Received: 15 July 2023 | Revised: 3 November 2023 | Accepted: 27 November 2023

DOI: 10.1002/nag.3668

RESEARCH ARTICLE

WILEY

A machine-learning supported multi-scale LBM-TPM model of unsaturated, anisotropic, and deformable porous materials

Mohamad Chaaban[1] | **Yousef Heider**[1,2] | **WaiChing Sun**[3] | **Bernd Markert**[1]

[1]Institute of General Mechanics, RWTH Aachen University, Aachen, Germany

[2]Institute of Mechanics and Computational Mechanics (IBNM), Leibniz University Hannover, Hannover, Germany

[3]Department of Civil Engineering and Engineering Mechanics, Columbia University, New York, New York, USA

Correspondence
Yousef Heider, Institute of General Mechanics, RWTH Aachen University, Aachen, Germany.
Email: heider@iam.rwth-aachen.de

Funding information
Deutsche Forschungsgemeinschaft, Grant/Award Number: 458375627

Abstract

The purpose of this paper is to investigate the utilization of artificial neural networks (ANNs) in learning models that address the nonlinear anisotropic flow and hysteresis retention behavior of deformable porous materials. Herein, the micro-geometries of various networks of porous Bentheimer Sandstones subjected to several degrees of strain from the literature are considered. For the generation of the database required for the training, validation, and testing of the machine learning (ML) models, single-phase and biphasic lattice Boltzmann (LB) simulations are performed. The anisotropic nature of the intrinsic permeability is investigated for the single-phase LB simulations. Thereafter, the database contains the computed average fluid velocities versus the pressure gradients. In this database, the range of applied fluid pressure gradients includes Darcy as well as non-Darcy flows. The generated output from the single-phase flow simulations is implemented in a feed-forward neural network, representing a path-independent informed graph-based model. Concerning the two-phase LB simulations, the Shan-Chen multiphase LB model is used to generate the retention curves of the cyclic drying/wetting processes in the deformed porous networks. Consequently, two different ML path-dependent approaches, that is, 1D convolutional neural network and the recurrent neural network, are used to model the biphasic flow through the deformable porous materials. A comparison in terms of accuracy and speed of training between the two approaches is presented. Conclusively, the outcomes of the papers show the capability of the ML models in representing constitutive relations for permeability and hysteretic retention curves accurately and efficiently.

KEYWORDS
anisotropic permeability, convolutional neural network, hysteretic retention curve, lattice Boltzmann method, multiphase fluid flow, recurrent neural network

1 | INTRODUCTION

The multiscale nature of hydro-mechanical behaviors of multiphase porous materials often requires incorporating microscopic information into the macroscopic constitutive modeling, which is overlooked in most common material models that rely on phenomenological or experimental observations, see, for example [1, 2] for review. As such, predicting the intrinsic permeability or retention behavior from the microstructure of the porous material has been an essential task for numerous science and engineering disciplines, such as geology, materials science, biomedical engineering, and rock mechanics. Lower-scale simulations based on the finite element, the finite volume, the lattice Boltzmann method (LBM), or the fast Fourier transformation could be implemented to real micro geometry (e.g., micro-CT images) or synthetic equivalent geometry to obtain effective macroscopic parameters or material models, see, for example [3–5]. These image-based multiscale approaches and the related inverse problems are well-established non-destructive techniques to extrapolate material properties, especially in Digital Rock Physics, see, for example [6].

While image-based multiscale 3D simulations are much cheaper than physical tests and achieve reliable constitutive information, they remain computationally expensive, especially when it comes to online multiscale schemes with microscale simulations at each integration point and time step of the macroscopic continuum model.[3,7] This paves the way for the involvement of more efficient approaches, such as artificial intelligence techniques using, for example, deep neural networks (DNN), to bridge the gap between the different scales. Building DNN-based constitutive models based on the lower-scale simulations can be found in different works, such as within crystal plasticity,[8,9] elasto-plasticity for composite materials,[10] multiphase porous media,[2,11,12] hyperelasticity with enforced constitutive conditions, as symmetry of the stress tensor, objectivity, material symmetry, polyconvexity, and thermodynamic consistency,[13] and other applications in ref. [14]. One of the most interesting topics in this regard is the construction of machine learning (ML)-based material models for interstitial fluid flow and path-dependent fluid retention behavior to capture cyclic wetting/drying processes, as will be intensively discussed in the current work.

The hysteretic response can be realized by studying the microscopic fluid-fluid interaction through image-based simulations. Alternatively, the LBM can be applied to representative volume elements (RVEs) of the respective materials. The LBM presents an important tool for simulating fluid flow through complex geometries such as porous media.[15] In order to link the micro- and macroscopic scales in an efficient but accurate way, this paper aims to utilize ML-based material models for the macroscopic scale, which are trained using lower-scale generated data. The sequential steps in this procedure are summarized as follows: (1) single- and multiphase LBM fluid flow simulations are achieved to generate, respectively, the datasets of the anisotropic intrinsic permeability and drying/wetting hysteretic retention curves. Afterward, (2) the ML models are trained using the aforementioned datasets, before (3) testing the trained models with unseen data. Once a reliable ML-based model is developed, this can be integrated as an alternative to the conventional material models in a multiphase continuum mechanical framework within macroscopic poromechanics, see for example [16–20] for references and applications related to porous media mechanics.

To give a brief review of the implementation of ANN within multiscale modeling of fluid flow in porous media, Wang et al.[21] combined deep learning (DL) with a local multiscale reduced-order model to simulate flow dynamics in porous media. As a solution for the shortage of observed data, they employed computational data for training the ML model. In multiscale modeling of subsurface fluid flow in heterogeneous porous media, Choubineh et al.[22] developed distinct convolutional neural networks (CNNs) to predict different multiscale basis functions for the mixed generalized multiscale finite element method. The solely required input for their models in the study was the permeability property. Wang and Sun[11] developed a hybrid data-driven model, which was generated from supervised ML and hybridized with classical constitutive laws, and implemented it on porous media with heterogeneous pore sizes to simulate the multiscale hydro-mechanical coupling effect. Bao et al.[23] combined a multiscale approach with a DNN model to investigate the effect of the pore-scale electrode structure on the device-scale electrochemical reaction uniformity within a redox flow battery. They utilized the LB equation for the simulation of flow transport and electrochemical reactions in the electrode sample, in addition to the quasi-Monte Carlo method for sampling. In the underlying paper, the ML-based model is used to capture the effect of deformation on the anisotropic intrinsic permeability as well as on the retention dynamics of the heterogeneous porous media.

In addition, a number of studies have utilized CNNs to estimate model parameters of porous materials. Alqahtani et al.[24] used the aforementioned approach for the fast computation of the porosity, coordination number, and average pore size of porous networks from two-dimensional grayscale μ-CT images in a supervised learning frame. In another study, Kamrava et al.[25] presented a novel deep convolutional neural network (DCNN) in combination with random-walk

particle-tracking simulations to determine the longitudinal dispersion coefficient in fluid flow through heterogeneous porous materials. In the work of Lähivaara et al.,[26] the implementation of CNNs for the computation of the material parameters of saturated porous media from synthetic ultrasound tomography data was presented. They generated data for training purposes of the ANNs by solving wave propagation in coupled poro-viscoelastic-acoustic media. Moreover, for the simulation of biphasic fluid flow, CNNs have been also incorporated with physics-informed neural networks (PINNs), that is, physics-informed deep convolutional neural network (PIDCNN). For example, Zhang et al.[27] developed a PIDCNN model to simulate transient biphasic Darcy flows in reservoir domains. They utilized the finite volume method (FVM) for the approximation of the residual of the partial differential equations (PDEs) in the loss function. Recently, Cai et al.[12] used a graph neural network (GNN) to predict SE(3)-equivariant formation factor and effective permeability of porous materials from micro-CT images. As input in their model, the micro geometry of the porous material is represented by a persistence-based Morse graph. A main added value in the current work is the generation of hysteretic retention curves of deformable unsaturated porous media via the implementation of 1D CNN, which has several advantages over other ML alternatives. Mainly, 1D CNNs achieve feature extraction and classification operations in one single process. In turn, this leads to reduced computational complexities and costs.[28]

Several works in the literature addressed the determination of the retention curves of unsaturated porous media based on ML models. Heider et al.[2] presented a meta-modeling approach that uses deep reinforcement learning (DRL) to determine optimal neural network settings for the ML constitutive laws for unsaturated poromechanics. They applied their approach to determine path-dependent retention curves and the anisotropic permeability tensor, which governs the flow in micropores. Zhang and Song[29] utilized molecular dynamics and ML to investigate nanoscale retention mechanisms of unsaturated clay. With the help of the ML-based curve fitting technique, function relations between mass water content, adsorptive pressure, and apparent soil-water interface area have been established. Lamorski et al.[30] determined the main wetting branch of the retention curve based on the main drying curve using ML. In their study, they developed and validated the direct and parametric models, by using the data points of the retention curve or the van-Genuchten (VG)[31] parameters of the drying curve, respectively. In this contribution, the hysteretic behavior of the path-dependent retention curve is captured using the ML-based model.

In the context of the above-mentioned works, this paper presents a novel multiscale approach that leverages the capabilities of ANN to enhance the understanding and modeling of deformable porous media in terms of single- and multiphase fluid flow. Regarding the former, the model is capable of computing the nonlinear, anisotropic, and deformation-dependent permeability character. As for the latter, the approach captures the path-dependent retention behavior of several deformed configurations of the porous domain. The aforementioned properties can be computed via material models that require heavy computational costs due to the challenging micro-geometry of porous media. The efficiency in the proposed approach is achieved via ML-based models that connect the microscopically determined material properties with the macroscopic continuum. From a practical perspective, this approach can facilitate the efficient and accurate simulation of fluid dynamics across a diverse array of applications. These include flow in geological formations, predicting contamination spread, flow in biological systems and bio-reactors, as well as energy-related systems such as fuel cells and hydrogen production systems.

The paper is arranged according to the following structure: Section 2, which focuses on LBM, includes a brief explanation of the theoretical framework for single- and biphasic fluid flow. Moreover, the computation of the deformation-dependent intrinsic permeability and retention curves of several deformed porous networks is carried out in this section. In addition, this section presents the investigation of the anisotropic permeability tensor and the consequent generation of the training data for the ML-based model. Thereafter, the ML-based models are implemented in saturated and unsaturated porous media. Regarding the former, Section 3 describes the informed-graph-based ANNs for the computation of the intrinsic permeability tensor. As for the latter, Section 4 describes the path-dependent ML models for the computation of the retention curves of biphasic fluid flow in unsaturated porous domains, that is, the 1D CNN and recurrent neural network (RNN) models. The difference in the performance of the two history-dependent ML models in terms of accuracy and speed of training is also clarified. In the end, concluding remarks are found in Section 5.

2 | LBM FOR FLUID FLOW SIMULATION AND PERMEABILITY COMPUTATION

Proceeding from the particle methods of the lattice gas automata, the LBM solves the Boltzmann equation[32] in a mesh-based approach. In this section, a succinct description of the LBM theory in terms of single- and multiphase fluid flow is presented. Subsequently, the LBM results of the intrinsic permeability and retention curves are presented.

2.1 | LBM for single-phase fluid flow

The Boltzmann equation in LBM governs the space and time dynamics of the so-called velocity distribution function $f(\mathbf{x}, \boldsymbol{\xi}, t)$. This function characterizes a set of particles located at a position $\mathbf{x}$ and time t with discrete lattice velocity vector $\boldsymbol{\xi}$. The exchange of momentum and energy amongst these particles occurs through the streaming and collision phases, namely,

$$\left.\frac{df}{dt}\right|_{\text{streaming}} = \left.\frac{df}{dt}\right|_{\text{collision}}, \quad \underbrace{\frac{\partial f}{\partial t} + \boldsymbol{\xi} \cdot \frac{\partial f}{\partial \mathbf{x}}}_{\text{streaming operator}} = \underbrace{\Omega(f)}_{\text{collision operator}}. \tag{1}$$

The local fluid density ρ and fluid velocity $\mathbf{v}_F$ can be computed as a function of $f(\mathbf{x}, \boldsymbol{\xi}, t)$ via

$$\rho(\mathbf{x}, t) = \int f(\mathbf{x}, \boldsymbol{\xi}, t)\, d\boldsymbol{\xi} \quad \text{and} \quad \mathbf{v}_F(\mathbf{x}, t) = \frac{1}{\rho} \int \boldsymbol{\xi}\, f(\mathbf{x}, \boldsymbol{\xi}, t)\, d\boldsymbol{\xi}. \tag{2}$$

For space discretization, a particle is restricted to stream in 19 possible directions in 3D, that is, D3Q19, namely,

$$\mathbf{e}_i = \begin{cases} (0,0,0) & i = 0 \\ (\pm 1,0,0), (0,\pm 1,0), (0,0,\pm 1) & i = 1, 2, \dots, 6 \\ (\pm 1,\pm 1,0), (\pm 1,0,\pm 1), (0,\pm 1,\pm 1) & i = 7, 8, \dots, 18, \end{cases} \tag{3}$$

where $\mathbf{e}_i$ is the direction of the velocity vectors $\boldsymbol{\xi}_i = c\, \mathbf{e}_i$. Herein, c is the ratio of the distance between the nodes Δx to the time step Δt.

With regard to the collision operator $\Omega(f)$ in (1), the Bhatnagar-Gross-Krook (BGK)[33] model is utilized since it can be numerically implemented in a straightforward manner and has been broadly utilized in fluid flow simulations.[34] The BGK collision operator Ω_{BGK} is formulated via

$$\Omega_{\text{BGK}} = -\frac{f_i - f_i^{eq}}{\tau}, \quad \text{with} \quad \tau := \frac{1}{2} + \nu_l\, c_s^{-2}. \tag{4}$$

In this, the relaxation time τ is a function of the lattice fluid viscosity ν_l and lattice speed of sound $c_s = 1/\sqrt{3}$. The BGK formulation prescribes the relaxation of the distribution functions f_i toward equilibrium distributions f_i^{eq} at a collision frequency τ^{-1}. The formulation of f_i^{eq} is expressed as follows

$$f_i^{eq} = w_i\, \rho \left(1 + \frac{\mathbf{e}_i \cdot \mathbf{u}_l}{{c_s}^2} + \frac{(\mathbf{e}_i \cdot \mathbf{u}_l)^2}{2{c_s}^4} - \frac{(\mathbf{u}_l \cdot \mathbf{u}_l)^2}{2{c_s}^2}\right), \tag{5}$$

where w_i presents the lattice weights, that is,

$$w_i = \begin{cases} 1/3 & i = 0 \\ 1/18 & i = 1, 2, \dots, 6 \\ 1/36 & i = 7, 8, \dots, 18. \end{cases} \tag{6}$$

Ultimately, the distribution functions are updated via

$$\underbrace{f_i(\mathbf{x} + \boldsymbol{\xi}_i \Delta t, t + \Delta t) - f_i(\mathbf{x}, t)}_{\text{streaming}} = \underbrace{\Omega_{\text{BGK}}}_{\text{collision}}. \tag{7}$$

In terms of the boundary conditions (BCs), the Zou-He[35] bounce-back boundary dynamics are prescribed. For more detailed descriptions of the LBM approach for single-phase fluid flow, the reader is referred to ref. [4].

2.2 | LBM for multiphase fluid flow

Amongst the various LBM models for modeling multiphase fluid flow through porous media, the pseudopotential Shan-Chen (SC) model[36] has been selected for this study. The aforementioned model enables the manipulation of the kinetic viscosity of the fluid phases and the fluid-fluid interaction. As a kind of extension of the formulation of single-phase fluid flow, the BGK model is expressed for each fluid phase $\beta = 1, 2$ in biphasic flow as

$$f_i^{\beta}(\mathbf{x} + \boldsymbol{\xi}_i \Delta t, t + \Delta t) - f_i^{\beta}(\mathbf{x}, t) = -\frac{f_i^{\beta}(\mathbf{x}, t) - f_i^{\beta, eq}(\mathbf{x}, t)}{\tau^{\beta}}. \tag{8}$$

The local equilibrium distribution function $f_i^{\beta, eq}$ is written as

$$f_i^{\beta, eq} = w_i \rho^{\beta} \left(1 + \frac{\mathbf{e}_i \cdot \mathbf{u}_l^{\beta, eq}}{{c_s}^2} + \frac{(\mathbf{e}_i \cdot \mathbf{u}_l^{\beta, eq})^2}{2{c_s}^4} - \frac{(\mathbf{u}_l^{\beta, eq})^2}{2{c_s}^2} \right). \tag{9}$$

Herein, $\mathbf{u}_l^{\beta, eq}$ presents the equilibrium velocity, that is,

$$\mathbf{u}_l^{\beta, eq} = \mathbf{u'}_l + (\tau^{\beta} \mathbf{f}^{\beta})(\rho^{\beta})^{-1} \quad \text{with} \quad \mathbf{u'}_l := (\rho^{\beta} \mathbf{u}_l^{\beta} / \tau^{\beta})(\rho^{\beta} / \tau^{\beta})^{-1}, \tag{10}$$

where $\mathbf{u'}_l$ denotes the common velocity. The total force acting on fluid phase β is characterized by $\mathbf{f}^{\beta}$. It is the combination of forces at the fluid-fluid and fluid-solid interfaces, that is, $\mathbf{f}_{ff}^{\beta}$ and $\mathbf{f}_{fs}^{\beta}$, respectively. In addition, the external force is presented by $\mathbf{f}_e^{\beta}$. The aforementioned forces are formulated as follows

$$\mathbf{f}_{ff}^{\beta}(\mathbf{x}) = -\psi^{\beta}(\mathbf{x}) G^{\beta\bar{\beta}} \sum_{\mathbf{x'}} \sum_{\bar{\beta}=1}^{2} -\psi^{\bar{\beta}}(\mathbf{x'}) \mathbf{e}_i, \quad \mathbf{f}_{fs}^{\beta}(\mathbf{x}) = -\rho^{\beta} G_{ads}^{\beta}(\mathbf{x}) \sum_i w_i s(\mathbf{x'}) \mathbf{e}_i, \quad \mathbf{f}_e^{\beta} = \rho^{\beta} g \approx 0. \tag{11}$$

For the pseudopotential $\psi^{\beta}(\mathbf{x})$ in $(11)_1$, several formulations can be found in the literature.[36–40] In our work, the relation found in the works of Martys and Chen[41] and Pan et al.[42] is used, that is, $\psi^{\beta} = \rho^{\beta}$. In this common form of ψ^{β}, the pseudopotential is not bound and small values for the density ρ^{β} need to be prescribed to avoid divergence of the interaction force $\mathbf{f}_{ff}^{\beta}$.[15] As for $\mathbf{x'}$, it presents the neighbor lattice position, that is, $\mathbf{x'} = \mathbf{x} + \mathbf{e}_i$. Moreover, $G^{\beta\bar{\beta}}$ is the coupling coefficient that dictates the fluid-fluid interaction. In terms of the fluid-solid interaction force $\mathbf{f}_{fs}^{\beta}$ in $(11)_2$, s equals 1 for the solid phase and 0 for the fluid phase. Also, the adsorption coefficient is presented by G_{ads}^{β}. As for $(11)_3$, the external force $\mathbf{f}_e^{\beta}$ is assumed negligible. For additional discussion about the numerical stability of LBM in multiphase fluid flow, see.[4,15]

The equation of state (EOS) of the SC multicomponent model[36] is used to compute the lattice pressure p_l at each node as follows

$$p_l = c_s^2 \sum_{\beta} \rho^{\beta} + \frac{c_s^2 \Delta t^2}{2} \sum_{\beta\bar{\beta}} G^{\beta\bar{\beta}} \psi^{\beta} \psi^{\bar{\beta}}. \tag{12}$$

2.3 | LBM computation of the intrinsic permeability tensor

For the computation of the intrinsic permeability components, the geometries of the Bentheimer networks of each level of deformation, or degree of strain, are input to the single-phase LBM model, described in Section 2.1. The main objective

behind these simulations is that a corresponding average lattice fluid velocity is computed for each prescribed pressure gradient of the fluid phase applied across the porous domain via each of the hydrodynamic axes $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$, that is, ∇p_1, ∇p_2 and ∇p_3, respectively. From the simulation results, the lattice intrinsic permeability tensor $\mathbf{K}_l^S$, expressed in lattice units [l.u.] , of each of the deformed Bentheimer networks has been determined under the assumption of a symmetric and positive definite permeability tensor using the approach of Kuhn et al.[43] Herein, they suggested two types of fluid flow simulations for each hydrodynamic direction: one with the prescription of a no-slip BC and another with the assignment of a natural slip BC for the surfaces parallel to the direction of the fluid flow. The reason behind such consideration is that the average computed velocity in the pressure direction of the no-slip BC is lower in comparison to the case of a natural slip BC. Hence, the difference between the two velocities gives an indication to the additional fluid flow in the direction that is orthogonal to the pressure gradient direction. Regarding the former, the diagonal components of the permeability tensor are computed via the following lattice-accommodated Darcy-like filter law:

$$(K_l^S)_{ii} = -\nu_l \frac{(u_l)_{i,\,\mathrm{avg}}}{\nabla_i\, p_l} \quad [\mathrm{l.u.}]\,, \quad \text{with} \quad i = 1,\, 2,\, 3. \tag{13}$$

In this, the lattice pressure gradient, presented by $\nabla_i\, p_l = \partial p_l / \partial x_i$ is induced between two opposing surfaces perpendicular to the flow direction to calculate the average lattice fluid velocity $(u_l)_{i,\,\mathrm{avg}}$. As for the latter, the unknown non-diagonal elements of the permeability tensor are computed as follows[43]

$$\begin{bmatrix} \nabla_2\, p_l & \nabla_3\, p_l & 0 \\ \nabla_1\, p_l & 0 & \nabla_3\, p_l \\ 0 & \nabla_1\, p_l & \nabla_2\, p_l \end{bmatrix} \begin{bmatrix} (K_l^S)_{12} \\ (K_l^S)_{13} \\ (K_l^S)_{23} \end{bmatrix} = \begin{bmatrix} -\nu_l\,(u_l)_{1,\,\mathrm{avg}} - (K_l^S)_{11}\,\nabla_1\, p_l \\ -\nu_l\,(u_l)_{2,\,\mathrm{avg}} - (K_l^S)_{22}\,\nabla_2\, p_l \\ -\nu_l\,(u_l)_{3,\,\mathrm{avg}} - (K_l^S)_{33}\,\nabla_3\, p_l \end{bmatrix}. \tag{14}$$

Other approaches for the computation of the anisotropic permeability tensor via LBM are also present, such as in the study of Galindo-Torres et al.[44] In their method, the pressure gradient is prescribed as a function of an angle of an octahedral plane. Therefore, for the determination of the nine unknown indices of the anisotropic permeability tensor, where they did not initially assume the symmetry of the tensor, they required flow simulations for three different octahedral angles. In this sense, they would have adequate equations to solve for the unknowns. The difference of the aforementioned method to the implemented approach by Kuhn et al.[43] in our study is that the flow simulations are done twice per hydrodynamic direction with different BCs to solve for the non-diagonal components as explained above.

Conclusively, the lattice permeability results are converted to their macroscopic counterpart as follows

$$K_{ij}^S = (K_l^S)_{ij}\,(\delta x_i\, \delta x_j) \quad \left[\mathrm{m}^2\right], \tag{15}$$

where δx_i and δx_j characterize the spatial resolution of the $\boldsymbol{\mu}$-CT images in the i and j directions, respectively. The μ-CT images of the Bentheimer networks are published by Moon and Andrew[45] at a resolution of $\delta x_1 = \delta x_2 = \delta x_3 = 8.96\,\mu$m per pixel. For the selection of the suitable RVE size of the geometry of the Bentheimer Sandstones, the reader is referred to our work in ref. [46]. For visualization of the output of LB simulations, exemplary results are shown in Figure 1.

2.4 | Investigation of the anisotropy and generation of database

In building an objective ML-based permeability model, it is crucial to understand in advance whether the model is anisotropic and whether the anisotropy changes during the deformation process. This can be figured out through the computation and comparison of the eigenvalues and eigenvectors of the permeability tensors, which represent the principal permeability components and the directions of these components, respectively. Hence, the anisotropic nature of the deformation-dependent intrinsic permeability is initially investigated. For this purpose, the permeability tensors are computed at three different pressure gradients, that is, $\nabla p = 2.4,\ 7.62$, and 15.19 [MPa/m] for the different degrees of the strain ε in the range of $[0, 0.5]$. The values of the pressure gradient have been selected in this manner to include moderate to high fluid velocities in the anisotropy analysis. Herein, a comparison between the principal values of each permeability tensor as well as the deformation-related changes of the principal directions is accomplished. In this regard, spectral

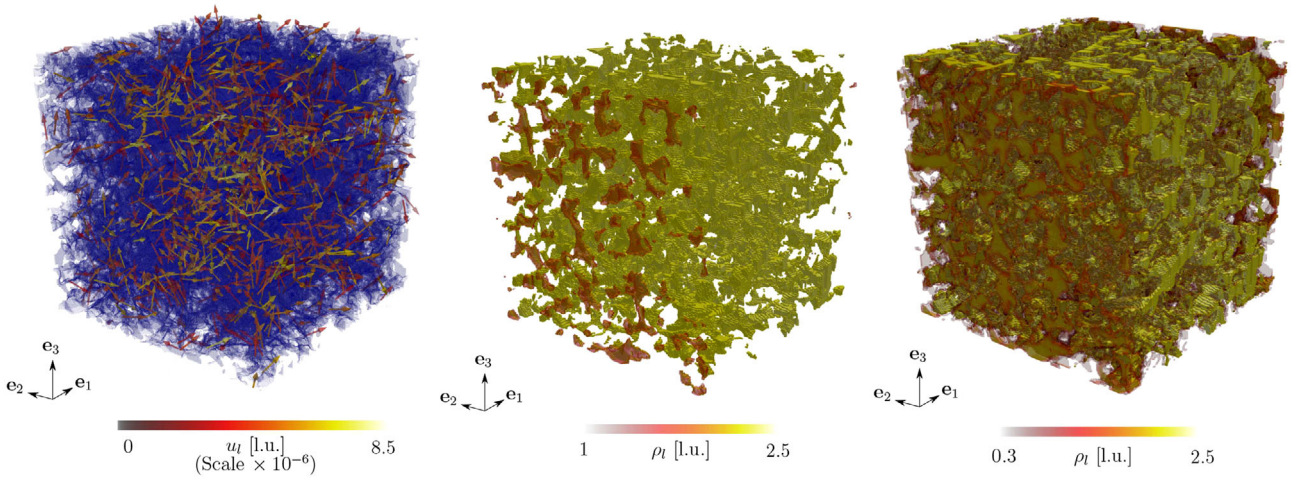

FIGURE 1 Exemplary simulation output for the porous domain with $\varepsilon = 0.02$: vector representation of the computed lattice velocity fields u_l when the pressure gradient was applied in the $\mathbf{e}_1$-direction for the single phase LB simulation. In this, the blue skeleton represents the solid phase (left); biphasic simulation of the drying process, where the non-wetting and wetting agents are shown in red and yellow, respectively, at two different time steps t_1 (middle) $< t_2$ (right). LB, lattice Boltzmann.

decomposition is utilized to determine the eigenvalues and eigenvectors of a given permeability tensor $\mathbf{K}^S$ as

$$\mathbf{K}^S = \sum_{i=1}^{3} K_i^S \left(\mathbf{n}_{ki} \otimes \mathbf{n}_{ki}\right), \tag{16}$$

where K_i^S are the eigenvalues of the permeability tensor and $\mathbf{n}_{ki}$ are the corresponding eigenvectors. The orthogonal eigenvectors, which can be normalized, enable the construction of the 3×3 rotation tensors, from which the three Euler angles can be calculated. It is noteworthy that other alternatives for the representation of rotation exist, such as the quaternions or Cardano angles. However, the Euler angles are used in our study due to their straight-forward numerical application and common usage[47] (the reader is also referred to our work in ref. [8] for alternative rotation representation). Herein, the rotation tensors can be expressed as

$$\mathbf{R}_k = \begin{bmatrix} \mathbf{n}_{k1} & \mathbf{n}_{k2} & \mathbf{n}_{k3} \end{bmatrix}, \tag{17}$$

where $\mathbf{R}_k \in SO(3)$. The rotation tensor with nine components can be expressed by three equivalent Euler angles, that is, $\{\varphi_K, \Theta_K, \psi_K\} \in \mathbb{E} \subset \mathbb{R}^+$.

Within the analysis of the relation between the principal permeabilities and the induced deformation, an increase in the values of the principal permeabilities K_1^S, K_2^S and K_3^S is witnessed as the volumetric strain increases to $\varepsilon \approx 0.1$, compare Figure 2. The increase in the principal permeabilities is caused by the increase of the porosity n^F, as shown in Figure 3. This is attributed to the method used in deforming the Bentheimer geometries in the published images of Moon and Andrew.[45] Herein, they implemented the elastic simulation technique, using the algorithm of Rutka andWiegmann,[48] to create the series of the compressed Bentheimer networks. In this, the solid phase, presented by the particles, is assumed to be compressible and linear elastic. At the early stages of compression, where low values of volumetric strain are applied, the particles are compressed or congregated together within the porous matrix. In turn, this gives a possibility for the increase of the voids, that is, the porosity n^F at this early stage of deformation. Nevertheless, the principal permeabilities decrease drastically for strain values $\varepsilon > 0.1$. This can be explained by the decrease of the porosity due to the rearrangement of the particles and compression of the pore channels while increasing the applied volumetric strain. Except for the principal permeability results at $\varepsilon \approx 0.3$, an evident difference between the values of the principal permeabilities K_i^S is witnessed for most of the strain degrees. An analogous observation is noted in the analysis of the Euler angles. For strain levels $\varepsilon < 0.2$, the attained principal directions are almost equal. However, as the strain ε exceeds 0.3, the principal directions diverge and a considerable difference is noted amongst them. This can be attributed to the alteration of the pore structure and flow paths due to the induced deformation, which consequently affects the prinicpal directions. In turn, this shows the need for consideration of the anisotropic character of the intrinsic permeability throughout the different deformation levels.

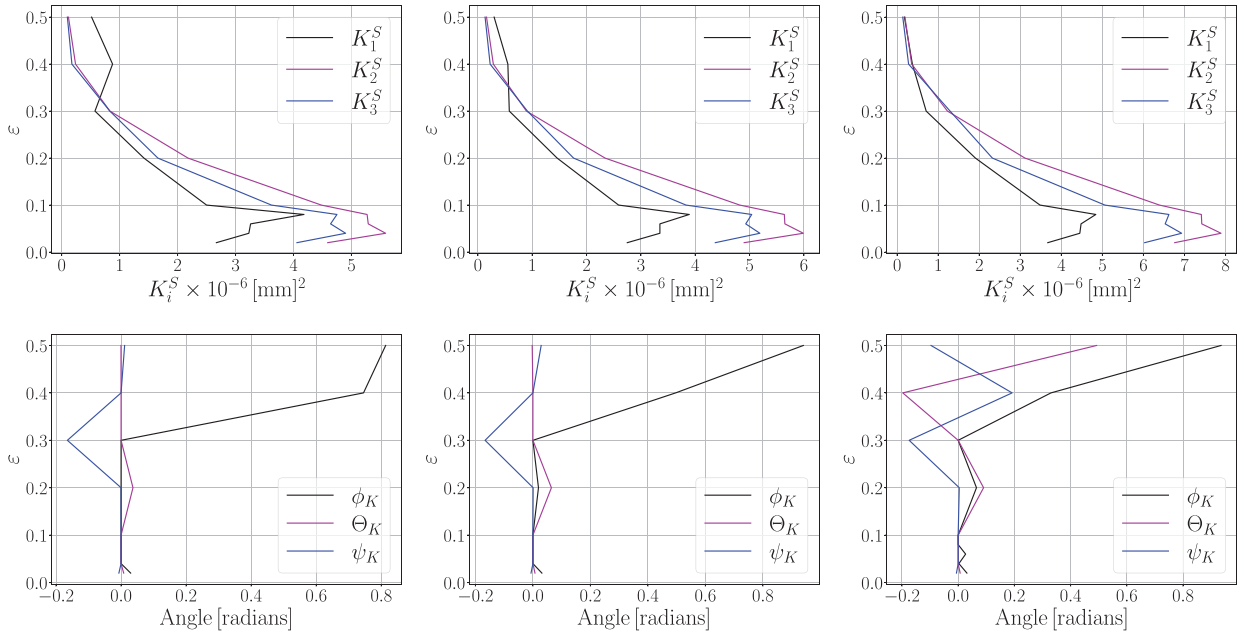

FIGURE 2 The strain degree ε versus the principal values K_i^S of the permeability tensor (top) and the corresponding Euler angles (bottom) for pressure gradients $\nabla p = 2.4\,\mathrm{MPa/m}$ (left), $7.62\,\mathrm{MPa/m}$ (middle), and $15.19\,\mathrm{MPa/m}$ (right).

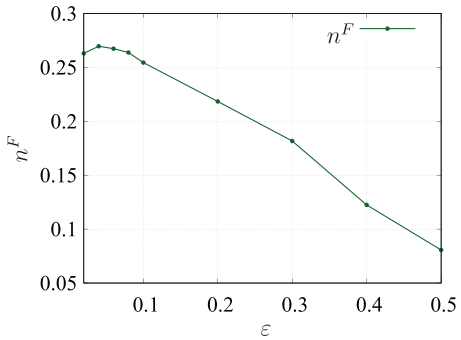

FIGURE 3 The change of porosity n^F with respect to the increase of strain ε . At the early increasing steps of the strain, that is, $\varepsilon \approx 0.02$, a slight increase in the porosity n^F is witnessed.

Afterward, the required database by the ML-based model for the prediction of the permeability has been generated. To this end, the sample sources are created from the findings of the LB computations on the microgeometry of the Bentheimer Sandstones. Initially, the μ-CT images correspond to the size of $1024 \times 1024 \times 1024$ voxels.[45] However, this causes high numerical costs on the microscopic numerical simulations. Therefore, a suitable size of the RVE of $350 \times 350 \times 350$ voxels is determined.[46] The size of the dataset for the ML model includes 2286 datapoints, whereby each datapoint corresponds to an increment of the pressure gradient on the porous domain. Herein, the LB model assigns for each strain degree several increasing increments of the pressure gradient of the fluid across the porous domain to calculate the resulting average velocity of the fluid. This is achieved for each type of the previously discussed BCs per hydrodynamic direction. Herein, the range of applied fluid pressure gradients includes fluid flows with Reynolds number in the range of [0.15, 1.36]. In this manner, the fluid flow exhibits both Darcy flow, which is characterized by laminar flow with Reynolds number less than 0.2,[49] and a non-Darcy flow.

2.5 | LBM for the deformation-dependent retention curves

Several phenomenological retention curve models, such as the VG, are not designed to capture the hysteresis of unsaturated porous materials' wetting and drying paths. Neglecting the hysteresis may affect the reliability of the numerical

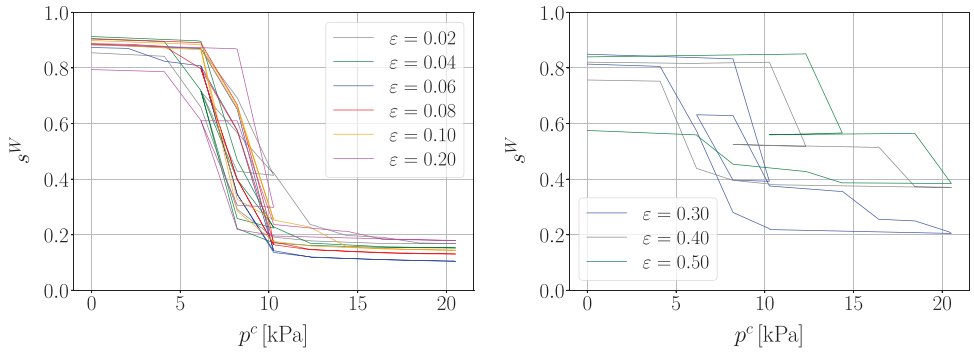

FIGURE 4 The retention curves showing the plots of saturation degree s^W versus the capillary pressure p^c for strain degrees $\varepsilon \leq 0.20$ (left) and $0.30 \leq \varepsilon \leq 0.50$ (right). For discussions regarding the effects of deformation on the retention curves, such as the hysteretic behavior, see our work in ref. [46].

models that involve cyclic drying/wetting processes, which leads for instance to inaccurate computation of both Bishop's effective stress of unsaturated porous media and the relative permeability. Furthermore, these retention curves are affected by the state of the material deformation, which adds an additional source of inaccuracy to the single-curve hand-crafted retention curve models. Considering deformable unsaturated porous media, a link is established in this section between the LBM, as a tool to generate databases from the lower-scale simulations, and the ML approach, used for generating constitutive laws for the continuum mechanical simulations. In particular, the data generated for the path-dependent retention behavior using the SC LBM model, discussed in Section 2.2, is used to generate an ML model for the retention behavior. In order to ensure hysteretic retention behavior, a cyclic pressure differential of the non-wetting fluid, that is, the invading fluid, has been prescribed to attain drying/wetting cycles across the domain. In turn, the saturation degree s^W is computed for every increment of the capillary pressure p^c, defined as the difference between the non-wetting and wetting effective pressures p^{nw} and p^w , respectively. The biphasic LBM simulations were accomplished for all deformed geometries of the strain degrees, see Figure 4 for a summary of the results. For more details regarding the LBM simulations and related discussions, such as the effect of deformation on hysteresis or the air-entry value, the reader is referred to our work in ref. [46].

The micro-geometry of the deformed porous media is available for nine different increments of the strain ε.[45] Nevertheless, the strain increment for $\varepsilon \leq 0.10$ is set at 0.02 , which represents a relatively small deformation increment. However, the increment highly increased to 0.10 for $\varepsilon > 0.10$. In turn, this caused a sudden shift in the trends of the retention curves for high values of strain, such as $\varepsilon = 0.40$ and 0.50. For this reason, the dataset did not prove sufficient for the RNN model for training, validation, and testing purposes. Therefore, additional sets of strain degrees were deemed necessary for input to the training of the RNN model. In order to artificially expand our database, RVEs for additional assumed strain degrees $\varepsilon = 0.12,\ 0.14,\ 0.16,$ and 0.18 are created. Consequently, RVEs within the available micro-geometry data of $\varepsilon = 0.10$ and 0.20 with similar porosity characteristics, such as the intrinsic permeability K^S , shown in Figure 2 and the porosity n^F , shown in Figure 3, were extracted for the aforementioned assumed strain degrees. For example, RVEs for $\varepsilon = 0.12$ and 0.14 were extracted from the micro-geometry of $\varepsilon = 0.10$, and RVEs for $\varepsilon = 0.16$ and 0.18 were extracted from the micro-geometry of $\varepsilon = 0.20$. Afterward, these geometries were input into the LBM biphasic fluid flow simulations for the generation of additional numerical data needed to input into the ML model, that is, capillary pressure and porosity of the deformed porous domain, to compute the corresponding saturation degree. Due to the little dataset provided for the deformed Bentheimer geometries for deformations $\varepsilon > 0.20$, we restricted the computation of the retention curves for the RNN model for strain degrees $\varepsilon \leq 0.20$. The dataset for the ML models for the computation of the retention curves includes 6240 data points.

3 | ML-BASED ANISOTROPIC PERMEABILITY MODEL

This section describes the supervised ML model used for the estimation of the deformation-dependent anisotropic intrinsic permeability. In particular, a multi-step ANN model is applied to determine the path-independent response, where an informed graph model is utilized in this process.

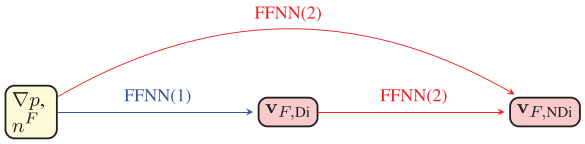

FIGURE 5 Representation of the informed directed graph model for the supervised ML-based permeability. The yellow node represents the root node, whereas the pink nodes are the target nodes. A two-step FFNN is applied to predict the velocities with no-slip BCs, that is, $\mathbf{v}_{F,\mathrm{Di}}$, and the velocities with natural slip boundaries, that is, $\mathbf{v}_{F,\mathrm{NDi}}$. BC, boundary conditions; FFNN, feed-forward regression neural network; ML, machine learning.

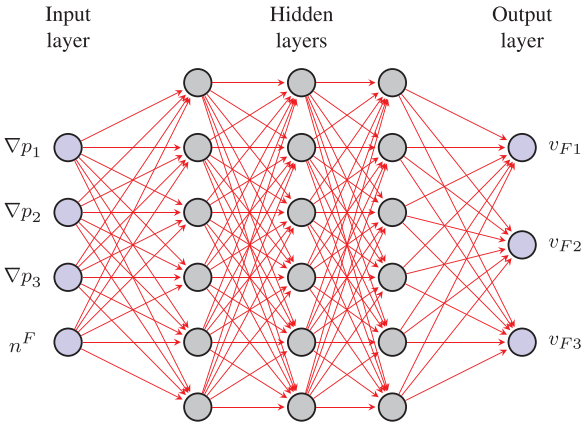

FIGURE 6 Illustration of the ANN structure in the ML-based permeability prediction model. Input parameters are represented by the pressure gradient ∇p_i and porosity n^F . The output parameters are the components of the fluid velocity v_{Fi} with $i = 1, 2, 3$. ANN, artificial neural networks; ML, machine learning.

3.1 | Informed-graph-based permeability model

The main scheme of the ML model in this section is summarized in two steps: (1) the prediction of the fluid velocity for the no-slip BC ($\mathbf{v}_{F,\mathrm{Di}}$) based on the input pressure gradient and porosity of the domain, and (2) the prediction of the fluid velocity for the natural slip BC ($\mathbf{v}_{F,\mathrm{NDi}}$) based on $\mathbf{v}_{F,\mathrm{Di}}$, the pressure gradient, and the porosity. Herein, a two-step feed-forward regression neural network (FFNN) is used for the prediction, compare Figure 5. The choice of implementing FFNN in the prediction of the intrinsic permeability is due to its capability in training path-independent material behavior.[8] Later in the prediction of the hysteretic retention behavior in Section 4, FFNN is replaced with RNN to replicate the path-dependent material behavior, which depends on the data history.

In this informed, directed graph approach, information streams from the root (input), that is, n^F and ∇p , to the first leaf (output), that is, $\mathbf{v}_{F,\mathrm{Di}}$, which represents the fluid velocity for the no-slip BC. This step is referred to as FFNN(1). Afterwards in step FFNN(2), we proceed with the root (n^F , ∇p) and first leaf ($\mathbf{v}_{F,\mathrm{Di}}$) for the prediction of the second leaf, that is, the fluid velocity for natural slip BC, represented by $\mathbf{v}_{F,\mathrm{NDi}}$.

3.2 | Training and testing of the ANN permeability model

In the aforementioned steps, the involved ANN iterates an array of predictions on the fluid velocities as well as in-between quantities, realized as hidden layers in Figure 6, to achieve enhanced accuracy of the output velocity results, see, for example [2, 11, 50] For all the strain degrees, the input layers incorporated the pressure gradients across the three hydrodynamic directions, namely, ∇p_1 , ∇p_2, and ∇p_3 , in addition to the porosity n^F of the deformed porous network, in order to calculate the corresponding fluid velocities v_{F1} , v_{F2}, and v_{F3} . Afterwards the computation of the components of the permeability tensor is achieved via solving the inverse problem in (13) and (14).

The pre-processing of the data obtained from the LBM simulations is done using the open-source Python data analysis library Pandas.[51] Herein, a random shuffling of the rows is carried out first. Thereafter, data splitting is applied in the

TABLE 1 A summary of the NN settings considered in the feed-forward neural network-based permeability model.

NN setting description	Abbreviation	Values
Neuron type subset	*NeuronType*	*MLP*
Hidden layers subset	*numHiddenLayers*	2
Number of neurons per layer	*numNeuronsPerLayer*	80
Dropout rate subset	*DropOutRate*	0.0
Optimizer type subset	*Optimizer*	*Adam*
Activation functions subset	*Activation*	*ReLU*, *sigmoid*
Batch sizes subset	*BatchSize*	8

following manner: 67% of the dataset is considered as the training subset and utilized for computing the weights and biases of the ANN, 11% of the dataset is considered as the validation subset to compute the errors during the training and to assist in tuning weights, whereas the remaining 22% of the dataset is chosen as the testing or unseen subset that is used for independent evaluation of the trained ML model. Before the obtained data is input into the ML model, a normalization procedure is carried out and the data is transformed into a range between 0 and 1 using the MinMaxScaler class of the sklearn.preprocessing toolkit.[52] In turn, the predicted data is rescaled to its original range after the training of the ML model is done. The settings of the ML model, such as the number of layers, choice of the optimizers, objective functions, and error estimations, are implemented with the help of the DL open-source code Keras[53] with built-in neural networks such as the multilayer perceptrons (MLPs). Herein, the system architecture entails an input layer, two hidden layers with 80 neurons for each, and the Rectified Linear Unit (*ReLU*) activation function, as well as an output layer with the sigmoid activation function. Concerning the optimization approach, the first-order gradient method, that is, ADAM,[54] is utilized. A summary of the chosen NN settings is given in Table 1. It is important to note that the choice of hyperparameters is not unique. In our research, we used the DRL algorithm presented in Heider et al.[2] to test whether alternative hyperparameter settings yield higher prediction accuracy. Our analysis consistently confirmed that the proposed hyperparameters (Table 1) are among the best choices in terms of prediction accuracy. For an in-depth examination of our hyperparameter setting methods and additional sources, see ref [2].

In this work, the estimation of the difference between the true data, namely, $\mathrm{X}_{\mathrm{true}}$, and predicted data $\mathrm{X}_{\mathrm{pred}}$ is achieved via the scaled mean squared error (scaled MSE), which is expressed for each experiment i with a total number of output data points N as follows

$$\text{scaled MSE}_i = \frac{1}{N}\sum_{j=1}^{N}\left[(\overline{\mathrm{X}}_{\mathrm{true}})_j - (\overline{\mathrm{X}}^{M}_{\mathrm{pred}})_j\right]^2 \quad \text{with} \quad \overline{\mathrm{X}}_{\mathrm{true}} := \text{MinMaxScaler}(X_{\mathrm{true}})\,. \tag{18}$$

Herein, the MinMaxScaler is only based on the true data and, afterwards applied on the predicted values to yield $\overline{\mathrm{X}}^{M}_{\mathrm{pred}}$. In the training, an *EarlyStopping* callback is assigned, which applies early stopping of the training if the loss does not improve for 100 weight updates (epochs).

Concerning the FFNN(1) step, the presented ML model required a total of 538 weight updates for the training, compare Figure 7 for the resulting accuracy and values of the loss function toward computing $\mathbf{v}_{F,\mathrm{Di}}$.

To have a closer look at the performance of the ML model, Figure 8, left, presents a qualitative comparison between the scaled MSE of the testing, validation, and training datasets. It shows that the errors in all cases are uniformly distributed and below 0.015. To quantitatively assess those results, we calculate the non-parametric, empirical cumulative distribution functions (eCDFs), see,.[55] In this, eCDFs are calculated for the MSE_i of testing (N_{testdata}), validation (N_{valddata}), and training ($N_{\mathrm{traindata}}$) datasets considering all the corresponding cases of the dataset (all), that is, $i \in [1, N^{\mathrm{all}}_{\mathrm{testdata}}]$, $i \in [1, N^{\mathrm{all}}_{\mathrm{valddata}}]$, $i \in [1, N^{\mathrm{all}}_{\mathrm{traindata}}]$. Thus, for a dataset N with MSE_i sorted in ascending order, the eCDF can be computed as

$$F_N(\mathrm{MSE}) = \begin{cases} 0, & \mathrm{MSE} < \mathrm{MSE}_1, \\ \dfrac{r}{N}, & \mathrm{MSE}_r \le \mathrm{MSE} < \mathrm{MSE}_{r+1},\ r = 1, \dots, N-1, \\ 1, & \mathrm{MSE}_N \le \mathrm{MSE}\,. \end{cases} \tag{19}$$

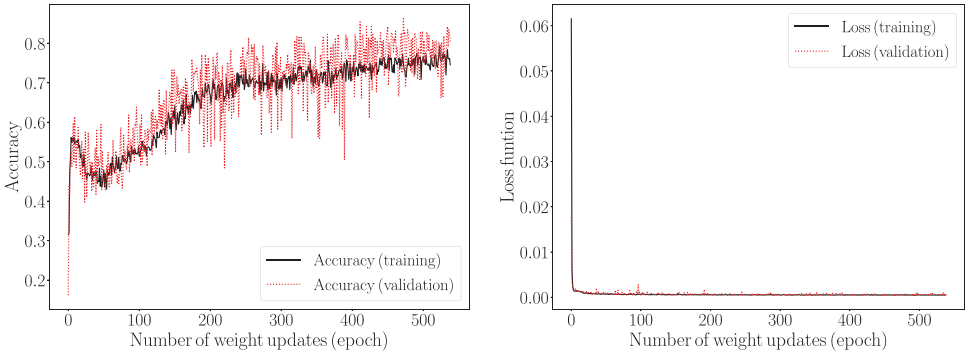

FIGURE 7 The accuracy (left) and values of the loss function (right) with respect to the number of weight updates of the training and validation steps within FFNN(1), where the ML model computed the velocities regarding the no-slip BCs based on the pressure gradient and porosity. BC, boundary conditions; FFNN, feed-forward regression neural network; ML, machine learning.

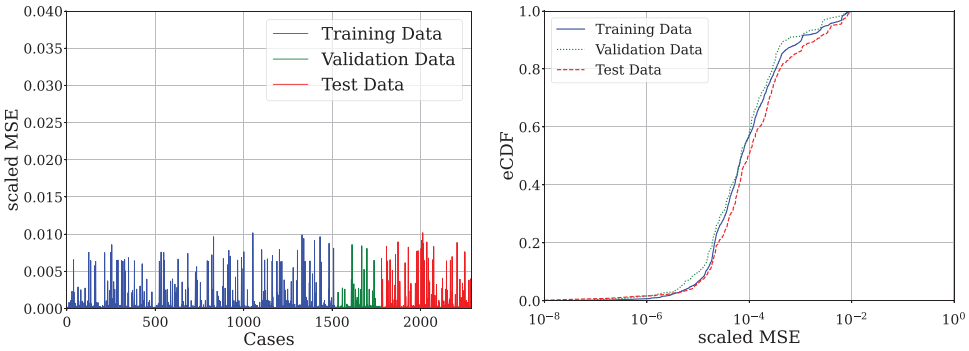

FIGURE 8 Scaled MSE of the different cases of training, validation, and testing datasets within FFNN(1) step (left). eCDF versus scaled MSE of the training, validation, and testing datasets within the FFNN(1) step (right). eDCF, empirical cumulative distribution functions; FFNN, feed-forward regression neural network; MSE, mean squared error.

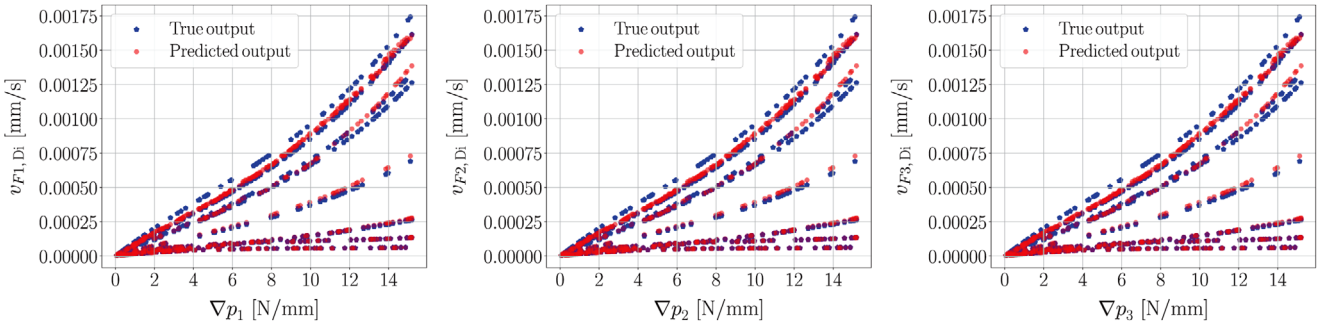

FIGURE 9 Comparison between the true output of the LBM simulations and the predicted output of the ML model regarding the fluid average velocity versus the pressure gradient, that is, $\mathbf{v}_{F1,\mathrm{Di}}$ w.r.t. ∇p_1 (left), $\mathbf{v}_{F2,\mathrm{Di}}$ w.r.t. ∇p_2 (middle) and $\mathbf{v}_{F3,\mathrm{Di}}$ w.r.t. ∇p_3 (right), of the no-slip BC. BC, boundary condition; LBM, lattice Boltzmann method; ML, machine learning.

Looking at the errors in Figure 8, right, it is clear that the trained model does not suffer from over-fitting, that is, the errors in the training and testing subsets are close to each other and in the range $\approx [10^{-2},\ 10^{-6}]$. In the eCDF error representation, the more the curve is vertical, the better the results, that is, close errors in all cases of the dataset. A more vertical curve can be obtained by, for example, considering a bigger dataset.

Following this, the trained ML model is applied to the unseen testing dataset in order to compare the predicted output with the true output of the average velocities for each hydrodynamic direction, compare Figure 9. In general, the predicted outputs show close results to the true output. Small deviations from the true output can be seen in the $\mathbf{v}_{F,\mathrm{Di}}$ plot, which is in agreement with the occurring errors that are visualized in Figure 8.

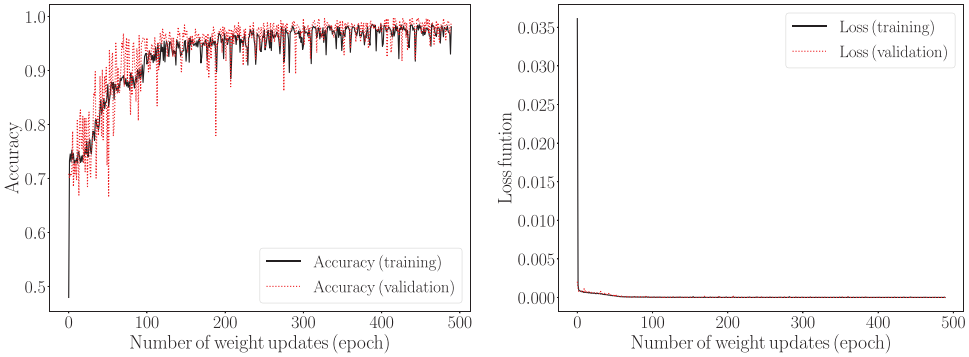

FIGURE 10 The accuracy (left) and values of the loss function (right) with respect to the number of weights updates (epochs) of the training and validation steps within FFNN(2). This ML model computes the velocities regarding the natural slip BCs based on the pressure gradient, porosity and the fluid velocities for the no-slip boundary conditions. BC, boundary conditions; FFNN, feed-forward regression neural network.

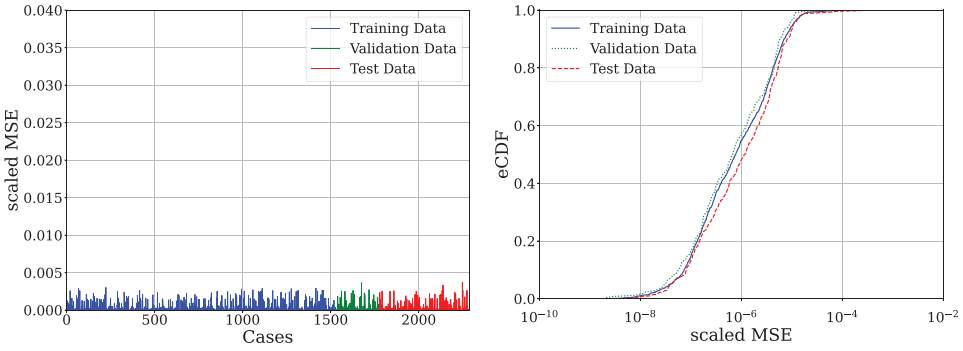

FIGURE 11 Scaled MSE of the different cases of training, validation, and testing datasets within FFNN(2) model (left). eCDF versus scaled MSE of the training, validation, and testing datasets within the FFNN(2) model (right). eCDF, empirical cumulative distribution functions; FFNN, feed-forward regression neural network; MSE, mean squared error.

The second step in building the supervised ML model is the prediction of the velocities $\mathbf{v}_{F,\mathrm{NDi}}$ related to the natural slip boundaries (cf. FFNN(2) in Figure 5). The NN settings and training steps for FFNN(1) are repeated for FFNN(2). Herein, a random shuffling of the rows of the dataset is carried out first. Thereafter, data splitting is applied with 67% for the training, 11% for the validation, and 22% for the testing. Moreover, a normalization procedure is carried out and the data is transformed into a range between 0 and 1. Similarly to FFNN(1), the accuracy and values of the loss function are computed at each epoch in the FFNN(2) for the training and validation datasets, compare Figure 10.

To have a closer look at the performance of the trained ML model, we apply the trained FFNN(2) model to predict the fluid velocity with natural slip boundaries $\mathbf{v}_{F,NDi}$, where the inputs are n^F and ∇p of the original dataset and the velocities $\mathbf{v}_{F,\mathrm{Di}}$ that result from FFNN(1) prediction. Figure 11, left, presents a qualitative comparison between the scaled MSE of the testing, validation, and training datasets. It shows that the errors in all training, validation, and testing are uniform and below 0.004. To quantitatively assess those results, we also calculate the non-parametric eCDFs. Looking at the errors in Figure 11, right, it is clear that the trained model does not suffer from over-fitting, that is, the errors in the training and testing subsets are close to each other and in the range $\approx [5 \times 10^{-3},\ 10^{-8}]$.

Finally, the trained FFNN(2) ML model is applied to the unseen testing dataset in order to compare the predicted output velocities ($\mathbf{v}_{F,\mathrm{NDi}}$) with the true ones, compare Figure 12. Following the informed graph 5, the input velocities $\mathbf{v}_{F,\mathrm{Di}}$ result from the FFNN(1) model. In general, the predicted outputs show close results to the true output. Small deviations from the true output can be seen in the $\mathbf{v}_{F,\mathrm{NDi}}$ plot, which is in agreement with the occurring errors that are visualized in Figure 11.

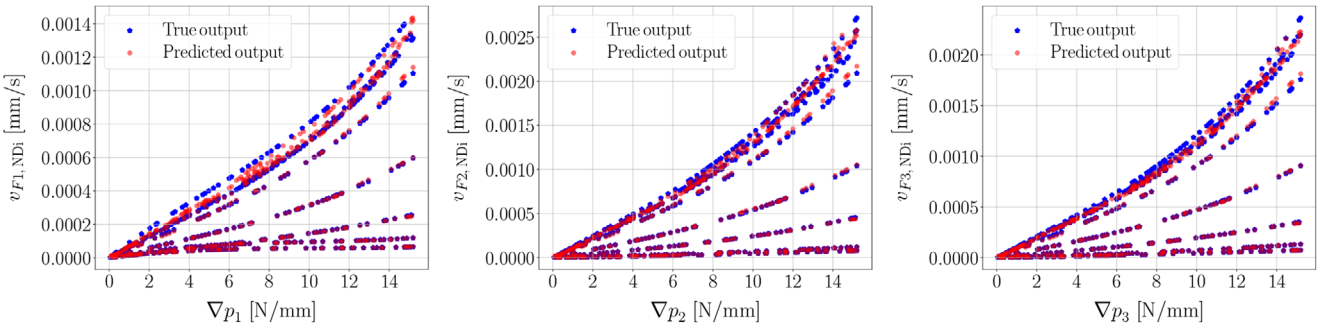

FIGURE 12 Comparison between the true output of the LBM simulations and the predicted output of the ML model regarding the fluid average velocity versus the pressure gradient, that is, $v_{F1,\mathrm{NDi}}$ w.r.t. ∇p_1 (left), $v_{F2,\mathrm{NDi}}$ w.r.t. ∇p_2 (middle) and $v_{F3,\mathrm{NDi}}$ w.r.t. ∇p_3 (right), of the natural slip BC based on the fluid velocities computed from FFNN(1). BC, boundary condition; FFNN, feed-forward regression neural network; LBM, lattice Boltzmann method; ML, machine learning.

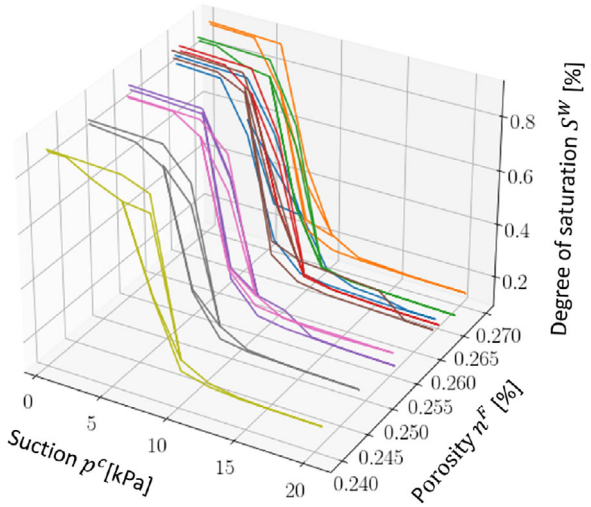

FIGURE 13 3D representation of the retention curves showing the plots of saturation degree s^W versus the capillary pressure p^c for strain degrees $\varepsilon \leq 0.20$ that correspond to different porosities n^F.

4 | ML MODELS FOR THE RETENTION BEHAVIOR OF BIPHASIC FLOW IN DEFORMABLE POROUS MEDIA

In contrast to the history-independent permeability model discussed in Section 3, the ML model for retention behavior should be able to mimic the path-dependent hysteretic nature of the saturation degree with respect to the capillary pressure. For this purpose, two neural network models, namely 1D CNN in Section 4.1 and RNN in Section 4.2, are employed to take into account the history and deformation dependency of the retention response. For the generation of the ML-based material model, a dataset for the retention curves that considers 10 various states of deformation, corresponding to $0.02 \leq \varepsilon \leq 0.2$, and includes scanning drying/wetting-intermediate cycles is prepared, with more details in Section 2.5. For an illustration of the dataset, Figure 13 includes a 3D representation of the deformation-dependent retention curves used in both 1D-CNN and RNN models

4.1 | 1D CNN model

We test in this section the applicability, efficiency, and accuracy of 1D CNN to generate a ML-based retention curve model. CNN, as a class of ANN used for DL, is widely used in applications like image classification and object or text detection. It requires mostly data that have a grid-like nature, such as images or time-series data, see, e.g,[56–61] for review

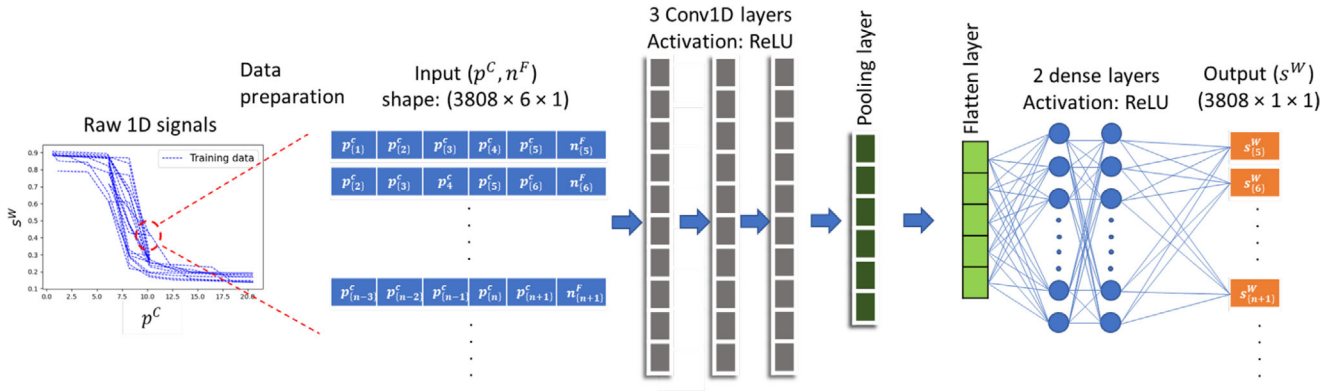

FIGURE 14 Illustration of the 1D CNN architecture and information flow that include data preparation, convolution operations, pooling operation, flatten, and fully-connected MLP layers. CNN, convolutional neural network; MLP, multilayer perceptron.

and applications. CNN is considered as a FFNN approach with weight-sharing characteristics that are achieved through convolutional and pooling (subsampling) layers. The major components of CNN are:

- *Convolution Operation*: This is carried out by multiplying the input data by a 2D filter (also called kernel or feature detector) to generate a feature map.
- *Pooling Operation*: This is applied after the convolution operation by multiplying the feature map with a 2D filter. The aim is to down-sample the feature map by calculating, for instance, their maximum or average values and send the important features only to the next CNN layer.
- *Multi-Layer Perceptron (MLP)*: the fully-connected (dense) or MLP layers have the pooled feature map as an input and a 1D feature vector as an output.

Based on this, we distinguished between the "CNN layers", the "pooling layers", and the "MLP layers" in the construction of the CNN model.

Unlike 2D CNNs that are designed to deal with data as images, 1D CNNs have the capability to deal with one-dimensional sequences of data, such as in the case of path-dependent retention curves. In comparison with the 2D CNNs, 1D CNNs have the advantage of lower computational complexity, see.[28] In this work, the 1D-CNN model is implemented using Keras with Python installed in a Jupyter notebook with Anaconda environment. The architecture of this 1D-CNN model is illustrated in Figure 14.

It includes the following features:

- Data preparation: In this, the raw data, which represent the capillary pressure p^c, the degree of saturation s^W, and porosity n^F at each time step t_n , is reshaped into segments of fixed length. Each segment contains a subset of the time series data, that is, $[p^c_{n-3}, p^c_{n-2}, p^c_{n-1}, p^c_n, p^c_{n+1}, n^F_{n+1}]$ as input and $[s^W_{n+1}]$ as output. The choice of five history values of p^c is consistent with the RNN approach discussed in the previous section. Following this, normalization of the input is carried out by computing the corresponding mean and standard deviation in order to improve the performance and stability of the network during training. Thereafter, each segment is passed through the network as a separate input, allowing the network to learn features and patterns at the segment level.
- Three convolutional layers (Conv1D) with filter (kernel) size = 3 for the first layer and 2 for the second and third layers are considered. The filter is moved one step at a time across the input data, that is, stride = 1, and no padding is added to the input data. The activation function in convolutional layers is chosen as *ReLU*.
- One max pooling layer with a pool size (sub-sampling) of 2 is used. This allows downsampling of the input sequence by taking the maximum value of every two adjacent values.
- One flatten layer is used to convert the 2D output of the pooling layer into a 1D array, which is used as an input to the fully-connected layers.
- Fully-connected MLP layers follow the flatten layer, which contain two hidden layers and the output array. The hidden layers include 80 neurons and *ReLU* as an activation function.

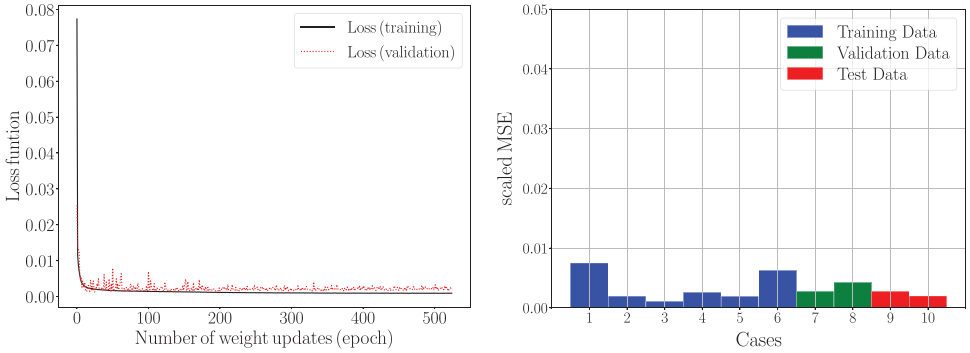

FIGURE 15 The values of the loss function with respect to the number of weight updates (epochs) of the training and validation steps (left) and the scaled MSE versus the number of states of deformations of the training, validation, and testing data (right) within the 1D CNN model. CNN, convolutional neural network; MSE, mean squared error.

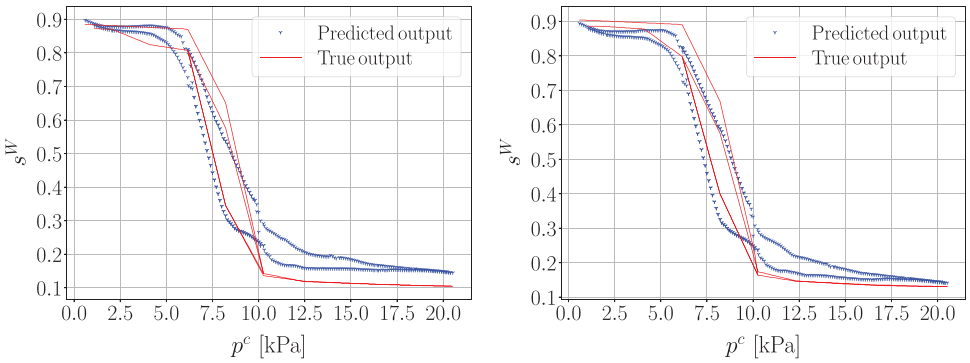

FIGURE 16 Comparison between the results of the LBM biphasic simulations, that is, true output, and the predicted output of the 1D CNN model regarding the saturation degrees versus the capillary pressure for deformation states $\varepsilon = 0.06$ (left) and $\varepsilon = 0.08$ (right). The ML-computed plots capture the scanning-drying and -wetting curves. CNN, convolutional neural network; LBM, lattice Boltzmann method; ML, machine learning.

- RMSprop (Root Mean Square Propagation) optimizer is used for training with a learning rate of "0.0002". Alternatively, we tested the more expansive ADAM optimizer, which resulted in similar trained model accuracy.

The computed values of the loss function over the number of weight updates (epochs) are shown in Figure 15 (left). In this, *EarlyStopping* technique is used to prevent overfitting during the training. In particular, the training ends when the loss metric monitored during training stops improving by a minimum delta ($\Delta_{\min} = 10^{-7}$) after a patience number of 50 epochs.

Figure 15, right, presents a comparison between the scaled MSE of the testing, validation, and training datasets. In this, a mean value of the scaled MSE of each state of deformation (a complete retention curve) is computed. The figure shows that the errors in all training, validation, and testing are below 0.008, whereas slightly fewer errors in the validation and testing cases.

Thereafter, we apply the final trained ML-based retention model to cases from the unseen test dataset, see Figure 16 for exemplary results for deformation states $\varepsilon = 0.06$ and 0.08 . The 1D CNN model proves its capability in capturing the hysteretic path-dependent retention behavior and the effect of deformation on the latter.

4.2 | RNN model

Alternative to the 1D CNN model for the history-dependent retention curve, a RNN-based retention curve model is proposed in the following. The chosen ML model for retention behavior depends on the data history for the computation of the saturation degree s^W. Figure 17 illustrates the difference between phenomenological models, such as the

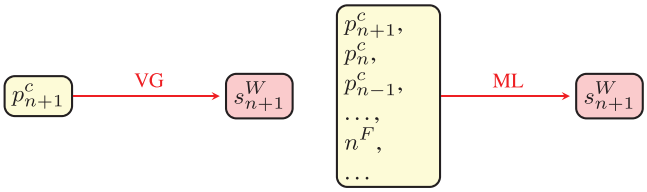

FIGURE 17 Graph representation of the VG retention model (left) and that of the machine learning RNN model (right).[2] RNN, recurrent neural network; VG, van-Genuchten.

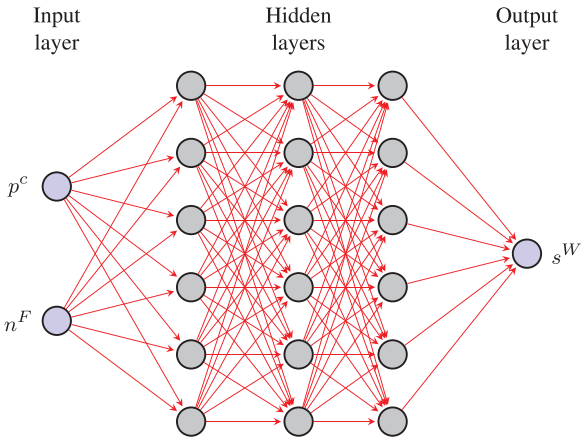

FIGURE 18 Illustration of the ANN structure in the ML-based retention curve model. The input parameters are the capillary pressure p^c and porosity n^F, and the output is the degree of saturation s^W.[2] ANN, artificial neural network; ML, machine learning.

van-Genuchten VG model, which is limited to yielding only one unique saturation degree s^W_{n+1} for each value of the capillary pressure $p_{c,n+1}$, and the proposed general ML-based RNN model, which utilizes the history of the capillary pressure, that is, $[p^c_{n-3}, p^c_{n-2}, p^c_{n-1}, p^c_n, p^c_{n+1}]$, as input parameters in addition to the deformation-dependent porosity n^F to determine the current saturation degree s^W_{n+1}, see[2] for analogy.

In this approach, the numerical data is similarly normalized in the pre-processing phase using the MinMaxScaler function. Within the RNN approach, we applied the LSTM neural network[62] implemented in Keras to take into account the history of the input variables in the prediction of the path-dependent retention curves. An illustration of the ANN structure in the ML-based retention curve model is shown in Figure 18. This shows the input parameters, which are the capillary pressure p^c and porosity n^F, and the output, which is the degree of saturation s^W.

In the model fitting, we distinguish between the training subset, chosen as 60% of the total database, the validation subset, chosen as 20% of the database (via the built-in "validation_split = 0.2" feature in Tensorflow), and the unseen testing subset, chosen as 20% of the database. As this data split is not unique, different ratios of the database can be examined to explore their effect on the accuracy of the trained model. In the presented results, the retention curves with states of deformation $\varepsilon \in \{0.02, 0.04, 0.1, 0.12, 0.16, 0.18\}$ are chosen for the training, the curves with the states of $\varepsilon \in \{0.14, 0.20\}$ for the validation, and the curves with the states of $\varepsilon \in \{0.06, 0.08\}$ for the testing.

In the training, a loss function based on the MSE of the scaled output data is used, see (18). The implemented NN settings are summarized in Table 2. They are chosen after carefully handcrafted testing and comparing different combinations of the settings. Applying the hyperparameter tuning approach presented in Heider et al.[2] shows that the proposed hyperparameters (Table 2) are among the best choices in terms of prediction accuracy of the trained model.

The loss function values over the number of weight updates (epochs) are presented in Figure 19, left. It is evident that the training and validation datasets result in coinciding curves, where no significant over-fitting can be observed. Figure 19, right, presents a comparison between the scaled MSE of the testing, validation, and training datasets. In this, a mean value of the scaled MSE of each state of deformation (a complete retention curve) is computed. The figure shows that the errors in all training, validation, and testing are below 0.008, whereas slightly more errors in the validation and testing cases (as also observed in Figure 19, left) are obtained.

TABLE 2 NN settings considered in the RNN-based retention model.

NN setting description	Abbreviation	Values
Neuron type subset	*NeuronType*	*LSTM*
Hidden layers subset	*numHiddenLayers*	2
Number of neurons per layer	*numNeuronsPerLayer*	100
Dropout rate subset	*DropOutRate*	0.0
Optimizer type subset	*Optimizer*	*Adam*
Learning rate	*lr*	0.01
Activation functions subset	*Activation*	*sigmoid*
Batch sizes subset	*BatchSize*	16

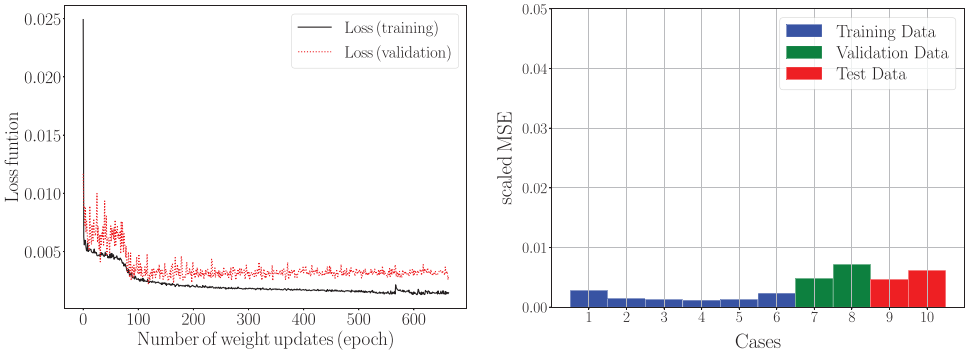

FIGURE 19 The values of the loss function with respect to the number of weight updates of the training and validation steps (left) and the scaled MSE versus the states of deformations of the training, validation, and test data (right) of the RNN model. MSE, mean squared error; RNN, recurrent neural network.

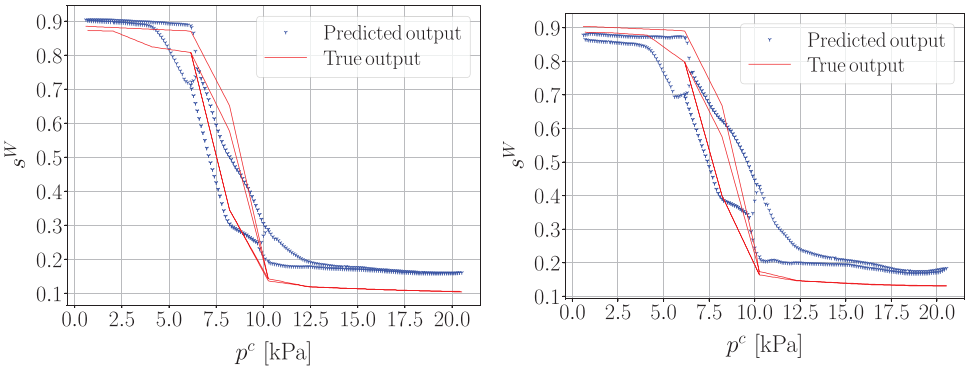

FIGURE 20 Comparison between the results of the LBM biphasic simulations, that is, true output, and the predicted output of the RNN model regarding the saturation degrees versus the capillary pressure for states of deformation $\varepsilon = 0.06$ (left) and $\varepsilon = 0.08$ (right). LBM, lattice Boltzmann method; RNN, recurrent neural network.

Similar to Section 4.1, the final trained ML-based retention model is applied to cases from the unseen testing dataset, compare Figure 20 for exemplary results for strain $\varepsilon = 0.06$ and 0.08 . As the results shown, the RNN model proves its capability in representing the deformation- and path-dependent hysteretic retention behavior.

The results presented in Sections 4.1 and 4.2 show the effectiveness of 1D CNN and RNN in processing sequential data related to the path-dependent retention model. The resulting ML models also show comparable accuracies as presented in Figures 15 and 19.

5 | DISCUSSION AND CONCLUSIONS

This work presented the implementation of ML-based models trained by the output data of LBM simulations in the multiscale study of interstitial single-phase flow and retention behavior of biphasic flow through deformable porous media. The advantage of this approach lies in the extraction of crucial material properties and responses from the microscale, for example, intrinsic permeability tensor and water saturation degree, and conveying these data to the macroscopic scale via ML-based models. Furthermore, the presented approach to calculate the anisotropic permeability tensor or the retention curve can be generalized to other types of porous media with different microgeometries and pore fluids with different fluid properties. As a first step, the LBM has been utilized to simulate single- and biphasic fluid flow through the deformable porous networks of the available micro-geometry of Bentheimer Sandstones. Concerning the former, investigation of the anisotropy of the intrinsic permeability was additionally achieved through comparison among the principal values of the permeability tensors and analyzing the changes of the principal directions due to deformation. As for the latter, the retention curves providing the relation between the saturation degree and capillary pressure for each strain degree have been determined. Consequently, the database from the LBM simulations was used for training the ML-based models.

For the ML-based anisotropic deformation-dependent permeability model, the FFNN approach has been utilized as the response is history independent. In this, a two-step FFNN model has been developed to first predict the fluid velocity of the no-slip boundary condition and then that of the natural slip boundary condition. The results proved the capability of the aforementioned ML model to accurately compute the output fluid velocity resulting from the fluid pressure gradient, although a nonlinear relationship exists between the two quantities. Based on that, the informed-graph-based FFNN provided accurate predictions of the fluid velocity in both Darcy and non-Darcy flow regimes.

As for the ML models used for determining the retention curves of the unsaturated deformable porous materials, the 1D CNN and RNN approaches have been used to simulate the path-dependent hysteretic nature of the saturation degree with respect to the capillary pressure. The advantage of such ML models is the computation of more than one saturation degree in the case of cyclical drying/wetting processes. This is not present in phenomenological models, such as the VG, where only one saturation degree is obtained for both processes. Moreover, ML-based models have been used for the first time to model hysteretic retention behavior. The 1D CNN and RNN models yielded comparable results in terms of accuracy for the retention curves. However, the training of the 1D CNN model required less computational costs than the RNN model. Nevertheless, both models proved their capability in predicting hysteretic retention curves for biphasic fluid flows through deformable porous materials.

Despite the fact that the ML models presented in this paper were able to compute the permeability tensor and retention data in a qualitatively accurate manner, enhancements can be made in future works. These include generating permeability datasets using multiphase flow in LBM and comparing the resulting unsaturated permeability tensor with the permeability tensor of fully saturated porous media. The future extensions also include the involvement of transfer learning in order to fix the data-insufficiency problem. In this, experimental retention data of deformable porous materials can be imported into the available database. In addition, our presented approach can be tested on another type of porous media, such as fibrous porous materials, for example, the gas diffusion layers of polymer electrolyte fuel cells, or open-celled foam materials with interconnected channels. In this regard, we plan to create a larger in-house database, which will help us test ML training methods like cross-validation effectively. We aim with this to overcome issues related to model generalization and, thus, enhance the reliability of our ML models. Our future research will additionally explore the application of Physics-Informed Neural Networks in conjunction with our data-driven material model. This combined approach can efficiently and accurately solve boundary-value problems, even with limited data, ensuring that the solutions adhere to the underlying physics of the system.

ACKNOWLEDGMENTS

Open access funding enabled and organized by Projekt DEAL.

The second author, Y. Heider, would like to gratefully thank the German Research Foundation (DFG) for the support of the project "Multi-field continuum modeling of two-fluid-filled porous media fracture augmented by microscale-based machine-learning material laws", grant number 458375627.

CONFLICT OF INTEREST STATEMENT

The authors declare that there is no conflict of interest.

DATA AVAILABILITY STATEMENT

Data available on request from the authors.

ORCID

Mohamad Chaaban https://orcid.org/0000-0002-2001-1407
Yousef Heider https://orcid.org/0000-0003-2281-2563
WaiChing Sun https://orcid.org/0000-0002-3078-5086

REFERENCES

1. Sun W, Andrade JE, Rudnicki JW, Eichhubl P. Connecting microstructural attributes and permeability from 3D tomographic images of in situ shear-enhanced compaction bands using multiscale computations. *Geophys Res Lett*. 2011;38(10).
2. Heider Y, Suh HS, Sun W. An offline multi-scale unsaturated poromechanics model enabled by self-designed/self-improved neural networks. *Int J Numer Anal Methods Geomech*. 2021;45(9):1212-1237. https://doi.org/10.1002/nag.3196
3. Wang K, Sun W. An updated lagrangian LBM–dDEM-FEM coupling model for dual-permeability fissured porous media with embedded discontinuities. *Comput Meth Appl Mech Eng*. 2019;344:276-305.
4. Chaaban M, Heider Y, Markert B. Upscaling LBM-TPM simulation approach of darcy and non-darcy fluid flow in deformable, heterogeneous porous media. *Int J Heat Fluid Flow*. 2020;83:108566.
5. Chaaban M, Heider Y, Markert B. A multiscale LBM-TPM–PFM approach for modeling of multiphase fluid flow in fractured porous media. *Int J Numer Anal Methods Geomech*. 2022;46(14):2698-2724. https://doi.org/10.1002/nag.3423
6. Andrä H, Combaret N, Dvorkin J, et al. Digital rock physics benchmarks—part I: Imaging and segmentation. *Comput Geosci*. 2013;50:25-32. https://doi.org/10.1016/j.cageo.2012.09.005
7. Wang K, Sun W. A semi-implicit discrete-continuum coupling method for porous media based on the effective stress principle at finite strain. *Comput Meth Appl Mech Eng*. 2016;304:546-583.
8. Heider Y, Wang K, Sun W. So (3)-invariance of informed-graph-based deep neural network for anisotropic elastoplastic materials. *Comput Meth Appl Mech Eng*. 2020;363:112875.
9. Heider Y, Sun W. Objectivity and accuracy enhancement within ann-based multiscale material modeling. *PAMM*. 2023;22(1):e202200203. https://doi.org/10.1002/pamm.202200203
10. Fuchs A, Heider Y, Wang K, Sun W, Kaliske M. DNN2: a hyper-parameter reinforcement learning game for self-design of neural network based elasto-plastic constitutive descriptions. *Comput Struct*. 2021;249:106505.
11. Wang K, Sun W. A multiscale multi-permeability poroplasticity model linked by recursive homogenizations and deep learning. *Comput Meth Appl Mech Eng*. 2018;334:337-380.
12. Cai C, Vlassis N, Magee L, et al. Equivariant geometric learning for digital rock physics: estimating formation factor and effective permeability tensors from morse graph. *Int J Multiscale Comput Eng*. 2023;21(5):1-24.
13. Linden L, Klein DK, Kalina KA, Brummund J, Weeger O, Kästner M. Neural networks meet hyperelasticity: a guide to enforcing physics. *J Mech Phys Solids*. 2023; 179:105363. https://doi.org/10.1016/j.jmps.2023.105363
14. Wessels H, Böhm C, Aldakheel F, et al. *Computational Homogenization Using Convolutional Neural Networks*. In: Benchmark problems, datasets and methodologies for the computational geosciences. Springer International Publishing; 2022:569-579. https://doi.org/10.1007/978-3-030-87312-7_55
15. Krüger T, Kusumaatmaja H, Kuzmin A, Shardt O, Silva G, Viggen EM. *The lattice Boltzmann Method*. Vol 10. Springer International Publishing; 2017:4-15.
16. Markert B. A constitutive approach to 3-D nonlinear fluid flow through finite deformable porous continua. *Transp Porous Media*. 2007;70(3):427.
17. Markert B, Heider Y, Ehlers W. Comparison of monolithic and splitting solution schemes for dynamic porous media problem. *Int J Numer Meth Eng*. 2010;82:1341-1383.
18. Heider Y. A review on phase-field modeling of hydraulic fracturing. *Eng Fract Mech*. 2021;253:107881.
19. Sweidan A, Niggemann K, Heider Y, Ziegler M, Markert B. Experimental study and numerical modeling of the thermo-hydro-mechanical processes in soil freezing with different frost penetration directions. *Acta Geotech*. 2022;17:231-255. https://doi.org/10.1007/s11440-021-01191-z
20. Nguyen C, Heider Y, Markert B. A non-isothermal phase-field framework for hydraulic fracture modeling in saturated porous media with convection-dominated heat transport. *Acta Geotech*. 2023;18:4515-4538. https://doi.org/10.1007/s11440-023-01905-5
21. Wang Y, Cheung SW, Chung ET, Efendiev Y, Wang M. Deep multiscale model learning. *J Comput Phys*. 2020;406:109071.
22. Choubineh A, Chen J, Coenen F, Ma F. An innovative application of deep learning in multiscale modeling of subsurface fluid flow: reconstructing the basis functions of the mixed GMsFEM. *J Pet Sci Eng*. 2022;216:110751.
23. Bao J, Murugesan V, Kamp CJ, Shao Y, Yan L, Wang W. Machine learning coupled multi-scale modeling for redox flow batteries. *Adv Theory Simul*. 2020;3(2):1900167.

24. Alqahtani N, Armstrong RT, Mostaghimi P. Deep learning convolutional neural networks to predict porous media properties. In: *SPE Asia Pacific Oil and Gas Conference and Exhibition*. OnePetro; 2018.
25. Kamrava S, Im J, de Barros FP, Sahimi M. Estimating dispersion coefficient in flow through heterogeneous porous media by a deep convolutional neural network. *Geophys Res Lett*. 2021;48(18):e2021GL094443.
26. Lähivaara T, Kärkkäinen L, Huttunen JM, Hesthaven JS. Deep convolutional neural networks for estimating porous material parameters with ultrasound tomography. *J Acoust Soc Am*. 2018;143(2):1148-1158.
27. Zhang Z, Yan X, Liu P, Zhang K, Han R, Wang S. A physics-informed convolutional neural network for the simulation and prediction of two-phase darcy flows in heterogeneous porous media. *J Comput Phys*. 2023;477:111919.
28. Kiranyaz S, Avci O, Abdeljaber O, Ince T, Gabbouj M, Inman DJ. 1D convolutional neural networks and applications: a survey. *Mech Syst Sig Process*. 2021;151:107398. https://doi.org/10.1016/j.ymssp.2020.107398https://www.sciencedirect.com/science/article/pii/S0888327020307846
29. Zhang Z, Song X. Nanoscale soil-water retention curve of unsaturated clay via MD and machine learning. Computers and Geotechnics. 2023;163:105678.
30. Lamorski K, Šimunek J, Sławiński C, Lamorska J. An estimation of the main wetting branch of the soil water retention curve based on its main drying branch using the machine learning method. *Water Resour Res*. 2017;53(2):1539-1552.
31. van Genuchten M. A closed-form equation for predicting the hydraulic conductivity of unsaturated soils. *Soil Sci Soc Amer J*. 1980;44:892-898.
32. Boltzmann L. *Lectures on Gas Theory*. Courier Corporation; 2012.
33. Bhatnagar PL, Gross EP, Krook M. A model for collision processes in gases. I. small amplitude processes in charged and neutral one-component systems. *Phys Rev*. 1954;94:511-525. https://doi.org/10.1103/PhysRev.94.511
34. Wolf-Gladrow DA. *Lattice-Gas Cellular Automata and Lattice Boltzmann Models: an introduction*. Springer; 2004.
35. Zou Q, He X. On pressure and velocity boundary conditions for the lattice boltzmann BGK model. *Phys Fluids*. 1997;9(6):1591-1598.
36. Shan X, Chen H. Lattice Boltzmann model for simulating flows with multiple phases and components. *Phys Rev E*. 1993;47(3):1815.
37. Shan X, Chen H. Simulation of nonideal gases and liquid-gas phase transitions by the lattice boltzmann equation. *Phys Rev E*. 1994;49(4):2941.
38. Raiskinmäki P, Shakib-Manesh A, Jäsberg A, Koponen A, Merikoski J, Timonen J. Lattice-Boltzmann simulation of capillary rise dynamics. *J Stat Phys*. 2002;107(1):143-158.
39. Raiskinmäki P, Koponen A, Merikoski J, Timonen J. Spreading dynamics of three-dimensional droplets by the lattice-Boltzmann method. *Comput Mater Sci*. 2000;18(1):7-12
40. Qian YH, Succi S, Orszag S. Recent advances in lattice Boltzmann computing. *Annual Reviews of Computational Physics III*. 1995:195-242.
41. Martys NS, Chen H. Simulation of multicomponent fluids in complex three-dimensional geometries by the lattice Boltzmann method. *Phys Rev E*. 1996;53(1):743.
42. Pan C, Hilpert M, Miller C. Lattice-Boltzmann simulation of two-phase flow in porous media. *Water Resour Res*. 2004;40(1):W01501.
43. Kuhn MR, Sun W, Wang Q. Stress-induced anisotropy in granular materials: fabric, stiffness, and permeability. *Acta Geotech*. 2015;10(4):399-419.
44. Galindo-Torres S, Scheuermann A, Li L. Numerical study on the permeability in a tensorial form for laminar flow in anisotropic porous media. *Phys Rev E*. 2012;86(4):046306.
45. Moon C, Andrew M. Bentheimer networks. 2019. http://www.digitalrocksportal.org/projects/223 https://doi.org/10.17612/1a36-rn45
46. Chaaban M, Heider Y, Markert B. A multiscale study of the retention behavior and hydraulic anisotropy in deformable porous media. *PAMM*. 2023;23(1):e202200129.
47. Bernardes E, Viollet S. Quaternion to euler angles conversion: a direct, general and computationally efficient method. *Plos One*. 2022;17(11):e0276302.
48. Rutka V, Wiegmann A. Explicit jump immersed interface method for virtual material design of the effective elastic moduli of composite materials. *Numer Algorithms*. 2006;43:309-330.
49. Xue L, Guo X, Chen H. *Fluid Flow in Porous Media*. World Scientific; 2020. https://doi.org/10.1142/11805
50. Banerjee R, Sagiyama K, Teichert G, Garikipati K. A graph theoretic framework for representation, exploration and analysis on computed states of physical systems. *Comput Meth Appl Mech Eng*. 2019;351:501-530.
51. McKinney W. Data structures for statistical computing in python. In: *Proceedings of the 9th Python in Science Conference*. Vol 445. Austin, TX; 2010:51-56.
52. Pedregosa F, Varoquaux G, Gramfort A, et al. Scikit-learn: machine learning in Python. *J Mach Learn Res*. 2011;12:2825-2830.
53. Chollet F, et al. (2015) Keras [www document]. GitHub https://github.com/keras-team/keras
54. Kingma DP, Ba J. Adam: a method for stochastic optimization. arXiv Preprint arXiv:1412.6980, 2014.
55. JE G. *Computational Statistics*. Springer; 2009.
56. Gu J, Wang Z, Kuen J, et al. Recent advances in convolutional neural networks. *Pattern Recognit*. 2018;77:354-377. https://doi.org/10.1016/j.patcog.2017.10.013
57. Eidel B. Deep CNNs as universal predictors of elasticity tensors in homogenization. *Comput Meth Appl Mech Eng*. 2023;403:115741.
58. Aldakheel F, Soyarslan C, Palanisamy H, Elsayed E. Machine learning aided multiscale magnetostatics. Mechanics of Materials 2023;184:104726

59. Dhillon A, Verma G. Convolutional neural network: a review of models, methodologies and applications to object detection. *Prog Artif Intell*. 2020;9:85-112. https://doi.org/10.1007/s13748-019-00203-0
60. Aldakheel F, Elsayed E, Tarek Z, Wriggers P. Efficient multiscale modeling of heterogeneous materials using deep neural networks. *Comput Mech*. 2023;72:155-171. https://doi.org/10.1007/s00466-023-02324-9
61. Tandale S, Stoffel M. Recurrent and convolutional neural networks in structural dynamics: a modified attention steered encoder–decoder architecture versus LSTM versus GRU versus TCN topologies to predict the response of shock wave-loaded plates. *Comput Mech*. 2023;72:765-786. https://doi.org/10.1007/s00466-023-02317-8
62. Hochreiter S, Schmidhuber J. Long short-term memory. *Neural Comput*. 1997;9(8):1735-1780.

How to cite this article: Chaaban M, Heider Y, Sun W, Markert B. A machine-learning supported multi-scale LBM-TPM model of unsaturated, anisotropic, and deformable porous materials. *Int J Numer Anal Methods*. 2024;48:889–910. https://doi.org/10.1002/nag.3668