Opto-Electronic Science

Supplementary information

2022, Vol. 1, No. 1

DOI: 10.29026/oes.2022.210012

Benchmarking deep learning-based models on nanophotonic inverse design problems

Taigao Ma[1], Mustafa Tobah[2], Haozhu Wang[3]* and L. Jay Guo[3]*

[1]Department of Physics, The University of Michigan, Ann Arbor, Michigan, 48109, USA; [2]Department of Materials Science and Engineering, The University of Michigan, Ann Arbor, Michigan, 48109, USA; [3]Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, Michigan, 48109, USA.

*Correspondence: HZ Wang, E-mail: hzwang@umich.edu; LJ Guo, E-mail: guo@umich.edu

This file includes:

Supplementary information for this paper is available at https://doi.org/10.29026/oes.2022.210012

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License.
To view a copy of this license, visit http://creativecommons.org/licenses/by/4.0/.

© The Author(s) 2022. Published by Institute of Optics and Electronics, Chinese Academy of Sciences.

Section 1: Network structures and training

This part gives extra information for constructing and training the neural networks used to deal with the template structures (color inverse design problem) and free-form structures (transmission spectrum inverse design problem).

Template structure: Multilayer Perceptrons (MLPs)

1) Tandem networks

To reduce confusions on notations, in both forward prediction and inverse design, we use X to denote colors, and Y to denote structures. A training data is a combination of one structure $y_i = (H_i, D_i, P_i, G_i)$, and one color $x_i = (x_i, y_i, Y_i)$. As shown in Fig. 1(a), the tandem networks are constructed by connecting outputs of the inverse neural networks (INNs) to the input of the forward neural networks (FNNs). FNNs are MLPs that have four-layer fully connected layers, with 64 neurons in each layer. FNNs take in the 4-dimensional structures and output the 3-dimensional color coordinates. INNs are also MLPs that have four-layer fully connected layers with 64 neurons in each layer, and they take in the 3-dimensional color coordinates while output the 4-dimensional structures.

Training tandem networks is a two-step process. The first step is to train the FNNs. During training, FNNs learn the mapping $Y \to X$ by minimizing the mean square error (MSE) loss of predicted colors with respect to the target color, which is shown in Eq. (1) in the main text. After training the FNNs, FNNs can be used as a surrogate model to predict the color for a given new structure input. The second step is to train the INNs. Now the parameters of FNNs are fixed, and the FNNs are connected to the output of INNs to supervise the learning of INNs. After INNs give the structure predictions, FNNs can immediately predict their colors. The INNs learn one branch of the mapping $X \to Y$ by minimizing the MSE loss of the predicted color given by this pre-trained FNNs, which is shown in Eq. (2) in the main text. Fig. S1(a) shows the loss curves during training of the FNNs.

In order to avoid over-fitting and pick up the best model, we use the technique of early stopping[1] (this technique is also used in all other models considered in this work) and validate the tandem networks' inverse design accuracy on the validation dataset. During validation, the one with the smallest MSE loss $L_{\mathrm{INNs}} = \frac{1}{N}\sum_i^N \left(\hat{x}_{i_{\mathrm{pred}}} - x_i\right)^2$ is picked up as the final model, where x_i is the target color, and $\hat{x}_{i_{\mathrm{pred}}}$ is the predicted color. The predicted color $\hat{x}_{i_{\mathrm{pred}}}$ related to the predicted structure is supposed to be calculated by electromagnetic (EM) simulation. To facilitate training, we use the pre-trained FNNs to calculate the color $\hat{x}_{i_{\mathrm{pred}}}$ related to the inverse predicted structure given by INNs. However, if we use the same FNNs during training and validation, the INNs will leverage the FNNs, and give predictions that are accurate only using forward model, but not accurate using RCWA simulations. To avoid this model bias, in the first step during training, two different FNNs will be trained separately. Later on, one of the FNNs is used to train the INNs, and another one is used to pick up the best model based on the validation dataset. We call these two FNNs training FNNs and validation FNNs, respectively. In Fig. S1(b), we show the loss curve during the training of tandem networks, as well as the validation loss using two different FNNs. Clearly, we can see the model strong bias when we use the same training FNNs for validation.

After sufficient training and validation, the INNs are used to inverse predict a possible structure for a given color target.

2) Variational Auto-Encoders:

As mentioned in the main text, we are using the conditional-VAEs (c-VAEs), which are introduced in ref.[2] (named as conditional generative model in this paper). In c-VAEs, the color targets X are treated as conditional input variables, the structures Y are treated as the output variables, and Z are treated as the latent variables. Usually, the latent variables Z are chosen to follow the normal distribution. The c-VAEs learn the inverse design by first drawing z from the prior distribution $p_\theta(z|x)$ on the condition of given colors input X, then giving predictions of structures y based on the distribution of $p_\theta(y|x,z)$. θ denote the network parameters. C-VAEs include three neural networks: the recognition networks $q_\phi(z|y,x)$, the generation networks $p_\theta(y|x,z)$, and the conditional prior networks $p_\theta(z|x)$. In terms of network structures, all three neural networks in the c-VAEs are MLPs that have 4-layer fully connected layers, with 64 neurons in each layer. The latent space is 3-dimensional normal distribution.

We find that connecting the pre-trained FNNs to c-VAE can improve the accuracy. Therefore, the overall network

structures for c-VAEs are shown in Fig. 2(b). We are using the same pre-trained training FNNs and validation FNNs from the tandem networks to supervise the training and validation process of c-VAEs. Therefore, we do not need to re-train the FNNs again.

During training, the recognition networks take in the structure information and conditional colors, and encode them together into the latent space z. The generation networks then decode the latent variables back into structure space based on the conditional colors. The conditional prior networks will learn a temporary mapping from the color space to structure space. These predicted structures from conditional prior networks will not be used for c-VAEs, but will be helpful during validation and inverse predicting, where the structure information is no longer available. The loss function is given in the main text in Eq. (3). During training, we find a larger $\alpha = 10$ in Eq. (3) works better in order to improve the accuracy.

The validation process is different from the training process. During validation, we need to inverse predict structures for given color inputs. However, because there are no structures as the inputs for the recognition networks (we need to inverse predict these structures), we cannot use the recognition networks to encode and find the distribution of the latent variables. Therefore, the first step during validation is using the conditional prior networks to provide the reconstructed structures for the recognition networks. Later on, we pass the reconstructed structures together with colors to the recognition networks, and encode into the latent variables. The generation networks will later decode the latent variables back to the final predicted structures based on the conditional color input. In order to pick up the best model, we validate the predicted structures' color accuracy by feeding the predicted structures into the validation FNNs, and choose the model with the smallest MSE loss $L_{\text{pred}} = \frac{1}{N}\sum_i^N \left(\hat{x}_{i_{\text{pred}}} - x_i\right)^2$, where the color $\hat{x}_{i_{\text{pred}}}$ is predicted by the validation FNNs. Figure S1(c) shows the loss curves during training and validation of c-VAEs.

After sufficient training and validation, the c-VAEs can be used to inverse predict the possible structure for a given color target. The prediction process is similar to generation in the validation process.

3) Generative Adversarial Networks

As mentioned in the main text, we are using conditional-GANs (c-GANs). The network construction is similar to ref.[3]. As shown in Fig. 1(c), the c-GANs also consist of two neural networks: the generator networks and the critic networks. The generator networks are MLPs that have four-layer fully connected layers, with 64 neurons in each layer, and it takes in both 3-dimensional color X and an extra 3-dimensional random noise Z, and outputs generated 4-dimensional structures Y. Because of the introduction of random variables Z, c-GANs are able to learn one-to-many mapping based on different conditions and give multiple predictions. The critic networks are also MLPs that have four-layer fully connected layers, with 64 neurons in each layer, but it takes in the 4-dimensional structures and the 3- dimensional color, and outputs a single value in the range of (0,1), which represents the possibility of being real.

The training procedure of c-GANs requires the co-training of the generator networks and the critic networks. In each iteration, the generator and the critic are updated consecutively. The loss function is given in the main text in Eq. (4). During training, the generator networks always generate fake structures to fool the critic, while the critic networks always try to distinguish real structures from fake structures based on the conditional color. Therefore, the generator networks will try to minimize this loss function, while the critic networks will try to maximize this loss function.

Again, we validate the accuracy of inverse prediction on the validation dataset to avoid over-fitting. To pick up the best model, we use the same validation FNNs, and calculate the MSE loss $L_{\text{pred}} = \frac{1}{N}\sum_i^N \left(\hat{x}_{i_{\text{pred}}} - x_i\right)^2$. This is done by connecting the output of the generator to the input of the validation FNNs. The color $\hat{x}_{\text{pred}}$ is predicted by FNNs, which corresponds to the inverse designed structures given by the generator networks based on a given color target x. The model with the smallest MSE loss is selected as the best c-GANs model. Fig. S1(d) shows the loss curves during the training of the c-GANs.

After sufficient training and validation, the generator is used to inverse predict a possible structure based on the conditional input of color target.

The training parameters for all three models as well as their training time are summarized in the Table S1. For Table 1 in the main text, we report the average performance when we train each model starting from five different random seeds.

Table S1 | The training hyperparameters as well as their estimated training time for the FNNs, the tandem networks, the c-VAEs, and c-GANs in the template structure inverse design.

	FNNs	Tandem networks	c-VAEs	c-GANs
Dimension of z	/	/	3	3
Learning rate	0.001	0.0005	0.001	Generator: 0.00005 Critic: 0.0001
Batch size	128	128	128	128
Optimizer	Adam	Adam	Adam	Adam
Training time	~1.4 h	~2 h	~3.5 h	~8.5 h

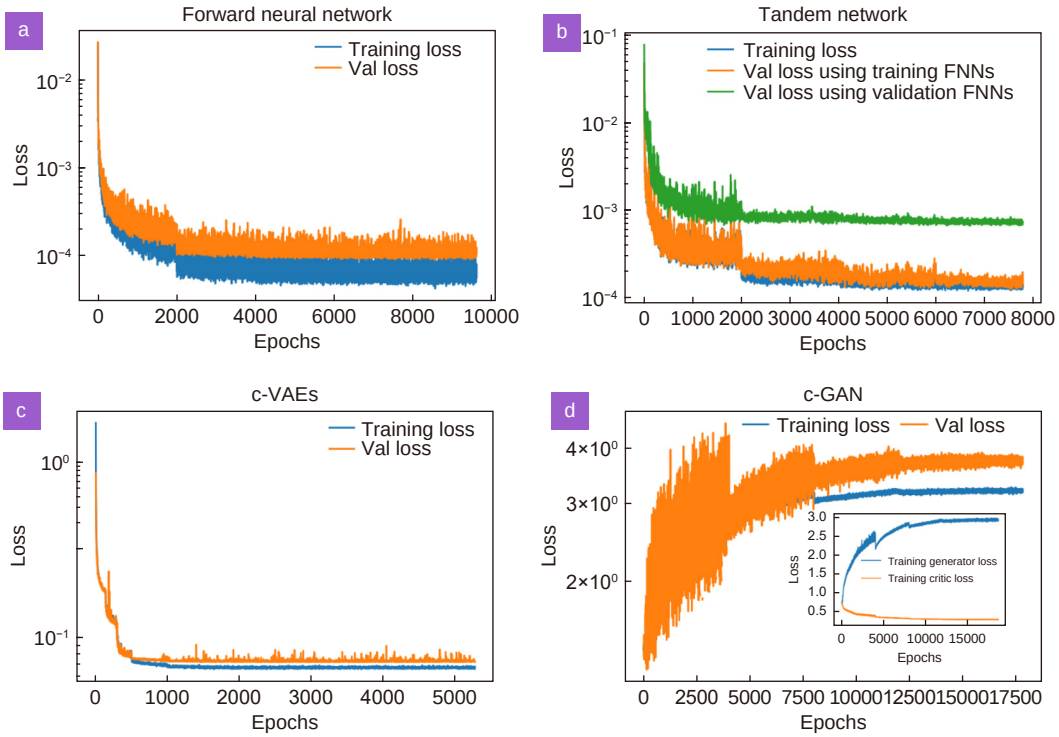

Fig. S1 | The training curve for each model in the template structure inverse design problem, where the blue and orange curve refer to training loss and validation loss. Notice that different models have different loss functions, so we cannot compare their relative values directly. (**a**) Training curve of the FNNs. (**b**) Training curve of the tandem networks. The extra green line is the validation loss by the validation FNNs. We can see that model bias is strong if we use the same FNNs model for both training and validation. (**c**) Training curve of c-VAEs. (**d**) Training curve of the c-GANs. The inset shows the generator loss and critic loss during training, where the generator loss is maximized, and the critic loss is minimized.

Free-form structure: Convolutional neural networks (CNNs)

For the transmission spectrum inverse design problem with free-form structures, instead of the MLPs, we use the CNNs to deal with the image information. As suggested in ref.[4], adding the loss of structural similarity index (SSIM) with a factor of 0.05 in the total loss function will help all models to learn the characteristics of images better. To increase the amount of data, we implement data augmentation by rotating each shape by 90/180/270, followed by exchanging the TE and TM responses. Because of the added SSIM loss, in order to guarantee the convergence of inverse design, data augmentation is only applied when training the forward neural network.

We build up our three neural network models using the same structures shown in Fig. 1, and use the same training, validation, and inverse predicting process described before. Because of the introduction of the convolution layer, the detailed constructions of each neural networks will be different. Figs. S2, S3, S4 show the CNNs used for tandem networks, c-VAEs, and c-GANs, respectively. We summarize the training parameters as well as their training time used in each model in Table S2. The training curves for each model are also shown in Fig. S5. For Table 2 in the main text, we report the average performance when we train each model starting from three different random seeds.

We provide all code and simulation data on GitHub[5], where detailed network structures and training procedure are

included. Usually, activation functions are also considered as a critical component of neural network architectures. We are using the most widely used activation functions that have been shown to work well on different inverse design problems, which can also be found on GitHub[5]. Most other commonly used activation functions that are proper for the problems can also lead to a similar conclusion.

Table S2 | The training hyperparameters as well as their estimated training time used for the FNNs, the tandem networks, the c-VAEs, and c-GANs in the free-form structure inverse design.

	FNNs	Tandem networks	c-VAE	c-GAN
Dimension of z	/	/	50	50
Kernel size	9	3	5	5
Learning rate	0.0005	0.0001	0.0001	Generator: 0.001 Critic: 0.0001
Batch size	1024	256	256	256
Optimizer	Adam	Adam	Adam	Adam
Training time	~1.2 days	~0.7 day	~1.4 days	~2.8 days

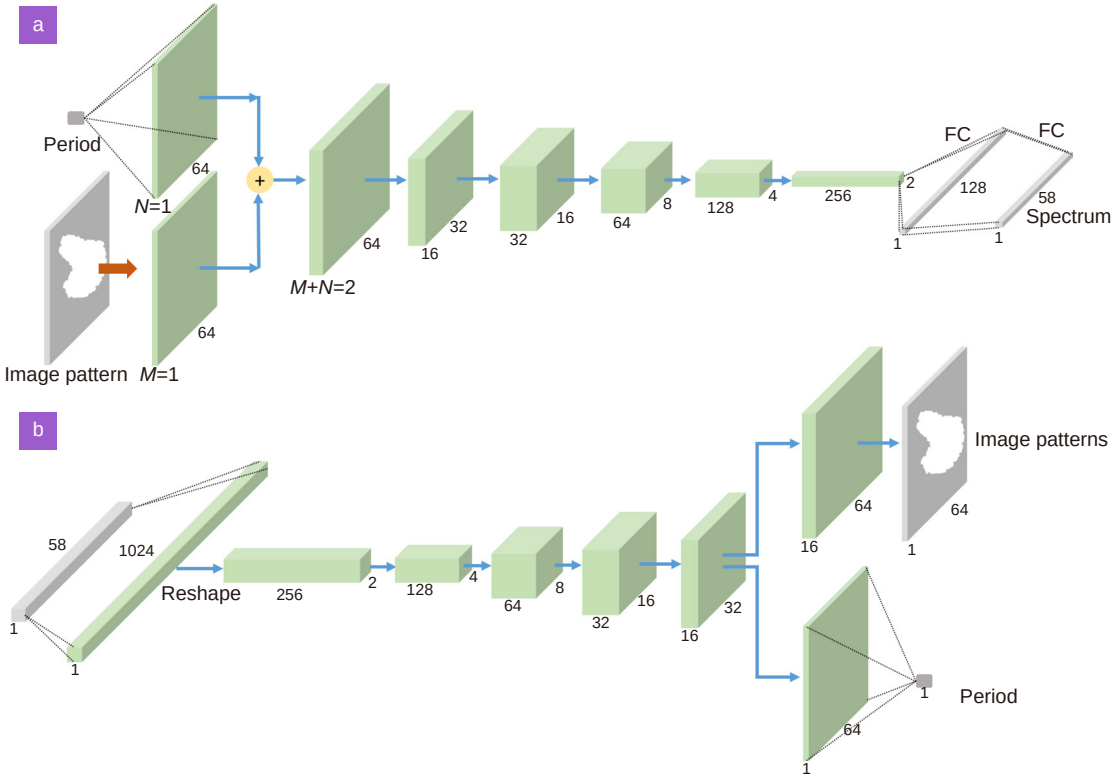

Fig. S2 | The CNNs of tandem networks for the free-form structures. (**a**) The structure of the FNNs. To combine period information into an image, we first map this one-by-one scaler of the period into a 4096-by-1 vector using a single FC, then reshape this vector into a 64×64 size image. This one-depth image transformed from the period is further concatenated with the 2D image pattern, which corresponds to the free-form structure after the convolution with kernel size k=1. We can change the depth of concatenation by changing the m and n. By default, the depth of period and image during concatenation is 1. Later on, 5-layer convolutional layers are used to recognize and extract image features into a 256-depth two-by-two matrix. After reshaping this matrix into a 1024-by-1 vector, two fully connected layers are connected to process features and mapping into the 58-dimensional spectrum. (**b**) The structure of the INNs, which map the spectrum information into the 2D image patterns and one-dimensional period. First, a fully connected layer transforms the 58-by-1 spectrum vector into a 1024-by-1 vector, which will be reshaped into a 256-depth 2-by2 matrix. After this, the transpose convolution is used to generate images layer by layer. After the four consecutive transpose convolution layers, the network will be divided into two parts. The upper branch takes another transpose convolution and generates 16-depth 64×64 matrix, and does another convolution to generate the predicted 2D image pattern. The lower branch also takes another transpose convolution action, but only generates the 1-depth 64×64 matrix. This matrix will be reshaped into a 4096-by-1 vector. A fully connected layer is used to transform this 4096-by-1 vector into the one-by-one predicted period. This is also the same structure of the conditional prior networks in c-VAEs. The training, validation, and generating process of tandem networks are similar in the template structures, and therefore not described again.

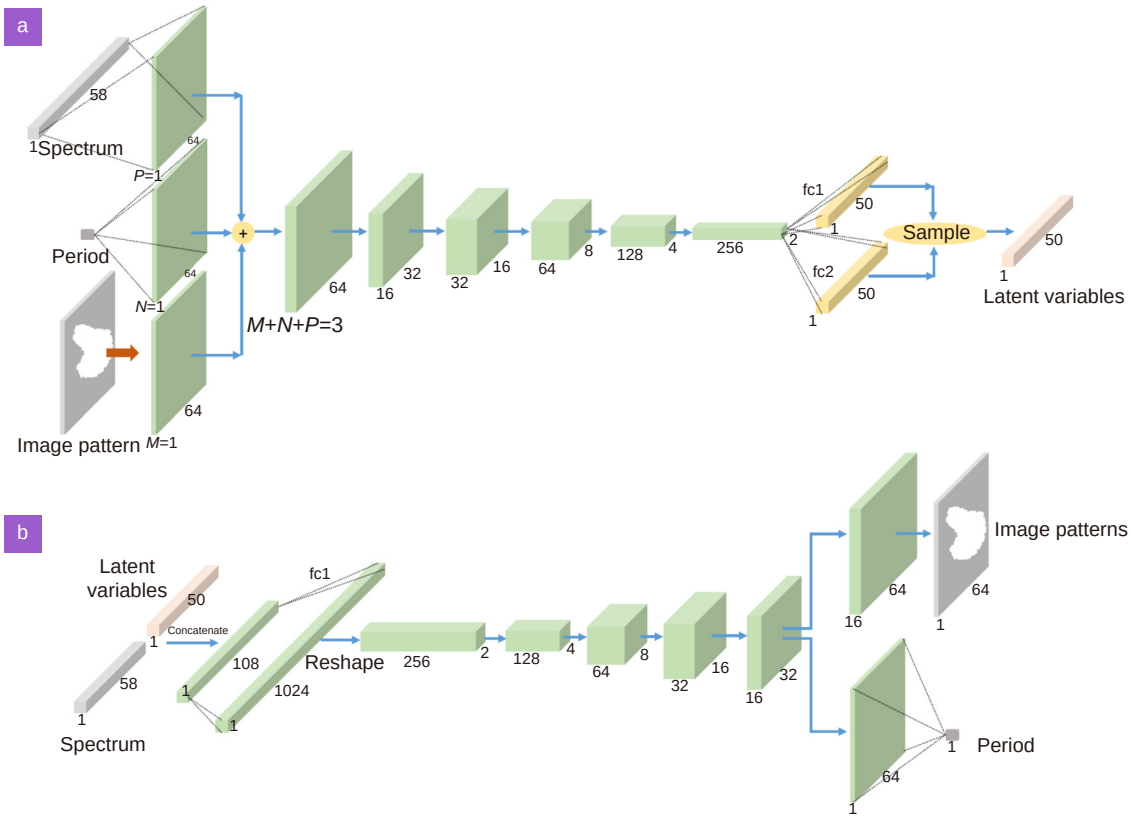

Fig. S3 | The CNN network structure of the c-VAEs for the free-form structures. C-VAEs include three neural networks: the recognition networks, the generation networks, and the conditional prior networks. Specifically, the conditional prior networks use the same constructions as the INNs, which is shown in Fig. S2(b). Therefore, we do not include them here. (**a**) The structure of the recognition networks, which are used to find the latent distribution z. The network structures are similar to the structures of the FNNs. We will first map 58-dimensional spectrum vector into a 4096-by-1 vector using a single FC, then reshape this it into a 64*64 size image. Similar procedures are done for the scalar of the period. These two images transformed from the spectrum and the period are further concatenated with the 2D image pattern, which corresponds to the free-form structure after the convolution with kernel size k=1. We can change the depth of concatenation by changing the m, n and p. By default, we set m=n=p=1. Later on, 5-layer convolutional layers are used to recognize and extract these image features into a 256-depth two-by-two matrix. Different from the FNNs, after reshaping this matrix into a 1024-by-1 vector, two separate fully connected layers are connected to process features and mapping into two different 50-dimensional vectors, which describe the mean value μ and variance v of the latent variables. These 50-d vectors are later used to construct the latent variables that follow the gaussian distribution. (**b**) The structure of generation networks, which generate predicted structures based on spectra and latent variables. The neural network structure is very similar to the INNs in Fig. S2(b). The only difference is that for the input, both spectrum vector and latent variables are concatenated into a 108-by-1 long vector.

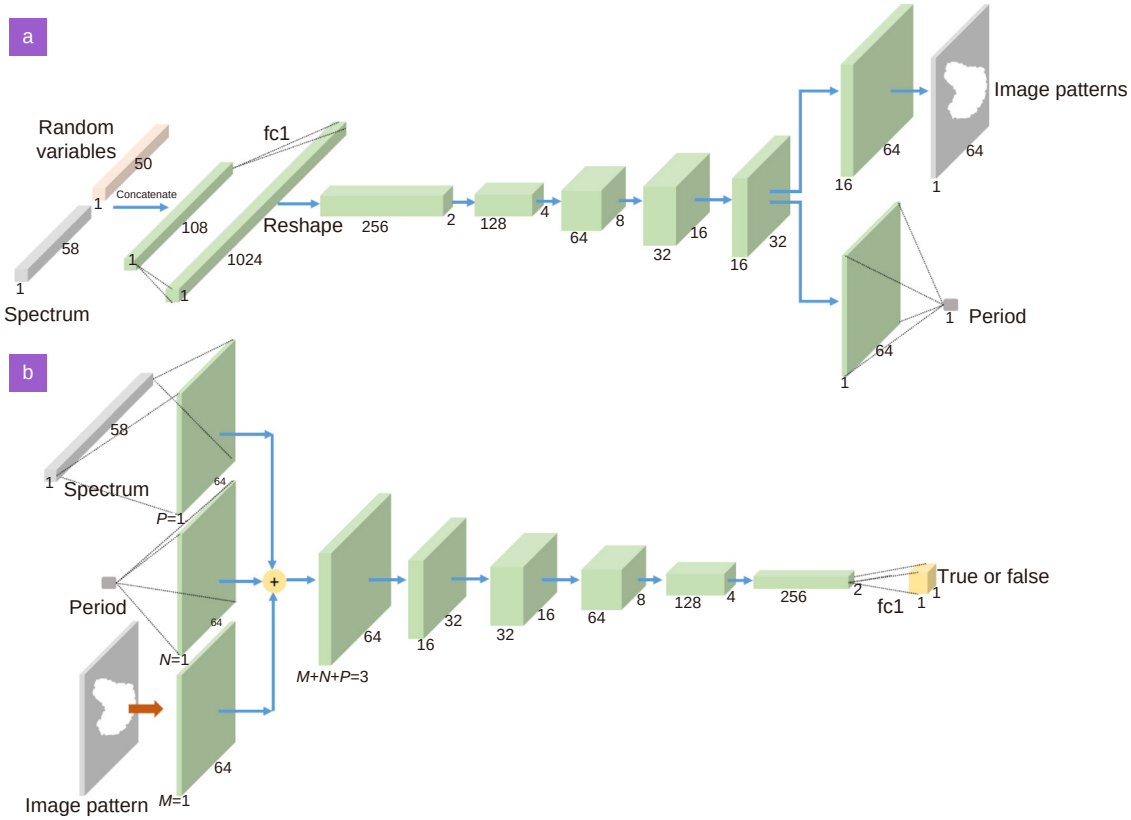

Fig. S4 | The CNN network structure of the c-GANs for the free-form structures. C-GANs include two neural networks: the generator and the critic. (**a**) The structure of the generator networks, which generate structures based on the spectra and normally-distributed random variables. The network structure is very similar to the INNs shown in Fig. S2(b). The only difference is that both spectrum vector and random variables are concatenated into a 108-by-1 long vector for the input. (**b**) The structure of the critic networks. The network structures are similar to the structures of the recognition network in the c-VAEs, which is shown in Fig. S3(a). The generator also takes in the structures as well as the spectra. Different from the recognition networks in c-VAEs, after reshaping this matrix into a 1024-by-1 vector, only one fully connected layer is connected to process these features by mapping them into a scalar. This scalar represents a scores that given the conditional input of spectra, if the input structures come from the original training dataset or the generator. During training, for the structures contained in the training dataset, critic networks will always give high scores, while for the generated structures given by the generator, critic networks will always give low scores. Therefore, after training, the generator network will try to generate structures similar to the training structures such that critic networks tend to give high scores.

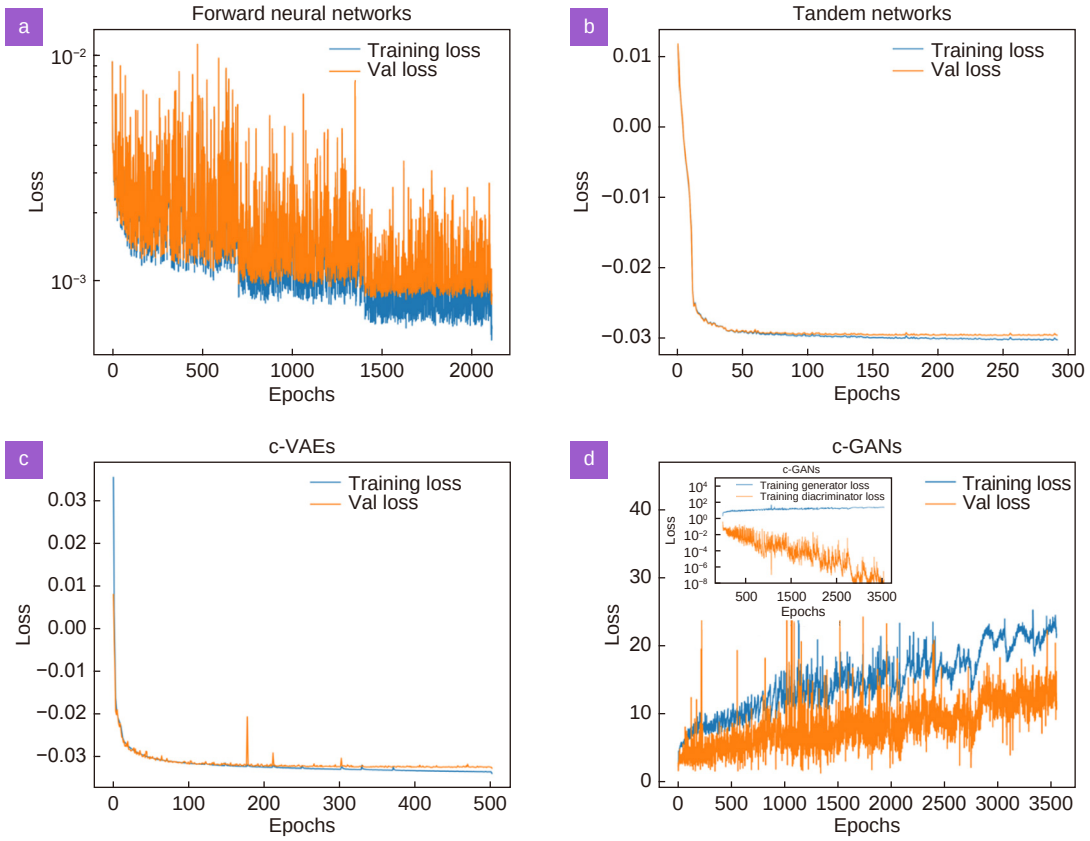

Fig. S5 | The training curves for each model in the free-form structure inverse design problem, where the blue and orange curves refer to training loss and validation loss. Notice that different models have different loss functions, so we cannot compare their relative values directly. (**a**) Training curve of the FNNs. (**b**) Training curve of tandem networks. (**c**) Training curve of c-VAEs. (**d**) Training curve of the c-GANs. The inset shows the generator loss and critic loss during training, where generator loss is maximized, and critic loss is minimized.

Section 2: Template structures

This part gives extra information for performance comparisons in the template structures.

Choosing the size of dataset

In both tasks, we are using a finite size of the training dataset. In principle, when the data volume is infinitely large such that a good coverage of the entire design space is guaranteed, these three networks would give accurate inverse predictions. Since the dataset is generated based on EM simulations, obtaining a sufficiently large dataset to achieve this ideal performance is usually impractical, and the performance when dataset size is limited is of more practical interest. We choose the size of the dataset in our manuscript so that these datasets can be collected in a reasonable amount of time but can still provide good performance. Thus, we believe our conclusions can faithfully reflect the accuracy performance of these deep learning-based inverse design models in practical settings.

R^2 scores and MAE of color

A detailed comparison of R^2 scores and MAE for the color inverse design is shown in Fig. S6, where Fig. S6(a) and Fig. S6(e) show the performance of FNNs, which are used to predict the color for a given structure input. In Fig. S6(a), the x axis is the ground truth of three target color coordinates, while the y axis is the predicted color coordinates given by the FNNs. Three figures show the three color coordinates of (x, y, Y), respectively. In Fig. S6(e), the x axis is the ground truth of the target color coordinates, while y axis gives the MAE of predicted color coordinates. The high R^2 score and low MAE mean that the FNNs are very accurate in predicting colors for a given structure.

In Fig. S6(b–d), we give more details of R^2 scores for the three models: Tandem networks, c-VAEs, and c-GANs, respectively. The x axis is the ground truth of the target color coordinates, while the y axis is the predicted color coordinates

given by the predicted structures from each model. The predicted color coordinates are calculated using RCWA. In Fig. S6(f–h), we give more details of MAE for the three models: Tandem networks, c-VAEs, and c-GANs, respectively. Again, the *y* axis is the MAE between target color coordinates and predicted color coordinates correspond to the predicted structures.

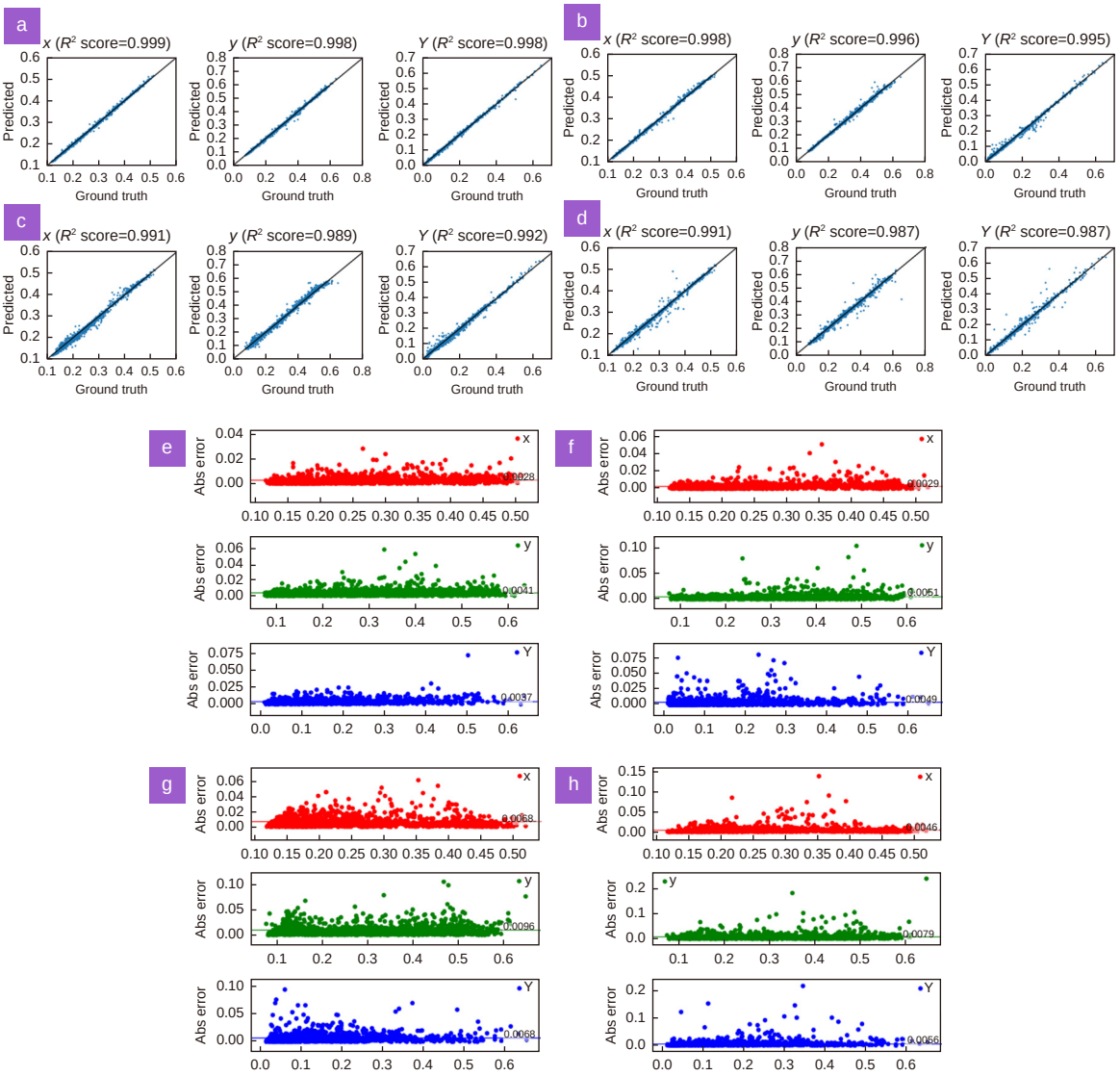

Fig. S6 | A detailed comparison of R^2 scores and MAE for the color inverse design. (**a**–**d**) The details of R^2 scores of FNNs, tandem networks, VAEs and GANs. (**e–h**) The absolute difference of color for FNNs, tandem networks, c-VAE and c-GAN

More examples of color inverse design

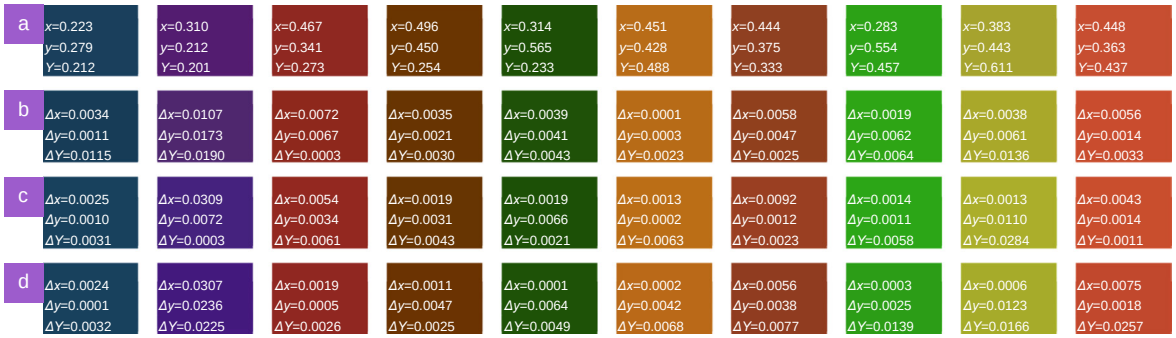

Fig. S7 | Another 10 examples of color inverse design. Row (**a**) is the target color, where the inset numbers are the color CIE coordinates. Row (**b**–**d**) correspond to the predicted colors correspond to the structures predicted by the tandem networks, the c-VAEs, and the c-GANs, respectively, and the inset numbers are the absolute CIE difference between the inverse predicted color and the targe color.

The diversity of inverse designing brown and yellow color

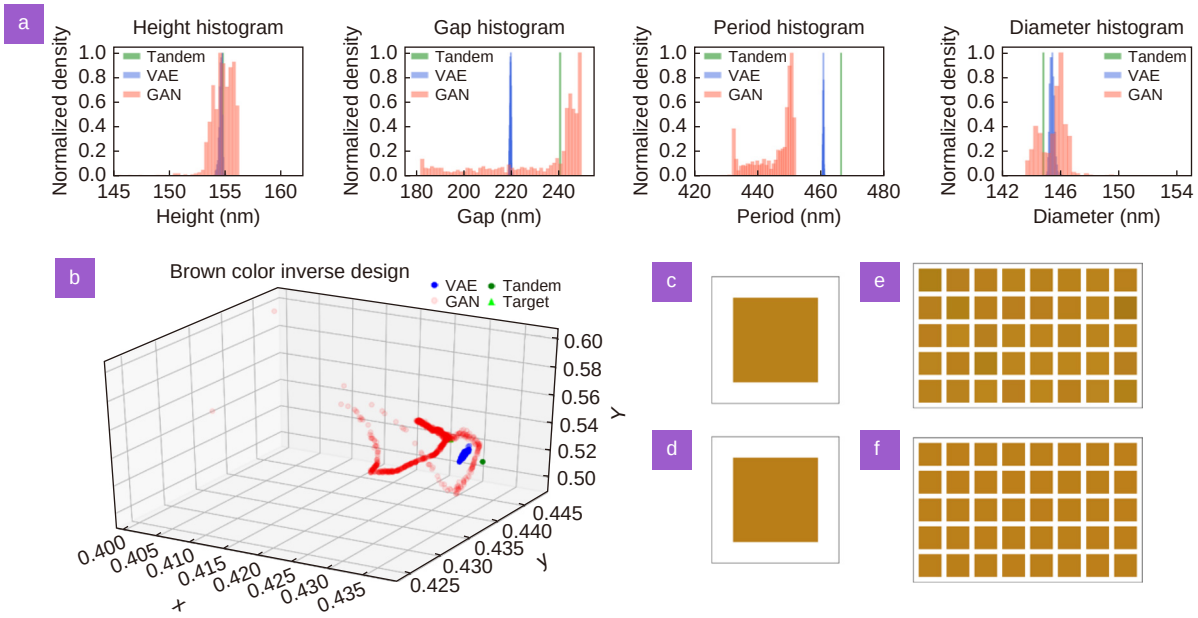

Fig. S8 | (**a**) The histogram distribution of 1000 inverse designed structure parameters for the brown color (**c**) with the CIE coordinates (*x*, *y*, *Y*)=(0.4320, 0.4404, 0.5415). (**b**) The 3-dimensional color distribution is related to 1,000 inverse designed structures. We can see all these predicted structures give fairly accurate brown color. (c) The target brown color with coordinates (*x*, *y*, *Y*)=(0.4320, 0.4404, 0.5415). (**d**) The color corresponding to the structure predicted by tandem networks. (**e**) The randomly selected 40 different colors correspond to the structures predicted by the VAE. (**f**) The randomly selected 40 different colors corresponding to the structures predicted by the GAN.

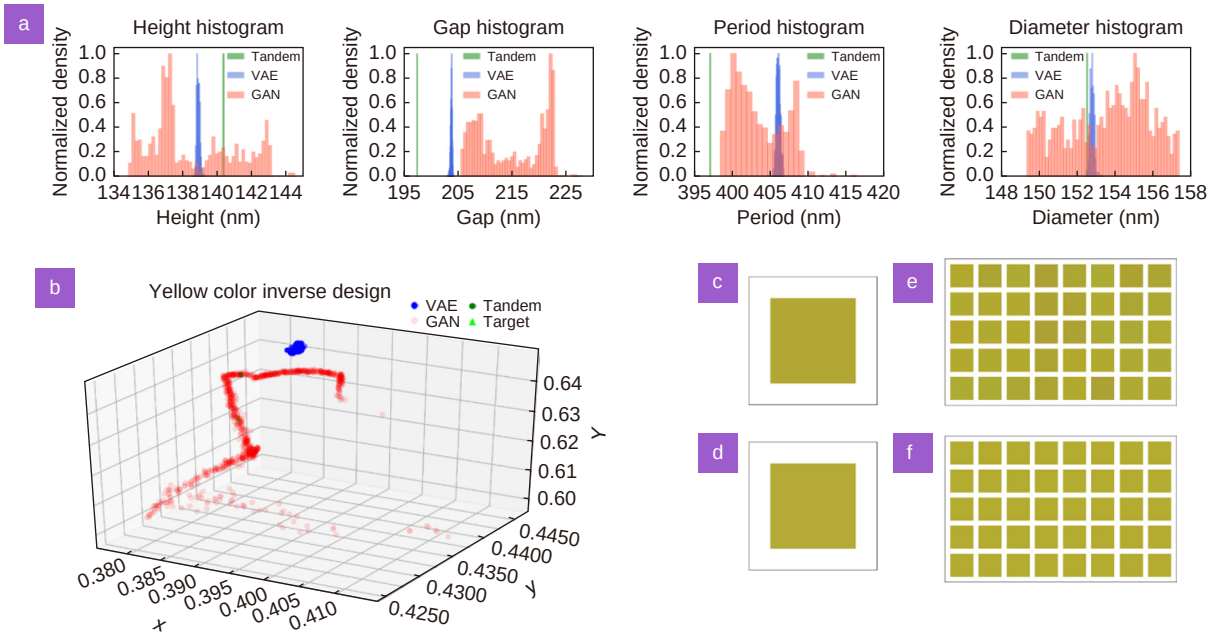

Fig. S9 | (**a**) The histogram distribution of 1000 inverse designed structure parameters for the yellow color (c) with the CIE coordinates (*x*, *y*, *Y*)=(0.3851, 0.4320, 0.6305). (**b**) The 3-dimensional color distribution related to 1,000 inverse designed structures. We can see all these predicted structures give fairly accurate yellow color. (**c**) The target yellow color with coordinates (*x*, *y*, *Y*)=(0.3851, 0.4320, 0.6305). (**d**) The color corresponding to the structure predicted by tandem networks. (**e**) The randomly selected 40 different colors corresponding to the structures predicted by the VAE. (**f**) The randomly selected 40 different colors corresponding to the structures predicted by the GAN.

The robustness analysis of reproducing a painting

In Fig. 3(c–h), we provide a real application of reconstructing the Vincent van Gogh's painting: Fishing Boats on the Beach at Saintes Maries-de-la-Mer. For the inverse designed structures given by each model, we are supposed to use the electromagnetic (EM) simulators to validate their colors. However, since there are millions of pixels inside this painting, we are using the trained evaluation forward model to calculate their colors. Now we will analyze the robustness of the painting reproduction. For each color pixel, we examine if the predicted structure is a *faulty design* (meaning the predicted structure does not satisfy the constraints of the physical systems). If it is, we change the corresponding pixel to white color. We analyze all the reconstructed images in Fig. 3(d–h), and show the results in Fig. S10(b–f). Based on the number of white pixels, we can see that there is a higher chance for tandem networks to give failed structures. Also, increasing the sampling times for VAEs and GANs can further improve the robustness slightly.

In Fig. S11(b), we show the color distribution corresponding to the white pixels in each model in the CIE chromatic diagram. These colors are the failed colors for each model (meaning the inverse designed structures are not physical). We can see that most of these failed colors are located at a region where the training dataset is pretty sparse (shown in Fig. S11(a)). Therefore, in order to provide a better platform for future researchers to benchmark and compare their own models and new implementations, we will also create new datasets that are uniformly distributed in the chromatic diagram and provide them on GitHub[5] in the future.

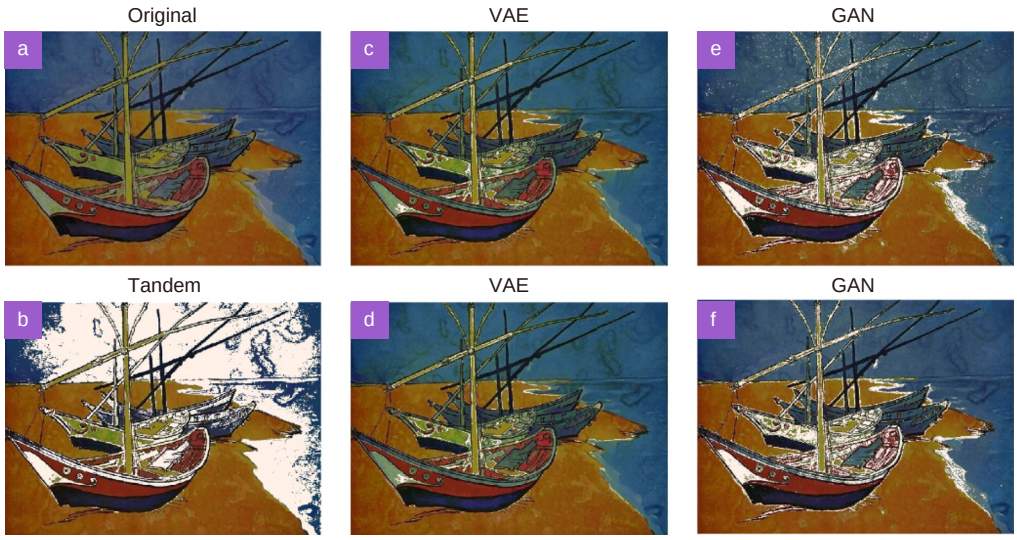

Fig. S10 | The robustness analysis in Fig. 3(c–h). The white color in each image means the model fails to inverse predict a physical structure at this color pixel. (**a**) The original image of the Vincent van Gogh's painting: Fishing Boats on the Beach at Saintes Maries-de-la-Mer. (**b–f**) The robustness analysis corresponds to the image in Fig. 3(d–h).

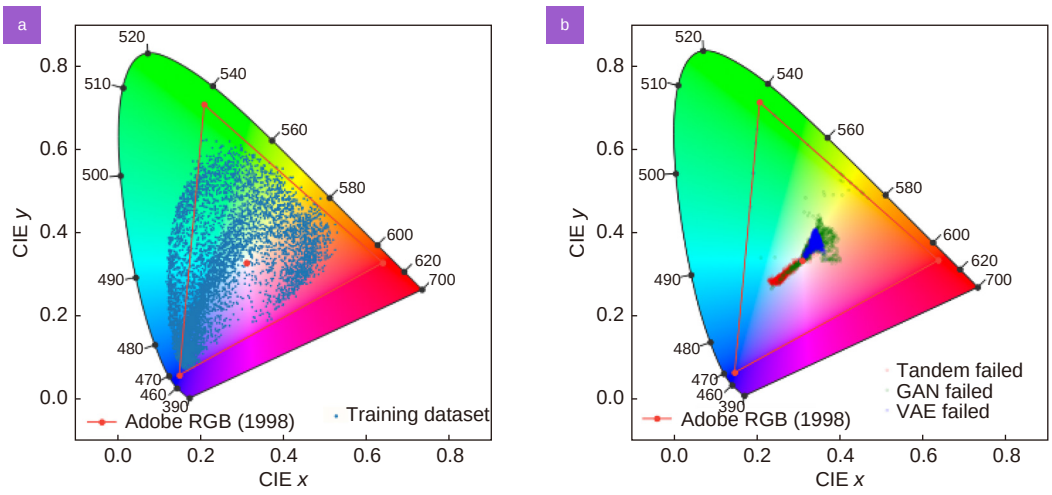

Fig. S11 | (**a**) The obtained color of training dataset embedded in the CIE 1931 chromatic diagram. (**b**) The target colors corresponding to the white pixel in Fig. S10 embedded in the CIE 1931 chromatic diagram. Most of those failed color are located at a region where the training dataset is pretty sparse.

The robustness of size of array for a pixel

As mentioned in the main text, the previous comparisons and discussions are based on the idea that the predicted structure is represented inside a periodic unit cell. The corresponding structure color is calculated using this periodic boundary conditions, assuming the same structure extends to infinity. This condition can be applicable if we are considering pure color printing. However, for the real application, such as image reconstruction, we cannot assume that the structure is periodic anymore. For each color pixel, we need to consider an array with finite number of unit cells[6]. In order to find the structure color associated with a specific nonperiodic structure, we use FDTD to simulate a large region, which contains different number of unit cells. Inside this region, these unit cells form an array. Specifically, we consider the size of array to be 2 by 2, 3 by 3, 4 by 4, and 5 by 5. During simulation, we set the simulation region to include the whole region of the array and change boundary conditions to perfect matching layers. The simulated reflection spectra are used to calculate the structure color.

Section 3: Free-form structures

This part gives extra information for performance comparisons in the free-form structures.

Image generation and post-process

The 2D patterns are randomly generated using the same algorithm in ref.[4]. During the generation of simulation samples, some sharp structures can be generated. In order to satisfy the fabrication limitation, we smooth the structures so that all sharp structures satisfy the minimum 20 nm radius curvature[7]. For simulations in RCWA, the region with image pixel equals to one is treated as silicon, while the region with the image pixel equals to zero is treated as air. However, for the inverse predicted structures given by the neural networks, the pixel values may not equal to zero or one. We post-process the predicted structures by setting up a binarization threshold of 0.5. Any pixel values greater than 0.5 is assigned to be one.

In the main text, we introduce the quantity of *irr* in order to evaluate the distribution of structures. We give several examples of different *irr* in Fig. S12, showing that by increasing the *irr*, the structure's irregularity starts to increase. Therefore, we can use this *irr* to reveal part of the structure distribution.

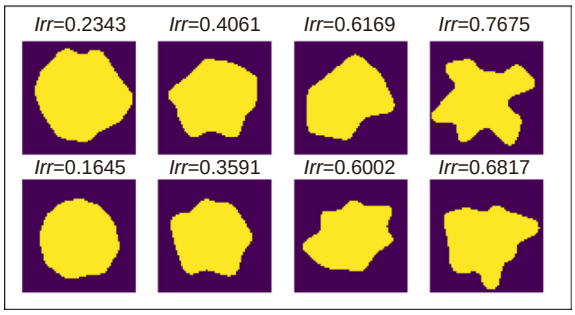

Fig. S12 | Several images demonstrating that *irr* can be used as a criterion to describe the irregularity of images. As we can see, small *irr* refers to a more regular-shaped images, while images are more irregular when *irr* is greater.

More examples of color inverse design

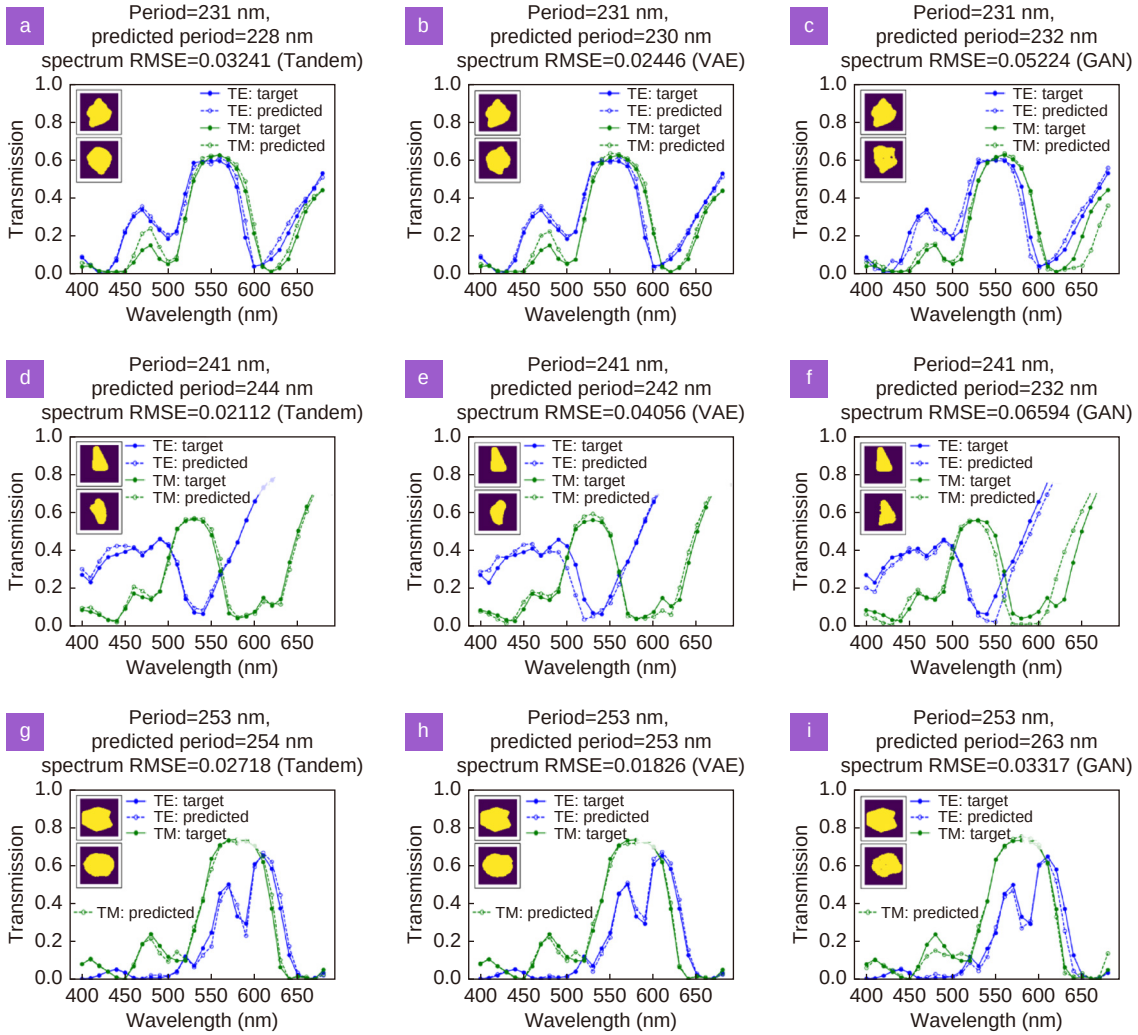

Fig. S13 | More examples of randomly selected transmission spectrum inverse design for the tandem networks (**a**, **d**, **g**), VAEs (**b**, **e**, **h**), and GANs (**c**, **f**, **i**). The inset shows the original structure (upper) in the test dataset and the inverse predicted structure (lower) by each model.

The diversity of another spectrum inverse design

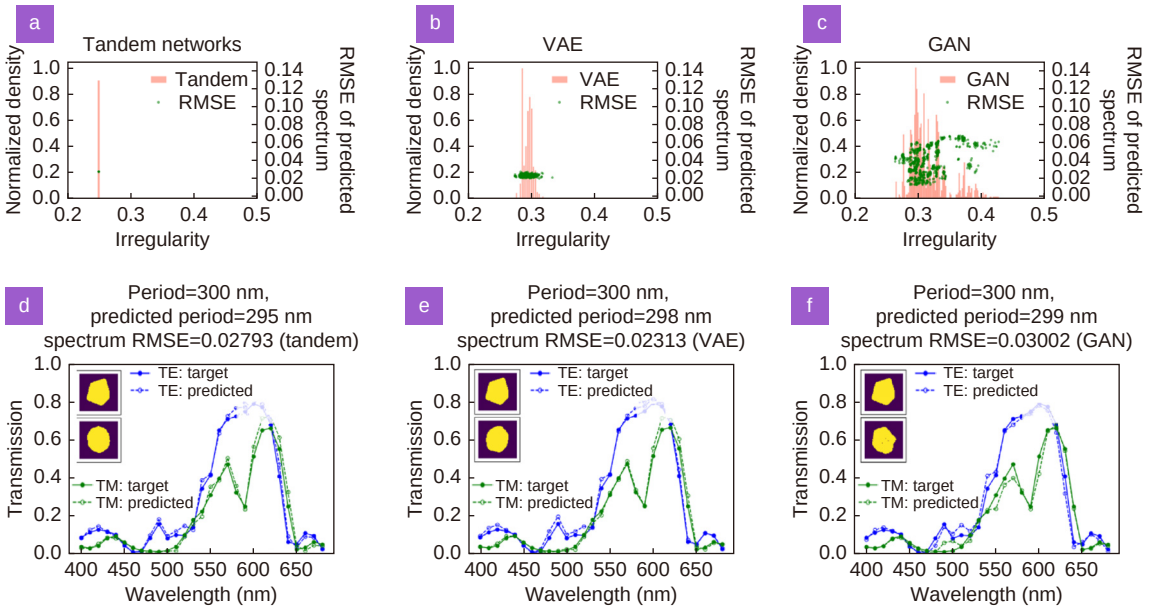

Fig. S14 | Another example of comparisons of diversity for three models in the template structure inverse design. (**a**–**c**) The red bar shows the distribution of the normalized density of irregularity for 1,000 inverse predicted structures by tandem networks, VAEs and GANs, while the green points are the scatter plot of spectrum RMSE VS the irregularity. According to the distribution of irregularity, we can see that tandem networks only give one structure prediction, where the VAE gives limited diversity, and the GAN gives multi-modal structure distributions that covers a wide region. (**d**–**f**) A randomly-chosen structure from 1,000 predictions as well as its spectrum predicted by tandem networks, VAEs and GANs. The inset shows the original structure (upper) in the test dataset and the predicted structure (lower) given by each model.

The robustness in term of fabrication variations

In order to look at the tolerance of fabrication variations of the predicted structures, we change the shape of predicted structures by shrinking, expanding, or smoothing by a small factor. In order to do this, we first do a gaussian convolution with different kernel size, and then do a binarization with different threshold. In Fig. S15, we give several examples when we change the kernel size and the binarization threshold. We can see that increasing the kernel size will make the edges of the structure smoother, while increasing or decreasing the binarization cutoff threshold can shrink or expand the image. When we consider the robustness in terms of fabrication error, we change the kernel size from 3 to 7 and change the binarization cutoff threshold from 0.1 to 0.5. Their corresponding simulated spectra in each case are also shown in Fig. S15 for reference.

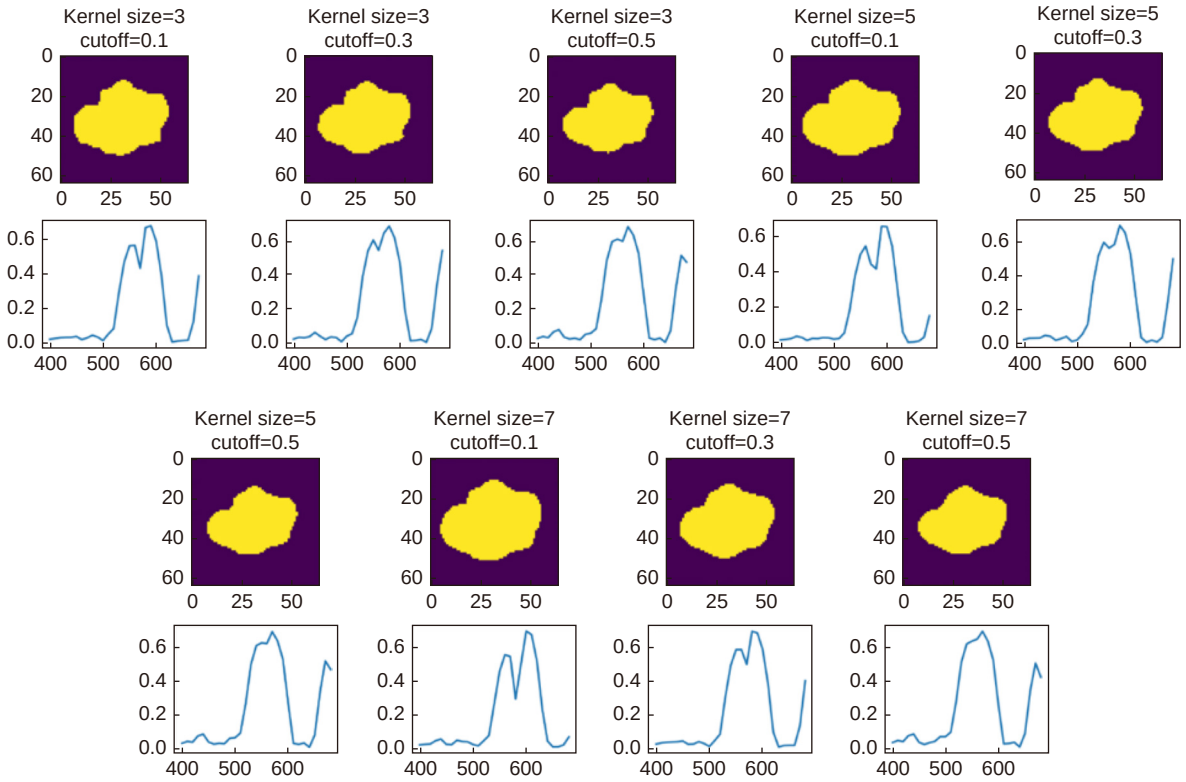

Fig. S15 | First row: the considered fabrication error by changing the kernel size and binarization cutoff. **Second row**: the corresponding transmission TE spectrum. The x axis is the wavelength in the unit of nm, and the y axis is the transmission spectrum.

References

1. Prechelt L. Early stopping - but when? In Orr GB, Müller KR. Neural Networks: Tricks of the Trade. 55 –69 (Springer, 2002); http://doi.org/10.1007/3-540-49430-8_3.
2. Sohn K, Yan XC, Lee H. Learning structured output representation using deep conditional generative models. In *Proceedings of the 28th International Conference on Neural Information Processing Systems* 3483–3491 (MIT Press, 2015).
3. Mirza M, Osindero S. Conditional generative adversarial nets. arXiv: 1411.1784 (2014).
4. Han X, Fan ZY, Liu ZY, Li C, Guo LJ. Inverse design of metasurface optical filters using deep neural network with high degrees of freedom. *InfoMat* **3**, 432–442 (2021).
5. https://github.com/taigaoma1997/benchmark_in_de.git
6. Yang WH, Xiao SM, Song QH, Liu YL, Wu YK et al. All-dielectric metasurface for high-performance structural color. *Nat Commun* **11**, 1864 (2020).
7. https://github.com/Toblerity/Shapely