Supplementary Material: 3D Microstructure Reconstruction of Aerogels via Conditional GANs

Prakul Pandit*
Institute for Frontier Materials on Earth and in Space
German Aerospace Center (DLR), Cologne, Germany
prakul.pandit@dlr.de

Sugan Kanagasenthinathan*
Institute for Frontier Materials on Earth and in Space
German Aerospace Center (DLR), Cologne, Germany
sugan.kanagasenthinathan@dlr.de

Ameya Rege
Department of Mechanics of Solids, Surfaces & Systems, University of Twente, The Netherlands
Institute for Frontier Materials on Earth and in Space, German Aerospace Center (DLR), Cologne, Germany
ameya.rege@utwente.nl

1 Supplementary Information

1.1 The DLCA Algorithm

The Diffusion limited cluster cluster aggregation (DLCA) algorithm simulates the formation of particle clusters through a random walk in spherical coordinates. Particles diffuse with a step size s at each iteration, with their movement governed by changes in the radial (r), polar (θ), and azimuthal (ϕ) components. The critical distance for clustering, ϵ, is defined as $\epsilon = 2.15 \times r$, where r is the characteristic particle radius. If two particles come within this critical distance, they are considered aggregated, and their positions become fixed relative to one another. It is important to highlight that the simulation is performed using periodic boundary conditions to mimic the material's bulk properties.

Once particles aggregate into clusters, these clusters themselves begin to move as single entities, following the same diffusion mechanism. When two clusters approach each other within the critical distance ϵ, they merge to form a larger cluster. This hierarchical aggregation process continues, with progressively larger clusters forming as the simulation evolves. The random walk in spherical coordinates ensures isotropic diffusion, accurately modeling the stochastic motion and clustering behavior of particles.

1.2 The CNN Model and Architecture

Table 1: Optimal hyperparameters for the CNN.

Hyperparameter	**Value**
Batch Size	128
Learning Rate	0.0001
Adam Optimizer Beta1	0.5
Step LR Scheduler Step	5

Table 1 presents the final hyperparameters of the trained CNN model, while Table 2 outlines the CNN architecture. Although three separate CNNs with this architecture and hyperparameters were trained independently for the 3D microstructural features, they could be consolidated into a single model. However, for this study, the CNNs were trained individually to maintain modularity and facilitate reconstruction based on a single feature due to ease of computation and training.

*Equal contribution.

Table 2: CNN architecture for predicting the structural properties of aerogels. H and W denote the height and width of the 2D input image, respectively.

Layer	**Details**	**Output Shape**
Input	–	$1 \times H \times W$
Conv Layer 1	3×3 kernels, 32 channels, stride=1, padding=1	$32 \times H \times W$
Activation 1	ReLU	$32 \times H \times W$
Max Pool 1	2×2 pooling	$32 \times \frac{H}{2} \times \frac{W}{2}$
Conv Layer 2	3×3 kernels, 64 channels, stride=1, padding=1	$64 \times \frac{H}{2} \times \frac{W}{2}$
Activation 2	ReLU	$64 \times \frac{H}{2} \times \frac{W}{2}$
Max Pool 2	2×2 pooling	$64 \times \frac{H}{4} \times \frac{W}{4}$
Conv Layer 3	3×3 kernels, 128 channels, stride=1, padding=1	$128 \times \frac{H}{4} \times \frac{W}{4}$
Activation 3	ReLU	$128 \times \frac{H}{4} \times \frac{W}{4}$
Max Pool 3	2×2 pooling	$128 \times \frac{H}{8} \times \frac{W}{8}$
Flatten	Reshape to vector	$128 \times \frac{H}{8} \times \frac{W}{8}$
Linear Layer 1	256 units	256
Activation 4	ReLU	256
Linear Layer 2	1 unit	1

1.3 cGAN Model and Architecture

Table 3 presents the seleted hyperparameters for the trained cWGAN after hyperparameter optimisation to achieve the best results. Moreover, the generator and discriminator architecture are presented in Table 4 and Table 5 respectively.

Table 3: Hyperparameters used for training the cGAN.

Hyperparameter	**Value**
Generator Learning Rate	0.001
Discriminator Learning Rate	0.0005
β_1 (Adam optimizer)	0.5
Batch Size	64
Gradient Penalty (λ)	15

Table 4: Generator architecture for the cGAN with a maximum convolutional width of 128. The generator applies BatchNorm3D and ReLU activation after each ConvTranspose3D layer, except the final output layer, which uses a sigmoid activation.

Layer	**Details**	**Padding**
Input	z, *labels*	–
Label Transformation	Embedding = Linear($label\_dim = 11$, $latent\_dim = 200$)	–
Concatenation	Concatenate z and Embedding	–
Reshape	z reshaped to ($batch\_size$, $latent\_dim + latent\_dim$, 1, 1, 1)	–
ConvTranspose3D Layer 1	$4 \times 4 \times 4$, stride=1, 128 out channels	0
ConvTranspose3D Layer 2	$4 \times 4 \times 4$, stride=2, 64 out channels	1
ConvTranspose3D Layer 3	$4 \times 4 \times 4$, stride=2, 32 out channels	1
ConvTranspose3D Layer 4	$4 \times 4 \times 4$, stride=2, 16 out channels	1
ConvTranspose3D Layer 5	$4 \times 4 \times 4$, stride=2, 8 out channels	1
ConvTranspose3D Layer 6	$4 \times 4 \times 4$, stride=2, $output\_dim$ out channels	1
Sigmoid Activation	Applied to final output layer	–

Table 5: Discriminator architecture for the cGAN with a maximum convolutional width of 128. The discriminator applies InstanceNorm3D and LeakyReLU activation after each Conv3D layer, except the final layer.

Layer	**Details**	**Padding**
Input	$x \in Binary\_Image^{128\times128\times128}$, $labels \in one\_hot\_labels^{11}$	–
Label Transformation	Embedding = Linear($label\_dim = 11$, $1 \times 128 \times 128 \times 128$)	–
Reshape	Embedding reshaped to ($batch\_size$, 1, 128, 128, 128)	–
Element-wise Multiplication	x multiplied with Embedding	–
Conv3D Layer 1	$4 \times 4 \times 4$, stride=2, 8 out channels	1
Conv3D Layer 2	$4 \times 4 \times 4$, stride=2, 16 out channels	1
Conv3D Layer 3	$4 \times 4 \times 4$, stride=2, 32 out channels	1
Conv3D Layer 4	$4 \times 4 \times 4$, stride=2, 64 out channels	1
Conv3D Layer 5	$4 \times 4 \times 4$, stride=2, 128 out channels	1
Conv3D Layer 6	$4 \times 4 \times 4$, stride=1, 1 out channel	0

1.4 Qualitative Analysis of the Reconstructed Microstructures

The density of the 3D images generated by the cGAN is calculated by thresholding the generator output into a 3D binary image and dividing the number of '1's (particles) by the total number of elements ($128 \times 128 \times 128$):

$$\text{Relative Density} = \frac{\text{Number of ones in the 3D array}}{\text{Total number of elements}} \tag{1}$$

The calculated densities and corresponding mean absolute percentage error (MAPE) are shown in Table 6.

Table 6: Comparison of Conditional and Calculated Densities with MAPE.

Generator Output	**Conditional Density**	**Calculated Density**	**MAPE (%)**
Figure 3b	0.03	0.03336	11.20
Figure 3d	0.06	0.06263	4.38
Figure 3f	0.09	0.09067	0.74

The results indicate that the generated outputs closely match the target densities, with low MAPE values. The highest error (11.2%) occurs at 0.03 density due to the generator's difficulty limiting particle generation for low-density structures, while higher densities achieve significantly better accuracy (0.74% at 0.09). This demonstrates that the cWGAN effectively generates density-specific 3D structures with high fidelity.