X) = \{v^0(X), v^1(X), ..., v^S(X)\}$. Note that the sum of volume fractions must be unity at all points, i.e., $\sum_{i=0}^{S} v^i(X) = 1$. Further, in this paper, we will assume that material mixing is not allowable, i.e., after optimization, only one of the volume fractions must take the value of unity while others must be zero. This is unlike functionally graded materials, where material mixing is allowed [55].

3.3. Representing Volume Fractions via a Neural Network

In classic SIMP based MMTO, the volume fraction composition $v(X) \in \mathbb{R}^{S+1}$ is represented using the underlying finite element mesh. In other words, the design variables are the element volume fractions v_e^i . Consequently, a mass-constrained MMTO problem is posed as:

$$\underset{v_{e}^{i}}{\text{minimize}} \qquad \boldsymbol{u}^{\mathsf{T}}\boldsymbol{K}(v_{e}^{i})\boldsymbol{u} \tag{1a}$$

subject to $K(v_e^i)u = f$ (1b)

$$\sum_{i=0}^{S} \sum_{e} \rho^{i} v_{e}^{i} A_{e} = m^{*}$$

$$(1c)$$

where $K(\cdot)$ is the finite element stiffness matrix, f is the imposed load. Equation 1c is the mass constraint [12]. A_e is the area of element e and m^* is the net mass constraint. Further observe that the number of design variables is the product of the number of materials and the number of finite elements.

In contrast, here the volume fraction fields are defined using a neural network (NN); see Figure 4. The NN has 2 inputs, namely the (x, y) coordinates of any point within the domain, and S + 1 outputs, corresponding to the S + 1 volume fractions at that point. Internally, the NN is a simple fully-connected feed-forward network consisting of a series of hidden layers (depth) each consisting of several neurons (height). Each of the neurons

is associated with an activation function [29] such as rectified linear unit (ReLU). Further, the NN is associated with scalar weights through which one can control the output (see description below). The output layer of the NN is associated with a softMax functions (sometimes called a *classifier* [8]) that is described below.



Figure 4: Neural network architecture used for MM-TOuNN.

To illustrate the construction of the NN, consider a simplified NN (see Figure 5) with only one hidden layer, and two outputs. Observe that each connection within the NN is associated with a weight. The output of any node is computed as follows:

- A weighted sum z₁^[1] is first computed as z₁^[1] = w₁₁^[1]x + w₂₁^[1]y + b₁^[1], where w_{ij}^[k] is the weight associated to the *jth* neuron in layer *k* from the *ith* neuron in the previous layer, and b_j^[k] is the bias associated with the *jth* neuron in the *kth* layer.
- Then, the output $a_1^{[1]}$ of the node is computed as $a_1^{[1]} = \sigma(z_1^{[1]})$ where σ is the chosen activation function. For example, the ReLU activation function is defined as: $\sigma(z) \equiv max(0, z)$. This and other activation functions are supported by various NN software libraries such as pyTorch [28].
- The output of a particular layer is then used as input for the subsequent layer and so on.
- The final output layer, as mentioned earlier, is associated with a special activation function called softMax that generates (in this case) two outputs:

$$v^{0} = \frac{e^{z_{1}^{[2]}}}{e^{z_{1}^{[2]}} + e^{z_{2}^{[2]}}}$$
(2)

$$v^{1} = \frac{e^{z_{2}^{[2]}}}{e^{z_{1}^{[2]}} + e^{z_{2}^{[2]}}}$$
(3)

Observe that, by construction, $0 \le v^0$, $v^1 \le 1$ and $v^0 + v^1 = 1$.

• Thus, once the activation functions are chosen, the fields $v^0(X)$ and $v^1(X)$ are entirely controlled by the weights and bias associated with the NN.



Figure 5: A simplified neural network.

The above observations can be easily generalized to the more complex NN in Figure 4. The major differences are that there are more hidden layers and more neurons per layer. Further, the softMax layer (that can be easily configured) generates S + 1 outputs such that:

$$v^{i} = \frac{e^{z_{i}^{[o]}}}{\sum\limits_{k=0}^{S} e^{z_{k}^{[o]}}}; \quad i = 0, 1, \dots S$$
(4)

As a consequence, we have the following observations:

- (i) **Analytic Fields**: The output fields $v^i(X)$ of the NN are analytically defined at all points X within the domain.
- (ii) **Scaling**: The softMax layer guarantees that $0 \le v^i(\mathbf{X}) \le 1$
- (iii) **Partition of Unity**: The softMax layer also guarantees that $\sum_{i=0}^{S} v^{i}(X) = 1$.
- (iv) **Design Variables**: The fields $v^i(X)$ are controlled by the weights and biases of the network, denoted by w. Thus, we express the unknown fields as v(X; w). The total number of design variables is equal to the number of weights in the hidden layers plus the number of weights in the output later.

With this representation, the optimization problem may be posed as:

$$\min_{w} u^{\mathsf{I}} K(w) u \tag{5a}$$

subject to

$$K(w)u = f \tag{5b}$$

$$\sum_{i=0}^{S} \sum_{e} \rho^{i} v_{e}^{i}(w) A_{e} = m^{*}$$
(5c)

Observe that, unlike in Equation 1, here the design variables are the weights and biases associated with the network.

3.4. Loss Function

We now consider the problem of solving the optimization problem. Neural networks are designed to minimize a loss function using optimization procedures such as Adam procedure [20]. We therefore convert the constrained minimization problem in Equation 5 into a loss function by relying on the penalty formulation [26]. Specifically, the loss function is defined as:

$$L(w) = \frac{u^{T} K u}{J^{0}} + \alpha \left(\frac{1}{m^{*}} \sum_{i=0}^{S} \sum_{e} \rho^{i} v_{e}^{i} A_{e} - 1\right)^{2}$$
(6)

where α is a penalty parameter, and J^0 is the initial compliance, used here for scaling. Note that the FE equality Equation 5b is satisfied automatically when solved for the displacement field. Further, we define the mass constraint deviation as:

$$\epsilon_m := \left| \left(\frac{1}{m^*} \sum_{i=0}^{S} \sum_{e} \rho^i v_e^i A_e - 1 \right) \right| \tag{7}$$

This will be used later on as one of the termination criteria.

As described in [26], starting from a small positive value for the penalty parameter α , a gradient driven step is taken to minimize the loss function. Then, the penalty parameter is increased and the process is repeated; details are provided later in the algorithm. Observe that, in the limit $\alpha \rightarrow \infty$, when the loss function is minimized, the equality constraint is satisfied and the objective is thereby minimized. Other methods such as the augmented Lagrangian [18] may also be used to convert the constrained problem into a loss function. The overall NN-based loss-function overall framework is captured in Figure 6, details of which are described in the remainder of the paper.



Figure 6: Overview of the proposed MM-TOuNN framework.

The effective modulus of elasticity $\overline{E}(X)$ in Figure 6 is defined as:

$$\bar{E}(\boldsymbol{X}) = \sum_{i=0}^{S} (v^{i}(\boldsymbol{X}))^{p} E^{i}$$
(8)

where *p* is the SIMP penalization constant (typically p = 3). The penalization discourages intermediate volume fractions, i.e., discourages material mixing, and the effective modulus is used to construct the stiffness matrix (see [34]) as discussed later. The physical density at any point, on the other hand, is defined as the weighted density of all materials, i.e.,

$$\bar{\rho}(\mathbf{X}) = \sum_{i=0}^{S} v^{i}(\mathbf{X})\rho^{i}$$
(9)

3.5. Finite Element Analysis

For finite element analysis, we discretize the domain using a regular 4 node quad element as illustrated in Figure 6. We then rely on the fast Cholesky factorization based on the CVXOPT library [2] for analysis. Note that the FE solver is treated as a black-box by the NN. During each iteration, the volume fractions $v_e =$ $\{v_e^0, v_e^1, \dots, v_e^S\}$; $\forall e \in \Omega_0^h$ are evaluated the center of each element. This is followed by the computation of the effective modulus of elasticity (see Equation 8). This is then provided to the FE solver that computes the displacement vector u, and, for convenience, the un-scaled compliance associated with each element:

$$J_e = \{u_e\}^T [K]_0 \{u_e\}$$
(10)

Here $[K]_0$ is the stiffness matrix assuming an Young's modulus of unity, and $\{u_e\}$ are the computed displacement vectors corresponding to element *e*. This will be used in sensitivity calculations as explained in the next section. The total compliance is given by:

$$J = \sum_{e} \bar{E}(\mathbf{X}_{e}) J_{e} \tag{11}$$

3.6. Sensitivity Analysis

We now turn our attention to sensitivity analysis, a critical part of any optimization framework. NNs rely on back-propagation [33] to analytically compute the sensitivity of loss functions with respect to the weights and bias [4], [32]. This is possible since the activation functions are analytically defined, and the output can be expressed as a composition of such functions.

Thus, in theory, once the network is defined, no additional work is needed to compute sensitivities; it can be computed automatically (and analytically) via backpropogation! However, in the current scenario, the FE implementation is outside of the NN (see Figure 6). Therefore, we need to compute some of the sensitivity terms explicitly.

Note that the sensitivity of the loss function with respect to a particular design variable w_i is given by:

$$\frac{\partial L}{\partial w_j} = \sum_{i=0}^{S} \sum_{e} \frac{\partial L}{\partial v_e^i} \frac{\partial v_e^i}{\partial w_j} \tag{12}$$

The second term $\frac{\partial v_e^i}{\partial w_j}$ can be computed analytically by the NN through backpropogation since the dependency is entirely part of the NN. On the other hand, the first term involves both the NN and the FE black-box, and

must therefore be explicitly provided. Note that:

$$\frac{\partial L}{\partial v_e^i} = \frac{1}{J^0} \frac{\partial}{\partial v_e^i} (\boldsymbol{u}^\mathsf{T} \boldsymbol{K} \boldsymbol{u}) + \frac{2\alpha \rho^i A_e}{m^*} \left(\frac{\sum\limits_{s=0}^{S} \rho^s \sum\limits_k v_k^s A_k}{m^*} - 1 \right)$$
(13)

Further, recall call that [7]

$$\frac{\partial}{\partial v_e^i} (\boldsymbol{u}^\mathsf{T} \boldsymbol{K} \boldsymbol{u}) = -\{u\}_e^\mathsf{T} \frac{\partial \boldsymbol{K}_e}{\partial v_e^i} \{u\}_e = -\{u\}_e^\mathsf{T} \frac{\partial \bar{E}_e}{\partial v_e^i} [K]_0 \{u\}_e = -p E^i (v_e^i)^{p-1} J_e \tag{14}$$

where J_e is the element-wise un-scaled compliance defined earlier. Thus one can now compute the desired sensitivity as follows:

$$\frac{\partial L}{\partial w_j} = -\frac{p}{J^0} \left[\sum_{s=0}^{S} E^s \left(\sum_k (v_k^s)^{p-1} J_k \frac{\partial v_k^s}{\partial w_j} \right) \right] + \frac{2\alpha}{m^*} \left(\frac{\sum_{s=0}^{S} \rho^s \sum_k v_k^s A_k}{m^*} - 1 \right) \left[\sum_{s=0}^{S} \rho^s \left(\sum_k A_k \frac{\partial v_k^s}{\partial w_j} \right) \right]$$
(15)

3.7. Algorithm

In this section, the proposed framework is described through Algorithm 1 described below:

- We will assume that the NN has been constructed with a desired number of layers, nodes per layer and activation functions. Here, we use ReLU6 activation functions. In this paper, the MM-TOuNN framework is implemented in Python, and pyTorch [28] is used to construct the NN.
- 2. The domain is sampled at the center of each element; this serves as the input to the NN for optimization.
- 3. The penalty parameters for optimization α is initialized with $\alpha_0 = 0.5$. A continuation strategy is adopted for the SIMP penalty parameter *p* [21], [31], [38], with $p_0 = 1.0$.
- 4. The weights and bias of the network are initialized and seeded using Glorot normal initialization [14]. The learning rate for the Adam optimizer is set at the recommended value of 0.01 [20].
- 5. The main iteration starts here.
- 6. The element volume fractions v_e^i are computed by the NN using the current set of weights w.
- 7. The volume fractions are then used to compute the effective modulus of elasticity for each element. These are used in the construction of the element stiffness matrices.
- 8. The computed stiffness matrices, in conjunction with the applied boundary condition are used by the FE solver to solve the structural problem, and to compute the un-scaled element compliances J_e defined in Equation 10.
- 9. Further, in the first iteration, a reference compliance J^0 is also computed for scaling purposes.
- 10. Then the loss function is computed using Equation 6

- 11. The sensitivities are computed using Equation 15.
- 12. Using these sensitivities the weights *w* are then updated using the built-in optimizer (Adam optimizer).
- 13. The penalty parameter α is updated as follows: $\Delta \alpha = 0.15$ with $\alpha_{max} = 100$.
- 14. The SIMP penalty parameter *p* is updated as follows: $\Delta p = 0.01$ with $p_{max} = 4$.
- 15. The process is then repeated until termination.
- 16. Two criteria must be satisfied for the algorithm to terminate. The first constraint is that the mass deviation error ϵ_m (see Equation 7) must be less than a user prescribed value ϵ_m^* . The second constraint is that the fraction of gray elements ϵ_g (see Equation 16) must be less than a user prescribed value ϵ_g^* ; an element is defined to be gray if any of its volume fractions v_e^i lies in the range [0.2, 0.8]; the second constraint ensures that the material/topology distribution has converged.

$$\epsilon_g = \frac{\left|e|\exists i, 0.2 \le v_e^i \le 0.8\right|}{N_e} \tag{16}$$

17. Upon termination, the domain is sampled at a finer resolution to compute the topology with a sharp boundary.

Algo	orithm 1 MM-TOuNN	
1: p	procedure MM-TOUNN(NN, E^i , ρ^i , Ω_0^h , m^*)	▷ NN, material properties, discretized domain, mass
2.	$\mathbf{r}_{a} = \{\mathbf{r}_{a}, \mathbf{u}_{a}\} \text{ob}$	▷ center of elements in FE mesh
2.	$\mathbf{i} = 0: \mathbf{a} = \mathbf{a}_0: \mathbf{n} = \mathbf{n}_0$	▷ Popalty factor initialization
3. 4:	$w \sim Xavier normal$	▷ Initialization of NN weights
5:	repeat	0
6:	$v_e = NN(x_e) \forall e$	\triangleright Call NN to compute v
7:	$ar{E}(oldsymbol{x}_e) = \sum\limits_{i=0}^{S} (v^i(oldsymbol{x}_e))^p E^i orall e$	▷ Effect modulus of elasticity
8:	$J_e \leftarrow FEA(v_e, \Omega_0^h)$	⊳ Solve FEA
9:	if $i == 0$ then $J^{0} = \sum_{i} \overline{E}(\boldsymbol{x}_{e}) J_{e}$ end if	
10:	Compute $L(w)$	▷ Loss function
11:	Compute ∇L	⊳ Sensitivity
12:	$oldsymbol{w} \leftarrow oldsymbol{w} + \Delta oldsymbol{w} (abla L)$	> Adam optimizer step
13:	$\alpha \leftarrow \min(\alpha_{max}, \alpha + \Delta \alpha)$	\triangleright Increment α
14:	$p \leftarrow \min(p_{max}, p + \Delta p)$	▷ Continuation
15:	$i \leftarrow i + 1$	
16:	until $\ \epsilon_g < \epsilon_g^*$ and $\ \epsilon_m \leq \epsilon_m^*$	Check for convergence
17:	$\boldsymbol{X}_{s} = \{X_{s}, Y_{s}\} \in \Omega_{0}^{h}; v_{s} = NN(\boldsymbol{X}_{s})$	Compute volume fractions fields at a fine resolution

4. Numerical Experiments

In this section, we conduct several experiments to illustrate the framework and algorithm. The default parameters are as follows:

• All materials are assumed to exhibit a Poisson ratio of $\nu = 0.3$; Young's moduli and densities are specified for each example.

- A mesh size of 60 × 30 is used for all experiments, unless otherwise stated; the force is assumed to be 1 unit.
- The neural network is composed of 5 hidden layers with 25 nodes (neurons) per layer unless otherwise specified. This corresponds to 1774 design variables. In addition, each of the input and output neurons are associated with two more design variables. Thus the total number of design variables correspond approximately to the number of elements in the default 60 × 30 mesh. The computed gradient clipping were clipped at 0.2 [27].
- The termination criteria are as follows: ε^{*}_g = 0.035; g^{*}_m = 0.05; a maximum of 500 iterations is also imposed.
- After termination, the density in each element is sampled on a 15×15 grid to extract the topology.
- All experiments were conducted on a Intel i7 8700 CPU @ 3.2 Ghz with 32 GB of RAM.

Through the experiments, we investigate the following.

- 1. *Validation*: We validate the MM-TOuNN framework by comparing the obtained topology and compliance with published results in [57] and [59], utilizing up to four materials.
- 2. *Convergence Study*: Typical convergence plots of the loss function, objective and mass constraint are reported and explained.
- 3. *Computational Cost*: The cost to solve the optimization problem as a function of number of candidate materials is studied. The cost division between the various components of the framework is summarized.
- 4. *NN Dependency*: Next, we vary the neural network size (depth and width), and study its impact on the computed topology and resulting compliance, for a specific problem.
- 5. Mesh Dependency: Similarly, we vary the mesh size and study its impact on the topology and compliance.
- 6. *Pareto Designs*: Here we demonstrate how a series of multi-material Pareto-optimal designs can be obtained by gradually reducing the desired mass.
- 7. *Geometric Patterns*: Here we demonstrate how geometric patterns (ex: symmetry and grid patterns) can be imposed in the framework.
- 8. *Post-Processing*: As explained in the algorithm, once the optimization is complement, a high resolution topology can be generated via a fine sampling of the volume fraction fields. We also illustrate how the material interface can be extracted by computing the gradients using the NN.
- 4.1. Validation

For validation, we consider the published results in [57]. First we perform MMTO on a MBB structure in Figure 7 with dimensions 60×30 ; boundary conditions are as illustrated where F = 1. The materials available

are tabulated in Table 1. Depending on the experiment, only a subset of these materials will be allowed. The domain is initialized with the heaviest material, i.e., the initial mass is 1800 units; the desired mass is 900 units.



Figure 7: MBB Structure

Material	Color code	E	ρ
1	Black	2	1
2	Red	1	0.6
3	Blue	0.6	0.4
4	Pink	0.01	0.1

Table 1: Material properties for validation.

For various material choices (from the available set in in Table 1), the computed topologies along with the compliance values (for MM-TOuNN and [57]) are summarized in Figure 8. We observe that the compliance values are comparable, with MM-TOuNN yielding significantly better results in some cases, but the topologies differ in every scenario. This merely reflects the fact that MMTO exhibits numerous local minima [16].



Figure 8: Validation - Comparing obtained topology and compliance with [57]

As a second validation experiment, consider the Michell structure in Figure 9 with dimensions 60×30 , and boundary conditions as illustrated where F = 1. The materials available are tabulated in Table 2. Once again, depending on the experiment, only a subset of these materials will be allowed. The initial mass is 1800 units, and the desired mass is 720 units.



Figure 9: Michell structure.

Material	Color code	Ε	ρ
1	Black	1	1
2	Red	0.6	0.7
3	Blue	0.2	0.4

Table 2: Material properties for validation.

For various material choices (from Table 2), the computed topologies along with the compliance values (for MM-TOuNN, [57] and [59]) are summarized in Figure 10. We observe that MM-TOuNN consistently results in better performing designs. Observe that although the material choice [1,2] is a subset of [1,2,3], two different topologies are obtained; the underlying reasons are unclear.

4.2. Convergence

In this next example, we study the convergence of the proposed algorithm using the mid-cantilever beam in Figure 11 with three materials in Table 3. Note that the loss function in Equation 6 is the sum of the relative



Figure 10: Optimization of Michell beam; Reference (A) is [57] and Reference (B) is [59].

compliance and mass constraint; the convergence of these two quantities is illustrated in Figure 12.



Figure 11: Mid-Cantilever Structure

Material	Color code	Ε	ρ
1	Red	380	19250
2	Cyan	210	7800
3	Black	110	4390

Table 3: Material properties for convergence study.

4.3. Computational Cost

We now discuss typical computational costs in MM-TOuNN. The framework in Figure 6 can be separated into the following components:

- Forward: Computing density values at the sampled points, a.k.a, forward propagation.
- *FEA*: Finite element analysis to compute displacements, compliance, etc.
- Weight-update: Computing the loss function, sensitivity and updating weights through back-propagation.

The objective of this experiment is to quantify the total computational cost, and individual costs for each of these three components, with varying number of materials; all other parameters are at default values. Further, it is well known that NN computations scale well on the GPU. Thus, in addition to reporting computational cost on the CPU, we run the same experiments on a GPU (NVIDIA Quadro GP 100 with 3584 CUDA cores) and report our observations.

The Michell problem in Figure 9 is used for the experiment; a total of 10 non void materials (and one void material), with $E_k = 2 - 0.1k$; k = 1, 2, ... 10 and $\rho_k = 0.5E_k$ are assumed to be available. From this set, the first *K* materials are chosen, and the desired mass fraction is 0.7. The computational time in seconds, and the number of iteration for *K* ranging from 1 to 10 are reported in Table 4. The compliance is almost a constant, i.e., adding additional materials does not improve the performance in this case. But more importantly, the computational cost does not grow with the number of materials; there appears to be no correlation between



Figure 12: Convergence plot: relative compliance and mass constraint, as a function of iteration count.

the two. Finally, the computational cost for the three components (forward propagation, FEA, and weightupdate), when averaged over all the experiments, account for 9%, 74%, and 17% respectively. In other words, FEA dominates the computational effort as expected.

K	1	2	3	4	5	6	7	8	9	10
Compliance	58.1	58.1	59.1	59.5	59.9	59.3	58.9	59.5	58.9	59.5
#Iterations	156	136	146	116	139	135	113	87	86	79
CPU Time (secs)	11.68	10.8	13.3	10.0	13.3	14.9	11.8	7.6	8.1	7.33
GPU Time (secs)	8.93	8.12	10.12	7.56	10.31	10.65	9.01	5.65	6.25	5.49

Table 4: Computational cost as a function of number of materials. for the Michell beam problem.

4.4. Neural Net Dependency

Next we study the impact of the NN size (depth and height) on the tip loaded cantilever problem (see Figure 13) with materials listed in Table 3. All parameters are kept constant while the NN size is varied; the target mass fraction is 0.5. The results are illustrated in Figure 13 where, for example, a NN size of 4×12 implies 4 hidden layers, each consisting of 12 fully connected neurons. Although there is a significant variation in the material distribution and topologies as the NN size is varied, the compliance values remains almost a constant.

4.5. Mesh Dependency

Analogous to the previous experiment, we now vary the mesh size, keeping all other parameters constant. As an example, we consider the Michell structure in Figure 9 with candidate materials in Table 3; the desired mass fraction is 0.5. Figure 14 illustrates the (raw) topologies for different mesh sizes. Once again, while the material distribution depends on the underlying mesh, the compliance values are consistent.



Figure 13: Optimized topologies for the tip-loaded cantilever beam, for varying neural net size.



Figure 14: Optimized topologies for the Michell structure 9, for varying mesh size.

4.6. Weight Initialization vs. Recycling

Next we investigate if it is advantageous to recycle the NN weights while generating a series of Paretooptimal designs [25]. As an example, we consider the mid-cantilever problem posed earlier in Figure 11 with material properties listed in Table 3. The domain is initialized with the heaviest (strongest) material and the NN weights are set via Global normal initialization [14]. Then the design is optimized for a desired mass fraction of 0.9; see Figure 15. From here on, we consider two different strategies: (A) In the first strategy, the weights from the previous optimization study are retained, i.e., recycled, and the next Pareto-optimal design for mass fraction of 0.8 is computed. (B) In the second strategy, the NN are reset via Global normal initialization (i.e., not recycled) and the Pareto curve is traced. The resulting set of designs for the two strategies are illustrated in Figure 15. We observe that the designs obtained *without* recycling perform sightly better than designs with recycling. Further, there was no computational advantage in recycling the weights, i.e., it is *not* advantageous to recycle the weights.

4.7. Imposing Constraints

Constraints such as symmetry are often imposed in TO via projection methods [46]. Similarly, in the proposed framework, we introduce input and output projection operators as illustrated in Figure 16.

For example, to enforce symmetry about, say, the x axis passing through (x_0, y_0) , the input y coordinate is



Figure 15: Pareto-optimal designs generated by gradually decreasing the desired mass fraction.



Figure 16: Constraints are imposed using input and/or output projections.

transformed via an input projection as follows:

$$y \leftarrow y_0 + |y - y_0| \tag{17}$$

As an example, Figure 17 illustrates imposing different types of symmetry for a tip-loaded cantilever beam, with material properties listed in Table 3. As expected, imposition of symmetry leads to a decrease in performance.

Next, we consider a constraint where a certain material is prescribed within a region Ω_N . To facilitate backward propagation, i.e., sensitivity analysis, we treat this as an output projection via:

$$v^{(i)}(\mathbf{X}) = \begin{cases} v^{(i)}(\mathbf{X}) + |1 - v^{(i)}(\mathbf{X})| & i = M \\ v^{(i)}(\mathbf{X}) - |v^{(i)}(\mathbf{X}) - \epsilon| & i \neq M \end{cases}, \quad \mathbf{X} \in \Omega$$
(18)

As an example, Figure 18a illustrates the Michell beam problem with material-3 is prescribed in an annular region as shown (with material properties listed in Table 3). Symmetry with respect to y axis is also prescribed. The computed topology is illustrated in Figure 18b. Imposing this constraint results in a 14% increase in



Figure 17: Imposing symmetry on a tip loaded cantilever beam for $m_f^* = 0.4$

compliance, compared to the design without this constraint in Figure ??.



Figure 18: Symmetry about Y-axis (input projection) along with a fixed annular region (output projection).

4.8. Post-Processing

As mentioned earlier, once the optimization is completed, due to the global representation of the activation functions, one can extract a high resolution boundary at no additional cost. Specifically, the optimized weights, i.e., w^* , are used to sample the volume fraction fields at a high resolution as illustrated in Figure 19.

Finally, the gradient of the underlying fields can be easily and accurately computed in NN through backpropagation. Figure 20 illustrates an optimized Michell structure and the gradients of the three material fields. The non-zero values of these gradients correspond to the interface-boundary.

5. Conclusion

The main contribution of this paper is a novel neural-network based multi-material topology optimization method. The method was validated by comparing it against published research. Some of the merits of the proposed method, including crisp material interfaces and reduced number of design variables, were also demonstrated. However, only a simple volume-constrained compliance minimization was considered in this paper; extensions to handle other objectives with multiple constraints are currently being explored; see [19], [23].



Figure 19: High resolution boundary is extracted by first optimizing, and then using the optimized weights to sample the density at a high resolution.



Figure 20: High resolution boundary and boundary gradient for Michell structure

Checkerboard patterns commonly occur in SIMP based topology optimization [11], [?], and restriction methods such as filtering, perimeter control, etc. are often used in this context [38]. However, in the current formulation, no checkerboard patterns were observed in any of the experiments. A possible reason is the use of global activation functions, but further investigation is needed.

One of the drawbacks in the framework is its tendency to converge to a local optima when a large number of candidate materials are used; this was observed, for example, in Figure 10. Continuation strategies can perhaps alleviate this problem. Insightful understanding of the effect of simulation parameters like mesh size, NN size, activation functions are desirable in making the presented formulation more accessible.

6. Replication of results

The Python code used in generating the examples in this paper is available at www.ersl.wisc.edu/software/MM-TOuNN.zip.

Acknowledgments

The authors would like to thank the support of National Science Foundation through grant CMMI 1561899. Prof. Krishnan Suresh is a consulting Chief Scientific Officer of SciArt, Corp.

Compliance with ethical standards

The authors declare that they have no conflict of interest.

References

- [1] Allaire, G., Dapogny, C., Delgado, G., and Michailidis, G. (2014). Multi-phase structural optimization via a level set method. *ESAIM Control, Optimisation and Calculus of Variations*, 20(2):576–611.
- [2] Andreassen, E., Clausen, A., Schevenels, M., Lazarov, B. S., and Sigmund, O. (2011). Efficient topology optimization in MATLAB using 88 lines of code. *Structural and Multidisciplinary Optimization*, 43(1):1–16.
- [3] Bandyopadhyay, A. and Heer, B. (2018). Additive manufacturing of multi-material structures.
- [4] Baydin, A. G., Pearlmutter, B. A., Radul, A. A., and Siskind, J. M. (2017). Automatic differentiation in machine learning: a survey. *The Journal of Machine Learning Research*, 18(1):5595–5637.
- [5] Bendsoe, M. P., Guedes, J. M., Haber, R. B., Pedersen, P., and Taylor, J. E. (2008). An Analytical Model to Predict Optimal Material Properties in the Context of Optimal Structural Design. *Journal of Applied Mechanics*, 61(4):930.
- [6] Bendsøe, M. P. and Sigmund, O. (1995). *Optimization of structural topology, shape, and material*, volume 414. Springer.
- [7] Bendsoe, M. P. and Sigmund, O. (2013). *Topology optimization: theory, methods, and applications*. Springer Science & Business Media.
- [8] Bridle, J. S. (1990). Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In *Neurocomputing*, pages 227–236. Springer.
- [9] Chandrasekhar, A. and Suresh, K. (2020). TOuNN: Topology optimization using neural networks. *Structural and Multidisciplinary Optimization (accepted; pre-print available at www.ersl.wisc.edu).*
- [10] Deng, S. and Suresh, K. (2015). Multi-constrained topology optimization via the topological sensitivity. *Structural and Multidisciplinary Optimization*, 51(5):987–1001.
- [11] Díaz, A. and Sigmund, O. (1995). Checkerboard patterns in layout optimization. *Structural Optimization*, 10(1):40–45.
- [12] Gao, T. and Zhang, W. (2011). A mass constraint formulation for structural topology optimization with multiphase materials. *International Journal for Numerical Methods in Engineering*, 88(8):774–796.
- [13] Gaynor, A. T., Meisel, N. A., Williams, C. B., and Guest, J. K. (2014). Multiple-material topology optimization of compliant mechanisms created via polyjet three-dimensional printing. *Journal of Manufacturing Science and Engineering*, 136(6).
- [14] Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249– 256.

- [15] Huang, X. and Xie, Y. M. (2009). Bi-directional evolutionary topology optimization of continuum structures with one or multiple materials. *Computational Mechanics*, 43(3):393–401.
- [16] Hvejsel, C. F. and Lund, E. (2011). Material interpolation schemes for unified topology and multi-material optimization. *Structural and Multidisciplinary Optimization*, 43(6):811–825.
- [17] Hvejsel, C. F., Lund, E., and Stolpe, M. (2011). Optimization strategies for discrete multi-material stiffness optimization. *Structural and Multidisciplinary Optimization*, 44(2):149–163.
- [18] Kervadec, H., Dolz, J., Yuan, J., Desrosiers, C., Granger, E., and Ayed, I. B. (2019a). Constrained deep networks: Lagrangian optimization via log-barrier extensions. *CoRR*, *abs/1904.04205*, 2(3):4.
- [19] Kervadec, H., Dolz, J., Yuan, J., Desrosiers, C., Granger, E., and Ayed, I. B. (2019b). Constrained deep networks: Lagrangian optimization via log-barrier extensions. *arXiv preprint arXiv*:1904.04205.
- [20] Kingma, D. P. and Ba, J. L. (2015). Adam: A method for stochastic optimization. In 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings. International Conference on Learning Representations, ICLR.
- [21] Li, L. and Khandelwal, K. (2015). Volume preserving projection filters and continuation methods in topology optimization. *Engineering Structures*, 85:144–161.
- [22] Liu, J., Gaynor, A. T., Chen, S., Kang, Z., Suresh, K., Takezawa, A., Li, L., Kato, J., Tang, J., Wang, C. C., et al. (2018). Current and future trends in topology optimization for additive manufacturing. *Structural and Multidisciplinary Optimization*, pages 1–27.
- [23] Márquez-Neila, P., Salzmann, M., and Fua, P. (2017). Imposing hard constraints on deep networks: Promises and limitations. *arXiv preprint arXiv:1706.02025*.
- [24] Mirzendehdel, A. M., Rankouhi, B., and Suresh, K. (2018). Strength-based topology optimization for anisotropic parts. *Additive Manufacturing*, 19:104–113.
- [25] Mirzendehdel, A. M. and Suresh, K. (2015). A pareto-optimal approach to multimaterial topology optimization. *Journal of Mechanical Design*, 137(10):101701.
- [26] Nocedal, J. and Wright, S. (2006). Numerical optimization. Springer Science & Business Media.
- [27] Pascanu, R., Mikolov, T., and Bengio, Y. (2012). Understanding the exploding gradient problem. *CoRR*, abs/1211.5063.
- [28] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems* 32, pages 8024–8035. Curran Associates, Inc.

- [29] Ramachandran, P., Zoph, B., and Le, Q. V. (2017). Searching for activation functions. *CoRR*, abs/1710.05941.
- [30] Ramani, A. (2010). A pseudo-sensitivity based discrete-variable approach to structural topology optimization with multiple materials. *Structural and Multidisciplinary Optimization*, 41(6):913–934.
- [31] Rojas-Labanda, S. and Stolpe, M. (2015). Automatic penalty continuation in structural topology optimization. *Structural and Multidisciplinary Optimization*, 52(6):1205–1221.
- [32] Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.
- [33] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.
- [34] Sanders, E. D., Aguiló, M. A., and Paulino, G. H. (2018). Multi-material continuum topology optimization with arbitrary volume and mass constraints. *Computer Methods in Applied Mechanics and Engineering*, 340:798–823.
- [35] Sigmund, O. (2001a). A 99 line topology optimization code written in matlab. *Structural and multidisciplinary optimization*, 21(2):120–127.
- [36] Sigmund, O. (2001b). Design of multiphysics actuators using topology optimization Part II: Two-material structures. *Computer Methods in Applied Mechanics and Engineering*, 190(49-50):6605–6627.
- [37] Sigmund, O. and Maute, K. (2013). Topology optimization approaches. *Structural and Multidisciplinary Optimization*, 48(6):1031–1055.
- [38] Sigmund, O. and Petersson, J. (1998). Numerical instabilities in topology optimization: A survey on procedures dealing with checkerboards, mesh-dependencies and local minima. *Structural Optimization*, 16(1):68– 75.
- [39] Sigmund, O. and Torquato, S. (1997). Design of materials with extreme thermal expansion using a threephase topology optimization method. *Journal of the Mechanics and Physics of Solids*, 45(6):1037–1067.
- [40] Stegmann, J. and Lund, E. (2005). Discrete material optimization of general composite shell structures. *International Journal for Numerical Methods in Engineering*, 62(14):2009–2027.
- [41] Suresh, K. (2013). Efficient generation of large-scale pareto-optimal topologies. *Structural and Multidisci*plinary Optimization, 47(1):49–61.
- [42] Svanberg, K. (1987). The method of moving asymptotes—a new method for structural optimization. International journal for numerical methods in engineering, 24(2):359–373.

- [43] Taheri, A. H. and Hassani, B. (2014). Simultaneous isogeometrical shape and material design of functionally graded structures for optimal eigenfrequencies. *Computer Methods in Applied Mechanics and Engineering*, 277:46–80.
- [44] Taheri, A. H. and Suresh, K. (2017). An isogeometric approach to topology optimization of multi-material and functionally graded structures. *International Journal for Numerical Methods in Engineering*, 109(5):668–696.
- [45] Thomsen, J. (1992). Topology optimization of structures composed of one or two materials. *Structural optimization*, 5(1-2):108–115.
- [46] Vatanabe, S. L., Lippi, T. N., Lima, C. R., Paulino, G. H., and Silva, E. C. (2016). Topology optimization with manufacturing constraints: A unified projection-based approach. *Advances in Engineering Software*, 100:97–112.
- [47] Vatanabe, S. L., Paulino, G. H., and Silva, E. C. (2013). Design of functionally graded piezocomposites using topology optimization and homogenization - Toward effective energy harvesting materials. *Computer Methods in Applied Mechanics and Engineering*, 266:205–218.
- [48] Vermaak, N., Michailidis, G., Parry, G., Estevez, R., Allaire, G., and Bréchet, Y. (2014). Material interface effects on the topology optimization multi-phase structures using a level set method. *Structural and Multidisciplinary Optimization*, 50(4):623–644.
- [49] Vidimče, K., Wang, S.-P., Ragan-Kelley, J., and Matusik, W. (2013). Openfab: a programmable pipeline for multi-material fabrication. ACM Transactions on Graphics (TOG), 32(4):1–12.
- [50] Wang, M. Y., Chen, S., Wang, X., and Mei, Y. (2005). Design of Multimaterial Compliant Mechanisms Using Level-Set Methods. *Journal of Mechanical Design*, 127(5):941–956.
- [51] Wang, M. Y., Wang, X., and Guo, D. (2003). A level set method for structural topology optimization. *Computer Methods in Applied Mechanics and Engineering*, 192(1-2):227–246.
- [52] Wang, M. Y. and Zhou, S. (2004). Synthesis of shape and topology of multi-material structures with a phase-field method. *Journal of Computer-Aided Materials Design*, 11(2-3):117–138.
- [53] Wang, X., Mei, Y., and Wang, M. Y. (2004a). Level-set method for design of multi-phase elastic and thermoelastic materials. *International Journal of Mechanics and Materials in Design*, 1(3):213–239.
- [54] Wang, X., Mei, Y., and Wang, M. Y. (2004b). Level-set method for design of multi-phase elastic and thermoelastic materials. *International Journal of Mechanics and Materials in Design*, 1(3):213–239.
- [55] Xia, Q. and Wang, M. Y. (2008). Simultaneous optimization of the material properties and the topology of functionally graded structures. *Computer-Aided Design*, 40(6):660–675.
- [56] Xie, Y. M. and Steven, G. P. (1993). A simple evolutionary procedure for structural optimization. *Computers & structures*, 49(5):885–896.

- [57] Yang, X. and Li, M. (2018). Discrete multi-material topology optimization under total mass constraint. *CAD Computer Aided Design*, 102:182–192.
- [58] Yin, L. and Ananthasuresh, G. K. (2001). Topology optimization of compliant mechanisms with multiple materials using a peak function material interpolation scheme. *Structural and Multidisciplinary Optimization*, 23(1):49–62.
- [59] Zuo, W. and Saitou, K. (2017). Multi-material topology optimization using ordered SIMP interpolation. Structural and Multidisciplinary Optimization, 55(2):477–491.