Soft Matter

ARTICLE TYPE

Cite this: DOI: 00.0000/xxxxxxxxxx

Machine learning-based structure-property predictions in silica aerogels

Rasul Abdusalamov,*[a†] Prakul Pandit,[b†] Barbara Milow,[b] Mikhail Itskov,[a] and Ameya Rege*[b]

Received Date
Accepted Date

DOI: 00.0000/xxxxxxxxxx

The structural features in silica aerogels are known to be modelled effectively by the diffusion-limited cluster-cluster aggregation (DLCA) approach. In this paper, an artificial neural network (ANN) is developed for predicting the fractal properties of silica aerogels, given the input parameters for a DLCA algorithm. This approach of machine learning substitutes the necessity of first, generating the DLCA structures and then simulating and characterising their fractal properties. The developed ANN demonstrates capabilities of predicting the fractal dimension for any given set of DLCA parameters within an accuracy of $R^2 = 0.973$. Furthermore, the ANN is subsequently inverted for predicting the input parameters for reconstructing a DLCA model network of silica aerogels, for a given desired target fractal dimension. There, the problem of uniqueness is solved by using a guided gradient descent approach. Model DLCA structures are generated from the constrained and unconstrained inversion, and are compared against several parameters, amongst them, the pore-size distributions. The constrained inversion of the ANN is shown to predict the DLCA model parameters for a desired fractal dimension within an error of 2%.

1 Introduction

The data-driven material modelling is gaining significant attention in the scientific community. New machine learning (ML) and data mining approaches are gaining precedence in predictive modelling of materials.[1] The primary objective of applying ML methods in materials modelling is to exploit the prospect of enabling, accelerating or even simplifying the development of novel materials by means of reverse engineering. ML makes it possible to integrate physics-informed know-how in the form of governing equations, boundary conditions or constraints to manage ill-poised problems. A classical material model can integrate ML to develop surrogate models and overcome problems with parameter identification and uncertainty quantification. Such data-driven approaches have been coupled with atomistic models,[2,3] micromechanical models[4–6] as well as multiscale models.[7] A particular challenge is the identification and prediction of optimised microstructure configurations to prescribe the best material properties for specific applications. Addressing this issue, for example, Liu et al.[8] developed a predictive ML-based model for designing magnetoelastic Fe-Ga alloy microstructures for desired target properties. A systematic framework consisting of random data generation, feature selection and classification algorithms was developed. Generating optimised nanoporous materials using artificial neural networks (ANNs) can potentially lead to a notable revolution in future materials design. However, so far there have been not many successful attempts of ANNs to generate novel materials as ML has only been primarily applied to predict material properties. Recently, Kim et al.[9] implemented a generative adversarial network (GAN) to predict new zeolite structures based on target user-defined properties. This was a remarkable progress in the application of ML in the development of porous materials.

Silica aerogels represent excellent examples of fractal structures. The structural and fractal properties of silica aerogels have been modelled by means of all-atom molecular dynamics simulation,[10,11] coarse-grained simulation,[12] and the diffusion-limited cluster-cluster aggregation (DLCA) method.[13,14] The latter method was shown to predict accurately the aggregation mechanism in the synthesised silica aerogels by means of the sol-gel process, which involves the assembly of small particles into clusters and networks. In this approach, the Brownian motion of the aerogel particles resulting in the formation of clusters and aggregates is modelled. Computationally, this approach of modelling is a "slow" process. This means that for simulating the structural and fractal properties of a reasonably large box size, e. g., 300^3 nm^3 by means of the DLCA, huge computational resources are required especially in context of all-atom simulations. Coupling these constraints with the requirement of modelling DLCA structures with target properties, the resulting trial and error process in modelling and simulating the properties makes the classical approach from a materials development perspective very ex-

[a] Department of Continuum Mechanics, RWTH Aachen University, Aachen, Germany. E-mail: abdusalamov@km.rwth-aachen.de
[b] Department of Aerogels and Aerogel Composites, Institute of Materials Research, German Aerospace Center, Cologne, Germany. E-mail: ameya.rege@dlr.de
† These authors contributed equally to this work.

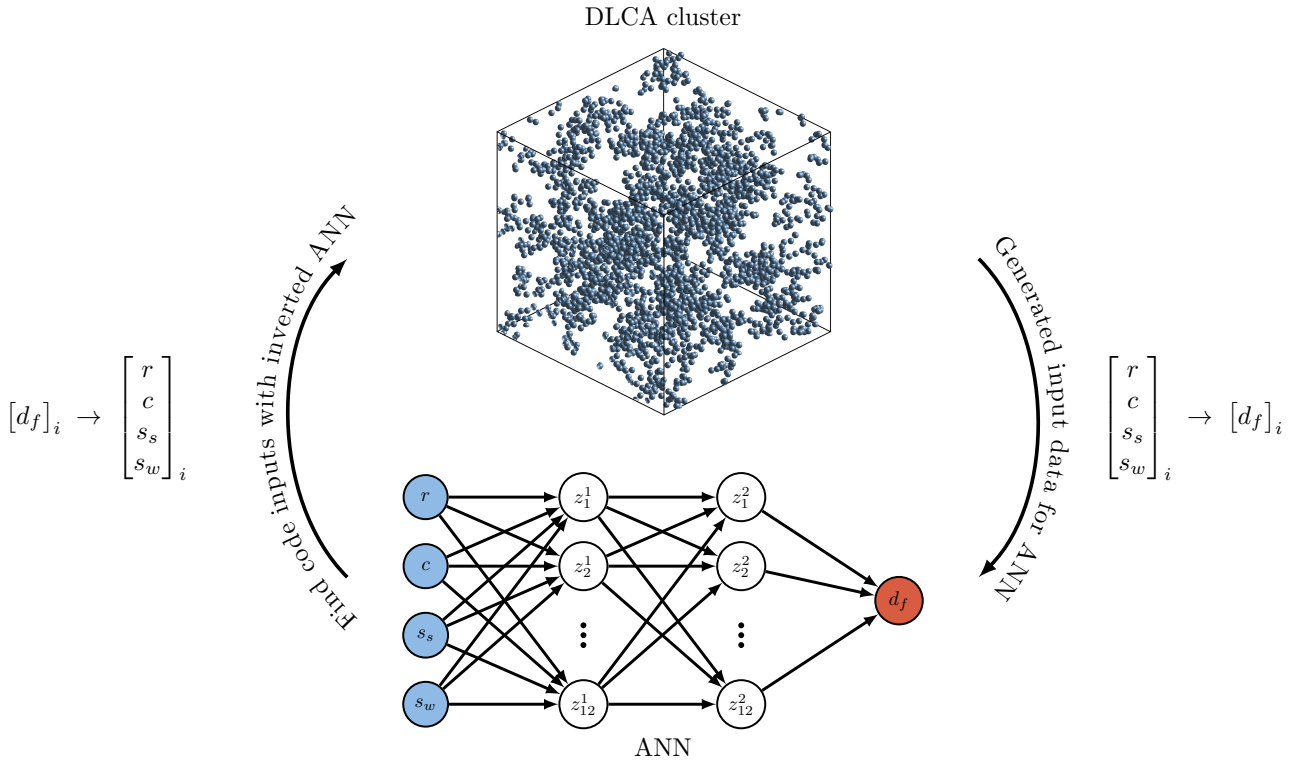


Fig. 1 Overview of the process: the DLCA algorithm is applied to generate clusters that are used to determine fractal dimensions for different input parameters. These structures form the training dataset of an ANN, which is developed and trained to predict fractal dimensions, without having to simulate the structures. The ANN is then inverted to predict the input structural properties for a given fractal dimension.

pensive.

In order to circumvent such problems, data-driven modelling approaches would be suitable. DLCA models have so far been used in a forward sense. This means that for a given silica aerogel, DLCA was applied to model its structural and fractal features, and the model results were then validated against the experimental data. For a large amount of silica aerogels, generating the fractal properties with minor changes in the model parameters is a very time-consuming process, involving a number of small angle X-ray scattering (SAXS) or small-angle neutron scattering (SANS) tests. If an acceptable amount of training data can be generated with DLCA, an ANN-based model could be useful in predicting the fractal properties given desired input parameters. It becomes even more interesting to generate a DLCA structure with the desired target output. This becomes a backward problem, and is traditionally realised in a trial-and-error process. To this end, a backward propagation of a feed-forward ANN could be used to reverse engineer the parameters that would alter the structural features in aeogels.

Accordingly, an ANN is first developed, where the DLCA model is employed to generate model silica aerogel structures that are used as the training and test data sets. Thus, the DLCA model parameters act as input neurons and the fractal dimension is the desired output of the ANN. In the second part, the ANN is inverted, such that for a target fractal dimension, the desired input parameters of the DLCA model are obtained. This is an example of reverse engineering of the modelling process for aerogels, a first-of-its-kind. An illustration of the forward and backward process is sketched in Fig. 1.

The paper is organised as follows. The methodology of using ANNs is detailed in section 2. A further description of the inversion process is later presented. In section 3 the forward and back propagation of the ANN, as well as their results, are presented. A summary and conclusion are finally presented in section 4.

2 Methodology

In a mathematical sense, ANNs can be characterised as non-linear functions mapping an input space to an output one.[15] The inspiration for ANNs comes from animal brains, which consist of interconnected neurons receiving and processing information. To make the ANN learn for accurately predicting desired output values given some inputs, it is necessary to provide a training dataset. Within the context of this paper, the generation of such a dataset by means of the DLCA model is described in subsection 2.1. The inter-connectivity of the neurons can be arranged in diverse ways. The architecture of the ANN we implemented is specified in subsection 2.2. The main goal of this work was to reconstruct the silica aerogel network structure from the provided target fractal properties. In order to realise this goal, the developed ANN was inverted using a gradient descent algorithm (see subsection 2.3). The overall process for the development of the network structure can be divided into three parts:

1. Setting up the DLCA model for simulating the aerogel structure and obtaining the structural parameters.

2. Developing an ANN and its hyper parameter optimisation to predict fractal dimensions from the structural parameters.

3. Inverting the trained network to predict the input parameters for specific fractal dimensions.

2.1 Numerical simulation of silica aerogels to generate training data

The DLCA method was used to generate 3125 clusters for training the ANN to predict the fractal dimension and to perform a micro-structure reconstruction. The methodology and implementation of DLCA modelling process has already been reported.[14] The model parameters for the numerical simulations were the particle radius, the particle concentration (an interpretation of the aerogel density), the step size of the seeds and the step size of the walkers. Each parameter was varied 5 times while calculating the results for the simulation over 5 repetitions to negate any random effects in the computational model. Subsequently, the fractal dimension of the clusters was determined and numerically averaged for all repetitions, resulting in a total of 625 data points. The DLCA clusters were generated in MATLAB and the output was exported for further processing.

Prior to training the ANN, the generated data was pre-processed in order to improve the ANN's prediction accuracy and restrict the input space in a certain domain so as to avoid gradient explosion while inverting the ANN. The data was feature scaled in the range of [-0.5, 0.5] to restrict the domain of input space while inverting the ANN followed by splitting the data-set into 60-25-15 percent for training, validation and testing of the network, respectively.

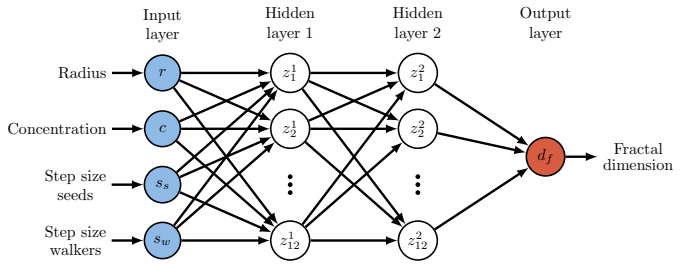


Fig. 2 Visualisation of the ANN architecture used with 4 inputs, 2 hidden layers and a single output.

2.2 ANN architecture and hyper-parameters optimisation

A common ANN consists of an input layer, hidden layers and an output layer having a certain number of neurons in each layer. The architecture of the chosen ANN was as follows: It consisted of an input layer with 4 inputs, 2 hidden layers and an output layer with one output (see Fig. 2). Each hidden layer consisted of 24 neurons, where $f(x) = \tanh(x)$ was chosen as an activation function. The choice of activation functions, the number of hidden layers and the number of neurons have a strong influence on the learning rate and efficiency of the ANN. The four input variables were also the input parameters to the DLCA model: the particle radius r, the concentration c, the step size of walkers s_w and the step size of seeds s_s; and the output variable was the fractal dimension d_f of the aggregate. The architecture and its hyper parameters were selected and optimised depending upon the type of problem and the available training data. In this work, the hyper-parameter optimisation for the ANN was realised using the grid search method, wherein the network was tested based on the Mean Square Error (MSE) metric and the quality of the fit was based on the R^2 score. The ANN was trained using the TENSORFLOW package in PYTHON. After analysing the optimisation through the grid search, a learning rate of 0.01 was selected to train the ANN for 2000 epochs. Here, the Adams optimiser was used.

The input nodes were programmed to be non-linearly activated in order to restrict the input domain within a certain interval during the inversion process through back-propagation. For this purpose, a symmetric inverse sigmoid activation function

$$f(a) = \mathrm{sig}^{-1}(a) = -\ln\left(\frac{1}{a+0.5} - 1\right)$$

$$\text{with} \quad a = \sum_{i=1}^{4} w_i x_i + b \,, \tag{1}$$

was selected, which restricted each input in the interval of [-0.5,0.5] when the network was inversed. Herein, x_i represents each input, w_i denote the corresponding weights and b the bias (see Fig. 3 for single neuron).

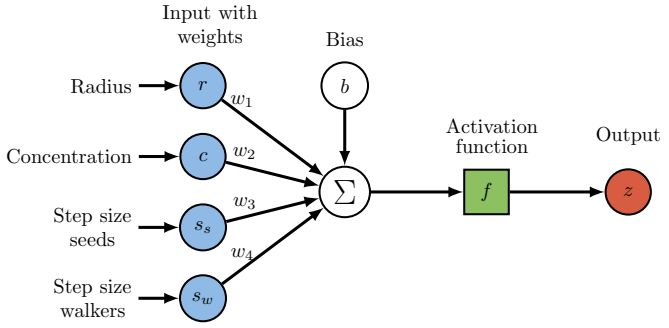


Fig. 3 Visualisation of a single neuron with an activation function.

2.3 ANN inversion for inverse design

The concept of using back-propagation in the inversion of a neural network was first introduced by R. J. Williams in 1986.[16] The concept was then applied by Kindermann et al.[17] in the inversion of neural networks to identify hand written digits by using gradient descent. The goal of a neural network in supervised learning is to achieve the minimum possible error for an input I with respect to a target T, and this error can be mathematically expressed as

$$E = (F(I) - T) < \tau \,, \tag{2}$$

where τ is the allowable tolerance level for the deviation of the neural network output $F(I)$ from the desired target T. While a neural network during forward mapping calculates an output for a given input pattern, the inversion of a neural network in-

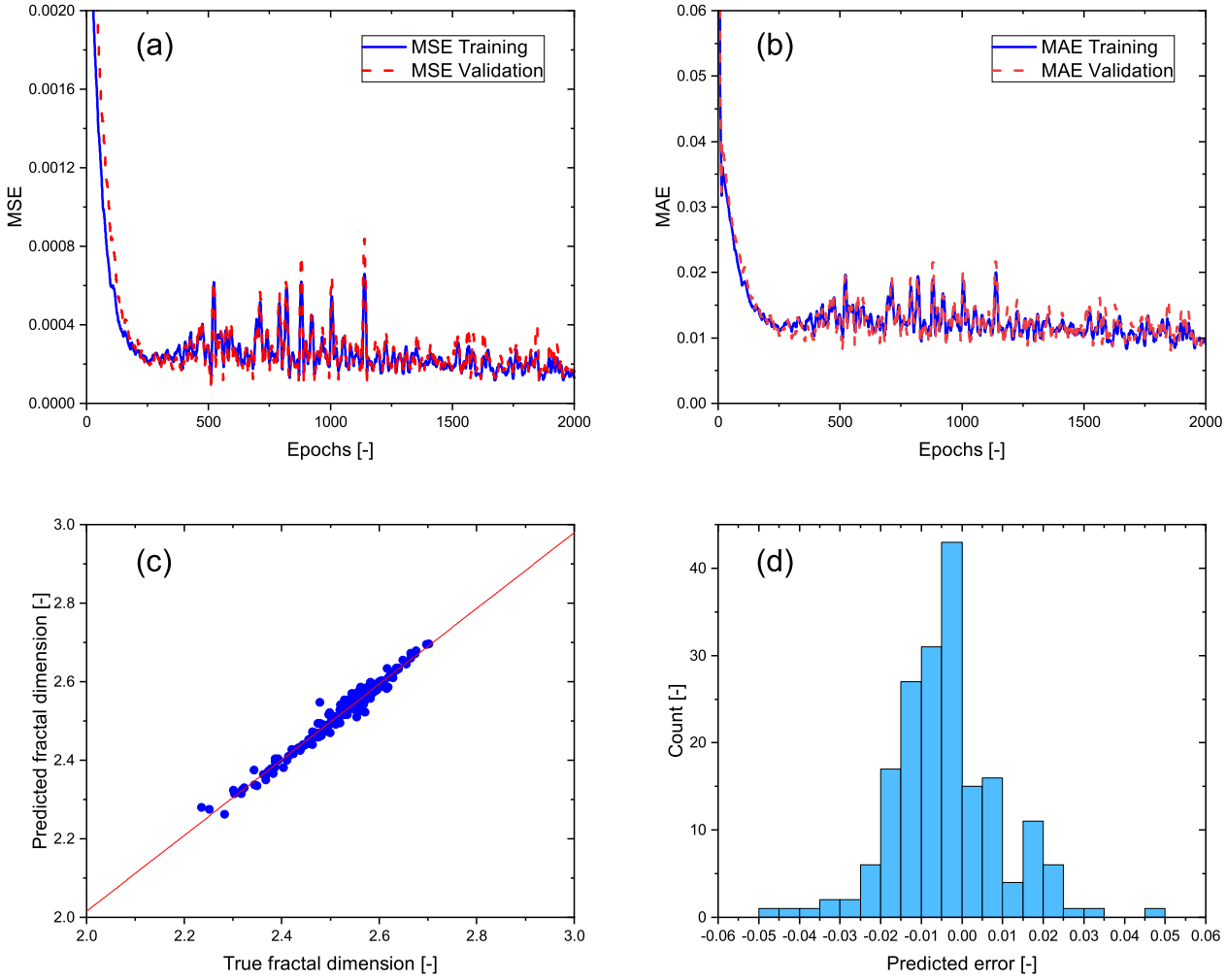


Fig. 4 (a) and (b) Training history for the ANN visualising the MSE and MAE, respectively, (c) and (d) results for prediction accuracy of the ANN showing the regression and the error plot for the fractal dimensions, respectively.

volves the calculation of input patterns that would lead to the desired output. This can be realised by implementing the gradient descent in the input space, wherein a set of input vectors $I^0, I^1, I^2, \ldots I^n$ are calculated such that the network error is minimised. The calculation of these input vectors was realised as

$$I^{l+1} = I^l - \eta \frac{\partial E}{\partial I^l} \,, \tag{3}$$

where η denotes the learning rate and influenced how quickly it reached the global minimum. This method is typically used for any neural network type, given the condition that the partial derivatives for the mappings within the network exists. However, in case of multi-layer perceptrons, as this particular ANN, the classical back propagation algorithm was used. As proposed by Kindermann et al.,[17] the error function was defined as

$$E = E_{LMS} = \sum_{k \in O} (t_k - a_k)^2 \,, \tag{4}$$

where O is the index of the output units, while $T = t_k$ and $F(I) = a_k$. As explained in subsection 2.2 a symmetric inverse sigmoid activation function was used such that the input layer had a non-linear activation. Thus, the inputs did not get arbitrarily large during the inversion of the network preventing gradient explosion. From Eq. (1), the net input of the input unit $i \in I$ can be calculated by taking the inverse as

$$\text{net}_i = f^{-1}(a_i) \,. \tag{5}$$

The idea of inversion with back-propagation involves carrying the error signals δ_i back to the input units. Using Eq (1), the input units were updated after determining the error signals δ_i as gradients of the error functions in regards to the network input. These error signals δ_i were evaluated as

$$\delta_i = -\frac{\partial E}{\partial \text{net}_i} = -\frac{\partial E}{\partial a_i} \frac{\partial a_i}{\partial \text{net}_i} \,. \tag{6}$$

The adaptation was then controlled by a learning rate η such as to perform increment/decrements in small steps. However, one of the major problems in inverting an ANN is uniqueness. Due to the fact that there exist several input patterns for a single output of fractal dimension, it would be difficult to generate the original model parameters through the ANN inversion. To this end, input constraints were added to the error function E, constraining the

Table 1 Convergence of input parameters by inverting the ANN without input constraints.

Epoch	Radius	Concentration	Seed Steps	Seed Walkers	Fractal Dimension
0	4.00	0.08	1.0	1.7	2.55424851
1000	3.762151	0.077013	1.060287	1.579455	2.54978388
2000	3.641224	0.075482	1.092810	1.519842	2.54829975
3000	3.575712	0.074648	1.111009	1.487991	2.54764679
4000	3.537936	0.074648	1.121699	1.469763	2.54730345
5000	3.537936	0.074166	1.121699	1.469763	2.54730345

range of the input parameters so as to provide a guided gradient descent during the back-propagation. An additional term was introduced in the final error function E' as

$$E' = (F(I) - T)^2 + (I - I_t)^2 \,, \tag{7}$$

where I specifies the initial input vector which iterated from I_0 to I_n and I_T denotes the target input vector which was approximated simultaneously along with the output target vector T. The error signals in Eq. (6) were rewritten as

$$\frac{\partial E'}{\partial \text{net}_i} = \frac{\partial E_{LMS}}{\partial \text{net}_i} + \frac{\partial (I - I_t)^2}{\partial a_i} \frac{\partial a_i}{\partial \text{net}_i} \,. \tag{8}$$

This leads to the extended updated rule

$$\Delta_i = \eta \delta_i + \kappa \zeta_i \quad \text{with} \quad \zeta_i = \frac{\partial (I - I_t)^2}{\partial a_i} \frac{\partial a_i}{\partial \text{net}_i} \,, \tag{9}$$

where ζ_i is the input constraint and κ determines its effect on the extended update rule.

3 Results and discussion

3.1 Predicting the fractal dimension

In order to measure the performance while training the ANN, the training loss in terms of the metrics, namely, MSE and mean absolute error (MAE), were plotted and visualised. It can be observed from Fig. 4 (a)-(b), that the proposed model was trained without resulting in an over or under fit. It was also observed that, initially up to *ca*. 300 epochs, the network's loss during validation was lesser than that during training. This can be attributed to the fact that the training error was averaged over a whole epoch, while the validation error was calculated only at the end of an epoch. Another reason for this behaviour can be explained by the large size of the training set as compared to the validation set, leading to a slightly higher loss. However, the validation loss converged with the training loss as the training continued, which confirms the ideal fit of the network. Thus, methods to prevent over-fitting, e. g., regularisation or early stopping were not required.

As discussed in the previous section, the model architecture with 2 hidden layers including 24 nodes each and a tanh activation function were selected based on hyper-parameter tuning through the grid search. The trained ANN was tested on unknown samples of the test set which were 15 percent of the total data set of 625 models. To determine the performance of the ANN on the test set, the MSE and MAE were calculated. Moreover to visualize the prediction accuracy, a scatter plot of the fractal dimension predicted by the ANN and the true fractal dimension from the corresponding DLCA simulations were realised and are presented in Fig. 4 (c). The trained ANN had a MAE of 0.09 and a R^2 score of 0.973 on the test set, which lies within the desired error limits and satisfies the requirements for the curve fit. Additionally, the resulting error of the prediction error can be visualised in Fig. 4 (d). It was observed that the error distribution had a mean approximately at -0.01 with certain outliers.

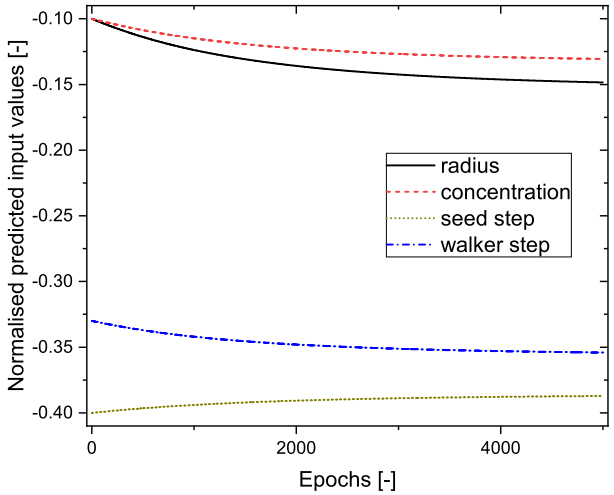


Fig. 5 Graphical illustration of the parameter convergence without input constraints.

3.2 Reconstruction of network structure input parameters

In order to predict the input parameters for a DLCA model given a desired fractal dimension, the trained ANN was then inverted using back-propagation in input space as described in subsection 2.3. The inversion algorithm was run with and without the input constraints to analyse how guided gradient descent effects the accuracy of determining input parameters.

For the initial case of inversion without input constraints, a fractal dimension of $d_f = 2.5724$ was provided to the algorithm. According to the initial numerical computations, the specific d_f was obtained for the DLCA parameters r, c, s_s and s_w equal to 3, 0.06997, 0.5, 1.125, respectively. These parameters were found to be very similar to those ones used to model silica aerogel networks.[14] The inversion algorithm was run using a learning rate of $\eta = 0.01$ for 5000 epochs. Table 1 illustrates the per epoch results of the inversion and the corresponding calculated fractal dimension at the respective epoch for the given case. Additionally, the convergence of the different parameters is visualised in Fig. 5. It was observed that although the ANN estimated the input parameters satisfying the desired fractal dimension, the value of these inputs was not equal to the desired model inputs, inferring

Table 2 Convergence of input parameters by inverting the ANN with defined input constraints.

Epoch	Radius	Concentration	Seed Steps	Seed Walkers	Fractal Dimension
0	6	0.04	0.60	2.29	2.5994848
1000	3.079717	0.068360	0.587438	1.242786	2.55851931
2000	3.078740	0.070022	0.536378	1.159080	2.57119181
3000	3.001809	0.069987	0.517222	1.134032	2.57224734
4000	3.012177	0.070175	0.507018	1.128261	2.59541093
5000	3.000253	0.069973	0.508511	1.127286	2.5724425

that the desired minimum in the solution plane was not achieved. Moreover, the lack of a constraint in the inversion algorithm could sometime lead also to a divergence away from the desired solution. This can happen when the initial assumption of inputs for the gradient descent is not close to the desired inputs to be calculated, thus requiring more epochs to achieve a minimum. Thus, the selection of the input vector I^0, and accordingly the learning rate, greatly influences the results of the inversion in this case.

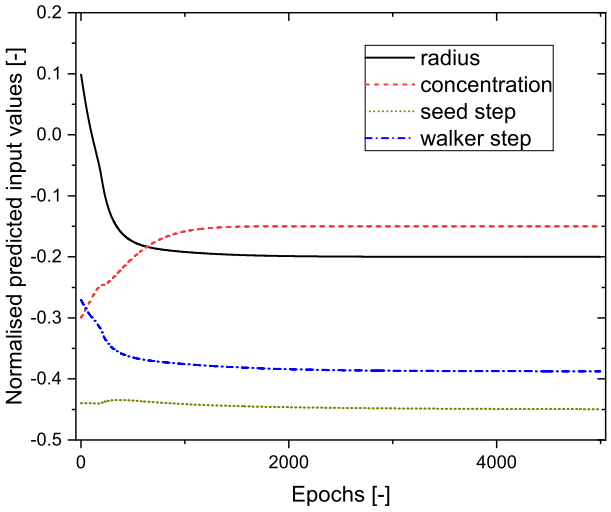


Fig. 6 Graphical illustration of the parameter convergence with input constraints.

For the second case, input constraints were provided in the inversion algorithm to guide the gradient descent towards the desired values. The variable κ influenced the control of these constraints on the inversion and its value was set to 0.03 with a learning rate of $\eta = 0.01$. The desired input values and fractal dimension were similar to the previous case with the inputs: $r = 3$, $c = 0.06997$, $s_s = 0.5$, $s_w = 1.125$, $d_f = 2.5724$. Table 2 exhibits that the inversion converged to the desired input values while also providing the desired fractal dimension. Moreover, these results were independent from the selected value of the initial assumption of inputs I^0 due to the existence of the constraints. The accuracy of the inversion, the independence from initial input assumption I^0 and the epochs required were greatly influenced by the factor κ. The visualisation of the inversion process is depicted in Fig. 6.

To compare the performance of the inversion algorithm with defined and undefined constraints, a set of 35 samples were randomly selected from the original DLCA simulation test data set. These samples were used to calculate the input parameters from the fractal dimensions determined through the respective DLCA

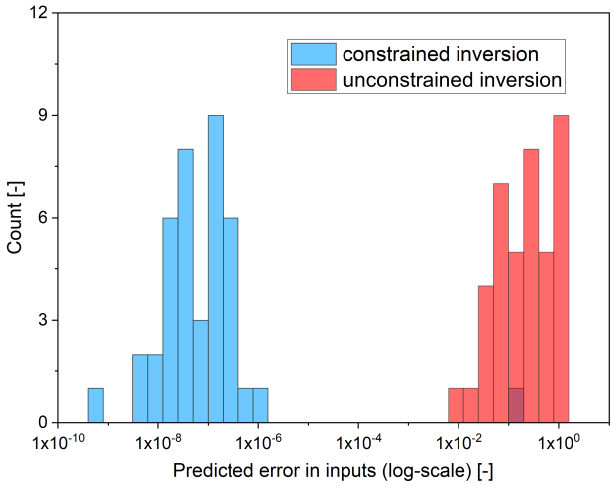


Fig. 7 MSE in the inputs calculated from the inversion of the ANN.

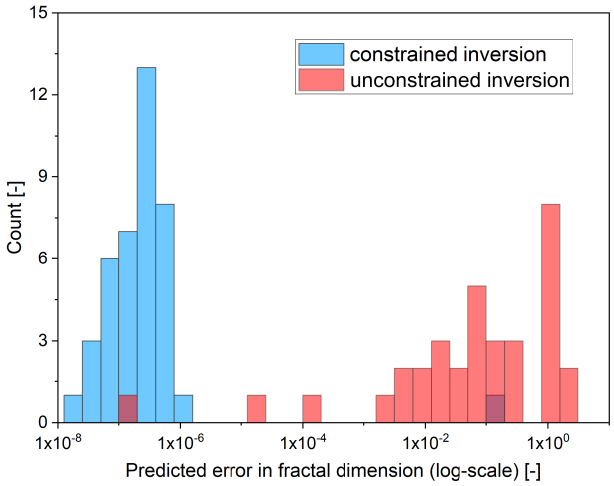
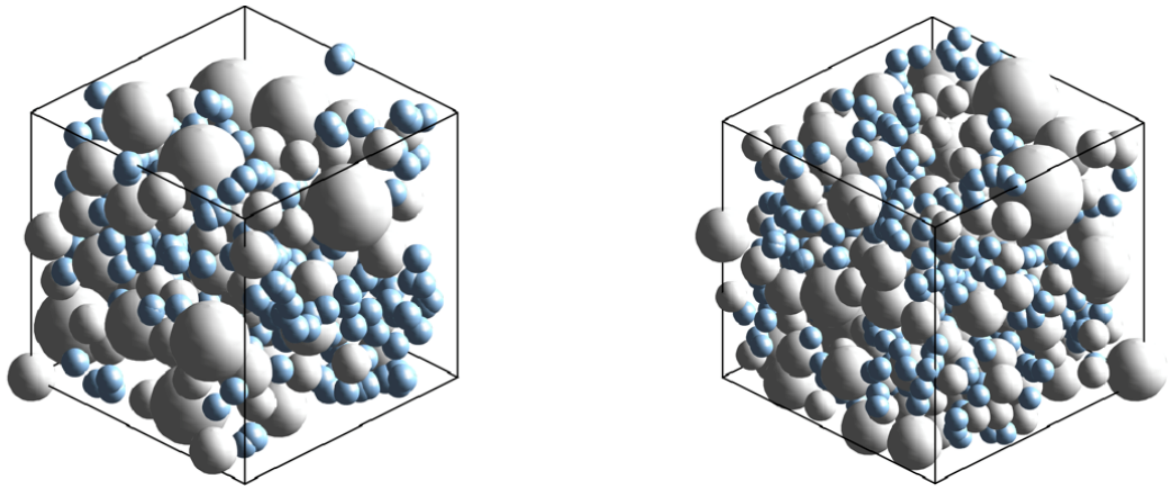


Fig. 8 MSE in the fractal dimensions calculated from the inverted inputs.

simulation. It can be observed from Fig. 7 that for the prediction of the input parameters through a defined fractal dimension, the MSE for a constraint inversion had a mean of around 10^{-7} on the logarithmic scale whereas for the same sample set for unconstrained inversion, a mean MSE of about 10^0 on the logarithmic scale was obtained. This behaviour can be attributed to the problem of non-uniqueness, i. e., the existence of multiple solutions (inputs) for a given fractal dimension. Thus, the availability of constraints during inversion makes it more accurate enabling the reverse engineering of silica aerogel structures with desired properties. The algorithms were also compared based on the obtained fractal dimension. These results can be visualised through the



(a) Cluster generated from the unconstrained inputs.

(b) Cluster generated from the constrained inputs.

Fig. 9 Visualization of the generated cluster structures using the inputs in Tables 1 and 2, for a given fractal dimension. The blue spheres indicate the particles and the grey ones indicate the pores.

error plots given in Fig. 8. The constrained inputs demonstrate a much better accuracy in predicting the fractal dimension. In the case of the unconstrained solution, the fractal dimension can strongly vary. It was noticed that depending on the initial guess, the deviation of the predicted fractal dimension can be quite high.

3.3 Comparison of predicted structures based on pore-size distributions

It is of special significance to validate the performance of the ANN in the structure-property prediction of the silica aerogels modelled through DLCA. To this end, the silica aerogel structures were reconstructed by means of the DLCA simulation by considering the input parameters obtained by both, the constrained and the unconstrained algorithm of the inversion of the ANN. Although the predicted fractal dimension for this particular case was similar, the generated clusters led to completely different morphological parameters. This can be seen from Figure 9, where the cluster particles are visualized in blue, and the pores are shown in grey. Structures with similar output properties, but with different input parameters, led to a clearly different interconnectivity of the particles. In the case of the unconstrained inputs, the concentration was smaller than that of the constrained inputs, which implies larger pores. By performing a grid-based search according to Bourcier et al.,[18] it was possible to determine the pore-size distribution of the generated structures (see Figure 10). However, the algorithm was slightly adjusted here, to avoid the determination of pores smaller than $1.25r$, where r denotes the particles radius. It can be inferred from the plot that the average pore size was larger in the case of the unconstrained inputs. This showed that by using a constrained inversion of the ANN, desired size- and pore-structure of the silica aerogels can be obtained. Since input parameters such as the density (concentration) and particle-size can be tailored during the synthesis of the aerogels, such a ML-based tool can be very useful in reverse engineering of the synthesis parameters. Furthermore, the proposed algorithm can be optimised by reducing the number of constraints in the inversion algorithm. The current ANN model requires the constraint of all model inputs in order to obtain accurate desired results. However, these constraints can be reduced significantly to only account for physical parameters which can be controlled in the laboratory environment during aerogel synthesis, as described above. Moreover, the inversion technique could also be used in the micro-structure reconstruction by training the network for other mechanical or morphological material properties, other than fractal dimension, e. g., yield stress or thermal properties.

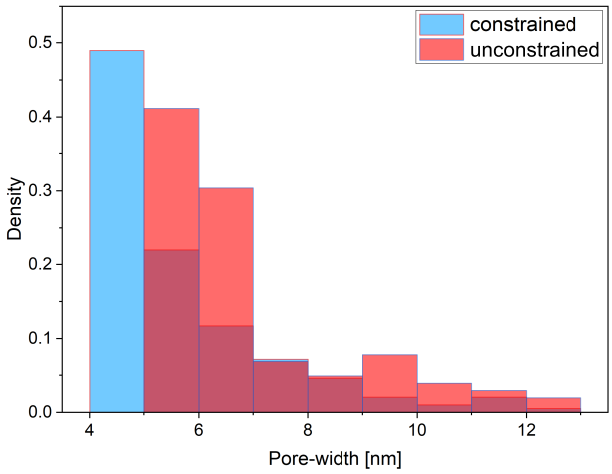


Fig. 10 Pore-size distribution of the generated unconstrained and constrained DLCA structures in Fig. 9.

4 Conclusions

In this paper, we proposed an ANN predicting the fractal dimension of silica aerogels and vice versa reconstructing their micro-

structures from a given fractal dimension. These micro-structures are obtained by means of the DLCA model with the parameters delivered by the ANN. In the forward-propagation model, an ANN was successfully trained using DLCA networks in order to the predict the fractal dimension for any given set of input parameters, resulting in an accuracy of $R^2 = 0.973$. The ANN was then inverted using the back-propagation method to perform a one to many mapping and successfully predict the necessary input DLCA model parameters for a desired fractal dimension. The network successfully calculated the DLCA model parameters from a desired fractal dimension within an error of 2%. The problem of non-uniqueness during this inversion was solved by including model constraints in the inversion algorithm. The developed ML-based tool is shown to be useful in tailoring the pore-structure of silica aerogels for a given target fractal dimension.

Conflicts of interest

There are no conflicts to declare.

References

1 F. E. Bock, R. C. Aydin, C. J. Cyron, N. Huber, S. R. Kalidindi and B. Klusemann, *Frontiers in Materials*, 2019, **6**, 110.

2 G. P. P. Pun, R. Batra, R. Ramprasad and Y. Mishin, *Nature Communications*, 2019, **10**, 2339.

3 M. Ceriotti, *The Journal of Chemical Physics*, 2019, **150**, 150901.

4 D. Reimann, K. Nidadavolu, H. ul Hassan, N. Vajragupta, T. Glasmachers, P. Junker and A. Hartmaier, *Frontiers in Materials*, 2019, **6**, year.

5 A. Koeppe, F. Bamer and B. Markert, *Acta Mechanica*, 2019, **230**, 3279–3293.

6 A. Ghaderi, V. Morovati and R. Dargazany, *Polymers*, 2020, **12**, 2628.

7 G. C. Y. Peng, M. Alber, A. Buganza Tepole, W. R. Cannon, S. De, S. Dura-Bernal, K. Garikipati, G. Karniadakis, W. W. Lytton, P. Perdikaris, L. Petzold and E. Kuhl, *Archives of Computational Methods in Engineering*, 2020.

8 R. Liu, A. Kumar, Z. Chen, A. Agrawal, V. Sundararaghavan and A. Choudhary, *Scientific Reports*, 2015, **5**, 1–12.

9 B. Kim, S. Lee and J. Kim, *Science Advances*, 2020, **6**, eaax9324.

10 A. Nakano, L. Bi, R. K. Kalia and P. Vashishta, *Physical Review Letters*, 1993, **71**, 85–88.

11 J. S. Rivas Murillo, M. E. Bachlechner, F. A. Campo and E. J. Barbero, *Journal of Non-Crystalline Solids*, 2010, **356**, 1325–1331.

12 L. D. Gelb, *The Journal of Physical Chemistry C*, 2007, **111**, 15792–15802.

13 A. Hasmy, E. Anglaret, M. Foret, J. Pelous and R. Jullien, *Physical Review B*, 1994, **50**, 6006–6016.

14 R. Abdusalamov, C. Scherdel, M. Itskov, B. Milow, G. Reichenauer and A. Rege, *The Journal of Physical Chemistry B*, 2021, **125**, 1944–1950.

15 *Neural Networks and Simulation Methods*, Taylor & Francis, 1993.

16 R. J. Williams, 8th Annual Conf. Cognitive Sci. Soc., 1986.

17 J. Kindermann and A. Linden, *Parallel computing*, 1990, **14**, 277–286.

18 C. Bourcier, W. Dridi, L. Chomat, E. Laucoin, B. Bary and E. Adam, SNA+ MC 2013-Joint International Conference on Supercomputing in Nuclear Applications+ Monte Carlo, 2014, p. 02107.