Supporting Information

Machine Learning-based Prediction and Inverse Design of 2D Metamaterial Structures with Tunable Deformation-Dependent Poisson's

Jie Tian[1], Keke Tang[1*], Xianyan Chen[2], and Xianqiao Wang[3*]

[1]School of Aerospace Engineering and Applied Mechanics, Tongji University, Shanghai 200092, China

[2]Department of Statistics, University of Georgia, Athens, GA 30602, USA

[3]School of ECAM, University of Georgia, Athens, GA 30602, USA

1． Molecular Dynamic Simulation

All the CG molecular dynamics simulations are performed in the commercial molecular dynamics package LAMMPS. In order to systemically and statistically calibrate the in-plane mechanical response of the CG models, the simulation box with 5×5 unit is $400\ \mathrm{nm} \times 400\ \mathrm{nm}$ in dimension, and periodic boundary conditions in the $x - y$ plane are imposed. The system is first energy minimized and then equilibrated in the NVT ensemble at a temperature of 1 K for 5,000 timesteps. After the equilibration, the sample is then compressed uniaxially along the x-direction using a strain-controlled loading method, in which the deformation is added every 10 timesteps by deforming the simulation box, and the equivalent strain rate is around 0.00002. The visualization and post-processing of simulation results are carried out via the OVITO and Python packages.

2. Machine Learning Algorithms

2.1 Data analysis

We employ both multidimensional scaling (MDS) and t-distributed stochastic neighbor embedding (T-SNE) to interpret the distribution of data in our database, in which both methods visualize high-dimensional data by giving each datapoint in a location in a two-dimensional map. Figure S1 shows the results of space visualization of our dataset. In both plots, there is no obvious data cluster observed, indicating that our dataset for porosity 50%, 60%, and 70% are well-distributed in the design space and suitable for a single machine learning algorithm.

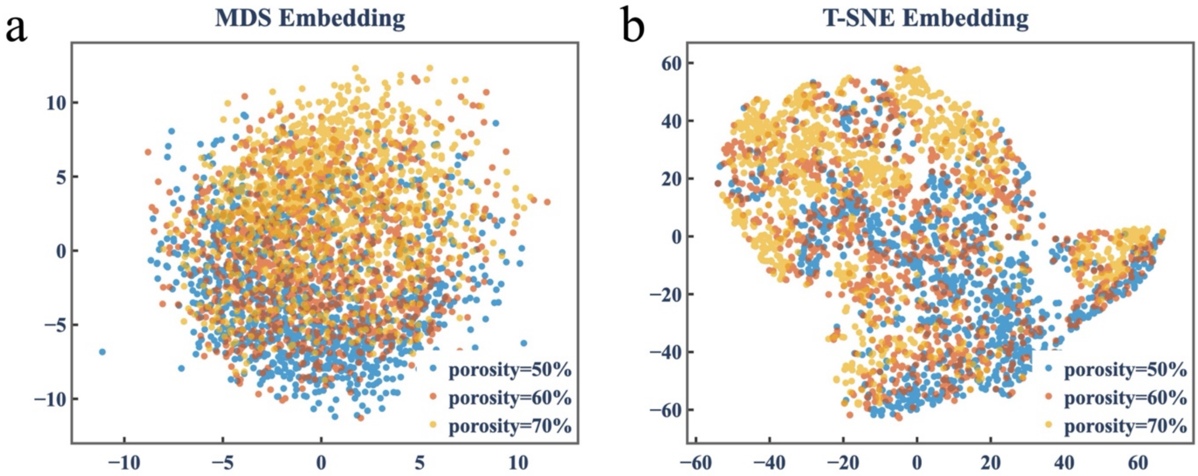

Figure S1: MDS and T-SNE methods are employed to analyze the distribution of data. Both plots show that no obvious data cluster is observed, indicating that our datasets for porosity 50%, 60%, and 70% are well-distributed in the design space and suitable for a single machine learning algorithm. (a) MDS plot; (b) T-SNE plot.

2.2 CNN training

Mean squared error (MSE) loss is helpful when calculating the gradient, and Infinity Norm L^{∞} loss is accurate for evaluating the distances between the predicted and ground-truth values applied in both training and test datasets

$$MSE = \sum_{i=1}^{n} \frac{(y_c^i - y_r^i)^2}{n} \tag{1}$$

$$L^{\infty} = \max\left(\left|y_c^i - y_r^i\right|\right) \tag{2}$$

where y_c^i represents the Poisson's ratio value predicted from CNN; y_r^i represents the ground-truth value from MD simulation; i is the feature number; n is the total feature number, which equals 100 here. The Adam optimization algorithm is adopted here to train the CNN algorithm. The CNN model is incapable of generating the 2D metamaterial structures, however it can predict the deformation-dependent Poisson's ratios of corresponding 2D metamaterial structures. The PyTorch is employed to train and test the CNN model. It takes approximately 48 hours to train the model on the RTX 3080Ti GPU.

2.3 Cycle-GAN

The architecture of Cycle-GAN can be illustrated with the Encoder and Decoder connected in red lines; take the model figure and the curve figure as data A and B from two different datasets, if A and B can be paired, then Encoder can transform A to B; in what follows, Decoder can transform B to A; this makes up the adversarial attack for training. However, it shall be noted that Cycle-GAN is stipulated for image processing because a certain apparent similarity between two pairs is necessitated. Several modifications are applied to overcome this limitation.

In training Procedure, we construct the loss function for Decoder as follows

$$Loss_{Encoder} = BCE\ Loss + MSE\ Loss + 0.1 * SmoothL1Loss \tag{3}$$

$Binary\ cross - entropy\ (BCE)$ loss is calculated from Discriminator; $MSE\ loss$ is determined by the error between input curve and the curve from Encoder; $SmoothL1Loss$ is a pixel-wise error between the actual image model and the model from Decoder. After the repetitive process of trial and error, we determine the coefficient of $SmoothL1Loss$ 0.1 from this supervised learning setup. Adam optimization algorithm is also adopted here. The PyTorch is employed to train and test the Cycle-GAN model. It takes around 60 hours to train the model on the RTX 3080Ti GPU.