Journal Name

ARTICLE TYPE

Cite this: DOI: 00.0000/xxxxxxxxxx

Supporting Information: Deep-learning enabled photonic nanostructure discovery in arbitrarily-large shape sets via linked latent space representation learning†

Sudhanshu Singh,[*a] Rahul Kumar,[*a] Soumyashree S. Panda[b] and Ravi S. Hegde[a]

Received Date
Accepted Date

DOI: 00.0000/xxxxxxxxxx

S-1 Shape set details

The spectral response of devices is heavily influenced by their geometric design. Therefore, the availability of a diverse range of geometric shapes is paramount for selecting the desired spectral response. To address this need, we have compiled a comprehensive dataset comprising 200,000 unique shapes. These shapes are categorized into 21 distinct classes and are represented as binary matrices with dimensions of 64 by 64 pixels. Each pixel in these matrices indicates the presence (1) or absence (0) of silicon material within the corresponding spatial location. In this dataset, the classes of all geometric data include double ellipses, rings, ellipses, triangles, plus, alphabetic H, L, C, rod, Perlin noise shapes, polygon, and void shapes of ellipse, double ellipse, ring, triangular, half moon, plus, 2-fold, polygon and perlin noise shapes. The Stanford Stratified Structure Simulator (S^4) was used for optical evolution. S^4 employs the Rigorous Coupled Wave Analysis (RCWA) to solve Maxwell's equation in layered periodic structures. The table below outlines the various shape classes included in the dataset, along with their respective generation procedures and the number of shapes belonging to each class. Notably, we have omitted the generation procedure for void shapes as it is the inverse of solid shapes and is thus self-evident. We generate a dataset of unit cell geometries represented as a (64×64) binary distribution of pixels. To do this, we generate 3 random points within the region $x = 0$ to 1 and $y = 0$ to 1 , assigning each point a z value between 0 and 1. We then use Gaussian interpolation to create a smooth surface passing through these points. This surface is converted into a 2D image by classifying pixels based on whether their z-values exceed 0.2. Pixels with z-values greater than 0.2 are classified as black pixels, while those with z-values less than or equal to 0.2 are classified as white pixels. This study focuses on observing the 0th order transmission and reflection of visible light wavelengths for s and p polarized incidents, resulting in four spectrum combinations: Transmission (s-pol), Transmission (p-pol), Reflection (s-pol), and Reflection (p-pol). The visible spectrum is discretized between 400 nm and 700 nm with 5 nm spacing, yielding 60 discrete wavelength samples for which the values of Transmission (s-pol), Transmission (p-pol), Reflection (s-pol), and Reflection (p-pol) are recorded. The investigation is conducted specifically on silicon material on a SiO_2 substrate. Therefore, with crystalline silicon as the material for a given geometry, four optical responses are obtained at each of the 60 wavelength samples.

S-2 Model hyperparameter optimization and CNN training data

The table presents six different model architectures that were trained for a linked variational autoencoder along with their corresponding total loss values. Every architecture represents a different autoencoder setup, which is essential to capture the underlying data distribution. Interestingly, every model has a distinct latent dimension; Model 6 and Model 3 have a same latent space dimension of 8. The combined losses for Models 6 and 3 are comparable to those of other models. The details of all six model's architecture are shown in **Figure S2**. Based on our analysis, we find that model 6 and 3 perform comparably well in terms of latent dimensionality, particularly when we take into account their comparable losses. But when we look more closely, we see that Model 6 has a slightly smaller loss than Model 3. As such, we choose to move forward with model 6 for further analysis.

The CNN uses cascaded layers to extract features from input patterns, whereas the fully connected network converts the CNN output to 8-dimensional μ and σ vectors. Each convolutional layer of the shape encoder includes rectified linear unit (ReLU) activations and L1 and L2 regularizers. Batch normalization is

[*a] *Department of Physics, Indian Institute of Technology, Gandhinagar, 382355*
[b] *Department of Information and Communication Technology, Pandit Deendayal Energy University, Gandhinagar, 382007*
[a] *Departmrnt of Electrical Engineering, IIT Gandhinagar, India, 382355 e-mail: hegder@iitgn.ac.in*
[*a] *The authors contributed equally and are considered as first author.*

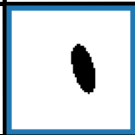
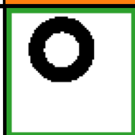
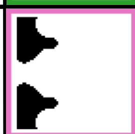
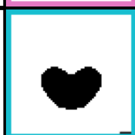
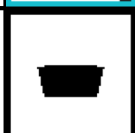
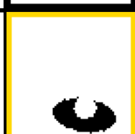
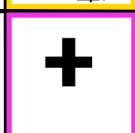
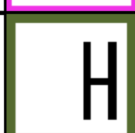
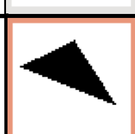
Class	Shape	Description for generation	Data size
Ellipse		Generates ellipses with random center coordinates, major and minor radii, and rotation angles. Gaussian interpolation is employed to interpolate points for smooth transitions. The output is a binary mask where pixels inside the ellipse are white and pixels outside are black.	5000
Double ellipse		Generates multiple ellipses per image with varied center coordinates, major and minor axes, and rotation angles. Gaussian interpolation is applied to random points to create smooth transitions, resulting in binary masks representing the combined ellipses.	5000
Ring		Generates rings with varying inner and outer radii. It selects random points within each ring and assigns random z values to them. Gaussian interpolation is then applied to these points, creating smooth transitions. Finally, a binary mask is generated where True represents points within the ring.	5000
Perlin noise		Generation involves creating a symmetric pattern by applying Perlin noise with random octaves, scale factor, and seed. Optional horizontal and vertical symmetry is applied with specified probabilities, resulting in a binary image with a varied distribution of black and white regions.	10000
2-fold		Generates symmetrical patterns by randomly selecting two points within a specified range for one half and mirrors them across the y-axis. Gaussian interpolation is performed on these points to create a smooth surface. A binary mask is then applied based on a threshold, creating a distinct pattern.	15000
Polygon		Generates symmetrical polygon shapes by randomly selecting vertices for one half of the polygon within a specified range. mirroring these vertices to achieve symmetry. Gaussian interpolation is applied to these vertices to create a smooth surface, and a binary mask is generated based on a threshold.	30000
Half-moon		Class ellipse is defined to generate half-moon shaped images within a specified boundary. It draws an ellipse with a cutout to form the half-moon shape, allowing random variations in size, aspect ration, rotational angle, and position within the image boundary.	10000
Plus		Generates plus signs with varying center coordinates and arm lengths. It then selects random points for interpolation within the plus sign area, applies Gaussian interpolation to these points, and creates a binary mask representing the plus sign shape.	10000
H-shape		Class 'H_shape' to generate H-shaped images within a specified boundary. It draws three rectangles representing the horizontal and verticals bars of the H shape, ensuring no intersection with the image edges, and provides functions to generate multiple instances	10000
L-shape		Generates by defining an outer rectangle and a smaller inner box within it. Random dimensions and positions are chosen for both shapes, and Gaussian interpolation is applied to create smooth transitions. The resulting binary masks represent the combination of the outer rectangle and the inner box.	15000
Triangle		Generates by three randomly generated vertices. It then performs linear interpolation using griddata to fill the triangles with varying intensity levels. Finally, it creates binary masks based on a threshold value and saves the generated images as text files.	5000
C-shape		Generates multiple U-shaped images with random parameters such as height, width, and center coordinates. The U shapes are drawn within a defined box using rectangles, resulting in binary masks representing the U shapes.	5000

Figure S1 Unveiling Shapes: A Comprehensive Exploration of Classes, Generation Methods, and Data Sizes in the Library.

Model 1

Layers	Param.	Options
Shape enc:		
Conv2d	4×4, 16	512 Batch size
Batch norm		
Conv2d	5×5, 32	
Conv2d	8×8, 64	
Flatten		
Dense	64	500 epochs
Sampling	32	
Spectra enc:		
Conv2d		
Dropout	43×1, 16	
Conv2d		
Dropout	16×4, 64	
Flatten		
Dense	64	
Sampling	32	
Shape dec:		
Dense	64	
Reshape	(1,1,64)	
Conv2DTran	16×16, 32	
Conv2DTran	2×2, 32	
Dropout		
Conv2DTran	2×2, 16	
Conv2DTran	1×1, 1	
Spectra dec:		
Reshape	(1,1, 32)	
Batch norm		
conv2DTran	43×1, 64	
Dropout		
Conv2DTran	18×2, 32	
Conv2DTran	1×3, 16	
Dropout		
Conv2DTran	1×1, 1	
Total loss	132843	
Time	225min	

Model 2

Layers	Param.	Options
Shape enc:		
Conv2d	4×4, 16	512 Batch size
Batch norm		
Conv2d	5×5, 32	
Conv2d	8×8, 64	
Flatten		
Dense	32	500 epochs
Sampling	64	
Spectra enc:		
Conv2d	43×1, 16	
Dropout		
Conv2d	16×4, 64	
Dropout		
Flatten		
Dense	64	
Sampling	64	
Shape dec:		
Reshape	(1,1,64)	
Conv2DTran	16×16, 32	
Conv2DTran	2×2, 32	
Dropout		
Conv2DTran	2×2, 16	
Conv2DTran	1×1, 1	
Spectra dec:		
Dense	64	
Reshape	(1,1,64)	
Batch norm		
conv2DTran	43×1, 64	
Dropout		
Conv2DTran	18×2, 32	
Conv2DTran	1×3, 16	
Dropout		
Conv2DTran	1×1, 1	
Total loss	141139	
Time	225min	

Model 3

Layers	Param.	Options
Shape enc:		
Conv2d	8×8, 32	512 Batch size
Batch norm		
Conv2d	9×9, 64	
Flatten		
Dense	256	500 epochs
Dense	8	
Sampling	8	
Spectra enc:		
Conv2d	43×1, 16	
Dropout		
Conv2d	18×4, 32	
Dropout		
Flatten		
Dense	8	
Sampling	8	
Shape dec:		
Reshape	(1,1,8)	
Conv2DTran	8×8, 64	
Conv2DTran	16×16, 32	
Dropout		
Conv2DTran	6×6, 16	
Conv2DTran	1×1, 1	
Spectra dec:		
Dense	32	
Reshape	(1,1,32)	
Batch norm		
conv2DTran	16×1, 64	
Dropout		
Conv2DTran	16×1, 32	
Conv2DTran	30×4, 16	
Dropout		
Conv2DTran	1×1, 1	
Total loss	173091	
Time	339min	

Model 4

Layers	Param.	Options
Shape enc:		
Conv2d	5×5, 16	512 Batch size
Batch norm		
Conv2d	4×4, 64	
Conv2d	8×8, 32	
Flatten		
Dense	64	500 epochs
Sampling	16	
Spectra enc:		
Conv2d	33×1, 16	
Dropout		
Conv2d	20×4, 64	
Dropout		
Flatten		
Dense	128	
Dense	64	
Sampling	16	
Shape dec:		
Reshape	(1,1,64)	
Conv2DTran		
Conv2DTran	8×8, 32	
Dropout	2×2, 32	
Conv2DTran	2×2, 16	
Conv2DTran	2×2, 64	
Conv2DTran	1×1, 1	
Spectra dec:		
Reshape		
Batch norm	1×1, 64	
conv2DTran	43×1, 64	
Dropout		
Conv2DTran	18×2, 32	
Conv2DTran	1×3, 16	
Dropout		
Conv2DTran	1×1, 1	
Total loss	138156	
Time	328min	

Model 5

Layers	Param.	Options
Shape enc:		
Conv2d	7×7, 32	512 Batch size
Batch norm		
Conv2d	3×3, 64	
Conv2d	5×5, 32	
Flatten		
Dense	64	500 epochs
Sampling	24	
Spectra enc:		
Conv2d	40×1, 16	
Dropout		
Conv2d	20×4, 64	
Dropout		
Flatten		
Dense	256, 64	
Sampling	24	
Shape dec:		
Dense	64	
Reshape	(1,1,64)	
Conv2DTran	3×3, 42	
Dropout		
Conv2DTran	3×3, 32	
Conv2DTran	4×4, 16	
Conv2DTran	2×2, 64	
Conv2DTran	2×2, 64	
Conv2DTran	1×1, 1	
Spectra dec:		
Reshape	(1,1,64)	
Batch norm		
conv2DTran	40×1, 64	
Dropout		
Conv2DTran	20×2, 32	
Conv2DTran	1×3, 16	
Conv2DTran	2×1, 63	
Dropout		
Conv2DTran	1×1, 1	
Total loss	132090	
Time	270min	

Model 6

Layers	Param.	Options
Shape enc:		
Conv2d	2×2, 16	512 Batch size
Batch norm		
Conv2d	2×2, 32	
Conv2d	10×10, 64	
Flatten		
Dense	32	500 epochs
Sampling	8	
Spectra enc:		
Conv2d	53×3, 16	
Dropout		
Conv2d	5×2, 32	
Dropout		
Conv2d	4×1, 64	
Flatten		
Dense	32	
Sampling	8	
Shape dec:		
Reshape	(1,1,32)	
Conv2DTran	4×4, 64	
Conv2DTran	10×10, 64	
Dropout		
Conv2DTran	2×2, 32	
Conv2DTran	2×2, 16	
Conv2DTran	1×1, 1	
Spectra dec:		
Dense	32	
Reshape	(1,1,32)	
Batch norm		
conv2DTran	4×1, 64	
Dropout		
Conv2DTran	5×2, 32	
Conv2DTran	53×3, 16	
Dropout		
Conv2DTran	1×1, 1	
Total loss	148419	
Time	269min	

Figure S2 Table displaying six model architectures trained for linked variational autoencoder, along with their corresponding total loss values.

applied between the first two convolutional layers. Residual blocks, including batch normalization, dropout, and ReLU activation, are implemented between the next two convolutional layers. The structure of a shape decoder is designed in reverse order, with transposed convolutional neural networks (TCNNs) to reconstruct the latent space vector into binary images. The sigmoid activation function is implemented in the last layer of the shape decoder output, and similarly, for the spectrum encoder, in each convolutional layer, tanh activations and L1 and L2 regularizers are implemented. A dropout layer with a 20% dropout probability is employed between each convolutional layer.

The optical responses of a spectrum decoder are designed in reverse order, with transposed convolutional neural networks (TCNNs) to reconstruct the latent space vector to the original spectrum. The sigmoid activation function is implemented in the last layer of the spectrum decoder output.The geometry latent space vectors pass through the spectrum decoder, and the spectrum latent space vectors pass through the shape decoder; in this way, both the latent spaces are linked to each other. The linked coupled variational autoencoder was trained for 500 epochs; a specific learning rate value of 0.001 was chosen. After combined training of linked-coupled VAE, the model would produce two types of predictions: forward (from geometry to spectrum, and inverse from spectrum to geometry).

S-2.1 Model training

The setup consists of dual variational autoencoders (VAEs), with one for shape encoding at the top (orange) and one for spectrum encoding at the bottom (blue). These VAEs aim to represent heterogeneous data, where X_1 represents shape data and X_2 represents optical response data. The data from X_1 and X_2 can be mapped into their respective latent space representations, denoted as Z_1 and Z_2, via a linked coupling VAE. For coupled training, we train the encoders and decoders of the two VAEs, and we also incorporate cross-reconstruction training losses. This approach enables the model to cross-reconstruct one type of data domain from another. Specifically, during training, the model performs reconstruction of shape and spectrum data in their respective latent spaces. Additionally, it handles cross-reconstruction, where shape data is reconstructed from the latent space of spectrum data and spectrum data is reconstructed from the latent space of shape data. This cross-reconstruction capability ensures that each data domain can be effectively translated into the other, enhancing the model's versatility and robustness. The prediction performance of the surrogate model was evaluated using a dataset containing 200,000 samples, which was split into two parts: 140,000 samples for the training dataset and 60,000 samples for the testing dataset, for both shape and spectrum. The mean squared error (MSE) for the surrogate model from shape to spectrum prediction was computed to be 0.72e-2 and 0.75e-2 on training and testing dataset respectively.

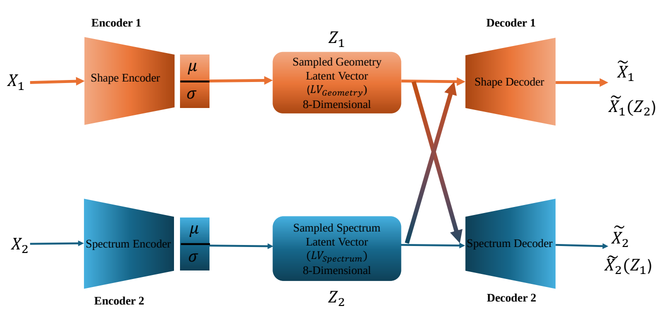


Figure S3 The heterogeneous data, represented by X_1 for shape and X_2 for the optical response, can be mapped into their respective latent space representations, Z_1 and Z_2, through a linked coupling VAE for compatibility, while the latent space representations can be reconstructed into the original images, denoted as $\tilde{X}_1$ and $\tilde{X}_2$ respectively. $\tilde{X}_1(Z_2)$ and $\tilde{X}_2(Z_1)$ are translated images.

S-2.2 Loss curve

The **Figure S4** illustrates the variation in individual losses during each of the training epochs of model 6. Monitoring six separate losses and a total loss is part of training model 6. A model's ability to accurately reconstruct input data for shape and spectrum is measured by its reconstruction losses. KL losses for spectrum and shape guarantee that the learned latent representations follow a predetermined distribution, which helps with regularization. The model's capacity to transfer data between diverse domains—for example, from shape to spectrum and vice versa—is evaluated by cross-reconstruction losses. The total loss, which combines these elements, represents the overall optimization purpose. By minimizing these losses, model 6 learns to consistently reconstruct input data, regularize latent representations, and capture meaningful relationships between distinct modalities, all of which are critical for convergence and performance evaluation during training.

S-3 Latent space representation

In examining the shape latent space, we notice a cohesive clustering of various geometric entities. These include ellipses (Blue), double ellipses (Orange), Perlin noise shapes (Pink), 2-fold shapes (Cyan), plus shapes (Magenta), triangles (Salmon), half moons (Gold), and their corresponding cavities. These entities come together into a unified cluster with overlapping boundaries, indicating shared geometric traits **Figure S5A**.

However, distinct clusters form for rings (Green), L-shapes (Light Gray), and C-shapes (Lime), with multiple clusters intertwining with other shapes. Within the H-shape category (Dark Olive Green), two distinct subclusters emerge based on the arrangement of transverse lines. Similarly, the polygon shape category (Black) shows two distinct subclusters: one for symmetrical polygons and another for asymmetrical polygons. This differentiation underscores the unique features within subsets of the H and polygon shapes, with each subcluster exhibiting clear separation and diverse structural configurations. Additionally, a small subcluster of multiple concentric rings see (**Figure S5B**)is observed within the ring category, further highlighting its unique shape characteristics within the latent space. We also employ t-Distributed Stochastic Neighbor Embedding (t-SNE), another robust algorithm for nonlinear dimensionality reduction. t-SNE is particularly effective at preserving the local structure of data and

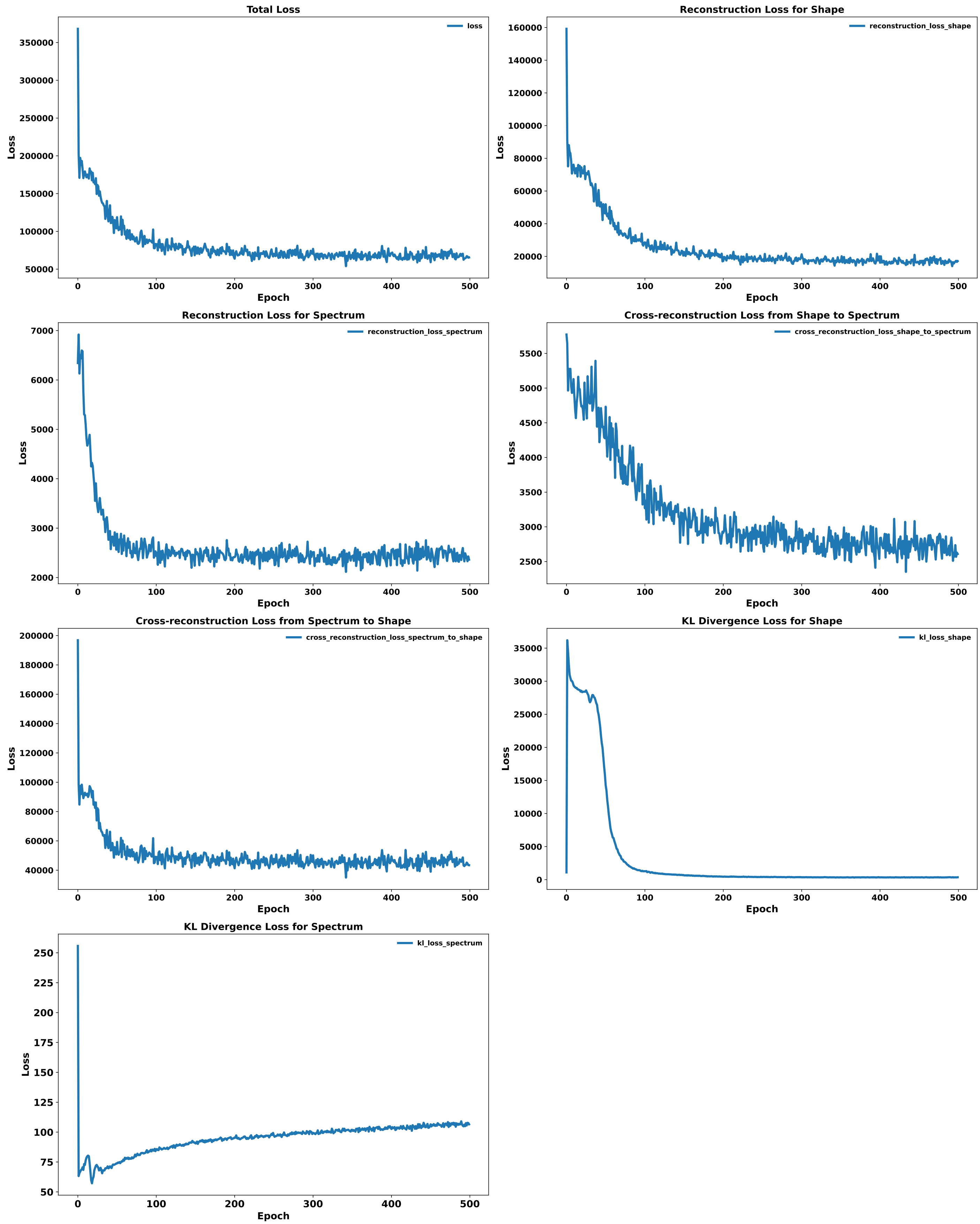


Figure S4 The figure shows how individual losses varied during the duration of model 6's training epochs. Every curve depicts a distinct loss measure that is monitored during the training phase, offering valuable insights about the model's convergence.

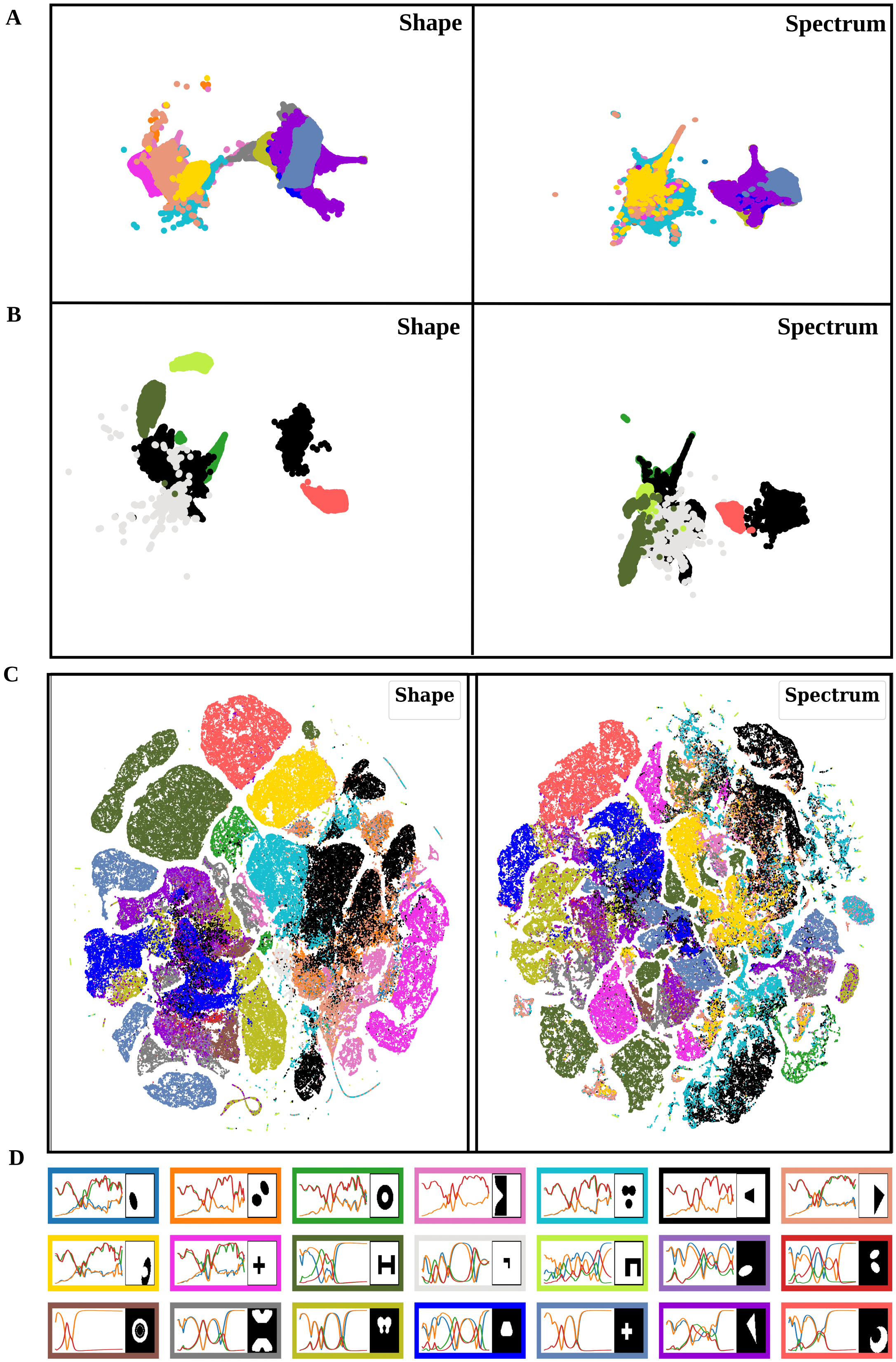


Figure S5 Visualization of the 8-dimensional shape and spectrum latent spaces projected into 2-dimensional space. A: Showcases some classes of shape and spectrum latent space of similar clusters, and B: Showcases some classes of shape and spectrum latent space of distinct subclusters. C: Visualization of shape and corresponding spectral latent space by utilising the t-SNE projection technique. D: Display points in shape and spectrum latent space are colour-coded using 21 classes.

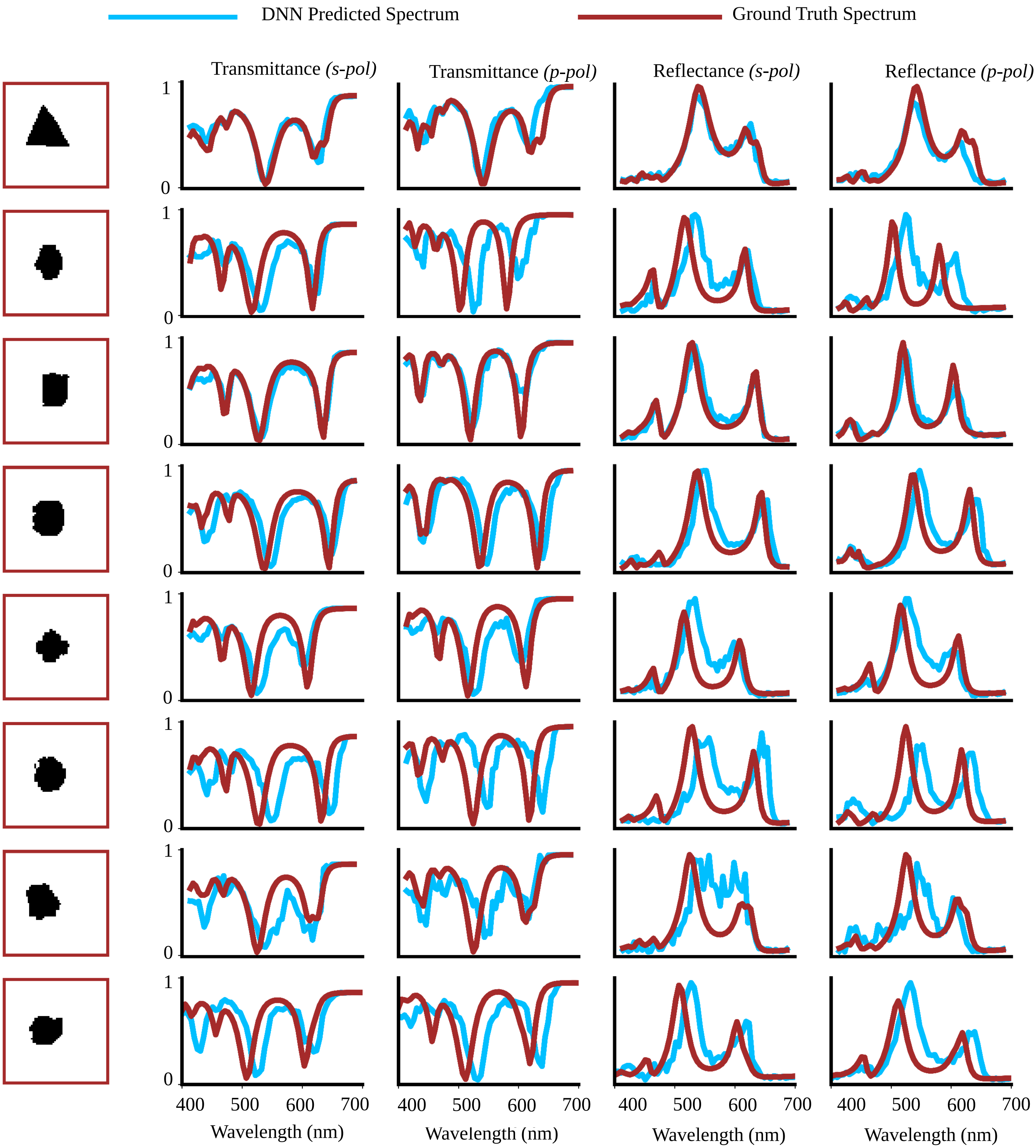


Figure S6 Similar points in the spectral latent space exhibit clustering, as evidenced by the generated images and their corresponding optical transmission and reflection spectra in both s-polarized and p-polarized light.

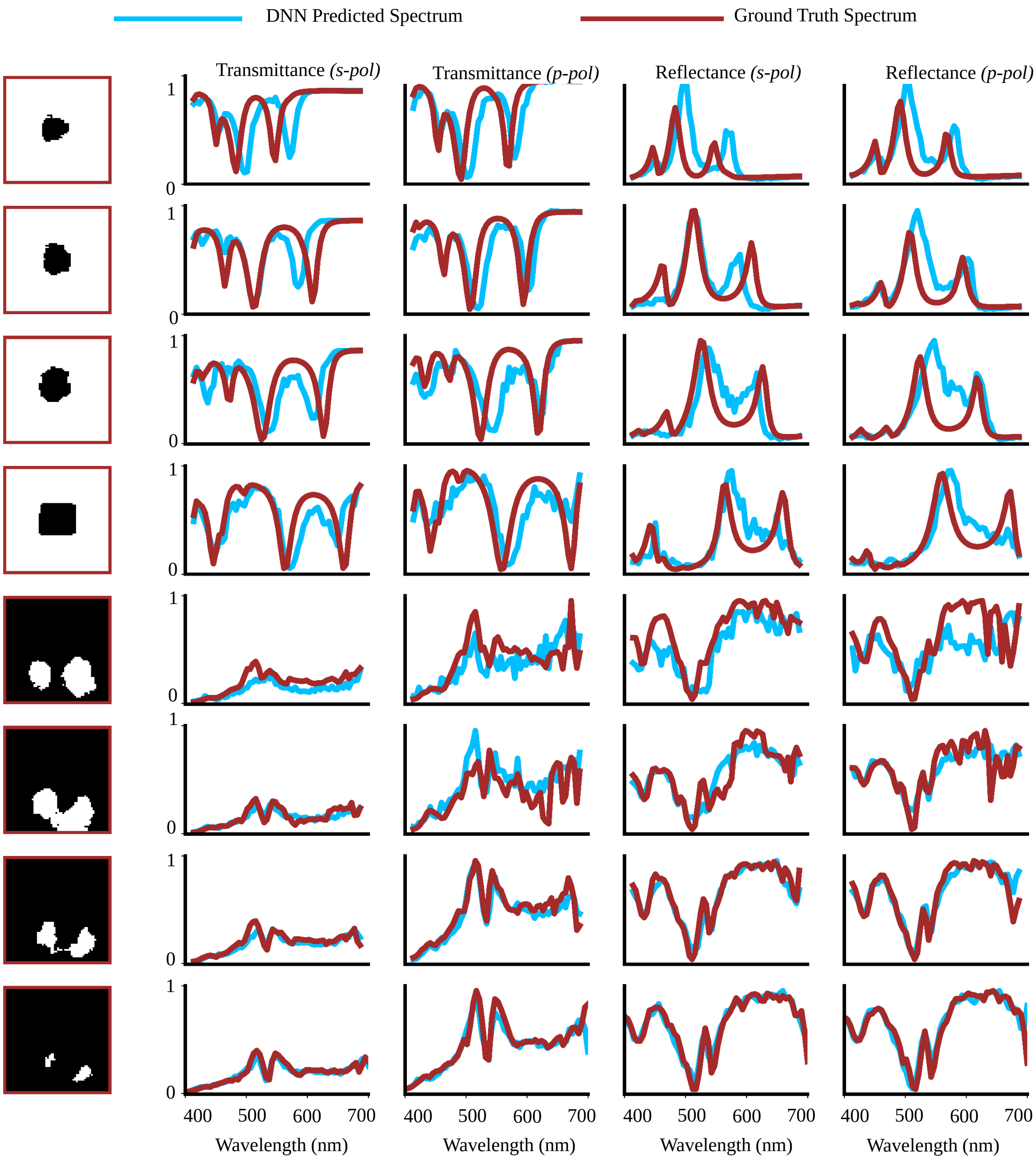


Figure S7 Smooth transitions from a plus shape to a half-moon cavity shape are observed in the spectrum latent space, as illustrated by the generated images and their corresponding transmission and reflection spectra in both s-polarized and p-polarized light. The top and bottom images depict the original spectrum.

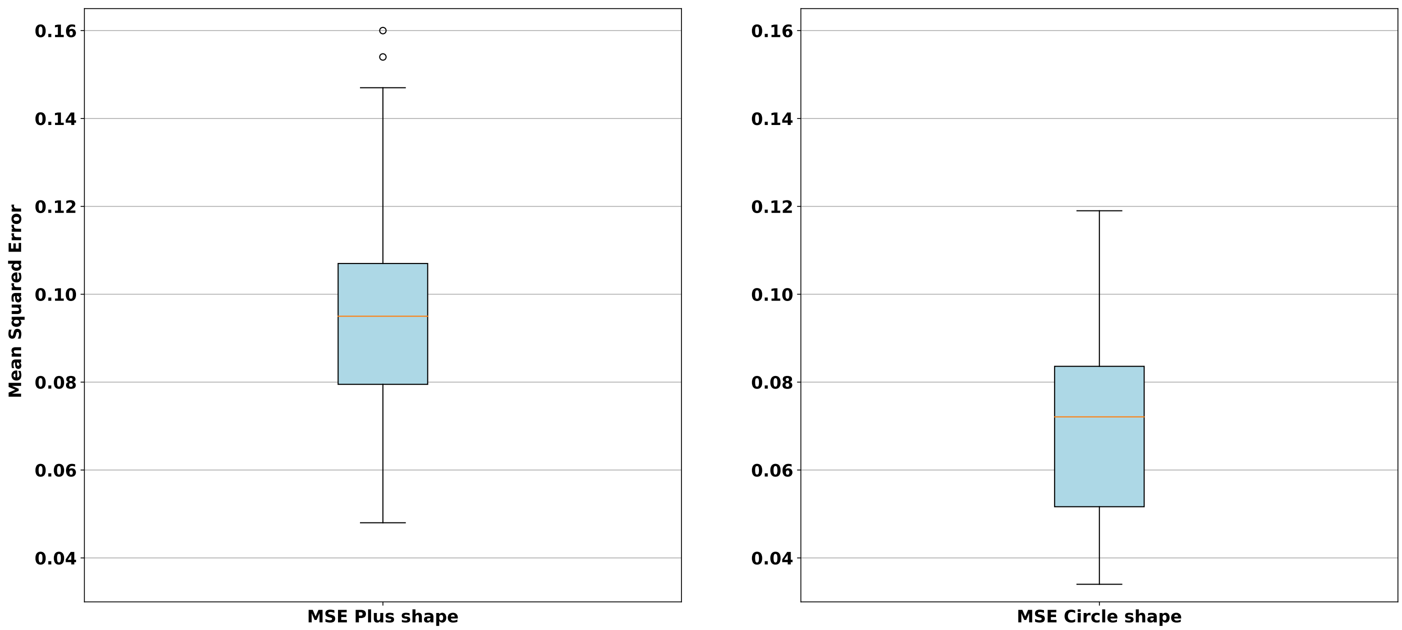


Figure S8 Mean Square Error (MSE) comparison between the target spectra for inverse design and the predicted spectra of the most noisy derived shapes, based on a sample size of 15.

maintaining the relationships between nearby points in the high-dimensional space. This makes t-SNE a powerful tool for visualising and clustering high-dimensional datasets. While t-SNE can reveal patterns in the global structure, its primary focus is on local neighbourhoods. **Figure S5C** shows a 2-D projection of an 8-Dimensional shape and spectrum latent space utilising t-SNE.

S-4 Interpolation at local and global levels for shape and spectrum

To assess the continuity of the spectral latent space at a local level, we initially choose a spectral response corresponding to a triangle shape. Subsequently, we pass this point through the spectrum encoder to acquire the corresponding data point in the spectral latent space. We then sample data points from a normal distribution centred around this chosen data point, with a slight standard deviation. These sampled data points are further processed through a spectrum decoder and a linked shape decoder. This process enables us to observe similar spectral responses and their corresponding shapes for transmittance and reflectance in s and p-polarized light, as illustrated in **Figure S6**. Given the one-to-many mapping, nearly similar spectral responses result in shapes such as triangles, pluses, rods, and ellipses.

For examining the continuity of the spectral latent space at the global level, we select the spectrum latent points corresponding to two distant shapes, a plus and a half-moon cavity. Employing linear interpolation between these points, we decode the resulting latent representations through a spectrum decoder and a linked shape decoder.

Figure S7 presents the reconstructed shapes and their corresponding predicted spectra alongside the original spectra (generated using the S^4 solver), revealing a smoother and more diverse transition from one shape to another. Notably, the top and bottom spectra represent the original spectra. Furthermore, due to the one-to-many relation, the spectral response of the plus and half-moon cavity shapes yields different shapes at the top and bottom.

S-5 Mean square error comparison for sensitivity between plus and circle Shapes

Assessment of sensitivity in derived shapes with the highest noise levels was conducted using Mean Square Error (MSE) calculations. Results indicate a greater deviation for the plus shape (median MSE: 0.095) compared to the circle shape (median MSE: 0.072) over 15 iterations, as depicted in **Figure S8**. Furthermore, the spread in MSE values is wider for the plus shape than for the circle shape. These findings suggest that the spectral response obtained from the derived circle shape is more resilient to fabrication tolerances.