UC Irvine
UC Irvine Previously Published Works

Title

Symmetry-enforcing neural networks with applications to constitutive modeling

Permalink

https://escholarship.org/uc/item/652279v3

Authors

Garanger, Kévin

Kraus, Julie

Rimoli, Julian J

Publication Date

2024-09-01

DOI

10.1016/j.eml.2024.102188

Copyright Information

Peer reviewed

Symmetry-enforcing neural networks with applications to constitutive modeling

Kévin Garanger[a], Julie Kraus[b], Julian J. Rimoli[a,*]

[a]*Department of Mechanical and Aerospace Engineering, University of California, Irvine, CA 92697, USA*
[b]*School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA*

Abstract

The use of machine learning techniques to homogenize the effective behavior of arbitrary microstructures has been shown to be not only efficient but also accurate. In a recent work, we demonstrated how to combine state-of-the-art micromechanical modeling and advanced machine learning techniques to homogenize complex microstructures exhibiting non-linear and history dependent behaviors [44]. The resulting homogenized model, termed smart constitutive law (SCL), enables the adoption of microstructurally informed constitutive laws into finite element solvers at a fraction of the computational cost required by traditional concurrent multiscale approaches. In this work, the capabilities of SCLs are expanded via the introduction of a novel methodology that enforces material symmetries at the neuron level, applicable across various neural network architectures. This approach utilizes tensor-based features in neural networks, facilitating the concise and accurate representation of symmetry-preserving operations, and is general enough to be extend to problems beyond constitutive modeling. Details on the construction of these tensor-based neural networks and their application in learning constitutive laws are presented for both elastic and inelastic materials. The superiority of this approach over traditional neural networks is demonstrated in scenarios with limited data and strong symmetries, through comprehensive testing on various materials, including isotropic neo-Hookean materials and tensegrity lattice metamaterials. This work is concluded by a discussion on the potential of this methodology to discover symmetry bases in materials and by an outline of future research directions.

1. Introduction

The determination of the effective mechanical properties of materials is a vast topic that has historically been essentially based on experimental methods [13],

---

*Corresponding author
   Email addresses:* `kevin.garanger@uci.edu` (Kévin Garanger), `jkraus@gatech.edu` (Julie Kraus), `jrimoli@uci.edu` (Julian J. Rimoli)

coupled with constitutive models specifically conceived for the type of material studied [49]. These constitutive models are fundamental to engineering-level analysis of the mechanics of materials, via their use in finite element methods [14]. Traditionally, they have been grouped into two categories — phenomenological and micro-mechanical —, although the distinction between the two is not always clearly defined. The former type of models tends to rely on empirical considerations and observations, while the latter type is based on physical considerations of the material behavior. What makes the distinction between the two types sometimes ambiguous is the fact that to be of practical use, even micro-mechanical models need to rely on some assumptions or simplifications. Some examples of phenomenological models include [39, 34, 17, 37] while examples of micro-mechanical models are given in [28, 31, 57, 16, 3].

A different approach consists of using computational modeling of the material behavior at the microscale. Various methods fall under this approach, such as computational homogenization, which is based on the principle of separation of scales [27, 62, 41, 25], and concurrent multiscale methods, where a strong coupling between scales is considered [9, 22, 63, 10]. A major drawback of these approaches is their computational cost, which tends to be prohibitively high for most three-dimensional problems [25], preventing their widespread use in engineering applications.

With the hope of benefiting from the efficiency of constitutive modeling and the accuracy of computational microscale modeling, data-driven methods based on machine learning have been proposed [2, 38, 33, 45, 59, 44, 35]. With these methods, mathematical models capable of approximating complex relationships, such as artificial neural networks, are trained from data generated by computational simulations. The use of very general and powerful models is justified by the need to capture intricate behaviors of materials that would be otherwise hard to efficiently model accurately from physical considerations only. Conceptually, this approach differs significantly from the derivation of a constitutive model based on physics or empiricism that already captures some characteristics of the material response by nature. This stark difference remains true even if some of these latter constitutive models are calibrated on data samples with methods inspired from machine learning, such as in [50].

Artificial neural networks were proposed decades ago to model the mechanical response of materials such as concrete [26], sand [19], or sandstone [47], although these initial approach relied on a limited amount of experimental data to calibrate relatively small networks. Subsequent early works revolving around neural networks for constitutive modelings include [24, 40, 54]. More recently, thanks to advances in micro-structure simulation and the increasing availability of computational resources, larger and more powerful models have been used to learn constitutive laws from simulated data for elastic [38, 45, 35] and inelastic materials [59, 44, 64]. In these works, constitutive models are learned by training neural networks with standard deep learning methods on pairs of inputs and outputs, which consist respectively of kinematic variables (e.g., strain tensor or deformation gradient) and stress variables.

Despite these promising results, one of the disadvantages of neural networks

for constitutive modeling is the fact that some of the properties that the material is known to satisfy, from mathematical or physical considerations, cannot in general be enforced. These properties notably include hyperelasticity, frame indifference (or objectivity), material stability, invariance to material symmetries, and compliance with thermodynamics laws [46]. This drawback of neural networks is linked to their inherent ability to approximate arbitrary functions, and therefore can be seen as an inevitable consequence of this otherwise powerful and desirable feature. To put in perspective this tension between the capability of a model to replicate intricate behavior and the intrinsic enforcement of known properties, micro-mechanical models can be thought as belonging to the other end of the spectrum, being derived from physical considerations but not always able to captures all aspects of a material's response.

In theory, a neural network large enough should be able to learn approximately the previously mentioned properties with sufficient data, but doing so may require an excessively large number of training samples and computational resources [12]. For material symmetry and frame indifference, the difficulty with obtaining a dataset large enough to implicitly represent these properties is partially alleviated by the use of data augmentation techniques. With these techniques, new samples are generated from existing ones by applying a transformation that preserves the invariant properties of interest [55]. However, data augmentation still does not solve the issue of computational cost and is not always applicable, as with the case of material symmetry if the basis of the material symmetry group is not known. In addition, the approach consisting of learning some properties from a dataset may not be acceptable if these properties needs to be satisfied exactly, as such approximations may result in an inconsistent model that violates some laws of physics. For example, the non-respect of frame indifference, itself a purely mathematical property, leads to a different work produced along a path depending on the frame of reference.

For these reasons, neural networks that are designed to satisfy these properties by construction have been proposed [43, 21, 20, 60, 4, 36]. In addition to alleviate the already mentioned issues with data augmentation, neural networks that are designed to satisfy some of these properties tend to also perform better and require smaller models [43, 48]. A common approach to guarantee hyperelasticity is to learn a strain energy function with a neural network potential [21, 20, 4, 36], rather than a strain-stress relationship, similarly to what has been done in molecular chemistry [7]. The frame indifference problem is generally solved by using an input representation that is independent of the frame of reference, such as the Cauchy-Green strain tensor [21, 20] or Green-Lagrange strain tensor [4]. However, the use of these tensors as inputs is incompatible with the enforcement of material stability via input convex neural networks, which is a recent method that requires the deformation gradient as input [36]. Finally, material symmetries have been enforced in previous works with one of two methods:

1. By using invariants of the strain tensor as inputs [43, 36, 42],
2. By using a process known as group symmetrization, where all the trans-

3

formed versions of an input with respect to a symmetry group are passed through a neural network and averaged [21, 20, 36].

One of the advantages of the first method compared to the second one is that the basis of the materials symmetry group does not need to be known. It can also be applied to infinite symmetry group, such as transverse isotropy [20]. The second method can be adapted to infinite groups by considering only a finite subgroup of the materials symmetries, but this results in the material symmetries being only approximately satisfied by the trained constitutive law. In addition, the larger the symmetry group, the more neural network outputs need to be computed with the second method, making it computationally more expensive than the first one for large groups. On the other hand, the first method requires identifying a set of invariants that is suited to the problem, which can be challenging, given that many different invariants exist [56, 53, 6, 18] and not all of them are appropriate for numerical computations [58]. What's more, there also seems to be cases where learning constitutive laws from invariants fails, unlike when learning from the full strain tensor [36].

In this work, a new method to enforce materials symmetry is proposed, in which the materials symmetries are exactly enforced at the neuron level, independently of the neural network architecture. This method relies on linear operations and activations that are equivariant to the action of the material symmetries. With traditional neural networks, features describe each individual scalar entry of a network's inputs and outputs, as well as the hidden layers' outputs. With the proposed approach, this concept is extended to tensors, by considering features represented by tensors. While the inputs and outputs of the networks introduced here are each individual tensors (a strain tensor and a stress tensor), the hidden layers' outputs are represented as arrays of tensors. This formalism does not change in essence the nature of the linear operations occurring between layers, but it allows to conveniently write symmetry-preserving operations in a concise and straightforward way, while reusing known results from linear elasticity applied to materials with symmetries. However, the proposed activation functions, since they are applied feature-wise, are not equivalent to activation layers of standard neural networks, which are applied to each scalar individually. The methodology to construct neural networks based on tensor features are detailed in section 2. The resulting neural networks are named *tensor feature equivariant neural networks* (TFENN). Although the examples presented in this work come from continuum mechanics, the proposed methodology is general and can be applied to any problem where the inputs and outputs can be represented as collections of symmetric tensors.

Both feed-forward and recurrent TFENNs are presented, allowing the learning of constitutive models from elastic and inelastic materials. To apply the proposed method, a basis of the symmetry must be known, which at first may seem like a drawback in comparison to invariants-based methods. However, this requirement can be turned into an advantage by considering a symmetry basis as a learnable parameter of the neural network, therefore allowing its discovery through training. Only rate-independent problems are considered in this

work, but the methodology is easily extended to viscoplasticity, for which the formulation of material symmetries is essentially the same. In section 3, the proposed methodology is applied to the learning of constitutive laws for several materials: a three-dimensional isotropic neo-Hookean material, the unit cell of a two-dimensional tensegrity lattice metamaterial, and a two-dimensional plastic microstructure. Results indicate that TFENNs are capable of learning complex constitutive models despite the apparently restrictive constraints placed on them. In addition, they tend to outperform standard neural networks with similar numbers of parameters and same numbers of layers, with the most significant improvements being for scenarios with scarce data and strong symmetries. The ability of standard neural networks to learn a dataset's symmetries is also investigated via numerical comparisons with TFENNs. An example of symmetry basis discovery is also presented, based on the tensegrity lattice metamaterial dataset. In addition, some of challenges and limitations of the proposed methodology are discussed. Finally, in section 4, main conclusions of the study are provided, as well as potential future research directions.

Notations

The following notations are used throughout the paper:

- $\mathbb{R}$: the set of real numbers.

- $[\![i, j]\!]$: the set of integers $\{i, i+1, \ldots, j\}$.

- S^n: the set of symmetric matrices of size $n \times n$.

- $\mathcal{O}(n)$: the set of orthogonal matrices of size $n \times n$.

- $\mathcal{T}_n^d$: the set of real tensors of order d and dimensions all equal to n.

- $\mathrm{diag}(v)$: the diagonal matrix with diagonal entries given by the vector v.

- $\mathrm{tr}(A)$: the trace of the matrix A.

- $\boldsymbol{I}_n$: the identity matrix of size n.

- $\boldsymbol{A} : \boldsymbol{B}$: the double contraction between two tensors of order at least 2 $\boldsymbol{A}$ and $\boldsymbol{B}$.

2. Neural networks with tensor features

2.1. Equivariant feedforward neural networks

TFENNs are meant to learn a constitutive relationship f between a pair of second-order tensors of same size, the first one being a kinematic input and the second one a stress output. In particular, since the primary focus of this work is the enforcement of material symmetries, the input and output tensors are assumed to be symmetric tensors $\boldsymbol{M} \in \mathrm{S}^d$ that, for any matrix $\boldsymbol{Q} \in \mathcal{O}(d)$, are

5

transformed via the group action: $\boldsymbol{M} \longmapsto \boldsymbol{Q}^{\mathrm{T}} \boldsymbol{M} \boldsymbol{Q}$, where d is the appropriate dimension. The Cauchy-Green strain tensor $\boldsymbol{C}$ and the second Kirchoff-Piola stress tensor $\boldsymbol{S}$, which are work conjugates up to a multiplicative constant, are an example of a pair of such tensors. For a history-independent material, using these tensors as input and output to a neural network allows to formulate the material symmetries of the constitutive relationship as

$$\forall \boldsymbol{Q} \in \mathcal{G}, \forall \boldsymbol{C} \in \mathrm{S}^d, \quad f(\boldsymbol{Q}^{\mathrm{T}} \boldsymbol{C} \boldsymbol{Q}) = \boldsymbol{Q}^{\mathrm{T}} f(\boldsymbol{C}) \boldsymbol{Q}, \tag{1}$$

where $\mathcal{G}$ is the group of material symmetries, a subgroup of $\mathcal{O}(d)$. If f satisfies eq. (1), it is said to be $\mathcal{G}$-equivariant, which is the property that is sought of TFENNs. To construct such neural networks, symmetric tensors are used to represent intermediate features resulting from internal operations of the network (see fig. 1), and said operations are designed to be $\mathcal{G}$-equivariant themselves.
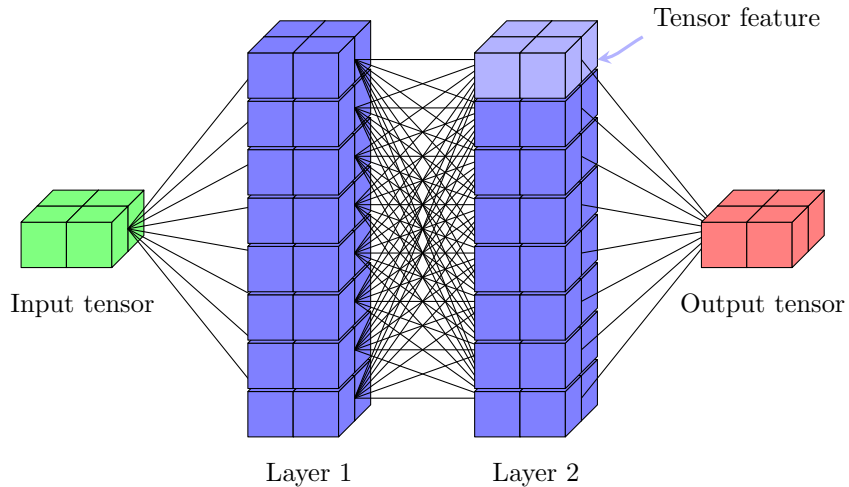
2.2. Neuron operations

Figure 1: Representation of a TFENN with two hidden layers of eight two-dimensional tensor features each. Each connection between two features is associated to a weight tensor and each feature is associated with a bias tensor. The weights of the connections leading to a given feature, the bias tensor of this feature, and the activation at the corresponding layer fully characterize a neuron operation.

At each layer of the proposed neural networks, a set of neuron operations is applied to a set of symmetric tensor features to produce a new set of symmetric tensor features. Assuming that a layer has a number of input (respectively output) features n_{in} (respectively n_{out}), and that the input (respectively output) features are denoted by $\boldsymbol{x}_j$ (respectively $\boldsymbol{h}_i$), the neuron operations is expressed as

$$\forall i \in [\![1, n_{\mathrm{out}}]\!], \quad \boldsymbol{h}_i = \phi \left(\sum_{j=1}^{n_{\mathrm{in}}} \boldsymbol{W}_{ij} : \boldsymbol{x}_j + \boldsymbol{b}_i \right). \tag{2}$$

6

In eq. (2), $\boldsymbol{W}_{ij} \in \mathcal{T}_d^4$ is a *weight* tensor with minor symmetries, $\boldsymbol{b}_i \in \mathrm{S}^d$ is a *bias* tensor, and $\phi : \mathrm{S}^d \longmapsto \mathrm{S}^d$ is a *tensorial activation function*. To ensure that the neuron operations are equivariant, each weight $\boldsymbol{W}$, each bias $\boldsymbol{b}$, and the activation ϕ are chosen such that

$$\forall \boldsymbol{Q} \in \mathcal{G}, \forall \boldsymbol{x} \in \mathrm{S}^d, \quad \begin{cases} \boldsymbol{W} : \left(\boldsymbol{Q}^{\mathrm{T}} \boldsymbol{x} \boldsymbol{Q}\right) = \boldsymbol{Q}^{\mathrm{T}} \left(\boldsymbol{W} : \boldsymbol{x}\right) \boldsymbol{Q}, & \text{(3a)} \\ \boldsymbol{b} = \boldsymbol{Q}^{\mathrm{T}} \boldsymbol{b} \boldsymbol{Q}, & \text{(3b)} \\ \phi(\boldsymbol{Q}^{\mathrm{T}} \boldsymbol{x} \boldsymbol{Q}) = \boldsymbol{Q}^{\mathrm{T}} \phi(\boldsymbol{x}) \boldsymbol{Q}. & \text{(3c)} \end{cases}$$

Clearly, eqs. (3a) to (3c) are sufficient conditions for the equivariance of the neural network resulting from the composition of multiple layers.

2.2.1. Weights

The form that $\boldsymbol{W}$ must take to satisfy eq. (3a) given $\mathcal{G}$ is a well-studied problem arising from the characterization of the elasticity tensor of linear elastic materials with symmetries. It was shown that the number of different equivalence classes for material group symmetries in two dimensions is 4 and 6 for hyperelastic and elastic materials respectively [29], and 8 [23] and 12 [61] in three dimensions. The tensor forms corresponding to each of these classes in the natural basis of symmetry are given in multiple references, for example in [5] for the 2D case and [1] for the 3D case.

2.2.2. Biases

If $\mathcal{G}$ is the trivial group, i.e. triclinic symmetry or no symmetry is assumed, any symmetric tensor $\boldsymbol{b}$ can be used to satisfy eq. (3b). Otherwise, the form of $\boldsymbol{b}$ is more constrained and but can be determined for each symmetry group easily by solving a set of linear systems. For example, with isotropic or cubic symmetry, $\boldsymbol{b}$ must be a multiple of the identity, while with orthotropic symmetry and $d = 2$, it must be diagonal.

2.2.3. Activations

Finally, to satisfy eq. (3c), the approach proposed in this work is to define ϕ such that it applies a *scalar* activation function σ to the eigenvalues $\boldsymbol{\lambda}$ of its argument. This is done by defining the mapping Φ from scalar activations to tensorial activations such that $\forall \boldsymbol{U} \in \mathcal{O}\left(d\right), \forall \boldsymbol{\lambda} \in \mathbb{R}^d$,

$$\Phi(\sigma) \left(\boldsymbol{U} \operatorname{diag}\left(\boldsymbol{\lambda}\right) \boldsymbol{U}^{\mathrm{T}}\right) = \boldsymbol{U} \operatorname{diag}\left(\begin{bmatrix} \sigma(\boldsymbol{\lambda}_1) & \cdots & \sigma(\boldsymbol{\lambda}_d) \end{bmatrix}\right) \boldsymbol{U}^{\mathrm{T}}. \tag{4}$$

Given eq. (4), it is obvious that Φ is well-defined (i.e. does not depend on the choice of $\boldsymbol{U}$) and that $\phi := \Phi\left(\sigma\right)$ satisfies eq. (3c).

2.3. Equivariant recurrent neural networks

The concept of tensor features can be extended to recurrent neural networks (RNNs) to model history-dependent constitutive models.

2.3.1. Standard neuron operations

The most popular RNN cells, such as the LSTM [32] and the GRU [11], are based on standard neuron operations. Some of these neurons output intermediate features and some are *gates* that control the flow of information through the network. These gates use logistic activation functions and their output is multiplied by the output of other neurons. For example, a possible layer operation of a GRU with scalar features is

$$r = \sigma \left(w^{\mathrm{ir}} x + w^{\mathrm{hr}} h' + b^{\mathrm{r}} \right), \tag{5}$$

$$z = \sigma \left(w^{\mathrm{iz}} x + w^{\mathrm{hz}} h' + b^{\mathrm{z}} \right), \tag{6}$$

$$\hat{h} = \psi \left(w^{\mathrm{ih}} x + b^{\mathrm{ih}} + r \odot \left(w^{\mathrm{hh}} h' + b^{\mathrm{hh}} \right) \right), \tag{7}$$

$$h = (1 - z) \odot \hat{h} + z \odot h', \tag{8}$$

where x is the vector of input scalar features, h' is the vector of output scalar features at the previous time step, σ and ψ are activation functions applied element-wise (respectively a logistic function and a hyperbolic tangent), r and z are gate output, $\hat{h}$ is a candidate output, and the various w and b are respectively weight matrices and bias vectors. In eqs. (7) and (8), $\odot$ denotes the element-wise product of vectors.

2.3.2. Tensorial neuron operations

To extend this architecture to tensor features, the following modifications are proposed:

- The input and output of the layer are replaced with arrays of tensor features, similarly to the approach proposed for feedforward neural networks.

- The output of gates are kept as vectors of scalars to remain faithful to the original purpose of gates which is to keep or forget specific features.

- The activation of eq. (7) is replaced with a tensorial activation function.

- The operation $\odot$ is replaced with an element-wise product between vectors of scalars and arrays of tensors.

These modifications lead to the following operations, written at the neuron level:

$$r_i = \sigma \left(\sum_{j=1}^{n_{\mathrm{in}}} \mathrm{tr} \left(w_{ij}^{\mathrm{ir}} x_j \right) + \sum_{j=1}^{n_{\mathrm{out}}} \mathrm{tr} \left(w_{ij}^{\mathrm{hr}} h_j' \right) + b_i^{\mathrm{r}} \right), \tag{9}$$

$$z_i = \sigma \left(\sum_{j=1}^{n_{\mathrm{in}}} \mathrm{tr} \left(w_{ij}^{\mathrm{iz}} x_j \right) + \sum_{j=1}^{n_{\mathrm{out}}} \mathrm{tr} \left(w_{ij}^{\mathrm{hz}} h_j' \right) + b_i^{\mathrm{z}} \right), \tag{10}$$

$$\hat{h}_i = \Phi(\psi) \left(\sum_{j=1}^{n_{\mathrm{in}}} W_{ij}^{\mathrm{ih}} : x_j + b_i^{\mathrm{ih}} + r_i \left(\sum_{j=1}^{n_{\mathrm{out}}} W_{ij}^{\mathrm{hh}} : h_j' + b_i^{\mathrm{hh}} \right) \right), \tag{11}$$

$$h_i = (1 - z_i) \hat{h}_i + z_i h_i', \tag{12}$$

where the various $\boldsymbol{W} \in \mathcal{T}_d^4$ are tensor weights, the $\boldsymbol{b} \in \mathrm{S}^d$ are bias tensors, and the $\boldsymbol{w} \in \mathrm{S}^d$ are matrix weights. The linear operations in gates are from S^d to $\mathbb{R}$, which explains the use of the trace operator, since linear forms on S^d are isomorphic to S^d via the isomorphism $\boldsymbol{s} \in \mathrm{S}^d \longmapsto \mathrm{tr}\,(\boldsymbol{s}\,\cdot)$.

The equivariance of the neuron operation defined by eqs. (9) to (12) can be ensured by choosing the weights and biases such that the operation in eq. (11) is equivariant and the operations in eqs. (9) and (10) are *invariant*. The invariance of these last operations means that a rotation of $\boldsymbol{x}$ and $\boldsymbol{h}'$ by the same orthogonal matrix results in unchanged output $\boldsymbol{r}$ and $\boldsymbol{z}$. All these conditions are satisfied by choosing the tensor weights $\boldsymbol{W}$ as in eq. (3a) and the bias tensors $\boldsymbol{b}$ and matrix weights $\boldsymbol{w}$ as in eq. (3b). The fact that the invariance of the operations in eqs. (9) and (10) requires this choice of weights is a consequence of the fact that $\forall \boldsymbol{Q} \in \mathcal{O}\,(d)\,, \mathrm{tr}\left(\boldsymbol{w}\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{x}\boldsymbol{Q}\right) = \mathrm{tr}\left(\boldsymbol{Q}\boldsymbol{w}\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{x}\right)$ and that $\forall \boldsymbol{y} \in \mathrm{S}^d, (\mathrm{tr}\,(\boldsymbol{x}\boldsymbol{y}) = 0\ \forall \boldsymbol{x} \in \mathrm{S}^d \implies \boldsymbol{y} = 0)$.

These conditions that the various $\boldsymbol{W}$, $\boldsymbol{w}$, $\boldsymbol{b}$ need to satisfy thus allows their full characterization as performed in sections 2.2.1 and 2.2.2.

2.3.3. Other neural network architectures

Although the presented approach is illustrated with a GRU, it can be applied without difficulty to other recurrent neural network architectures. This includes models with hidden state variables like LSTM by simply using tensor features for these variables.

3. Results

The results of applying TFENNs to multiple materials is presented in this section. These materials include a neo-Hookean 3D hyperelastic material, a 2D unit of an elastic tensegrity metamaterial, and a 2D representative volume element of an elasto-plastic microstructure. Standard neural network architectures with the same number of layers and a similar or higher number of parameters are also trained on these problems for comparison.

3.1. Implementation details

3.1.1. Neural network implementation

All neural networks and training scripts are implemented in Python with the automatic differentiation package Jax [8] and other packages based on it [15, 30]. An important detail is the fact that the tensorial activations as defined in eq. (4) are implemented via an eigenvalue decomposition of the input, which is only differentiable when eigenvalues are simple. This initially led to numerical issues when attempting to compute the gradient of TFENNs. However, using double precision floating point numbers and ensuring that no weights of the neural network are zero-initialized solved these issues in practice for the presented results. A better option for future work would be to express eq. (4) (which itself is differentiable everywhere) as a composition of differentiable matrix functions. For activations based on polynomial, exponential, or logarithm functions, this

could be done by using the matrix versions of these functions. However, not all of them are currently implemented in Jax.

3.1.2. Random deformations generation

Random deformation gradients $\boldsymbol{F}$ are generated from positive eigenvalues sampled with a uniform distribution (hence defining a diagonal matrix $\boldsymbol{D}$) and from random rotation matrices $\boldsymbol{R}$, both combined by setting $\boldsymbol{F} = \boldsymbol{RD}$. In two dimensions, rotation matrices are generated by sampling random angles from a uniform distribution between 0 and 2π. In three dimensions, they are constructed from random rotations axes sampled from a uniform distribution on the unit sphere and random rotation angles sampled as in two dimensions.

3.1.3. Data scaling

An important step in the training of neural networks is the scaling of the input and output data. Here, three different scaling schemes are used:

1. **Scalar component-wise scaling**: each component of the input and output tensors is scaled by its mean and standard deviation over the training set. This method is used to scale the input-output pairs provided during training to the standard neural networks, but is not suitable for the TFENNs, as it would break the possible symmetries of the data.
2. **Tensor symmetry-preserving scaling**: input and output tensors are shifted and scaled by adding and multiplying them with multiples of the identity matrix, such that the mean and standard deviation of the diagonal coefficients of the scaled tensors are respectively zero and one. This scaling ensures that material symmetries are preserved and is therefore used to scale the data provided to TFENNs.
3. **Global scaling**: identical to the scalar component-wise scaling, except that the standard deviation is computed over all components of the tensors. This method is used for the computation of the mean squared error loss during training and validation to provide a same objective in all cases and a comparable outcome. In other words, after a neural network predicts a scaled output tensor according to the first or second scheme, this tensor is rescaled according to the third scheme before being compared to the target tensor, itself also scaled according to the third scheme. Given that the loss function involves a subtraction between the predicted and target tensors, the shifting of the tensors has no impact on the loss and only the scaling is relevant. The scaling involves a standard deviation computed over all components of the tensors to ensure that the components of the tensors are represented in the loss according to their original magnitude. All values of the validation loss reported in this section are computed using the third scheme.

3.1.4. Loss function

The loss function used to train all neural networks is the mean squared error between the predictions and targets. That is, for a set of N pairs of inputs and

outputs $\{(\boldsymbol{X}_i, \boldsymbol{Y}_i)\}_{i=1}^{N}$, which are either tensors or sequences of tensors grouped in a single array, the loss function is defined as

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \|\boldsymbol{Y}_i - \boldsymbol{f}(\boldsymbol{X}_i)\|_F^2 \,, \tag{13}$$

where $\|\cdot\|_F$ denotes the Frobenius norm.

3.2. Neo-Hookean material

The first material considered is an isotropic neo-Hookean hyperelastic material. The model used for this material is derived from the strain energy

$$W = \frac{\lambda}{2} (\log \det \boldsymbol{F})^2 - \mu \log \det \boldsymbol{F} + \frac{\mu}{2} \left(\operatorname{tr} \left(\boldsymbol{F}^{\mathrm{T}} \boldsymbol{F} \right) - 3 \right), \tag{14}$$

where λ and μ are model parameters. Equation (14) results in the following constitutive model, expressed with respect to the Cauchy-Green and second Piola-Kirchoff tensors $\boldsymbol{C}$ and $\boldsymbol{S}$:

$$\boldsymbol{S} = \left(\frac{1}{2} \lambda \log \det \boldsymbol{C} - \mu \right) \boldsymbol{C}^{-1} + \mu \boldsymbol{I}_3. \tag{15}$$

A dataset of $\boldsymbol{C}$-$\boldsymbol{S}$ pairs was generated by sampling random deformation gradients $\boldsymbol{F}$ as described in section 3.1.2, and by subsequently computing $\boldsymbol{C} = \boldsymbol{F}^{\mathrm{T}} \boldsymbol{F}$, and $\boldsymbol{S}$ using eq. (15). Different architectures of neural networks with both scalar features and tensor features (using isotropic symmetry-preserving parameters) were trained on a dataset of 20 000 samples. These networks are compared with respect to the validation loss measured on a distinct dataset of 4000 samples during training in fig. 2. It can be seen that TFENNs achieve a significantly lower validation loss than standard networks with a lower number of parameters. These neural networks were also trained on a smaller training sets of 10 000 and 5000 samples and on larger ones of 40 000 and 80 000 samples. The validation loss obtained at the end of training for these sets is shown in fig. 3. An important observation is that the decrease in performance of TFENNs for fewer training samples is less significant than the one of standard networks. Even with a dataset of 5000 samples, TFENNs achieve a validation loss comparable to the one obtained with the best standard neural network trained on 80 000 samples.

To demonstrate the ability of TFENNs to preserve material symmetries to a much higher degree than standard networks, the trained networks were given random input tensor and random rotated versions of these inputs, and the resulting predictions were compared numerically. Given a neural network model f, a set of N random input tensors $\{\boldsymbol{C}_1, \ldots, \boldsymbol{C}_N\}$, and a set of N random rotation matrices $\{\boldsymbol{R}_1, \ldots, \boldsymbol{R}_N\}$ from the material symmetry group, the resulting error in symmetry enforcement is given by

$$\epsilon_{\mathrm{sym}} = \frac{2}{N} \sum_{i=1}^{N} \frac{\left\| f\left(\boldsymbol{C}_i\right) - \boldsymbol{R}_i f\left(\boldsymbol{R}_i^{\mathrm{T}} \boldsymbol{C}_i \boldsymbol{R}_i\right) \boldsymbol{R}_i^{\mathrm{T}} \right\|_2}{\left\| f\left(\boldsymbol{C}_i\right) \right\|_2 + \left\| f\left(\boldsymbol{R}_i^{\mathrm{T}} \boldsymbol{C}_i \boldsymbol{R}_i\right) \right\|_2}. \tag{16}$$
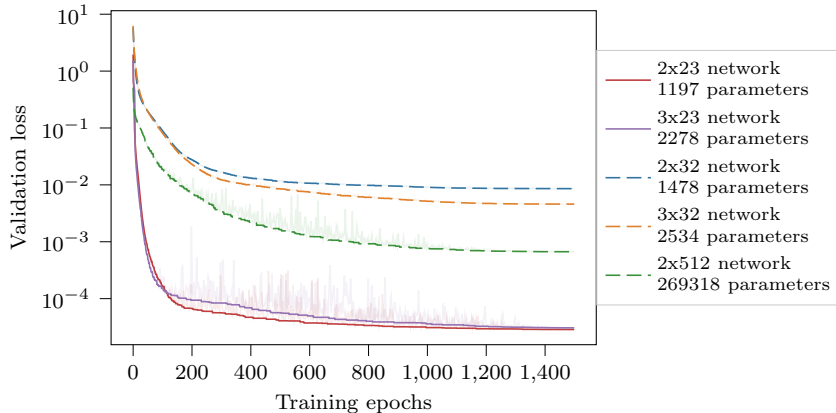
Figure 2: *Validation loss progress of neural networks trained on a neo-Hookean dataset with 20000 training samples.* The minimum validation loss up to the current epoch for tensor feature-based and standard networks is shown by solid and dashed lines, respectively. The current validation loss is represented by the dimmed lines.

The results of this test with $N = 200$ on different neural networks are shown in table 1. It can be seen that traditional neural networks only marginally enforce material symmetries after training, with only a modest improvement of the symmetry error in comparison to the randomly initialized networks before training. On the other hand, TFENNs achieve a symmetry error that is on par with the numerical precision of the computations, demonstrating that material symmetries are enforced to a high degree.

Table 1: Symmetry enforcement error of neural networks trained on the neo-Hookean dataset.

Network		Symmetry error	
Features	**Architecture**	**Pre-training**	**Post-training**
Scalar	3x32	9.26×10^{-1}	4.91×10^{-1}
Scalar	2x32	1.29	4.89×10^{-1}
Scalar	3x64	1.37	4.83×10^{-1}
Scalar	2x64	1.59	4.80×10^{-1}
Scalar	3x128	9.08×10^{-1}	4.78×10^{-1}
Scalar	2x128	1.34	4.78×10^{-1}
Tensor	3x23	2.39×10^{-15}	1.12×10^{-15}
Tensor	2x23	1.26×10^{-15}	9.98×10^{-16}

3.3. Tensegrity lattice unit cell

A two-dimensional unit cell of an elastic tensegrity metamaterial, shown in fig. 4a, is considered as a second example. The cell is composed of pinned bars and cables. Bars are modeled as linear elastic springs up to their Euler's buckling load, after which their load is assumed constant. Cables are modeled
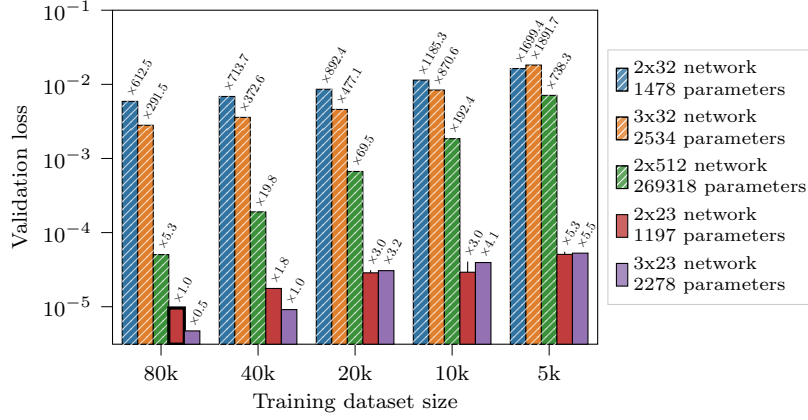
Figure 3: *Final validation loss of neural networks trained on full and reduced neo-Hookean datasets.* The validation loss of TFENNs and standard networks is shown by solid and hatched bars, respectively. The ratio of the achieved validation loss to the one achieved by the tensor feature-based network with the least number of parameters, highlighted with a bar with a wider edge, is shown by the numbers above the bars.
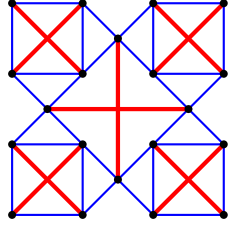
(a) Tensegrity cell [51]. Bars and cables are respectively represented by thick red lines and thin blue lines.
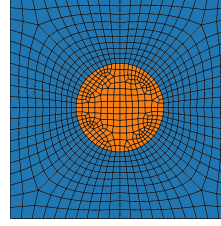
(b) Mesh of the elasto-plastic microstructure used in finite elements analysis [44].

Figure 4: Unit cells used in the tensegrity lattice and elasto-plastic microstructure problems.

as linear elastic springs which can only support tension. For a given deformation gradient $\boldsymbol{F}$, the twelve nodes at the boundary of the cell are fixed according to $\boldsymbol{F}$ while the nodes inside the cell are unconstrained. The equilibrium configuration of the cell is found by using a conjugate gradient method with a Polak-Ribière update formula [52]. Given the boundary nodes' positions in the undeformed configuration $\boldsymbol{X} \in \mathbb{R}^{2 \times 12}$, the forces at the nodes in the deformed configuration $\boldsymbol{f} \in \mathbb{R}^{2 \times 12}$, and the area of the undeformed cell A_0, the equivalent strain tensor is computed with

$$\boldsymbol{S} = \frac{1}{A_0} \boldsymbol{X} \boldsymbol{f}^{\mathrm{T}} \boldsymbol{F}^{-T}. \tag{17}$$

Equation (17) is found by equating the formulations of the work done from deformations of the cell both in terms of strain/stress and displacements/forces.

Here again, various neural networks were trained on different dataset sizes.

The resulting validation loss of neural networks trained on the tensegrity cell dataset is shown in fig. 5. In addition, results with various enforced symmetries for TFENNs are provided. TFENNs again outperform traditional neural networks with a similar number of parameters by a large margin. Unsurprisingly, the best performing TFENN is the one with cubic symmetry enforced, as it is the most restrictive set of symmetries that the tensegrity cell verifies. However, imposing orthotropic symmetry, which is less constraining but still verified by the tensegrity cell, still results in an improvement over a scalar feature formulation. The same can be said when using a triclinic formulation, which does not enforce any symmetry but relies on weight tensors with major symmetry, and when using a completely unconstrained formulation of the tensor weights. The difference in performance between the TFENNs with cubic symmetry and standard neural networks is even more pronounced for smaller datasets, showing again the high data efficiency of TFENNs. It can be noted that in some cases, the TFENN with no enforced symmetry performs better than the one with triclinic or even orthotropic symmetry. It is difficult to draw a general conclusion from this observation as this behavior is not consistent across all dataset sizes and the difference in performance is relatively small. In any case, it is worth noting that these three formulations still largely outperform the standard network, therefore supporting the concept of neural networks based on tensor features.

A test was also performed with a TFENN enforcing isotropic symmetry, which is not verified by the tensegrity cell, and naturally resulted in the network being incapable of learning the cell's response. This test is not represented in fig. 5 as the final validation loss, ranging between 8.429×10^{-1} and 1.13 for the various dataset sizes, is much higher than the displayed ones. From this result, it can be guessed that in the case of unknown dataset symmetry, multiple TFENNs with different enforced symmetries may be trained, and the one with the lowest validation loss selected, thereby offering a mean of discovering a dataset symmetry.
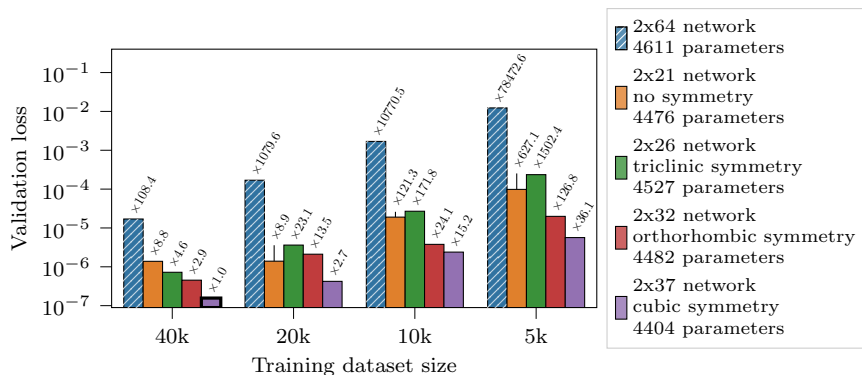

Figure 5: *Final validation loss of neural networks trained on full and reduced tensegrity cell datasets with different enforced symmetries.* Plotting conventions as in fig. 3.

3.4. Elasto-plastic microstructure

The last example considered is a two-dimensional microstructure composed of a hard elastic inclusion in a soft elastic-plastic matrix. 10 000 sequence samples of 200 loading steps each were generated via a finite element simulation, as detailed in [44]. The microstructure, which has cubic symmetry, is shown in fig. 4b. In these cases, the input and output tensors are respectively the infinitesimal strain tensor and the average Cauchy stress tensor.

A TFENN enforcing cubic symmetry and multiple standard neural networks were trained on different sizes of datasets, up to 8000 samples, and were tested on the remaining 2000 samples. All networks have two hidden layers and the TFENN has the least number of parameters. The final validation loss of all networks on the different dataset sizes is reported in fig. 6. The conclusions from this example are similar to the ones obtained from previous experiments: TFENNs provide a significant improvement in data efficiency and prediction accuracy, with a reduced number of parameters. The resulted observed so far therefore extend to history-dependent models.
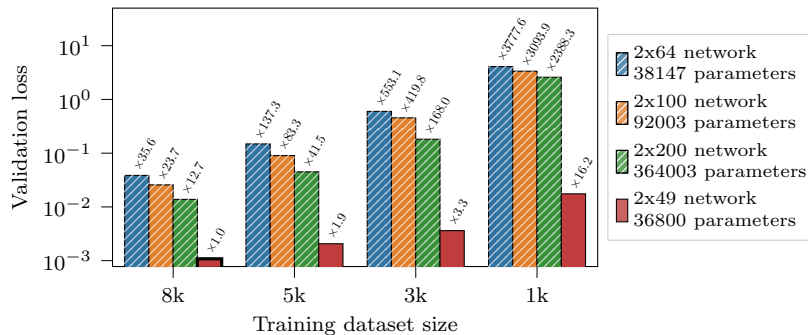

Figure 6: *Final validation loss of neural networks trained on full and reduced elasto-plastic microstructure datasets.* Plotting conventions as in fig. 3.

3.5. Discussion

Three examples of various complexity were considered in this section, both for history-independent and history-dependent models. The presented results show that TFENNs, which enforce material symmetries down to floating point precision, are capable of learning the response of various materials and microstructures with a high accuracy and a relatively low number of training samples. TFENNs are particularly efficient in comparison to standard neural networks with small datasets. Indeed, with all examples, the relative improvement of TFENNs over standard neural networks is more pronounced with fewer training examples.

Even when no symmetry is enforced and the tensor weights and biases are unconstrained, TFENNs still outperform standard neural networks, as shown with the case of the tensegrity cell. Given that the only difference between

TFENNs with no enforced symmetry and standard neural networks is the use of tensorial activation functions, this result suggests that these activation functions alone play a significant role in the performance of TFENNs.

3.6. Symmetry basis discovery

In all three previous examples, the basis of symmetry of the material was known and used in the implementation of the TFENNs. However, in practice, the basis of symmetry of a material may not always be known, which may seem like a limitation of TFENNs. To address this issue, a method to discover the basis of symmetry of a material from data is proposed in this section. This method consists of transforming the input and output tensors of the TFENNs during training via a learnable transformation matrix. That is, if the neural network operation is denoted by f, a $d \times d$ rotation matrix $\boldsymbol{R}$ is defined and the prediction of the network combined with the rotation is given by

$$\hat{\boldsymbol{y}} = \boldsymbol{R} f(\boldsymbol{R}^{\mathrm{T}} \boldsymbol{x} \boldsymbol{R}) \boldsymbol{R}^{\mathrm{T}}. \tag{18}$$

Clearly, if f is constrained to enforce the material symmetries assuming according to a chosen arbitrary basis, for the resulting model to learn the material's response, then $\boldsymbol{R}$ has to be a rotation from the actual basis of symmetry of the material to the chosen basis for the representation of f. To discover an appropriate matrix that satisfies this property, $\boldsymbol{R}$ is computed from a set of learnable parameters of the neural network and added during training. In two dimensions, this parameter is a single angle θ and in three dimensions, a possible choice is a set of four parameters defining a quaternion $\boldsymbol{q}$. For the three-dimensional case, $\boldsymbol{q}$ is normalized to obtain the resulting rotation matrix $\boldsymbol{R}$ and a term penalizing the deviation of $\boldsymbol{q}$ from a unit quaternion can be added to the loss function.

This method was tested on the tensegrity cell dataset, which was transformed into five different arbitrarily rotated versions of it. A TFENN enforcing cubic symmetry with the added learnable rotation was trained five times with different random initial parameters on each of these datasets. The final validation loss and the final value of the learned angle for each rotated dataset and initialization seed are reported in table 2. It can be seen that all but one case result in a low final validation loss with a learned rotation close to the actual rotation of the dataset (up to a 45° rotation, which preserves the cubic symmetry of the tensegrity cell). The model that fails to find the correct rotation also has a high final validation loss, making the failure easy to detect. In addition, the initial angle difference, which depends on the random initialization seed, is reported in table 2, as it shows that the ability of the model to find the correct rotation is not particularly correlated with the magnitude of the initial angle difference. Therefore, it can be concluded that this method is effective at discovering the basis of symmetry of a material from data with a few training experiments based on different random initializations.

16

Table 2: Final validation loss and learned rotation angle of TFENNs trained on rotated versions of the tensegrity cell dataset. The only failing case occurs for a rotation of 20° and a random initialization seed of 2.

True rotation	Seed	Final validation loss	Predicted rotation (error)	
			Pre-training	Post-training
20°	0	9.25×10^{-6}	$-31.727°$ $(6.727°)$	$-24.993°$ $(0.007°)$
	1	1.71×10^{-4}	$53.716°$ $(11.284°)$	$64.964°$ $(0.036°)$
	2	1.02	$-139.904°$ $(20.096°)$	$-142.341°$ $(17.659°)$
	3	1.21×10^{-5}	$-16.344°$ $(8.656°)$	$-25.005°$ $(0.005°)$
	4	7.71×10^{-5}	$-41.065°$ $(16.065°)$	$-24.983°$ $(0.017°)$
175°	0	1.57×10^{-4}	$-31.727°$ $(18.273°)$	$-50.053°$ $(0.053°)$
	1	4.68×10^{-6}	$53.716°$ $(13.716°)$	$39.999°$ $(0.001°)$
	2	1.38×10^{-5}	$-139.904°$ $(0.096°)$	$-140.005°$ $(0.005°)$
	3	7.99×10^{-6}	$-16.344°$ $(11.344°)$	$-4.989°$ $(0.011°)$
	4	1.12×10^{-5}	$-41.065°$ $(8.935°)$	$-49.971°$ $(0.029°)$
222°	0	5.25×10^{-5}	$-31.727°$ $(16.273°)$	$-47.999°$ $(0.001°)$
	1	5.18×10^{-5}	$53.716°$ $(11.716°)$	$42.009°$ $(0.009°)$
	2	6.12×10^{-5}	$-139.904°$ $(1.904°)$	$-137.993°$ $(0.007°)$
	3	1.11×10^{-4}	$-16.344°$ $(13.344°)$	$-2.994°$ $(0.006°)$
	4	1.08×10^{-4}	$-41.065°$ $(6.935°)$	$-47.999°$ $(0.001°)$
298°	0	4.83×10^{-5}	$-31.727°$ $(14.727°)$	$-17.008°$ $(0.008°)$
	1	2.43×10^{-5}	$53.716°$ $(19.284°)$	$73.024°$ $(0.024°)$
	2	1.76×10^{-5}	$-139.904°$ $(12.096°)$	$-152.010°$ $(0.010°)$
	3	3.76×10^{-6}	$-16.344°$ $(0.656°)$	$-16.993°$ $(0.007°)$
	4	1.12×10^{-4}	$-41.065°$ $(20.935°)$	$-61.939°$ $(0.061°)$
340°	0	5.28×10^{-6}	$-31.727°$ $(11.727°)$	$-20.000°$ $(0.000°)$
	1	8.65×10^{-6}	$53.716°$ $(16.284°)$	$70.001°$ $(0.001°)$
	2	3.18×10^{-5}	$-139.904°$ $(15.096°)$	$-155.011°$ $(0.011°)$
	3	8.93×10^{-6}	$-16.344°$ $(3.656°)$	$-19.999°$ $(0.001°)$
	4	5.00×10^{-3}	$-41.065°$ $(21.065°)$	$-19.971°$ $(0.029°)$

4. Conclusion

In this work, a novel type of neural network named TFENN was introduced. These neural networks are designed to work with input-output pairs of symmetric tensors and to be equivariant to the action of subgroups of the orthogonal group, with the main application being the constitutive modeling of materials with symmetries. This equivariance is exact for both finite and infinite groups, unlike some other methods that have been proposed in the literature. TFENNs rely on a representation of the inputs, outputs, and intermediate features of the networks as symmetric tensors of same order and dimensionality. The equivariance of the network is then enforced at the neuron-level by two techniques:

1. Weights and biases are constrained to vector subspaces that depend on the desired enforced symmetry, a technique analogous to weight sharing

which reduces the total number of free parameters.

2. The activation functions are applied to the eigenvalues of their input tensors.

These methods are easily extended to encompass the case of recurrent neural networks, which are used to model history-dependent problems. Despite the fact that this method for enforcing equivariance may seem overly constraining, TFENNs are shown to be capable of learning complex constitutive models on a variety of material models and geometries, elastic or inelastic. In fact, the various examples presented in this work show that TFENNs perform better than standard fully-connected neural networks, in particular when the training data is scarce. This better performance can intuitively be partially attributed to the fact that TFENNs preserve material symmetries by construction and thus do no need to learn them from data, which means that the learning capacity of the network is fully utilized toward learning the constitutive model instead. Since when no particular symmetry is enforced, TFENNs still tend to outperform traditional neural networks, the role of tensor-wise activation functions seem to be of particular importance. A possible avenue for future research is thus to investigate the effect of these activation functions on new problems, particularly for materials with no symmetry, since they could not benefit from the restricted linear operations used on materials with symmetries. The impact of imposing the major symmetry of the weight tensors is also still unclear, and could be investigated further, particularly for materials that are not hyperelastic. Another promising direction would be to combine the methodology presented in this work with techniques that enforce other material properties, such as potential networks used to learn strain energy fields or input convex neural networks for enforcing material stability. Finally, given the fairly general nature of the proposed method and its successful application to constitutive modeling, it would be interesting to apply it to other fields involving tensorial data with symmetry properties.

Acknowledgments

Data availability

The implementation of TFENNs' neurons is available at `https://github.com/kgaranger/TFENN`. The implementation of the training procedures along with the data used in this work are available at `https://github.com/kgaranger/TFENN-examples`.

References

[1] S. Abramian et al. "Recovering the Normal Form and Symmetry Class of an Elasticity Tensor". In: *Journal of Elasticity* 142.1 (July 2020), pp. 1–33. ISSN: 1573-2681. DOI: `10.1007/s10659-020-09784-7`.

[2] David Amsallem and Charbel Farhat. "Interpolation Method for Adapting Reduced-Order Models and Application to Aeroelasticity". In: *AIAA Journal* 46.7 (July 2008), pp. 1803–1813. DOI: `10.2514/1.35374`.

[3] S.L. Angioni, M. Meo, and A. Foreman. "A comparison of homogenization methods for 2-D woven composites". In: *Composites Part B: Engineering* 42.2 (Mar. 2011), pp. 181–189. ISSN: 1359-8368. DOI: `10.1016/j.compositesb.2010.10.004`.

[4] Faisal As'ad, Philip Avery, and Charbel Farhat. "A mechanics-informed artificial neural network approach in data-driven constitutive modeling". In: *International Journal for Numerical Methods in Engineering* 123.12 (Mar. 2022), pp. 2738–2759. ISSN: 1097-0207. DOI: `10.1002/nme.6957`.

[5] N Auffray, B Kolev, and M Olive. "Handbook of bi-dimensional tensors: Part I: Harmonic decomposition and symmetry classes". In: *Mathematics and Mechanics of Solids* 22.9 (Sept. 2017), pp. 1847–1865. ISSN: 1081-2865. DOI: `10.1177/1081286516649017`.

[6] D. Balzani et al. "A polyconvex framework for soft biological tissues. Adjustment to experimental data". In: *International Journal of Solids and Structures* 43.20 (Oct. 2006), pp. 6052–6070. ISSN: 0020-7683. DOI: `10.1016/j.ijsolstr.2005.07.048`.

[7] Jörg Behler. "Constructing high-dimensional neural network potentials: A tutorial review". In: *International Journal of Quantum Chemistry* 115.16 (Mar. 2015), pp. 1032–1050. DOI: `10.1002/qua.24890`.

[8] James Bradbury et al. *JAX: composable transformations of Python+NumPy programs*. Version 0.3.13. 2018. URL: `http://github.com/google/jax`.

[9] Jeremy Q. Broughton et al. "Concurrent coupling of length scales: Methodology and application". In: *Phys. Rev. B* 60 (4 July 1999), pp. 2391–2403. ISSN: 2469-9969. DOI: `10.1103/PhysRevB.60.2391`.

[10] F. Casadei, J.J. Rimoli, and M. Ruzzene. "A geometric multiscale finite element method for the dynamic analysis of heterogeneous solids". In: *Computer Methods in Applied Mechanics and Engineering* 263 (Aug. 2013), pp. 56–70. ISSN: 0045-7825. DOI: `10.1016/j.cma.2013.05.009`.

[11] Kyunghyun Cho et al. *On the Properties of Neural Machine Translation: Encoder-Decoder Approaches*. Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST-8). Nov. 3, 2014. arXiv: `1409.1259 [cs.CL]`.

[12] Taco Cohen and Max Welling. "Group Equivariant Convolutional Networks". In: *Proceedings of The 33rd International Conference on Machine Learning* (New York, New York, USA). Vol. 48. Proceedings of Machine Learning Research. issn=2640-3498. June 2016, pp. 2990–2999. URL: `https://proceedings.mlr.press/v48/cohenc16.html`.

[13] James W Dally and William F Riley. *Experimental stress analysis*. 3rd ed. College House, 2005. ISBN: 978-0976241300.

[14] René De Borst et al. *Nonlinear finite element analysis of solids and structures*. 2nd ed. Wiley, 2012.

[15] DeepMind et al. *The DeepMind JAX Ecosystem*. 2020. URL: `http://github.com/deepmind`.

[16] V.S. Deshpande and A.G. Evans. "Inelastic deformation and energy dissipation in ceramics: A mechanism-based constitutive model". In: *Journal of the Mechanics and Physics of Solids* 56.10 (Oct. 2008), pp. 3077–3100. ISSN: 0022-5096. DOI: `10.1016/j.jmps.2008.05.002`.

[17] V.S. Deshpande and N.A. Fleck. "Isotropic constitutive models for metallic foams". In: *Journal of the Mechanics and Physics of Solids* 48.6 (June 2000), pp. 1253–1283. ISSN: 0022-5096. DOI: `10.1016/S0022-5096(99)00082-4`.

[18] Vera Ebbing, Jörg Schröder, and Patrizio Neff. "Construction of polyconvex energies for non-trivial anisotropy classes". In: *Poly-, Quasi- and Rank-One Convexity in Applied Mechanics*. Springer Vienna, Aug. 2010, pp. 107–130. ISBN: 978-3-7091-0174-2. DOI: `10.1007/978-3-7091-0174-2_4`.

[19] Glenn W Ellis, Chengwan Yao, and Rongda Zhao. "Neural network modeling of the mechanical behavior of sand". In: *Engineering Mechanics. Proceedings of the Ninth Conference* (College Station, TX, USA). ASCE. 1992, pp. 421–424. ISBN: 0872628671.

[20] Mauricio Fernández, Felix Fritzen, and Oliver Weeger. "Material modeling for parametric, anisotropic finite strain hyperelasticity based on machine learning with application in optimization of metamaterials". In: *International Journal for Numerical Methods in Engineering* 123.2 (Dec. 2022), pp. 577–609. DOI: `10.1002/nme.6869`.

[21] Mauricio Fernández et al. "Anisotropic hyperelastic constitutive models for finite deformations combining material theory and data-driven approaches with application to cubic lattice metamaterials". In: *Computational Mechanics* 67.2 (Dec. 2020), pp. 653–677. ISSN: 1432-0924. DOI: `10.1007/s00466-020-01954-7`.

[22] Jacob Fish and Kamlun Shek. "Multiscale analysis of composite materials and structures". In: *Composites Science and Technology* 60.12 (Sept. 2000), pp. 2547–2556. ISSN: 0266-3538. DOI: `10.1016/S0266-3538(00)00048-8`.

[23] Sandra Forte and Maurizio Vianello. "Symmetry classes for elasticity tensors". In: *Journal of Elasticity* 43.2 (May 1996), pp. 81–108. ISSN: 1573-2681. DOI: 10.1007/BF00042505.

[24] Tomonari Furukawa and Genki Yagawa. "Implicit constitutive modelling for viscoplasticity using neural networks". In: *International Journal for Numerical Methods in Engineering* 43.2 (Dec. 1998), pp. 195–219. ISSN: 1097-0207. DOI: 10.1002/(SICI)1097-0207(19980930)43:2<195::AID-NME418>3.0.CO;2-6.

[25] Marc G. D. Geers et al. "Homogenization Methods and Multiscale Modeling: Nonlinear Problems". In: *Encyclopedia of Computational Mechanics*. 2nd ed. Vol. 6. Chichester, UK: Wiley, Dec. 2017. Chap. 5, pp. 1–34. ISBN: 9781119176817. DOI: 10.1002/9781119176817.ecm2107.

[26] Jamshid Ghaboussi, JH Garrett Jr, and Xiping Wu. "Knowledge-based modeling of material behavior with neural networks". In: *Journal of Engineering Mechanics* 117.1 (Jan. 1991), pp. 132–153. ISSN: 0733-9399. DOI: 10.1061/(ASCE)0733-9399(1991)117:1(132).

[27] José Miranda Guedes and Noboru Kikuchi. "Preprocessing and postprocessing for materials based on the homogenization method with adaptive finite element methods". In: *Computer Methods in Applied Mechanics and Engineering* 83.2 (Oct. 1990), pp. 1430–198. ISSN: 0045-7825. DOI: 10.1016/0045-7825(90)90148-F.

[28] Z. Hashin and S. Shtrikman. "A variational approach to the theory of the elastic behaviour of multiphase materials". In: *Journal of the Mechanics and Physics of Solids* 11.2 (Mar. 1963), pp. 127–140. ISSN: 0022-5096. DOI: 10.1016/0022-5096(63)90060-7.

[29] Q. -C. He and Q. -S. Zheng. "On the symmetries of 2D elastic and hyperelastic tensors". In: *Journal of Elasticity* 43.3 (June 1996), pp. 203–225. ISSN: 1573-2681. DOI: 10.1007/BF00042501.

[30] Jonathan Heek et al. *Flax: A neural network library and ecosystem for JAX*. Version 0.7.5. 2023. URL: http://github.com/google/flax.

[31] R. Hill. "Elastic properties of reinforced solids: Some theoretical principles". In: *Journal of the Mechanics and Physics of Solids* 11.5 (Sept. 1963), pp. 357–372. ISSN: 0022-5096. DOI: 10.1016/0022-5096(63)90036-X.

[32] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (Nov. 15, 1997), pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.

[33] Rubén Ibañez et al. "A Manifold Learning Approach to Data-Driven Computational Elasticity and Inelasticity". In: *Archives of Computational Methods in Engineering* 25.1 (Oct. 2018), pp. 47–57. ISSN: 1886-1784. DOI: 10.1007/s11831-016-9197-9.

[34] Gordon R. Johnson and Tim J. Holmquist. "Response of boron carbide subjected to large strains, high strain rates, and high pressures". In: *Journal of Applied Physics* 85.12 (June 1999), pp. 8060–8073. ISSN: 0021-8979. DOI: 10.1063/1.370643.

[35] A.R. Khoei and M. Kianezhad. "A machine learning-based atomistic-continuum multiscale technique for modeling the mechanical behavior of Ni3Al". In: *International Journal of Mechanical Sciences* 239, 107858 (Feb. 2023). ISSN: 0020-7403. DOI: 10.1016/j.ijmecsci.2022.107858.

[36] Dominik K. Klein et al. "Polyconvex anisotropic hyperelasticity with neural networks". In: *Journal of the Mechanics and Physics of Solids* 159, 104703 (Feb. 2022). ISSN: 0022-5096. DOI: 10.1016/j.jmps.2021.104703.

[37] Jeong-Hoi Koo et al. "Dynamic characterization and modeling of magneto-rheological elastomers under compressive loadings". In: *Smart Materials and Structures* 19.11, 117002 (Sept. 2010). DOI: 10.1088/0964-1726/19/11/117002.

[38] B. A. Le, J. Yvonnet, and Q.-C. He. "Computational homogenization of nonlinear elastic materials using neural networks". In: *International Journal for Numerical Methods in Engineering* 104.12 (June 2015), pp. 1061–1084. ISSN: 1097-0207. DOI: 10.1002/nme.4953.

[39] S. Leclercq and C. Lexcellent. "A general macroscopic description of the thermomechanical behavior of shape memory alloys". In: *Journal of the Mechanics and Physics of Solids* 44.6 (June 1996), pp. 953–980. ISSN: 0022-5096. DOI: 10.1016/0022-5096(96)00013-0.

[40] M. Lefik and B.A. Schrefler. "Artificial neural network as an incremental non-linear constitutive model for a finite element code". In: *Computer Methods in Applied Mechanics and Engineering* 192.28 (July 2003), pp. 3265–3283. ISSN: 0045-7825. DOI: 10.1016/S0045-7825(03)00350-5.

[41] Matthias Leuschner and Felix Fritzen. "Reduced order homogenization for viscoplastic composite materials including dissipative imperfect interfaces". In: *Mechanics of Materials* 104 (2017), pp. 121–138. ISSN: 0167-6636. DOI: 10.1016/j.mechmat.2016.10.008.

[42] Lennart Linden et al. "Neural networks meet hyperelasticity: A guide to enforcing physics". In: *Journal of the Mechanics and Physics of Solids* 179, 105363 (June 2023). ISSN: 0022-5096. DOI: 10.1016/j.jmps.2023.105363.

[43] Julia Ling, Reese Jones, and Jeremy Templeton. "Machine learning strategies for systems with invariance properties". In: *Journal of Computational Physics* 318 (Aug. 2016), pp. 22–35. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2016.05.003.

[44] Hernan J. Logarzo, German Capuano, and Julian J. Rimoli. "Smart constitutive laws: Inelastic homogenization through machine learning". In: *Computer Methods in Applied Mechanics and Engineering* 373, 113482 (Jan. 2021). ISSN: 0045-7825. DOI: 10.1016/j.cma.2020.113482.

[45] Xiaoxin Lu et al. "A data-driven computational homogenization method based on neural networks for the nonlinear anisotropic electrical response of graphene/polymer nanocomposites". In: *Computational Mechanics* 64.2 (2019), pp. 307–321. ISSN: 1432-0924. DOI: 10.1007/s00466-018-1643-0.

[46] Filippo Masi et al. "Thermodynamics-based Artificial Neural Networks for constitutive modeling". In: *Journal of the Mechanics and Physics of Solids* 147, 104277 (Feb. 2021). ISSN: 0022-5096. DOI: 10.1016/j.jmps.2020.104277.

[47] D. Millar and E. Clarici. "Investigation of back-propagation artificial neural networks in modelling the stress-strain behaviour of sandstone rock". In: *Proceedings of 1994 IEEE International Conference on Neural Networks*. Vol. 5. IEEE. 1994, pp. 3326–3331. DOI: 10.1109/ICNN.1994.374770.

[48] Benjamin Kurt Miller et al. *Relevance of rotationally equivariant convolutions for predicting molecular properties*. Machine Learning for Molecules Workshop at NeurIPS 2020, NeurIPS workshop on Interpretable Inductive Biases and Physically Structured Learning. Nov. 24, 2020. arXiv: 2008.08461 [cs.LG].

[49] Niels Saabye Ottosen and Matti Ristinmaa. *The mechanics of constitutive modeling*. 1st ed. Elsevier, 2005. ISBN: 9780080446066.

[50] Surajit Pal, G. Wije Wathugala, and Sukhamay Kundu. "Calibration of a constitutive model using genetic algorithms". In: *Computers and Geotechnics* 19.4 (1996), pp. 325–348. ISSN: 0266-352X. DOI: 10.1016/S0266-352X(96)00006-7.

[51] Hossein Salahshoor Pirsoltan. "Topics in stress-induced instabilities and phase transitions in lattice-based solids". PhD thesis. Georgia Institute of Technology, 2018. URL: http://hdl.handle.net/1853/62233.

[52] Elijah Polak and Gérard Ribière. "Note sur la convergence de méthodes de directions conjuguées". In: *Revue française d'informatique et de recherche opérationnelle. Série rouge* 3.R1 (1969), pp. 35–43.

[53] J. Schröder, P. Neff, and D. Balzani. "A variational approach for materially stable anisotropic hyperelasticity". In: *International Journal of Solids and Structures* 42.15 (July 2005), pp. 4352–4371. ISSN: 0020-7683. DOI: 10.1016/j.ijsolstr.2004.11.021.

[54] Y. Shen et al. "Neural Network Based Constitutive Model for Rubber Material". In: *Rubber Chemistry and Technology* 77.2 (May 2004), pp. 257–277. ISSN: 0035-9475. DOI: doi.org/10.5254/1.3547822.

[55] Connor Shorten and Taghi M Khoshgoftaar. "A survey on image data augmentation for deep learning". In: *Journal of big data* 6, 60 (July 2019), pp. 1–48. DOI: 10.1186/s40537-019-0197-0.

[56] David J. Steigmann. "On Isotropic, Frame-Invariant, Polyconvex Strain-Energy Functions". In: *The Quarterly Journal of Mechanics and Applied Mathematics* 56.4 (Nov. 2003), pp. 483–491. ISSN: 0033-5614. DOI: `10.1093/qjmam/56.4.483`.

[57] C.T. Sun and J.L. Chen. "A micromechanical model for plastic behavior of fibrous composites". In: *Composites Science and Technology* 40.2 (1991), pp. 115–129. ISSN: 0266-3538. DOI: `10.1016/0266-3538(91)90092-4`.

[58] Anh-Tuan Ta et al. "A constructive approach of invariants of behavior laws with respect to an infinite symmetry group – Application to a biological anisotropic hyperelastic material with one fiber family". In: *International Journal of Solids and Structures* 51.21 (Oct. 2014), pp. 3579–3588. ISSN: 0020-7683. DOI: `10.1016/j.ijsolstr.2014.06.019`.

[59] Ling Wu et al. "A recurrent neural network-accelerated multi-scale model for elasto-plastic heterogeneous materials subjected to random cyclic and non-proportional loading paths". In: *Computer Methods in Applied Mechanics and Engineering* 369, 113234 (July 2020). ISSN: 0045-7825. DOI: `10.1016/j.cma.2020.113234`.

[60] Kailai Xu, Daniel Z. Huang, and Eric Darve. "Learning constitutive relations using symmetric positive definite neural networks". In: *Journal of Computational Physics* 428, 110072 (Mar. 2021). ISSN: 0021-9991. DOI: `10.1016/j.jcp.2020.110072`.

[61] Huo Yong-Zhong and Gianpietro Del Piero. "On the completeness of the crystallographic symmetries in the description of the symmetries of the elastic tensor". In: *Journal of Elasticity* 25.3 (May 1991), pp. 203–246. ISSN: 1573-2681. DOI: `10.1007/BF00040927`.

[62] J. Yvonnet, D. Gonzalez, and Q.-C. He. "Numerically explicit potentials for the homogenization of nonlinear elastic heterogeneous materials". In: *Computer Methods in Applied Mechanics and Engineering* 198.33 (July 2009), pp. 2723–2737. ISSN: 0045-7825. DOI: `10.1016/j.cma.2009.03.017`.

[63] H. W. Zhang, J. K. Wu, and J. Lv. "A new multiscale computational method for elasto-plastic analysis of heterogeneous materials". In: *Computational Mechanics* 49.2 (Aug. 2011), pp. 149–169. ISSN: 1432-0924. DOI: `10.1007/s00466-011-0634-1`.

[64] Yin Zhang et al. "Learning constitutive relations of plasticity using neural networks and full-field data". In: *Extreme Mechanics Letters* 52, 101645 (Apr. 2022). ISSN: 2352-4316. DOI: `10.1016/j.eml.2022.101645`.