A micromechanics-based recurrent neural networks model for path-dependent cyclic deformation of short fiber composites

Citation for the original published paper (version of record):

Friemann, J., Dashtbozorg, B., Fagerström, M. et al (2023). A micromechanics-based recurrent neural networks model for path-dependent cyclic deformation of short fiber composites. International Journal for Numerical Methods in Engineering, In Press. http://dx.doi.org/10.1002/nme.7211

Received: 4 June 2021 | Revised: 18 December 2022 | Accepted: 21 January 2023

DOI: 10.1002/nme.7211

RESEARCH ARTICLE

WILEY

A micromechanics-based recurrent neural networks model for path-dependent cyclic deformation of short fiber composites

J. Friemann[1] | B. Dashtbozorg[2] | M. Fagerström[1] | S. M. Mirkhalaf[3]

[1]Department of Industrial and Materials Science, Chalmers University of Technology, Gothenburg, Sweden

[2]Department of Biomedical Engineering, Eindhoven University of Technology, Eindhoven, The Netherlands

[3]Department of Physics, University of Gothenburg, Gothenburg, Sweden

Correspondence
S. M. Mirkhalaf, Department of Physics, University of Gothenburg, Gothenburg, Sweden.
Email: mohsen.mirkhalaf@physics.gu.se

Funding information
Vetenskapsrådet, Grant/Award Number: 2019-04715; VINNOVA

Abstract

The macroscopic response of short fiber reinforced composites (SFRCs) is dependent on an extensive range of microstructural parameters. Thus, micromechanical modeling of these materials is challenging and in some cases, computationally expensive. This is particularly important when path-dependent plastic behavior is needed to be predicted. A solution to this challenge is to enhance micromechanical solutions with machine learning techniques such as artificial neural networks. In this work, a recurrent deep neural network model is trained to predict the path-dependent elasto-plastic stress response of SFRCs, given the microstructural parameters and the strain path. Micromechanical mean-field simulations are conducted to create a database for training the validating the model. The model gives very accurate predictions in a computationally efficient manner when compared with independent micromechanical simulations.

KEYWORDS

cyclic deformation, deep learning, micromechanics, path-dependent plasticity, recurrent neural networks, short fiber composites

1 | INTRODUCTION

Short fiber reinforced composites (SFRCs) are popular materials for engineering applications, partly due to being able to be manufactured through injection molding. The discontinuous nature of the fibers prevents loads from being transferred between themselves and thus, the matrix's role is transferring loads. The high stiffness of the fibers results in the potential of high stresses inside the composite. These high stresses often lead to yielding in the matrix. This interplay between the composite constituents leads to a non-linear elasto-plastic mechanical response of the SFRC. Non-linear analysis of SFRCs is therefore highly relevant for predicting the behavior of structural parts.

A wide variety of microstructural parameters, such as fiber volume fraction and fiber orientation distribution (among others), plays an important role in the mechanical performance of these materials. Thus, different homogenization schemes have been extensively investigated in different micromechanical models. Currently, the most accurate homogenization technique is using computational homogenization performed on realistic representative volume elements (RVEs). Despite accurate predictions, it is not always feasible to use this approach due to some challenges such as difficult RVE generation and expensive simulations.[1-4] Alternatively, two-step and mean-field methods have been developed and used during the last decades (see more information in e.g., References 5-7).

More recently, data-driven modeling approaches are being investigated. Following the successful application of machine learning techniques in image processing, language processing and so forth these techniques are currently finding a large number of applications in materials science as well (see e.g., Reference 8 for mechanical response, Reference 9 for electrical response, and Reference 10 for thermal conductivity). The movement of different engineering fields toward the usage of ML techniques is facilitated by (i) the availability of unprecedented amount of data from experiments and simulations, (ii) the quick growth of computer power, and (iii) the availability of advanced open-source libraries such as TensorFlow and PyTorch.

One of the widely used approaches in machine learning is artificial neural networks (ANN).

In recent years, a number of ANN models are developed to replace as a surrogate for constitutive models.[11,12] These ANN models are developed based on two main categories of data, namely experimental results, and physics-based simulations.[13] ANN models developed based on experiments typically describe phenomena for which a thorough physical understanding is not available yet. For simulation-based ANN models, two different approaches could be followed: (I) In the first approach, the material is considered homogenous and described by a phenomenological constitutive model. These constitutive models typically have a number of parameters which are calibrated using experimental results;[14] (II) The other approach is to consider the heterogeneities of the material at lower length scales and obtain the macroscopic material behavior by homogenizing the behavior of the constituents at the sub-scale. This kind of ANN models is trained using information from the material microstructure and the homogenized material behavior.[12]

An ANN typically consists of a large, layered network of artificial neurons whose individual properties are very simple, but their connected behavior can give rise to very complex phenomena. Neural networks have successfully been utilized for constitutive modeling of different materials[15,16] and have shown great promises. Mozaffar et al.[11] showed that path dependent plasticity can be properly predicted using recurrent neural networks (RNN). Two dimensional RVEs of a composite material were analyzed and used to create the required database for training and validating an RNN model. Gorji et al.[17] developed a machine learning model to replicate prediction of an anisotropic non-linear material model with anisotropic hardening. RNN were used to develop the data-driven model. After training and validating the network, it was used for modeling multi-axial behavior of a two-dimensional foam. More recently, Bonatti and Mohr[18] proposed a self-consistent recurrent neural network as a surrogate for non-linear material behavior. Linearized minimal state cell is introduced to alleviate the dependency of the network performance to increment size. Trained and validated network is then integrated into an explicit finite element framework and is used for component-level analysis. Although limited to two-dimensional analyses, Wu et al.[19] indeed developed a capable ANN model as a surrogate model for micromechanical modeling of an elasto-plastic composite. Two dimensional full-field (RVE) finite element simulations were performed to generate the required database for training and validating an RNN with promising results. As an extension, Wu and Noels[20] further developed an RNN surrogate model with the capability of also recovering the evolution of the local micro-structure state variables subjected to complex loading paths. It should, however, be mentioned that these latter contributions did not consider a varying microstructure (neither a varying fiber volume fraction nor different fiber orientation distributions) as input to the surrogate model. In fact, to the knowledge of the authors, all previous studies that have developed micromechanics-based ANN models for general 3D SFRCs with arbitrary fiber orientations and volume fractions are limited to the elastic regime, compare for example, Reference 12.

Therefore, in this study, the main aim has been to develop a more generic micromechanics-based ANN model for SFRCs, capable of considering as well path-dependent cyclic elasto-plastic behavior as arbitrary and fully three-dimensional stress–strain states, varying fiber volume fractions and different fiber orientation distributions. The main focus is given to the capability of ANN for reproducing general non-linear behavior of these materials, rather than analyzing a specific SFRC and direct comparisons to experiments. Although the same constituent materials are assumed, a variety of orientation distributions, fiber volume fractions and loading-paths are considered in the model. As for all machine learning problems, the issue of properly choosing the training data is central. To generate sufficiently general strain inputs, a continuous random walk-scheme in 6D strain space is proposed. The resulting strain paths are each associated with an orientation tensor describing the fiber distribution, and a fiber volume fraction. For the sampling of uniformly distributed orientation tensors, a simple but effective algorithm utilizing constraints on the tensors' eigenvalues is employed. As a result and in contrast to previously developed ANN models by others, we hereby present a rather generic model, applicable to a wide range of SFRCs.

To efficiently generate the large data set necessary for considering 3-dimensional micro-structures with a variety of fiber orientation distributions and fiber volume fractions, the ANN training inputs (the arbitrary stress–strain histories) are all obtained via mean-field (MF)-simulations. The results indicate that it is feasible to use ANN to model elasto-plasticity for general 3D analysis of SFRC. The proposed method for generating strain paths shows promise and leads to a training set that is general enough to enable prediction of arbitrary, even highly complex, loading histories. The obtained ANN model is able to accurately reproduce non-linear elasto-plastic stress–strain curves for SFRCs with arbitrary fiber orientation distributions and for a reasonable range of fiber volume fractions.

The remaining of this article is structured as follows. Section 2 describes the constituents of the studied SFRCs and generating the database. Section 3 explains the design and training of the ANN model. Results obtained from the ANN model and comparisons to micromechanical simulations are given in Section 4. Section 5 gives a discussion on the developed ANN model and obtained results. Finally, Section 6 gives some concluding remarks.

2 | MICROMECHANICAL SIMULATIONS

In this section, first, the constituent phases (matrix and reinforcements) of the analyzed composites are explained. Then, the method for generating the required data, for training and validating the artificial neural network model, is explained.

2.1 | Constituents of the studied SFRCs

We focus on a specific pair of matrix and reinforcements, namely a polymeric matrix (which behaves similarly as a Polyamide 6.6) and short glass fibers. With the same matrix and reinforcements, we will consider different fiber orientation distributions and fiber volume fractions under arbitrary loading conditions. Kammoun et al.[21] proposed describing this material with linear elastic fibers and a matrix obeying J_2-plasticity with isotropic linear-exponential hardening. The hardening stress is formulated as

$$\kappa = H\overline{\varepsilon^{\mathrm{p}}} + H_\infty \left(1 - e^{-m\overline{\varepsilon^{p}}}\right), \tag{1}$$

where H is the linear hardening modulus, H_∞ is referred to as the hardening modulus, $m > 0$ is the hardening exponent, and $\overline{\varepsilon^{\mathrm{p}}} \geq 0$ is the accumulated plastic strain. The yield function is given by

$$\Phi(\boldsymbol{\sigma}, \kappa) = \sigma_{\mathrm{eq}} - (\sigma_{\mathrm{y}} + \kappa) \leq 0, \tag{2}$$

where σ_{y} is the yield stress and σ_{eq} is the von Mises equivalent stress defined using the deviatoric stress:

$$\sigma_{\mathrm{eq}} = \sqrt{\frac{3}{2}\boldsymbol{\sigma}_{\mathrm{dev}} : \boldsymbol{\sigma}_{\mathrm{dev}}}, \quad \boldsymbol{\sigma}_{\mathrm{dev}} = \boldsymbol{\sigma} - \frac{1}{3}\mathrm{tr}(\boldsymbol{\sigma})\boldsymbol{I}, \tag{3}$$

where $\boldsymbol{I}$ is the second order identity tensor. Kammoun et al.[21] presents some of the model parameters of the matrix material in terms of the matrix yield stress σ_{y}. The yield stress itself is not given as classified information. In the current work, the value of $\sigma_{\mathrm{y}} = 25$ GPa is used. This seems to be a typical value for the yield stress for this type of material.[22] The used material parameters are given in Table 1.

2.2 | Generation of database

The generation of the network training data requires three steps, (i) generating strain paths that cover a representative spectrum of loading conditions (described in Section 2.2.1); (ii) generating a second order orientation tensor with a corresponding fiber volume fraction for each strain path (described Section 2.2.2); (iii) conducting strain-controlled

TABLE 1 The material parameters used for modeling the SFRCs.

	Parameter	Value
Fiber	Young's modulus E_F	76 GPa
	Poisson's ratio ν_F	0.22
	Length l	240 µm
	Diameter d	10 µm
Matrix	Young's modulus E_M	3.1 GPa
	Poisson's ratio ν_M	0.35
	Yield stress σ_y	25 MPa
	Linear hardening modulus H	150 MPa
	Hardening modulus H_∞	20 MPa
	Hardening exponent m	325

micromechanical mean-field simulations (described in Section 2.2.3). When training the network, the generated strain paths, orientation data, and fiber volume fractions are fed as inputs, and the corresponding stress responses are the quantities being predicted. In Section 2.2.4, the implementation of the aforementioned steps is explained.

2.2.1 | Generation of strain paths

In order to be able to build a network that can learn the proper path-dependency of plasticity, representative strain paths need to be generated. The paths used for training need to be varied enough for the network to be able to recognize complicated paths without losing the ability to predict common situations such as uniaxial stress. Random sampling and different algorithms have been used and proposed by other authors to generate adequate strain data, see for example, References 11, 17-19, 23. One approach is sampling a set of random points in strain space and subsequently using an interpolator to connect the points to form paths radiating from the origin.[11]

Randomly sampled data is a useful approach in ensuring a good distribution of data. We propose a random walk with superimposed noise as a means to sample representative strain paths. The random walk, hereby referred to as the drift directions, represents long-term trends in the strain components. The noise on the other hand, with a smaller amplitude, represents local variations in the evolution of the strain. The noise is added by taking repeated steps in the drift directions with added perturbations. The developed algorithm is explained in details in what follows.

As the strain tensor has 6 independent components, a random walk in $\mathbb{R}^6$ is performed. In order to sample 6-dimensional unit direction vectors uniformly from a unit 6D-sphere (analogous to uniform point picking on a 3D-sphere), the following procedure is employed. First, the components of the 6-dimensional vector are sampled independently from each other. The samples are picked from a normal distribution with mean $\mu = 0$ and standard deviation $\sigma^2 = 1$. Thereafter, the vector is normalized such that its magnitude is 1. Since the product of independent identical normal distributions is a normal distribution (which possesses radial symmetry), the resulting probability distribution will be uniform on the unit sphere after the normalization. Let N be the number of steps in the strain time series (total length $N + 1$ including the initial position at the origin). Subsequently, let n_1 be the number of drift directions sampled for the random walk and let n_2 be the number of steps per drift direction. These parameters are supplied with the constraint $n_1 n_2 = N$. Also, let γ be a number between 0 and 1 that determines the relative amplitude of the perturbations. The algorithm begins with sampling an ordered list of n_1 vectors, these vectors are the drift directions. Every vector in the list is repeated n_2 times such that the order is preserved, that is, the list with now $n_1 \cdot n_2 = N$ elements becomes: $\boldsymbol{v}_1, \dots, \boldsymbol{v}_{\boldsymbol{1}}, \dots, \boldsymbol{v}_{n_1}, \dots, \boldsymbol{v}_{n_1}$. Following this, N additional vectors are sampled and put into a second ordered list, each one of these new vectors are scaled by the factor γ. A third list is constructed by taking the element-wise vector sum of the two lists, each containing N elements. A 6-dimensional time series is obtained by taking the cumulative sum of list number three. The time series describing the evolution of the strain components finally follows by re-normalizing such that the largest magnitude element is some desired value ε_max. The presented algorithm (reduced to 2 dimensions for the sake of easy visualization) is displayed in Figure 1.

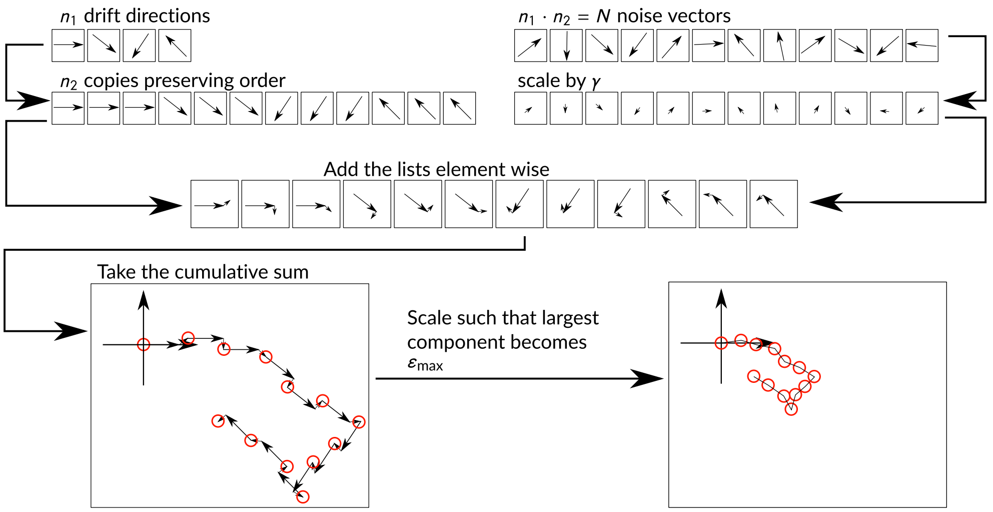
n_1 drift directions
$n_1 \cdot n_2 = N$ noise vectors
n_2 copies preserving order
scale by γ
Add the lists element wise
Take the cumulative sum
Scale such that largest component becomes $\varepsilon_{\max}$

FIGURE 1 Schematic representation of the proposed strain path generation algorithm (using $n_1 = 4$ and $n_2 = 3$).

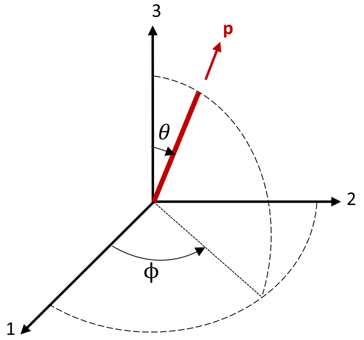

FIGURE 2 Orientation of a fiber (**p**) in a 3D configuration and the corresponding angles (taken from Reference 4).

2.2.2 | Uniform sampling of orientation tensors

The orientation of an inclusion (fiber) is represented by a unit vector **p** which can be equivalently described using two angles, as shown in Figure 2. The unit vector **p** is given by

$$\mathbf{p} = \begin{bmatrix} \sin\{\theta\}\cos\{\phi\} \\ \sin\{\theta\}\sin\{\phi\} \\ \cos\{\theta\} \end{bmatrix}. \tag{4}$$

An orientation tensor is a function of orientation distribution function (ODF) $\psi(\mathbf{p})$.[24] This function has the following properties:

$$\psi(\mathbf{p}) = \psi(-\mathbf{p}), \tag{5}$$

$$\int_{\Omega} \psi\,\{\mathbf{p}\}\,\mathrm{d}\Omega = 1, \tag{6}$$

where Ω is the whole domain of the RVE. The components of the second order orientation tensor (**a**) are given by

$$a_{ij} = \int_{\Omega} p_i p_j \psi(\mathbf{p}) d\mathbf{p}. \tag{7}$$

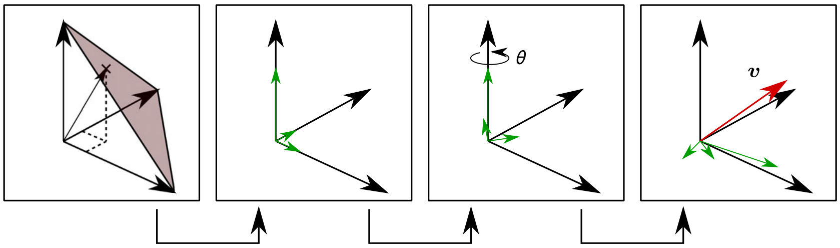

FIGURE 3 The proposed algorithm for uniform sampling of second order orientation tensors.

The trace of the second order orientation tensor must be unity,[24] and its eigenvalues must be non-negative. To uniformly sample from the set of all possible orientation tensors, one may uniformly sample triplets of eigenvalues and thereafter perform a coordinate transformation through a randomly chosen rotation. A visualization of the proposed sampling algorithm can be seen in Figure 3. Uniform sampling of three eigenvalues between 0 and 1 that sum to 1 can be achieved by sampling uniformly from the standard 2-simplex. This is can be done by sampling 2 points uniformly in (0, 1), and using the length of the three segments these two points make up together with 0 and 1 as the three eigenvalues.[25]

Remark 1. Based on the spectral theorem, a second order tensor can be rotated to be diagonal, where the diagonal elements are the eigenvalues. Thus, doing it in reverse, starting from uniformly sampled diagonalized tensors, rotated with uniformly sampled rotations will yield uniformly sampled orientation tensors.

The uniform random sampling of the rotation is performed with an algorithm by Arvo,[26] that is both fast and easy to implement:[27] First, an angle θ is uniformly sampled from $(0, 2\pi)$, and a single axis rotation $\boldsymbol{R}(\theta)$ is performed. Thereafter, a unit vector $\boldsymbol{v} = [\cos\varphi\sqrt{z}, \sin\varphi\sqrt{z}, \sqrt{1-z}]^{\mathrm{T}}$ is sampled, where φ is sampled uniformly from $(0, 2\pi)$ and z is sampled uniformly from $(0, 1)$. A negatively scaled Householder transformation $-\boldsymbol{H}(\boldsymbol{v}) = 2\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}} - \boldsymbol{I}$ is performed, where $\boldsymbol{I}$ is the identity matrix. It is possible to show that this transformation will map the axis of rotation to any arbitrary point on the unit sphere. A uniformly sampled rotation $\boldsymbol{R}'$ is thus received through the composition of the transformations; $\boldsymbol{R}'(\theta, \varphi, z) = -\boldsymbol{H}(\varphi, z)\boldsymbol{R}(\theta)$.

A very similar approach to sampling orientation tensors has been employed in a recent study.[12] Similarly to what is done in the current work, the sampling of orientation tensors was performed by sampling their spectral decomposition. However, while the current work samples eigenvalues uniformly, Mentges et al.[12] sampled eigenvalues using Bingham distributions.

2.2.3 | Mean-field analyses

Micro-mechanics is the study of the mechanics of heterogeneous materials where the microscopic structure of the material is taken into account. Micromechanical modeling could be divided into two main categories of models: *full-field* and *mean-field* models. In full-field models, microscopic fields are accounted for at all different microscopic points. On the other hand, mean-field models consider an average strain and average stress in each of the micro-structure phases. In mean-field models, typically, explicit analytical relations are obtained for the average response of the micro-structure. In terms of the computational cost, mean-field models are more efficient than full-field models. A large number of studies have proposed mean-field models for different materials (see e.g., References 28,29 for SFRCs, Reference 30 for semi-crystalline polymers, and Reference 31 for an overview of different methods for unidirectional composites). Classical mean-field theories, such as those by Eshelby,[32] Hashin and Shtrikman,[33,34] Hill,[35] Budiansky,[36] and Mori-Tanaka,[37] are typically the basis of different mean-field models. To get more insight about mean-field modeling of SFRCs, an interested reader is referred to the recent work by Hessman et al.[38]

The stress response pertaining to a given strain path, fiber distribution and fiber volume fraction is computed via micromechanical simulations. The simulations are performed using a mean-field approach where Mori-Tanaka theory is used for homogenization. The mean-field module of the software DIGIMAT-MF[39] is used to conduct the simulations. The mean-field homogenization procedure in DIGIMAT-MF is conducted in two steps. First, the RVE of the composite under study is divided to pseudo-grains (PGs), and each PG is individually homogenized. In the second step, the effective response of the whole RVE is computed by homogenizing all PGs. Figure 4 shows a schematic

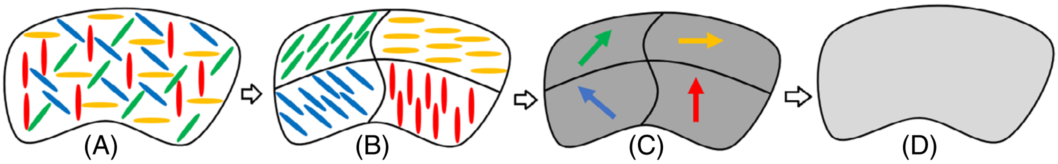

FIGURE 4 Schematic representation of the two-step homogenization procedure performed in DIGIMAT-MF, re-produced after;[5,39] (A) to (B) decomposition, (B) to (C) first homogenization step, (C) to (D) second homogenization step.

representation of the two-step homogenization approach. Each pseudo-grain is a two-phase composite including matrix material and identical inclusions with the same shape, aspect ratio and orientation. Let us consider the domain each PG as $\omega_i, (i = 1, \dots, N)$, where N is the number of PGs. The matrix domain in each PG is $\omega_{i,1}$ and the reinforcement domain is $\omega_{i,2}$. For elasto-plastic non-linear response of SFRCs (an elasto-plastic matrix and elastic fibers), each PG is homogenized using a Hill-type incremental formulation. For each phase, a fictitious reference material is assumed so that the rate stress–strain relations are given by:

$$\dot{\boldsymbol{\sigma}}(\mathbf{x}, t) = \hat{\mathbb{C}}_{i,r} : \dot{\boldsymbol{\varepsilon}}(\mathbf{x}, t) \quad \forall \mathbf{x} \in \omega_{i,r}, \quad r = 1, 2, \tag{8}$$

where $\hat{\mathbb{C}}_r$ is the reference tangent moduli of the two phases. The average strain rate of matrix and reinforcements are related to the average strain rate of PG by:

$$\langle \dot{\boldsymbol{\varepsilon}}(\mathbf{x}, t) \rangle_{\omega_{i,1}} = \left[v_2 \mathbb{B} + v_1 \mathbb{I} \right] : \langle \dot{\boldsymbol{\varepsilon}}(\mathbf{x}, t) \rangle_{\omega_i}, \tag{9}$$

$$\langle \dot{\boldsymbol{\varepsilon}}(\mathbf{x}, t) \rangle_{\omega_{i,2}} = \mathbb{B} : \langle \dot{\boldsymbol{\varepsilon}}(\mathbf{x}, t) \rangle_{\omega_{i,1}} = \mathbb{B} : \left[v_2 \mathbb{B} + v_1 \mathbb{I} \right] : \langle \dot{\boldsymbol{\varepsilon}}(\mathbf{x}, t) \rangle_{\omega_i}, \tag{10}$$

where, v_1 and v_2 are the volume fractions of matrix and reinforcements, $\mathbb{B}$ is the strain concentration tensor, and $\mathbb{I}$ is the fourth order identity tensor. Using the Mori-Tanaka model,[37] the strain concentration tensor is given by:

$$\mathbb{B} = \left\{ \mathbb{I} + \mathbb{E} : [\left(\hat{\mathbb{C}}_{i,1} \right)^{-1} : \hat{\mathbb{C}}_{i,2} - \mathbb{I}] \right\}^{-1}. \tag{11}$$

The homogenized tangent moduli of the PG is obtained as

$$\mathbb{C}_{\omega_i}(t) = \left[v_2 \hat{\mathbb{C}}_{i,2}(t) : \mathbb{B}(t) + v_1 \hat{\mathbb{C}}_{i,1}(t) \right] : \left[v_2 \mathbb{B}(t) + v_1 \mathbb{I} \right]^{-1}, \tag{12}$$

which relates the rate of the homogenized PG stress and strain rates:

$$\langle \dot{\boldsymbol{\sigma}}(\mathbf{x}, t) \rangle_{\omega_i} = \mathbb{C}_{\omega_i}(t) : \langle \dot{\boldsymbol{\varepsilon}}(\mathbf{x}, t) \rangle_{\omega_i}. \tag{13}$$

So far, each PG (ω_i) is homogenized.

In the second step, the homogenized response of all PGs (the final macroscopic response) is obtained. The average of micro-stress rate field is given by

$$\langle \dot{\boldsymbol{\sigma}}(\mathbf{x}, t) \rangle_{\Omega} = \langle \langle \dot{\boldsymbol{\sigma}}(\mathbf{x}, t) \rangle_{\omega_i} \rangle_{i,\psi_i} = \langle \mathbb{C}_{\omega_i}(t) : \langle \dot{\boldsymbol{\varepsilon}}(\mathbf{x}, t) \rangle_{\omega_i} \rangle_{i,\psi_i}, \tag{14}$$

where $\langle \cdot \rangle_{i,\psi_i}$ represents the ODF-weighted average over all PGs. Considering the Voigt assumption, we have

$$\langle \dot{\boldsymbol{\varepsilon}}(\mathbf{x}, t) \rangle_{\omega_i} = \langle \dot{\boldsymbol{\varepsilon}}(\mathbf{x}, t) \rangle_{\Omega} \quad \forall \omega_i. \tag{15}$$

Thus, Equation (14) is re-written as

$$\langle \dot{\boldsymbol{\sigma}}(\mathbf{x}, t) \rangle_{\Omega} = \langle \mathbb{C}_{\omega_i} \rangle_{i,\psi_i} \langle \dot{\boldsymbol{\varepsilon}}(\mathbf{x}, t) \rangle_{\Omega}, \tag{16}$$

and as a result, the homogenized macroscopic tangent moduli of the RVE is obtained as

$$\overline{\mathbb{C}}(t) = \langle \mathbb{C}_{\omega_i}(t) \rangle_{i,\psi_i}. \tag{17}$$

Digimat allows for a first and second order homogenization. Second order homogenization shows a significant improvement in accuracy over first order homogenization when there is a large difference between matrix and inclusion stiffness and the matrix exhibits little hardening.[28] In the present work these conditions are fulfilled and second order homogenization is therefore chosen. This was a brief introduction to the mean-field homogenization method in Digimat-MF. For more elaborated details on the modeling approach, an interested reader is referred to Reference 5.

2.2.4 | Implementation

The generation of training data is controlled via a small program written in Julia programming language.[40] The process of running Digimat simulations is automated by running the program in batch-mode.

The script first generates the strain data and orientation tensors used as input according to the presented algorithms. The total number of steps in the strain paths are set to $N = 2000$. A pseudo-time variable is also created that starts at 0 s and ends at 1 s with N increments. The time range is completely arbitrary due to the rate independence of the plasticity, and is simply chosen to match the default settings of Digimat. The motivation for long time sequences as training data is that an ANN trained on long sequences would be able to make more accurate predictions compared to shorter input sequences. This follows from the fact that the short sequences are contained in the long ones owing to the construction of the generation algorithm. However, the length of the sequences is a trade-off between generality and computational cost.

For each generated strain path, the generation parameters are sampled randomly. The noise multiplicative factor γ is chosen between 0 and 1 from a uniform distribution. The number of drift directions n_1 is chosen uniformly from the set $\{1, 2, 5, 10, 20, 25, 50,100, 200\}$. This set of number of drift directions is chosen arbitrarily, motivated only by allowing a fairly large range of path complexities, where $n_1 = 1$ would mean essentially a perturbed linear function and $n_1 = 200$ would result in a highly complex path. Furthermore, the maximum admissible strain component $\varepsilon_{\max}$ is sampled uniformly between 0.01 and 0.05. This range captures a full range of reasonable loading conditions. The lower bound allows loading that does not lead to plasticity and the upper limit of 0.05 is kept since failure is extremely likely to occur beyond that point which is outside the scope of this work. As a final parameter regarding the strain paths, the possibility of uniaxial strains is implemented. This is realized by setting all but one of the strain components of a strain path to constant 0 with a probability of 10%.

For every strain path, a corresponding orientation tensor is generated according to the proposed algorithm. The case of uniaxial fiber alignment is chosen with a probability of 10% during generation, that is, one eigenvalue is set to exactly 1 while the remaining two are set to exactly 0. As an additional parameter associated with the fiber distribution, the fiber volume fraction v_F is chosen randomly between 10% and 15%. The volume fraction is allowed to vary since it will be dependent on the position after injection molding. The range of fiber volume fraction is kept small to make the training process of the network easier. All parameters of the strain path and orientation tensor generation are displayed in Table 2.

A data set with 40,000 samples is generated. Each sample consists of one strain path, one orientation tensor, one fiber volume fraction, and the resulting stress path. The computations are carried out on a personal computer with an 8 core

TABLE 2 The parameters pertaining to the generation of strain paths and orientation tensors.

Parameter	Value
N	2000
γ	Uniform random (0, 1)
n_1	Uniformly from $\{1, 2, 5, 10, 20, 25, 50,100, 200\}$
$\varepsilon_{\max}$	Uniform random (0, 0.05)
$\mathcal{P}$ (uniaxial strain)	0.1
$\mathcal{P}$ (uniaxial fibers)	0.1
v_F	Uniform random (0.1,0.15)

(Intel Core i7-10700KF) 3.8 GHz processor and 16 GB of memory. The entire generation process takes about two weeks. The vast majority of the computational time is dedicated to the micromechanical simulations that compute the stresses.

3 | DESIGN AND TRAINING OF THE ARTIFICIAL NEURAL NETWORK

ANN are biologically inspired computational networks consist of many connected artificial neurons. One of the simplest network structures, but commonly used for a wide variety of problems, is the feed forward artificial neural network (FFANN). The FFANN consists of an ordered layer structure where outputs from the neurons of one layer make up the inputs of the subsequent layer's neurons, hence the name feed-forward. The first layer is the input layer and consists of n_{in} neurons. Following the input layer, N_{h} so-called hidden layers follow. Each hidden layer has $n_1, \dots, n_{N_{\mathrm{h}}}$ number of neurons respectively. The final layer is the output layer and has n_{out} number of neurons. Each neuron of any one layer is connected to every neuron of the following layer. Every connection in the network is supplied with a different weight. The weights scale the respective outputs before they are being fed as input to the following neurons. The final component that makes up the basic structure of the network is the bias signals. There is a set of bias signals between every layer of the network. Every set of biases connects to the neurons of the layer directly following it, and is added to the inputs.

Any given neural network is designed and trained for a fixed number of inputs and outputs. Ideally, some time-dependent phenomenon could be modeled by an FFANN with $N \cdot N_T$ input neurons, where N is the number of variables that are fed to the network and N_T is the number of time steps. However, such a network would not be able to predict what happens at time step $N_T + 1$, or even predicting the behavior for a time series shorter than N_T since that would require a different number of input neurons. In many applications, the predictions are time-dependent and require the information from previous steps that is, path-dependent plasticity with different possible loading conditions. The solution for time-series type of problems, using ANN with internal variables that update every time the feeding data to the network. Such networks are called RNN.

3.1 | Recurrent neural networks

Recurrent neural networks, also known as RNNs, are a class of neural networks that allow previous outputs to be used as inputs while having hidden states. RNNs save the output of processing nodes and feed the result back into the model to learn predicting the outcome of a layer. In the RNN models, each node acts as a memory cell, continuing the computation and implementation of operations. RNNs are useful in time series prediction only because of the feature to remember previous inputs as well (long short term memory (LSTM)), however, RNNs are prone to gradient vanishing and are challenging to train. As a way to circumvent the issue of vanishing gradients the RNN architecture is to allow for derivative truncation. The idea is to let the network only pass information a finite distance in time before *forgetting* it. Successful implementation of a mechanism enabling forgetting information is LSTM networks introduced by *Hochreiter* and *Schmidhuber*,[41] and later refined by *Gers* and *Cummins*.[42] The LSTM architecture introduces a new hidden state which keeps track of which information is relevant for long-term behavior. By forgetting things that have no influence on long-term behavior, the predicament of vanishing gradients is mitigated. Another RNN implementation that remedies the vanishing gradient problem is the gated recurrent unit (GRU) introduced by Cho et al.[43] The GRU contains two gate functions (reset and update gates) whereas an LSTM has three gates (input, output, and forget gates). The GRU controls the flow of information like the LSTM unit, but without having to use a memory unit. It just exposes the complete hidden content without any control. GRU is computationally more efficient compared to LSTM.

As shown in Figure 5, GRU supports gating and a hidden state to control the flow of information. A GRU has two gates: the update gate and the reset gate. The update gate, z_t, is responsible for determining the amount of previous information from prior time steps that needs to be passed along the next state. The reset gate, r_t, is on the other hand controls how much of the past information is needed to neglect.

3.2 | The designed and trained model

The computational programming language MATLAB (MathWorks Inc, Natick, MA)[45] is used for the implementation of the artificial neural network.

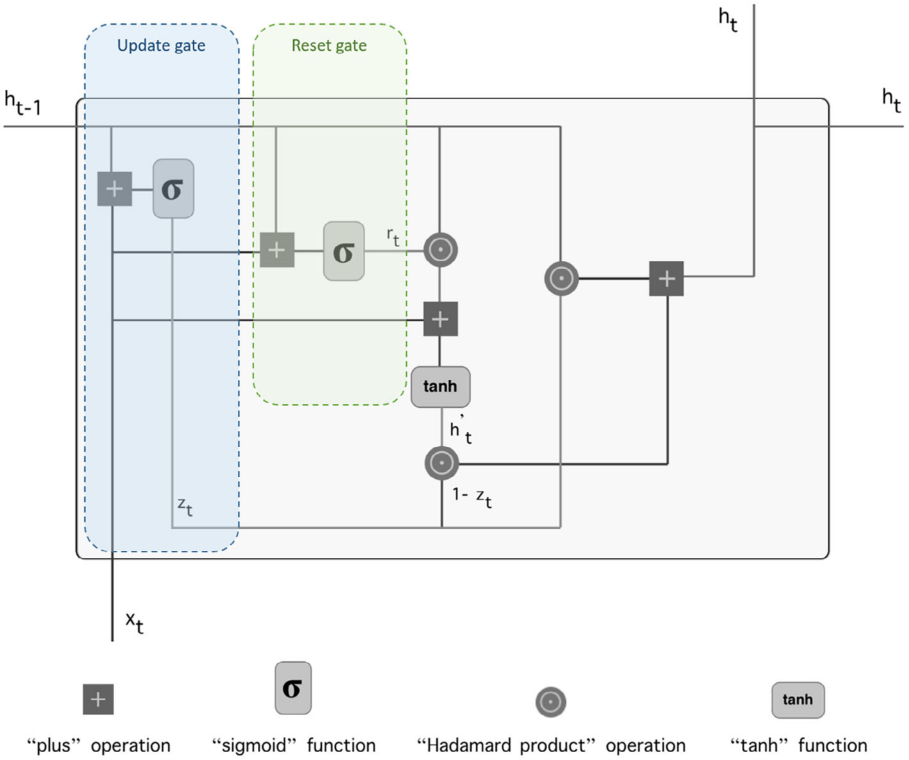

FIGURE 5 Illustration of a gated recurrent unit, where x_t and h_t are input vector and hidden layer information, respectively. The index $t-1$ in h_{t-1} signifies that it holds the information of the previous unit which is multiplied by its weight (Figure adapted from Reference 44).

To capture the path-dependency of plasticity, RNN are utilized. RNNs are a class of neural networks that allow previous outputs to be used as inputs while having hidden states which makes them powerful for modeling sequence data such as time series. An improved version of standard RNN, GRUs,[43] is used as the main neuron type in the network.

The input signals of the network consist of time series of length $N_T = 2001$, with $F = 13$ different features. The features comprise the 6 components of the second order orientation tensor, the fiber volume fraction, and the 6 independent components of the strain tensor at each time step. The time-independent features are simply fed to the network at every time step without modification, while the strain tensor components are changing with time. The output signals are the 6 independent components of the stress tensor. The loss is computed by comparing the output to the stress time series obtained from the micromechanical simulation corresponding to the input signal. We used the ADAM optimizer,[46] a first order gradient-based optimizer with a learning rate of 0.0005.

As illustrated in Figure 6, the network structure begins with an input layer of time sequence data, followed by three GRU layers. The input layer has no activation. In order to give the signal the correct output dimension, and to match the magnitude of the output training data, a fully connected feed-forward layer is placed after the GRU layers. This layer has unit activation such that it can reproduce stresses of arbitrary magnitude and sign. It should be mentioned that two non-linear activation functions are included in GRU units: (i) hyperbolic tangent function for updating hidden state and (ii) the Sigmoid activation function is applied on the gates. For network regularization and overfitting prevention, a dropout layer,[47] with a dropout probability of 50%, is added between the final GRU layer and the feed-forward layer. A fully connected layer with 6 neurons is the output layer (last layer) and it gives output stress time series.

The 40,000 generated data samples are randomly split into training (80%), validation (19.75%) and test (0.25%) sets. The entire training dataset is passed to the same neural network multiple times in an iterative manner (Epochs). At each epoch, the training data is shuffled to ensure that the training process is independent of the structure of the input data. The validation set is used to evaluate the network performance, and to tune the hyper-parameters such as batch size, learning rate and regularization parameters to control the behavior of the model and to achieve the highest performance. The validation set loss at each epoch indicates if the model generalizes its prediction capability to data outside the training set. A validation cost that stops decreasing, or starts to increase, is an indication of network overfitting. In order to

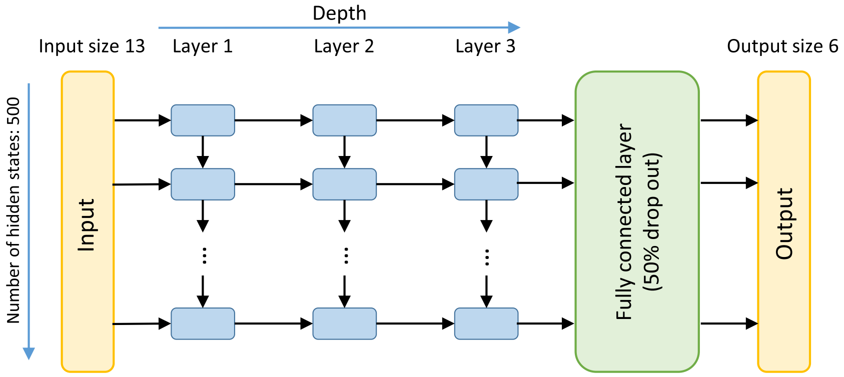

FIGURE 6 The proposed network architecture in which the first layer is the input of time sequence data of dimension 13, followed by three GRU layers and a dropout layer with a dropout probability of 50%. The final layer is a fully connected layer with 6 neurons that emits the output time series.

prevent overfitting, we used an early stopping approach. Early stopping is an optimization technique used to reduce overfitting without compromising model accuracy. The main idea behind early stopping is to stop training once the model performance stops improving on the validation dataset and the model tends to overfit. The batch size is a very important hyper-parameter that influences the convergence rate substantially. Simply taking as large mini-batch size as possible does not necessarily guarantee convergence to the global minimum.[48] During hyper-parameters tuning, a batch size of $N_{\text{batch}} = 32$ is chosen.

In order to improve the prospects of convergence, piece-wise learning rate decay is used. The scheduling is motivated by the following: by setting the learning rate too low from the beginning, convergence may be too slow, or the optimizer gets stuck in a local minimum early. On the other hand, setting a high learning rate might help the optimizer arrive in the vicinity of the global minimum. However, the large step size may prevent the optimizer from making the final stretch without overstepping the minimum, that may be in a small dip in an otherwise flat area of the cost function. By scheduling the learning rate one may get the best of both worlds. The learning rate is scheduled to decrease by 10% every 10 epochs.

Convergence is usually faster if the average of the input variables is close to zero and the input variance is close to one.[49] The input features are therefore subjected to z-score normalization. In order to prevent overfitting we used L_2-regularization,[50] which modifies the cost function such that it penalizes weights of large magnitude. The penalty term is the squared sum of all the weights multiplied by a factor λ that controls the amount of regularization. The value of $\lambda = 0.0001$ is used in the training of the current network. The idea is that weights with large magnitude make the model more complex and reduce its ability to generalize and is therefore penalized. In order to counteract exploding gradients that lead to numerical issues, gradient clipping is used, which re-scales a gradient if its norm is larger than a set threshold value.

For the cost function for the sequence regression, we used mean squared error:

$$C = \frac{1}{N_{\text{batch}}} \sum_{n=1}^{N_{\text{batch}}} C_n, \quad \text{where} \quad C_n = \frac{1}{2N_T} \sum_{i=1}^{F} \sum_{j=1}^{N_T} (\hat{y}_{ij}^n - y_{ij}^n)^2. \tag{18}$$

Here $\hat{y}_{ij}^n$ and y_{ij}^n are the nth prediction and target values in a training batch respectively.

The test set, as unseen data by the network, is used to evaluate the network performance after the training has been finished. The test set is intentionally kept very small in this work since it is only to be used for final validation before proper model testing is carried out. The random nature of the generated paths is not representative of the model use cases. Instead, the model is rigorously tested on more conventional data specifically generated for testing (see Section 4.3.2).

4 | RESULTS

This section begins with introducing a physically relevant error metric for evaluating the performance of the developed ANN model. Following this, the results of applying the trained model on the test set are presented. Thereafter, the obtained model is tested by subjecting a few virtual samples to some representative load cases. Micromechanical simulations are performed in DIGIMAT-MF and the output is compared to the ANN model prediction. The goal is to demonstrate that the proposed ANN model can reproduce mean-field predictions with a good accuracy. The required rate independence of the model is also demonstrated. It is also shown that the ANN model is able to make accurate predictions outside the parameter range which was used for training.

4.1 | Evaluation metrics

Simply referring to the cost defined in Equation (18) as an error metric is not sufficient, since a numerically small cost does not necessarily imply a small error if the scale of the output is also small. To give the error a physical significance, new quantities need to be introduced. To remove the dimensionality, the root mean square error is divided by the yield stress of the matrix (σ_y). This metric is from now on referred to as the mean relative error (MeRE). This metric gives an overall estimate of the model accuracy, it does however not account for large localized errors. Hence, the maximum relative error (MaRE) is also introduced. The MaRE is defined by the maximum absolute error divided by the matrix yield stress. The two error metrics of a time series of length N_T are computed for each stress component through the following equations:

$$\mathrm{MeRE} = \frac{\sqrt{\frac{1}{N_T}\sum_{t=1}^{N_T}(\sigma_t - \hat{\sigma}_t)^2}}{\sigma_y} \quad \text{and} \quad \mathrm{MaRE} = \frac{\max\limits_t |\sigma_t - \hat{\sigma}_t|}{\sigma_y}, \tag{19}$$

where $\hat{\sigma}_t$ is a predicted stress component at step t and σ_t is the desired stress component at step t.

4.2 | Obtained model

The network was trained during 500 epochs on an Nvidia V100 GPU with 32 GB of VRAM. The training took just above 81 hours using 32,000 training samples. At the first training epoch, the model yielded a cost of 12981.8057 MPa^2 on the validation set, which reduced to a final cost of 7.4021 MPa^2 when the training session terminated. A log-log plot of the training- and validation set cost functions is displayed in Figure 7. The fact that validation cost consistently keeps decreasing during training lets us be confident that the trained network does not suffer from overfitting. It may look pathological that the validation cost becomes smaller than the training cost. It is however correct, and has to do with the way MATLAB computes the cost. It computes the cost with dropout on the training data but without dropout on the validation data.

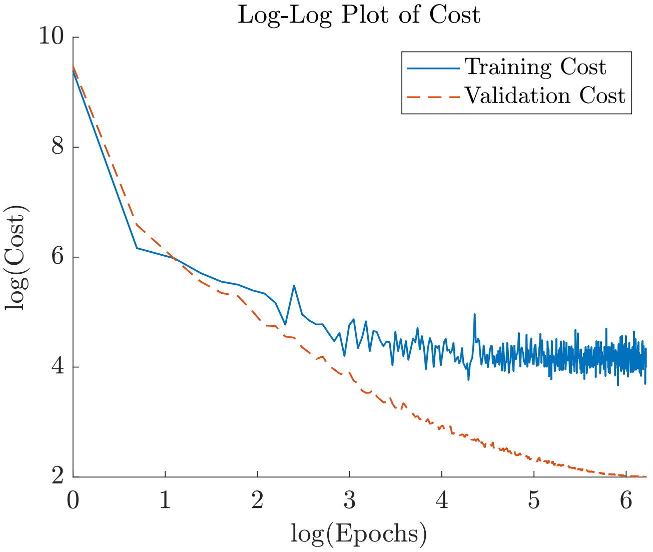

FIGURE 7 A log-log plot of the training- and validation cost functions is shown. The training cost flattens out quite early while the validation cost keeps decreasing.

TABLE 3 The average MeRE and MaRE computed over the test set.

	σ_{11}	σ_{22}	σ_{33}	σ_{12}	σ_{23}	σ_{13}
MeRE	0.0582	0.0528	0.0487	0.0469	0.0378	0.0431
MaRE	0.1255	0.1137	0.1121	0.1020	0.0874	0.0980

TABLE 4 The second order orientation tensors and fiber volume fractions for the virtual samples.

Sample	a_{11}	a_{22}	a_{33}	a_{12}	a_{13}	a_{23}	v_F
1	0.477	0.188	0.335	−0.080	−0.071	−0.183	0.130
2	0.094	0.692	0.214	−0.103	0.012	−0.255	0.144
3	0.649	0.139	0.212	0.011	−0.117	−0.154	0.131
4	0.392	0.225	0.382	−0.142	0.080	0.152	0.139
5	0.000	0.919	0.081	0.015	0.005	0.273	0.109
1D	1.00	0.000	0.000	0.000	0.000	0.000	0.120
2D	0.500	0.500	0.000	0.000	0.000	0.000	0.120
3D	0.333	0.333	0.333	0.000	0.000	0.000	0.120

The average of the MeRE and MaRE is computed over the 100 time series for the test set. The time to make a prediction of a stress time series was always less than 1 second, typically taking about 0.1 s. In comparison, the time to predict the stress response corresponding to the strain signals in the test set using DIGIMAT-MF was typically around a minute. The average errors for all 6 stress components are displayed in Table 3. The errors are small, which further indicates that the model does not suffer from overfitting and generalizes well.

Remark 2. It should be emphasized that the matrix yield stress (25 MPa), which is much lower than the maximum composite stresses obtained in the simulations, is used for defining the error metrics.

4.3 | Virtual sample testing

Five orientation tensors together with a corresponding fiber volume fraction were sampled uniformly according to the previously proposed rules. Additionally, samples with a uniaxial fiber distribution (1D), a random planar fiber distribution (2D), and a spatially uniform fiber distribution (3D) were created. These three samples all had the fiber volume fraction of 12%. The corresponding data related to the samples is presented in Table 4.

In order to demonstrate the general shape of the orientation tensors, ellipsoid representations of two of the orientation tensors are shown in Figure 8.

The stress–strain responses of the samples for some representative stress states were obtained through strain controlled simulations in DIGIMAT-MF. A load cycle is defined by changing the control strains piece-wise monotonically from 0 to ε_c to $-\varepsilon_c$ to 0, where ε_c is the maximum control strain. Only the control strains were strictly imposed, DIGIMAT-MF computed the remaining strain components such that a desired stress state was achieved. The micromechanical simulation acted as a ground truth, and the model error was computed by comparison with the ANN model output. The model used the strains in the simulations as input, together with the respective parameters corresponding to the different virtual samples.

4.3.1 | General evaluation of the model

General testing of the model is performed by running representative load cases against uniformly sampled fiber orientations and volume fractions within the training range. Five different types of loading conditions are simulated for samples 1 to 5 (see Figure 9 and Table 4). All loadings are strain controlled load cycles where the control

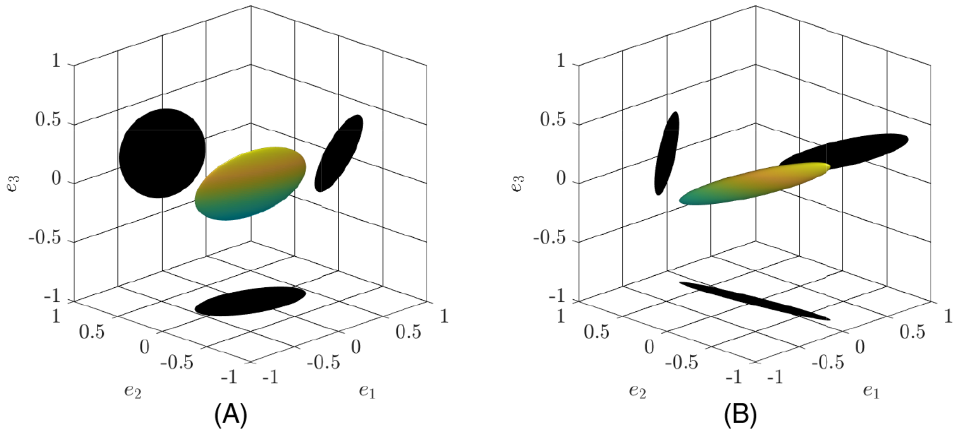

FIGURE 8 The ellipsoid representations of the orientation tensor of Sample 1 (A) and Sample 3 (B) are displayed. The black shadows are the projections on the principal coordinate planes.

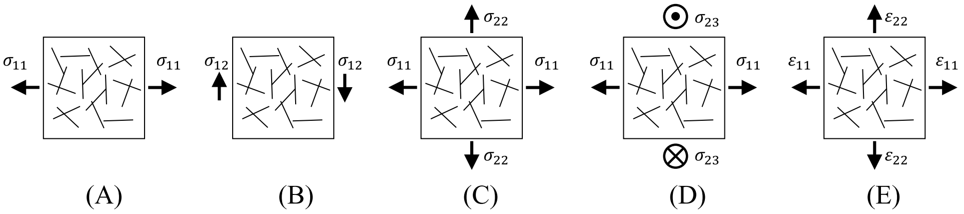

FIGURE 9 Five different cyclic load cases considered for the general evaluation of the model. (A) Uniaxial stress σ_{11}, (B) pure shear stress σ_{12}, (C) biaxial stress $\sigma_{11}+\sigma_{22}$, (D) biaxial stress $\sigma_{11}+\sigma_{23}$ and (E) biaxial plane strain $\varepsilon_{11}+\varepsilon_{22}$.

strains are set to change piece-wise monotonically from 0 to 0.035 to −0.035 to 0, and DIGIMAT-MF computes the other strain components such that the desired stress state is achieved. As a consequence, the stress predictions made by the trained ANN model may not be exactly uniaxial for these cases. As the ANN predictions are strictly driven by the complete multiaxial strain histories resulting from the corresponding DIGIMAT-MF predictions, several non-zero stress components may occur (besides the one in the loading direction). As a means to quantify any deviation from a purely uniaxial stress case, MeRE and MaRE values are therefore reported for each stress component also for these uniaxial cases. Finally, for plane strain loading, the out of plane strain components simply are set to constant 0.

The first type of loading is a uniaxial stress in the σ_{11}-direction, controlled by imposing ε_{11}. The second type of loading is a pure shear stress state in the σ_{12}-direction, controlled by imposing ε_{12}. The third type is a biaxial stress state in the σ_{11}-σ_{22}-directions, controlled by ε_{11} and ε_{22}. The fourth type of loading is a bi-axial stress state in the σ_{11}-σ_{23}-directions, controlled by ε_{11} and ε_{23}. The final loading condition is plane strain in the ε_{11}-ε_{22}-plane. The exhaustive results of all the tests of the five samples is found in the form of the MeRE in Table 5 and MaRE in Table 6. The model shows great promise with a MeRE that never surpasses 25% of the matrix yield stress during the tests. The component wise average MeRE is also low, less than 7% of the matrix yield stress. The largest recorded MaRE was less than 50% of the matrix yield stress, while the component wise average was less than 15%.

4.3.2 | Single cycle tests

Tests consisting of one load cycle were performed on samples 1–5. Five different stress states were investigated, (i) uniaxial stress in the σ_{11}-direction controlled by imposing ε_{11}, (ii) pure shear in the σ_{12}-direction controlled by imposing ε_{12}, (iii) a bi-axial stress state in the σ_{11}-σ_{22}-directions controlled by imposing ε_{11} and ε_{22}, (iv) a bi-axial stress state in the σ_{11}-σ_{23}-directions controlled by imposing ε_{11} and ε_{23}, and finally (v) a plane strain state in the ε_{11}-ε_{22}-plane. The maximum control strain was $\varepsilon_c = 0.035$ for these tests. The number of steps was simply matched with the output of

TABLE 5 MeRE of every stress component for the five samples and five loading conditions together with the feature wise average and maximum MeRE over all tests.

Sample	Load	Steps	σ_{11}	σ_{22}	σ_{33}	σ_{12}	σ_{23}	σ_{13}
1	Uniaxial	804	0.0971	0.0301	0.0391	0.0124	0.0080	0.0155
	Pure shear	808	0.0117	0.0123	0.0177	0.0247	0.0068	0.0077
	Biaxial (σ_{11}-σ_{22})	804	0.0624	0.0510	0.0667	0.0403	0.0481	0.0297
	Biaxial (σ_{11}-σ_{23})	810	0.0955	0.0384	0.0571	0.0805	0.0553	0.1232
	Plane strain	810	0.1217	0.0424	0.0889	0.0683	0.0318	0.0709
2	Uniaxial	804	0.0308	0.0346	0.0288	0.0299	0.0273	0.0148
	Pure shear	802	0.0163	0.0258	0.0226	0.0410	0.0106	0.0329
	Biaxial (σ_{11}-σ_{22})	802	0.0612	0.0781	0.0881	0.0584	0.0456	0.0331
	Biaxial (σ_{11}-σ_{23})	812	0.0552	0.1168	0.0731	0.0443	0.1030	0.0566
	Plane strain	810	0.0143	0.0417	0.0130	0.0285	0.0440	0.0357
3	Uniaxial	808	0.1515	0.0359	0.0503	0.0144	0.0151	0.0144
	Pure shear	804	0.0279	0.0101	0.0139	0.0225	0.0167	0.0209
	Biaxial (σ_{11}-σ_{22})	803	0.1136	0.0515	0.0515	0.0250	0.0357	0.0381
	Biaxial (σ_{11}-σ_{23})	806	0.0953	0.0373	0.0457	0.0449	0.0449	0.1059
	Plane strain	810	0.1252	0.0324	0.0370	0.0792	0.0197	0.0428
4	Uniaxial	804	0.0644	0.0249	0.0265	0.0309	0.0147	0.0190
	Pure shear	805	0.0237	0.0131	0.0208	0.0415	0.0119	0.0062
	Biaxial (σ_{11}-σ_{22})	803	0.0666	0.0630	0.1209	0.0423	0.0549	0.0577
	Biaxial (σ_{11}-σ_{23})	805	0.1073	0.0624	0.0475	0.2442	0.0091	0.0806
	Plane strain	810	0.0473	0.0356	0.0255	0.0320	0.0336	0.0279
5	Uniaxial	802	0.0459	0.0301	0.0349	0.0142	0.0278	0.0079
	Pure shear	808	0.0140	0.0121	0.0220	0.0242	0.0100	0.0237
	Biaxial (σ_{11}-σ_{22})	809	0.0862	0.2027	0.1096	0.0922	0.0825	0.0347
	Biaxial (σ_{11}-σ_{23})	807	0.0417	0.0307	0.0297	0.0180	0.0377	0.0119
	Plane strain	810	0.0423	0.0570	0.0323	0.0262	0.0725	0.0134
Average			0.0648	0.0468	0.0465	0.0472	0.0347	0.0370
Maximum			0.1515	0.2027	0.1209	0.2442	0.1030	0.1232

the micromechanical simulations. However, to investigate the rate-independence, the proposed tests were repeated with varying degree of linear interpolation/extrapolation in time on the stress–strain data.

The resulting stress–strain curves from the tests show very good agreements between the network predictions and micromechanical simulations. As some representative results, two stress–strain curves for (a) uniaxial test performed on sample 1, and (b) biaxial test performed on sample 5 are shown in Figure 10.

Remark 3. Since the network is not trained on uniaxial stress, it is of interest to confirm if it could properly handle this load case. The non-trivial strain path was computed by Digimat. For example, when uniaxial stress in the 11 direction was desired, the strain path in 11 was imposed, and the remaining components are computed such that the uniaxial stress state is achieved.

These curves highlight that the developed ANN model captures well the fundamental characteristics of the underlying model. Specifically, the model correctly displays a hardening process that saturates for large plastics strains, and the expected loading/unloading behavior.

TABLE 6 MaRE of every stress component for the five samples and five loading conditions together with the feature wise average and maximum MaRE over all tests.

Sample	Load	Steps	σ_{11}	σ_{22}	σ_{33}	σ_{12}	σ_{23}	σ_{13}
1	Uniaxial	804	0.1764	0.0541	0.0843	0.0375	0.0336	0.0503
	Pure shear	808	0.0225	0.0305	0.0477	0.0614	0.0166	0.0230
	Biaxial (σ_{11}-σ_{22})	804	0.2109	0.1195	0.1520	0.0795	0.1030	0.0643
	Biaxial (σ_{11}-σ_{23})	810	0.1340	0.0980	0.1809	0.1605	0.1789	0.2246
	Plane strain	808	0.0225	0.0305	0.0477	0.0614	0.0166	0.0230
2	Uniaxial	804	0.0621	0.0706	0.0534	0.0640	0.0426	0.0394
	Pure shear	802	0.0434	0.0689	0.0434	0.0823	0.0254	0.0699
	Biaxial (σ_{11}-σ_{22})	802	0.1998	0.1842	0.2319	0.1377	0.0841	0.0963
	Biaxial (σ_{11}-σ_{23})	812	0.1140	0.2093	0.1177	0.1073	0.1977	0.1199
	Plane strain	810	0.0418	0.1026	0.0335	0.0485	0.0746	0.0709
3	Uniaxial	808	0.2652	0.0795	0.0968	0.0386	0.0370	0.0267
	Pure shear	804	0.0524	0.0309	0.0418	0.0532	0.0298	0.0397
	Biaxial (σ_{11}-σ_{22})	803	0.3613	0.1071	0.1443	0.0512	0.0738	0.0792
	Biaxial (σ_{11}-σ_{23})	806	0.1742	0.0764	0.0936	0.0974	0.1523	0.1741
	Plane strain	810	0.2292	0.0751	0.0804	0.1409	0.0357	0.0822
4	Uniaxial	804	0.1288	0.0513	0.0686	0.0523	0.0305	0.0336
	Pure shear	804	0.0525	0.0374	0.0443	0.0948	0.0224	0.0108
	Biaxial (σ_{11}-σ_{22})	803	0.2014	0.1659	0.3256	0.0822	0.1505	0.1411
	Biaxial (σ_{11}-σ_{23})	805	0.2184	0.1374	0.0942	0.4278	0.0304	0.1527
	Plane strain	810	0.0982	0.0631	0.0672	0.0580	0.0766	0.0559
5	Uniaxial	802	0.0849	0.0991	0.1197	0.0313	0.0592	0.0164
	Pure shear	808	0.0262	0.0378	0.0488	0.0700	0.0174	0.0556
	Biaxial (σ_{11}-σ_{22})	809	0.1745	0.4861	0.2483	0.1759	0.2170	0.0649
	Biaxial (σ_{11}-σ_{23})	807	0.1111	0.1050	0.0705	0.0455	0.1024	0.0250
	Plane strain	810	0.1129	0.1654	0.0640	0.0783	0.1540	0.0355
Average			0.1327	0.1074	0.1040	0.0935	0.0785	0.0712
Maximum			0.3613	0.4861	0.3256	0.4278	0.2170	0.2246

A demonstrative stress time series of all six stress components for a plane strain test is presented in Figure 11. This figure also shows good agreement between the network predictions and the micromechanical simulations. The prediction of the σ_{13} looks poor, however by noting the scale of the stress it is apparent that the relative error is still small. The model shows great promise with a MeRE that never surpasses 25% of the matrix yield stress during the tests. The component-wise average MeRE is also low, less than 7% of the matrix yield stress. The largest recorded MaRE was less than 50% of the matrix yield stress, while the component-wise average was less than 15%.

Figure 12 shows the typical behavior of the MeRE and MaRE when the number of steps in the time series is varied. It is seen that the errors stay almost constant in the range between 200 and 20,000 steps. But it is quickly growing for sequences shorter or longer than these bounds. This holds true for most of the tests, but some tests showed stricter bounds for a constant error. None of the performed tests showed errors that varied to a significant degree when loaded with sequences of lengths between 500 and 8000 steps.

The limits on the sequence lengths can be explained by the structure of the training data. As the algorithm that was used to generate the training data has a theoretical maximum strain rate, and an average strain rate that is considerably lower than the maximum, it's reasonable that there is a minimum sequence length for which the neural network model

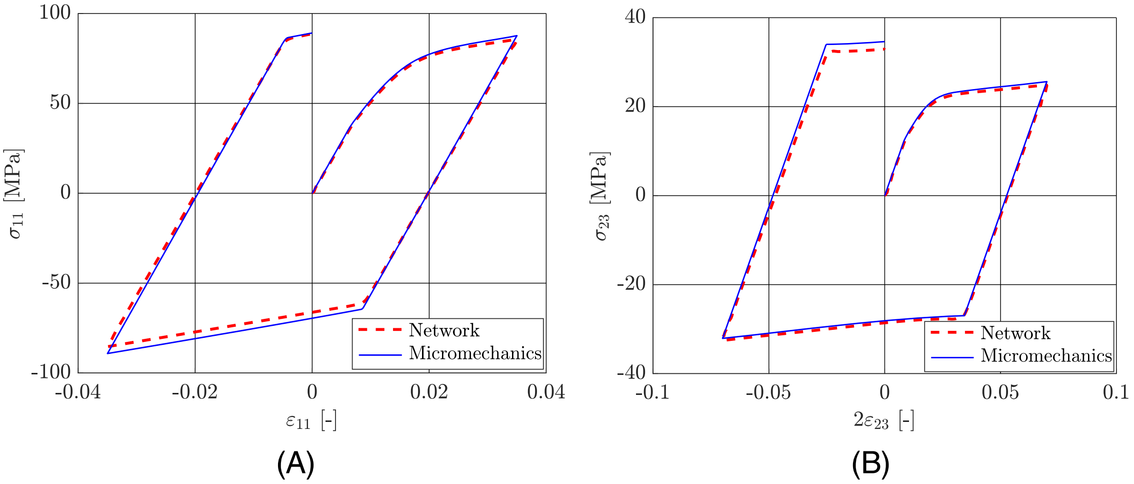

FIGURE 10 Comparison of the stress–strain predictions by the network with micromechanics simulations for (A) uniaxial test performed on sample 1, (B) σ_{11}-σ_{23} biaxial test performed on sample 5.

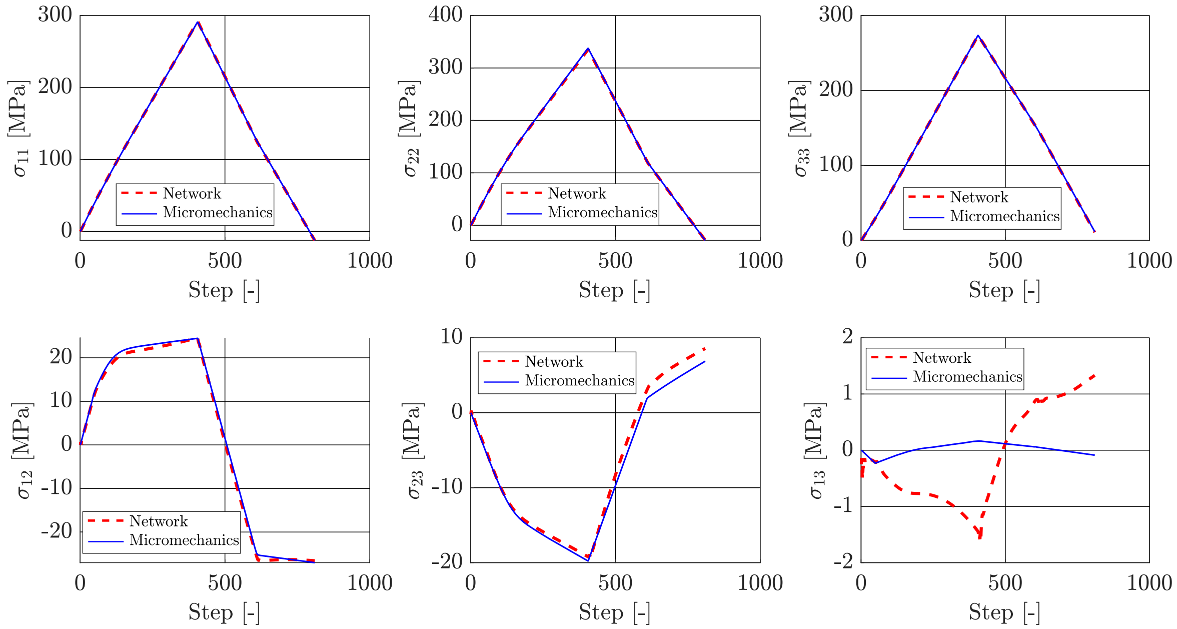

FIGURE 11 Comparison of the network predictions with micromechanics simulations for the time series data of all 6 stress components for a plane strain state load cycle on sample 5.

can make accurate predictions. This explains the lower bound. Since the sequence length of the training data is fixed, the network is not trained to remember the accumulated plastic strain forever. The loss of accuracy for very long sequences can possibly be explained by the network forgetting about plastic strain accumulated early in the sequence when it approaches the sequence's end. Bearing this in mind, it may still be said that the network model displays rate-independence for a large range of sequence lengths.

4.3.3 | Repeated cyclical loading

To test how complex the load histories can be without forgoing too much accuracy, a test with an increasing number of load cycles was performed. Samples 1D, 2D, and 3D were subjected to a uniaxial stress state in the σ_{11}-direction. The

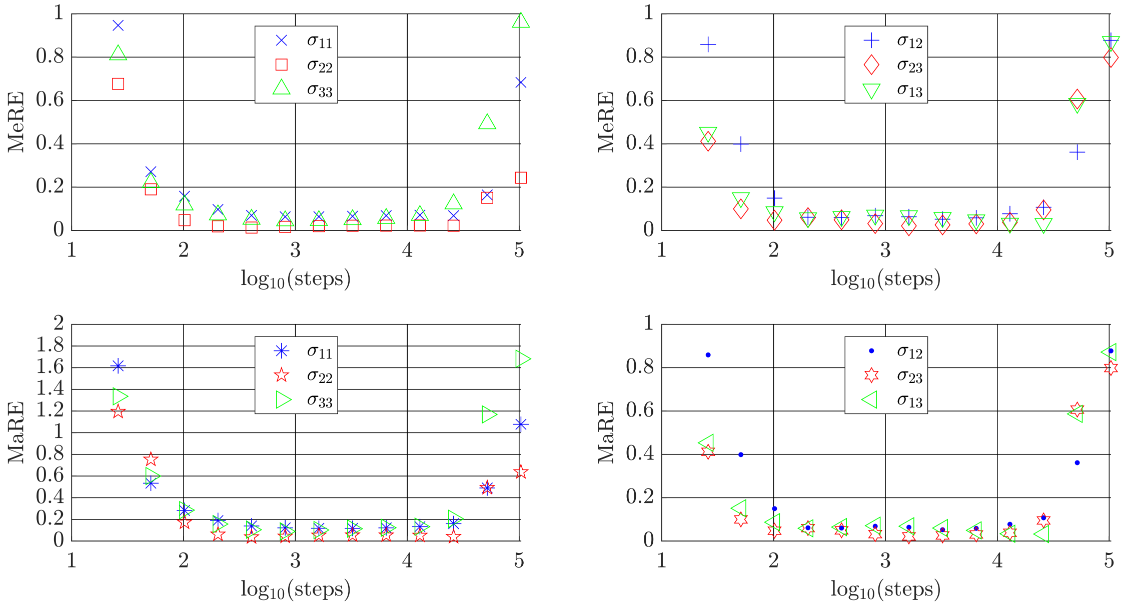

FIGURE 12 Dependency of the error to sequence length (for a plane strain load cycle on sample 1).

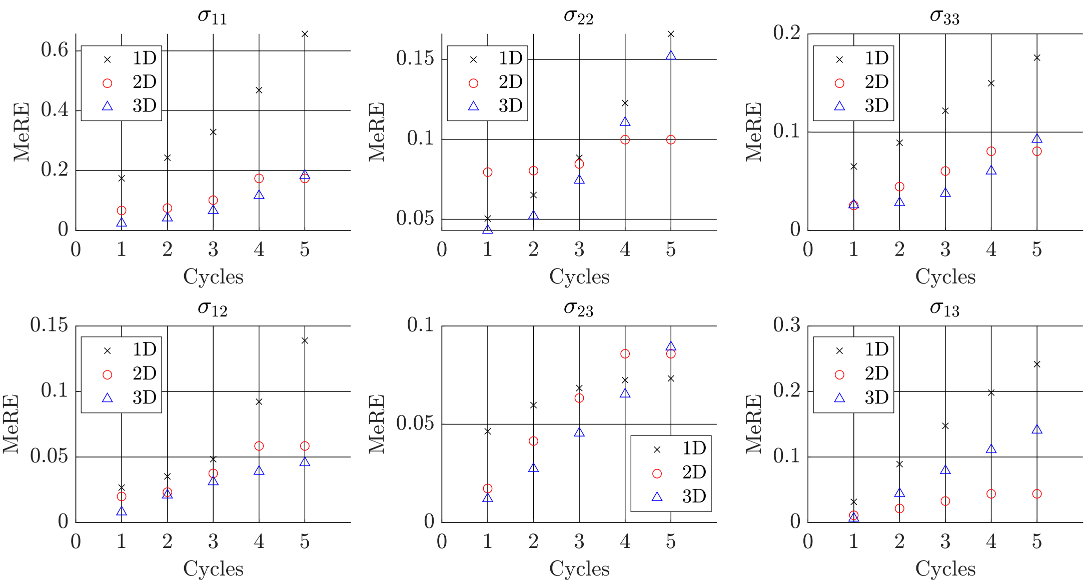

FIGURE 13 MeRE of cyclical tests on unidirectional (1D), random planar (2D), and spatially uniform (3D) samples as a function of the number of cycles.

maximum control strain was $\varepsilon_c = 0.04$, and tests of 1 to 5 cycles were performed. The calculated MeRE corresponding to each of the six strain components as a function of the number of load cycles is displayed in Figure 13, while the MaRE is found in Figure 14. The component-wise errors for a uniaxial fiber distribution, a uniform 2D distribution, and a uniform 3D distributions are given. The error is found to increase almost linearly with the number of load cycles. It can be seen that the uniaxial fiber distribution typically suffers from the greatest error. The sequence length is proportional to the number of cycles, where a single cycle consists of approximately 800 steps. Since it was shown in the previous section that the error of sequences between approximately 500 and 8000 steps is not affected by changing the sequence length, it can be concluded that the error stems from the complexity of the load path.

Two representative stress–strain curves from the conducted simulations are displayed in Figure 15. It is possible to see how the prediction error increases slightly for every additional cycle. It is also interesting to see that once yielding has

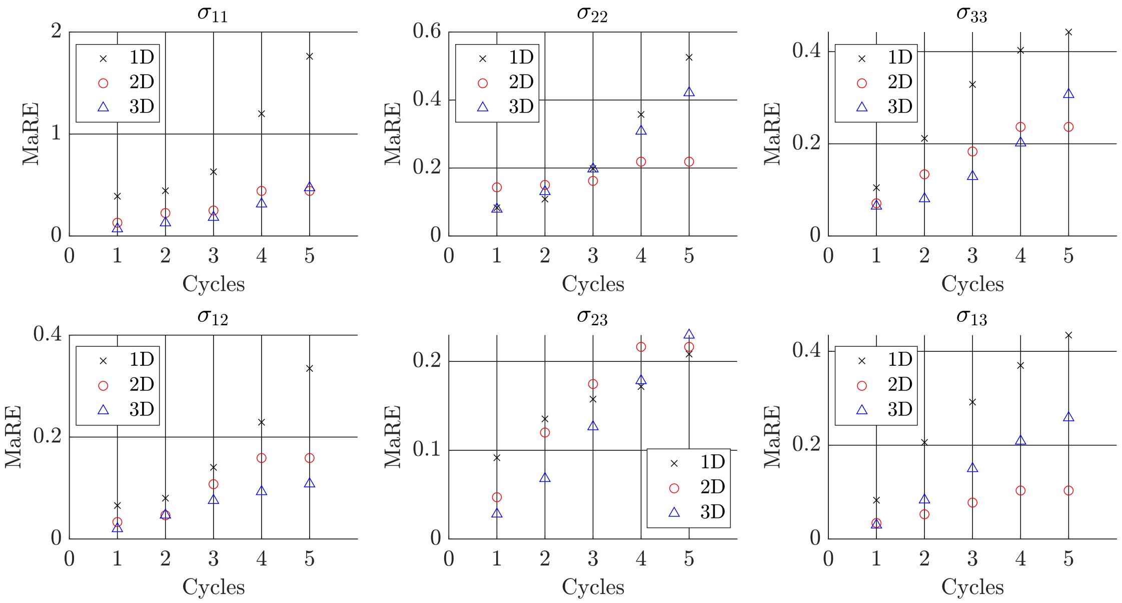

FIGURE 14 MaRE of cyclical tests on unidirectional (1D), random planar (2D), and spatially uniform (3D) samples as a function of the number of cycles.

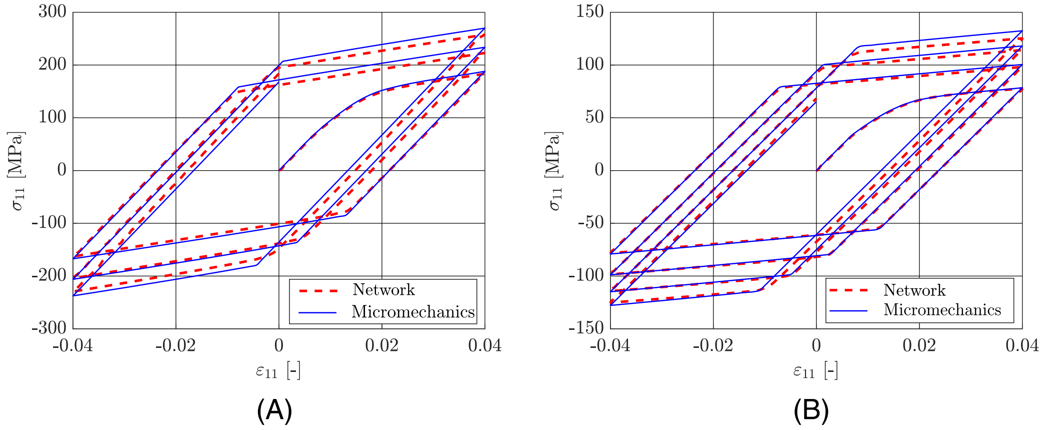

FIGURE 15 Comparisons of the stress–strain curves obtained by the network and micromechanics simulations for uniaxial tests performed on samples with (A) a unidirectional fiber distribution, and (B) a uniform 3D fiber distribution.

occurred, the error seems to stay fairly constant. This speaks in favor of the model accurately capturing material properties such as the tangent modulus.

4.4 | Extrapolation ability

An additional set of tests consisting of a uniaxial load cycle was performed on a sample with a random 3D fiber distribution. Fiber volume fractions of 0.1% 2.5%, 5%, 7.5%, 10%, 12.5%, 15%, 17.5%, and 20% were investigated. For each volume fraction a maximum control strain (ε_c) of 5%, 7.5%, and 10% was investigated.

The computed errors in the prediction of the σ_{11} component, for different fiber volume fractions and different maximum control strains, are shown in Figure 16. It seems like for fiber volume fractions in the permissible range (10%–15%), strains between 5% and 7.5% do not lead to a very significant increase in the error. However, strains between 7.5% and 10% lead to a larger increase in the error for admissible fiber volume fractions. It can also be seen that the model generalizes very well to fiber volume fractions lower than 10% and between 15% and 20% for strains in the admissible range (≤5%).

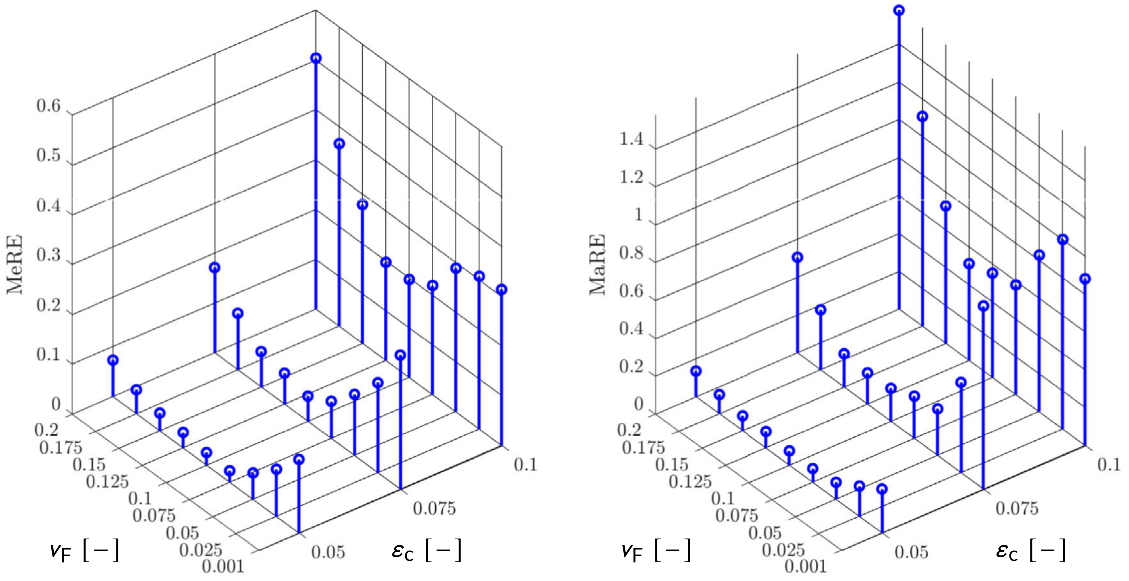

FIGURE 16 The error (MeRE and MaRE) in the σ_{11} component as a function of maximum control strain and fiber volume fraction.

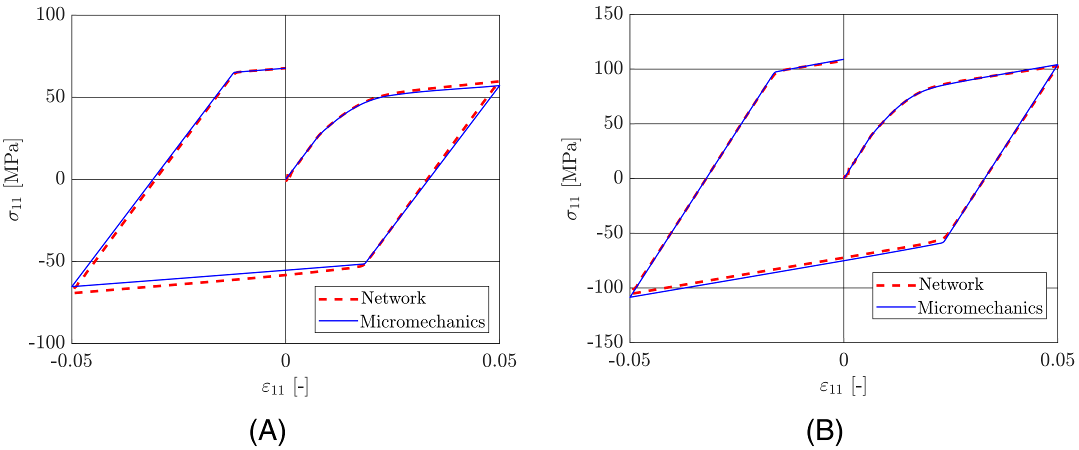

FIGURE 17 Comparison of stress–strain curves obtained by the network and in micromechanical simulations, (A) a uniaxial test performed on a sample with a fiber volume fraction of 2.5%, (B) a uniaxial test performed on a sample with fiber volume fraction of 20%.

The impressive extrapolation ability of the model for the admissible strains can be seen in the plot of two representative stress–strain curves in Figure 17. On the other hand, the detrimental effects on the prediction error for maximum strains larger than 5% are amplified for fiber volume fractions outside the admissible range.

5 | DISCUSSION

The acquired neural network, albeit exhibiting good performance, can not without a doubt be said to be optimal. In order to achieve an optimal network architecture, the hyper-parameters would need to be investigated more systematically. The complexity of the problem however results in very long network training times. A detailed survey of the hyper-parameters would require more dedicated hardware to enable parallel training of networks with multiple sets of parameters. It is possible to argue for the number of layers and neurons being appropriate. Previous attempts at predicting elasto-plasticity with GRU's showed that networks with more than 3 layers and 500 neurons per layer did not display any significant improvement in performance.[11]

The chosen method of generating training data may be deemed successful. The created network shows good performance, and it seems that it is able to accurately predict the response to a wide variety of mechanical loads. There is however room for improvement. One obvious thing to investigate is the inclusion of characteristic training data that represent physically relevant phenomena such as hydrostatic loading. Alternatively, in future investigations it would be of interest to consider more smooth strain paths for training. This has been successfully implemented by others, using Gaussian processes.[23]

A natural question to ask is what added value the proposed neural network model offers compared to the micromechanical model that it was trained on. To begin with, the proposed model offers the ability to change fiber orientation, and fiber volume fraction without having to perform additional simulations to homogenize the composite for the new set of parameters. This is important in the context of injection molding where the fiber density and orientation vary highly throughout the molded part. Second, the neural network model offers computational speed. As an example, using DIGIMAT-MF to compute the response to complex general 3D-loading (like the ones in the training data) may take up to a minute. In contrast, the network model makes the prediction in less than a second. Finally, the network contains all the information on loading/unloading and accumulated plastic strain/hardening purely through the strain history. Therefore, the process of determining these quantities repeatedly is not necessary. As a consequence, a lot of time is saved if the model is used in a framework where iterative calculations are required. This is important when implementing the network as a constitutive model in an FEM setting. It should be mentioned that the ANN model is capable of calculating the tangent stiffness for the surrogate model. At each time step, all the required derivatives, for the calculations of the tangent stiffness, are available in the RNN. It was however out of the scope of this study to conduct coupled multi-scale simulations.

In the present work, an ANN model has been trained on micromechanical simulations with the hope of the model being able to infer the physics of elasto-plasticity purely by observing stress–strain relations. As an alternative, a network may be trained with physically motivated constraints in order to ensure laws such as energy conservation. Linka et al.[51] introduced constitutive artificial neural networks. These networks first compute the variants of the strain tensor and combine them with micromechanical descriptors to compute a set of generalized invariants. The generalized invariants are in turn used to compute a strain energy functional. The stress and constitutive tensor are then easily obtained from the strain energy through differentiation. By computing the stress from an energy functional it is ensured that certain physical laws remain unviolated. The network is trained as usual with a strain input, with the goal of matching a predetermined stress–strain relation. One perk with that approach, in addition to certain laws of physics already being built into the network structure, is the reduced need for training data. As the model proposed in the present work yields some physically dubious results, such as accumulating plastic strain during hydrostatic loading, it would be of interest to investigate the possibility of introducing mechanically motivated constraints into future extensions of the network.

6 | CONCLUSIONS

In this study, a deep neural network model that predicts the elasto-plastic response of a SFRCs was developed. The material consists of elastic fibers with a J_2 elastoplastic matrix obeying a linear exponential hardening law. The network was trained on data from micromechanical simulations utilizing the Mori-Tanaka mean-field method for homogenization. The strain-paths used as input to the simulations were generated by utilizing a random walk in a 6D-strain space with bias directions. The created model allows for an arbitrary fiber distribution by defining a second order fiber orientation tensor. Additionally, it is possible to vary the fiber volume fraction between 10% and 15%, but it is possible to go slightly outside these bounds without introducing significant error.

The proposed model is computationally efficient and shows promising results. The model makes accurate predictions for a wide range of fiber orientation distributions and loading types. The model displays accurate yielding behavior with the appropriate hardening law (including saturation), and shows a clear difference in loading/unloading. It was furthermore demonstrated that the model correctly possesses rate-independence for a large range of loading rates. The developed model is a groundwork for general 3D micromechanics-based modeling of complex path-dependent behavior of SFRCs. In the next steps, this model will be extended by supplementary data for other matrices and fibers, and eventually by RVE finite element simulations for the sake of higher accuracy. Moreover, the possibility of improving the modeling by utilizing physical constraints in the model seems very promising. As an additional future outlook, a finite element implementation of the model is of interest.

ACKNOWLEDGMENTS

S. M. Mirkhalaf is thankful for the financial support from the Swedish Research Council (VR Grant: 2019-04715). M. Fagerström gratefully acknowledges the financial support through Vinnova's strategic innovation programme LIGHTer. Also, the authors would like to thank e-Xstream for providing a license of software Digimat.

DATA AVAILABILITY STATEMENT

The codes developed for this study, and some of the data, are available on Github (https://github.com/SM-Mirkhalaf/Elastoplastic-SFRCs-ANN-model). More data will be provided to an interested reader upon request.

ORCID

S. M. Mirkhalaf https://orcid.org/0000-0002-3735-5791

REFERENCES

1. Pan Y, Iorga L, Pelegri AA. Numerical generation of a random chopped fiber composite RVE and its elastic properties. *Compos Sci Technol*. 2008;68(13):2792-2798.
2. Bargmann S, Klusemann B, Markmann J, et al. Generation of 3D representative volume elements for heterogeneous materials: a review. *Prog Mater Sci*. 2018;96:322-384.
3. Mirkhalaf SM, Eggels EH, Anantharanga AT, Larsson F, Fagerström M. Short fiber composites: computational homogenization vs orientation averaging. Proceedings of the 2019 International Conference on Composite Materials Melbourne, Australia; 2019.
4. Mirkhalaf SM, Eggels EH, van Beurden TJH, Larsson F, Fagerström M. A finite element based orientation averaging method for predicting elastic properties of short fiber reinforced composites. *Compos B Eng*. 2020;202:108388.
5. Doghri I, Tinel L. Micromechanical modeling and computation of elasto-plastic materials reinforced with distributed-orientation fibers. *Int J Plast*. 2005;21(10):1919-1940.
6. Mirkhalaf SM, van Beurden TJH, Ekh M, Larsson F, Fagerström M. An FE-based orientation averaging model for elasto-plastic behavior of short fiber composites. *Int J Mech Sci*. 2022;219:107097.
7. Castricum BA, Mirkhalaf SM, Fagerström M, Larsson F. A hierarchical coupled multi-scale model for short fiber composites. Proceedings of the World Congress in Computational Mechanics and ECCOMAS Congress; 2021.
8. Le BA, Yvonnet J, He QC. Computational homogenization of nonlinear elastic materials using neural networks. *Int J Numer Methods Eng*. 2015;104(12):1061-1084. doi:10.1002/nme.4953
9. Lu X, Giovanis DG, Yvonnet J, Papadopoulos V, Detrez F, Bai J. A data-driven computational homogenization method based on neural networks for the nonlinear anisotropic electrical response of graphene/polymer nanocomposites. *Comput Mech*. 2019;64(2):307-321.
10. Wei H, Zhao S, Rong Q, Bao H. Predicting the effective thermal conductivities of composite materials and porous media by machine learning methods. *Int J Heat Mass Transf*. 2018;127:908-916.
11. Mozaffar M, Bostanabad R, Chen W, Ehmann K, Cao J, Bessa MA. Deep learning predicts path-dependent plasticity. *Proc Natl Acad Sci U.S.A.* 2019;116(52):26414-26420.
12. Mentges N, Dashtbozorg B, Mirkhalaf SM. A micromechanics-based artificial neural networks model for elastic properties of short fiber composites. *Compos B Eng*. 2021;213:108736.
13. Liu X, Tian S, Tao F, Yu W. A review of artificial neural networks in the constitutive modeling of composite materials. *Compos B Eng*. 2021;224:109152.
14. Hahn HT, Tsai SW. Nonlinear elastic behavior of unidirectional composite laminae. *J Compos Mater*. 1973;7(1):102-118.
15. Huang D, Fuhg JN, Weißenfels C, Wriggers P. A machine learning based plasticity model using proper orthogonal decomposition. *Comput Methods Appl Mech Eng*. 2020;365:113008.
16. Zhang A, Mohr D. Using neural networks to represent von Mises plasticity with isotropic hardening. *Int J Plast*. 2020;132:102732.
17. Gorji MB, Mozaffar M, Heidenreich JN, Cao J, Mohr D. On the potential of recurrent neural networks for modeling path dependent plasticity. *J Mech Phys Solids*. 2020;143:103972. https://www.sciencedirect.com/science/article/pii/S0022509620302076
18. Bonatti C, Mohr D. On the importance of self-consistency in recurrent neural network models representing elasto-plastic solids. *J Mech Phys Solids*. 2022;158:104697. https://www.sciencedirect.com/science/article/pii/S0022509621003161
19. Wu L, Nguyen VD, Kilingar NG, Noels L. A recurrent neural network-accelerated multi-scale model for elasto-plastic heterogeneous materials subjected to random cyclic and non-proportional loading paths. *Comput Methods Appl Mech Eng*. 2020;369:113234.
20. Wu L, Noels L. Recurrent neural networks (RNNs) with dimensionality reduction and break down in computational mechanics; application to multi-scale localization step. *Comput Methods Appl Mech Eng*. 2022;390:114476.
21. Kammoun S, Doghri I, Adam L, Robert G, Delannay L. First pseudo-grain failure model for inelastic composites with misaligned short fibers. *Compos A Appl Sci Manuf*. 2011;42(12):1892-1902.
22. Wu S, Wang B, Zheng G, et al. Preparation and characterization of macroscopically electrospun polyamide 66 nanofiber bundles. *Mater Lett*. 2014;124:77-80.
23. Logarzo HJ, Capuano G, Rimoli JJ. Smart constitutive laws: inelastic homogenization through machine learning. *Comput Methods in Appl Mech Eng*. 2021;373:113482.
24. Advani SG, Tucker CL. The use of tensors to describe and predict fiber orientation in short fiber composites. *J Rheol*. 1987;31(8):751-784.

25. Devroye L. *Non-Uniform Random Variate Generation*. Springer-Verlag; 1986.
26. Arvo J. Fast random rotation matrices. In: Kirk D, ed. *Graphics Gems III (IBM Version)*. Elsevier; 1992:117-120.
27. Brannon RM. *Rotation, Reflection, and Frame Changes*. IOP Publishing; 2018:2053-2563. doi:10.1088/978-0-7503-1454-1
28. Doghri I, Brassart L, Adam L, Gérard JS. A second-moment incremental formulation for the mean-field homogenization of elasto-plastic composites. *Int J Plast*. 2011;27(3):352-371.
29. Selmi A, Doghri I, Adam L. Micromechanical simulations of biaxial yield, hardening and plastic flow in short glass fiber reinforced polyamide. *Int J Mech Sci*. 2011;53(9):696-706.
30. Mirkhalaf M, van Dommelen JAW, Govaert LE, Furmanski J, Geers MGD. Micromechanical modeling of anisotropic behavior of oriented semicrystalline polymers. *J Polym Sci B Polym Phys*. 2019;57(7):378-391.
31. Heidari-Rarani M, Bashandeh-Khodaei-Naeini K, Mirkhalaf SM. Micromechanical modeling of the mechanical behavior of unidirectional composites—a comparative study. *J Reinf Plast Compos*. 2018;37(16):1051-1071.
32. Eshelby JD, Peierls RE. The determination of the elastic field of an ellipsoidal inclusion, and related problems. *Proc R Soc Lond A*. 1957;241(1226):376-396.
33. Hashin Z, Shtrikman S. On some variational principles in anisotropic and nonhomogeneous elasticity. *J Mech Phys Solids*. 1962;10(4):335-342.
34. Hashin Z, Shtrikman S. A variational approach to the theory of the elastic behaviour of multiphase materials. *J Mech Phys Solids*. 1963;11(2):127-140.
35. Hill R. A self-consistent mechanics of composite materials. *J Mech Phys Solids*. 1965;13(4):213-222.
36. Budiansky B. On the elastic moduli of some heterogeneous materials. *J Mech Phys Solids*. 1965;13(4):223-227.
37. Mori T, Tanaka K. Average stress in matrix and average elastic energy of materials with misfitting inclusions. *Acta Metall*. 1973;21(5):571-574.
38. Hessman PA, Welschinger F, Hornberger K, Böhlke T. On mean field homogenization schemes for short fiber reinforced composites: unified formulation, application and benchmark. *Int J Solids Struct*. 2021;230-231:111141.
39. eX stream. Digimat user's manual, MSC software, Belgium SA; 2020.
40. The Julia Programming Language. Accessed: January 27, 2020. https://julialang.org/
41. Hochreiter S, Schmidhuber J. Long short-term memory. *Neural Comput*. 1997;9(8):1735-1780.
42. Gers FA, Schmidhuber J, Cummins F. Learning to forget: continual prediction with LSTM. *Neural Comput*. 2000;12(10):2451-2471.
43. Cho K, van Merriënboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder–decoder for statistical machine translation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) Doha, Qatar, Association for Computational Linguistics; 2014:1724–1734.
44. El-Amir H, Hamdy M. Sequential models. *Deep Learning Pipeline*. Springer; 2020:415-446.
45. MATLAB version 9.9.0.1538559 (R2020b) Update 3. The Mathworks, Inc., Natick, MA; 2020.
46. Kingma D, Ba J. Adam: a method for stochastic optimization. Proceeding of the International Conference for Learning Representations; 2015.
47. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. *J Mach Learn Res*. 2014;15(56):1929-1958.
48. Keskar NS, Mudigere D, Nocedal J, Smelyanskiy M, Tang PTP. On large-batch training for deep learning: generalization gap and sharp minima. arXiv preprint arXiv:1609.04836, 2017.
49. LeCun YA, Bottou L, Orr GB, Müller KR. Efficient BackProp. In: Montavon G, Orr GB, Müller K-R, eds. *Neural Networks: Tricks of the Trade*. Lecture Notes in Computer Science. Vol 7700. Springer; 2012:9-48.
50. Murphy K. *Machine Learning: A Probabilistic Perspective*. MIT Press; 2012.
51. Linka K, Hillgärtner M, Abdolazizi KP, Aydin RC, Itskov M, Cyron CJ. Constitutive artificial neural networks: a fast and general approach to predictive data-driven constitutive modeling by deep learning. *J Comput Phys*. 2020;429:110010.

How to cite this article: Friemann J, Dashtbozorg B, Fagerström M, Mirkhalaf SM. A micromechanics-based recurrent neural networks model for path-dependent cyclic deformation of short fiber composites. *Int J Numer Methods Eng*. 2023;1-23. doi: 10.1002/nme.7211