MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Generative Deep Learning Model for Inverse Design of Integrated Nanophotonic Devices

Tang, Yingheng; Kojima, Keisuke; Koike-Akino, Toshiaki; Wang, Ye; Wu, Pengxiang; TaherSima, Mohammad; Jha, Devesh K.; Parsons, Kieran; Qi, Minghao

TR2020-135 October 07, 2020

Abstract

We propose a novel Conditional Variational Autoencoder (CVAE) model for designing nanopatterned integrated photonic components. In particular, we show that prediction capability of the CVAE model can be significantly improved by adversarial censoring and active learning. Moreover, generation of nanopatterned power splitters with arbitrary splitting ratios and 550 nm broadband optical responses from 1250 nm to 1800 nm have been demonstrated. Nanopatterned power splitters with footprints of 2.25 x 2.25 um2 and 20 x 20 etch hole positions are the design space, with each hole position assuming a radius from a range of radii. Designed nanopatterned power splitters using methods presented herein demonstrate an overall transmission of about 90% across the operating bandwidth from 1250 nm to 1800 nm. To the best of our knowledge, this is the first time that a state-of-the-art CVAE deep neural network model has been successfully used to design a physical device.

Lasers and Photonics Reviews

© 2020 MERL. This work may not be copied or reproduced in whole or in part for any commercial purpose. Permission to copy in whole or in part without payment of fee is granted for nonprofit educational and research purposes provided that all such whole or partial copies include the following: a notice that such copying is by permission of Mitsubishi Electric Research Laboratories, Inc.; an acknowledgment of the authors and individual contributions to the work; and all applicable portions of the copyright notice. Copying, reproduction, or republishing for any other purpose shall require a license with payment of fee to Mitsubishi Electric Research Laboratories, Inc. All rights reserved.

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

ORIGINAL ARTICLE

Generative Deep Learning Model for Inverse Design of Integrated Nanophotonic Devices

Yingheng Tang[1,2] | **Keisuke Kojima**[1*] | **Toshiaki Koike-Akino**[1] | **Ye Wang**[1] | **Pengxiang Wu**[1] | **Youye Xie**[1] | **Mohammad H. Tahersima**[1] | **Devesh K. Jha**[1] | **Kieran Parsons**[1] | **Minghao Qi**[2]

[1]Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA 02139, USA.

[2]School of Electrical and Computer Engineering and Birck Nanotechnology Center, Purdue University, West Lafeyette, IN 47907, USA.

Correspondence
* Keisuke Kojima
Email: kojima@merl.com

Funding information

We propose a novel Conditional Variational Autoencoder (CVAE) model for designing nanopatterned integrated photonic components. In particular, we show that prediction capability of the CVAE model can be significantly improved by adversarial censoring and active learning. Moreover, generation of nanopatterned power splitters with arbitrary splitting ratios and 550 nm broadband optical responses from 1250 nm to 1800 nm have been demonstrated. Nanopatterned power splitters with footprints of $2.25 \times 2.25\,\mu\mathrm{m}^2$ and 20×20 etch hole positions are the design space, with each hole position assuming a radius from a range of radii. Designed nanopatterned power splitters using methods presented herein demonstrate an overall transmission of about 90% across the operating bandwidth from 1250 nm to 1800 nm. To the best of our knowledge, this is the first time that a state-of-the-art CVAE deep neural network model has been successfully used to design a physical device.

KEYWORDS
metamaterials, inverse design, photonic integrated circuits, neural network, deep learning, adversarial learning, nanophhotonics

Optimization of nanopatterned photonic devices is an active area of research where several optimization methods have been proposed. For example, direct binary search (DBS) [1, 2, 3], genetic algorithm [4, 5], and particle swarm optimization [6], etc. have been used for various optimization applications; each shown to have unique advantages. All of these methods, however, usually involve a large amount of time and computing resources. For example, a DBS optimization setup with 20×20 hole vectors has an extremely large number of possible combinations (2^{400}), which requires many electro-magnetic (EM) simulations such as finite-difference time-domain (FDTD) computations. In order to reduce such heavy EM simulation load, machine learning (including deep learning) is becoming more popular to assist the design process of photonic devices to accelerate the underlying optimization process in the past few years. Recent successes of deep learning in modeling complex input-output relationships in spatial-temporal data have inspired the idea of intuitive physics engines that can learn physical dynamics in mechanics [7, 8, 9], material discovery [10, 11, 12], particle physics [13], and optics [14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25].

Deep neural network (DNN) approaches may require large amounts of data for training and could often be notoriously difficult to train. However, once the network is trained properly, the execution of forward propagation on the trained network can be much faster than expensive EM simulations (e.g., orders of magnitude faster than EM simulations). A critical criterion for the application of such learning-based models for physical systems is their generalization capability to problems beyond the training data set. Such generalization capability would enable the use of artificial neural network (ANN) models as a forward design optimization engine that is trained on a limited partially optimized data set.

In the field of integrated photonics, there have been several attempts in using machine learning methods for nanopatterned photonic design. We previously developed an artificial intelligence integrated optimization process using NN that can accelerate optimization by reducing the number of numerical simulations involved in the optimization process [24, 25]. Moreover, Tahersima et al. [14] used DNN

models in the forward/backward directions, i.e., correlate target performance data (such as transmission spectra) and device design. The DNN models used (i.e., residual networks), however, involved one-to-one mapping that generates only one device for each performance criterion. Previous demonstrations have shown methods for design/optimization of binary pixels (i.e., fixed-size etch hole being present or not). Here, we demonstrate a multilevel pixel structure (i.e., multi etch hole dimensions) that is a more complex optimization problem than previous studies and requires more sophisticated optimization algorithms.

In the field of metamaterials, a few studies have been proposed to use generative networks for designing nanopatterned metamaterial. A generative network can provide a series of well-optimized patterns based on random initial conditions. Liu et al. and An et al. have applied the Generative Adversarial Networks (GAN) [15, 26] and Ma et al. has employed the Variational Autoencoder (VAE) [23] for such applications. Inspired by these works, we first developed a model that uses a Conditional Variational Autoencoder (CVAE) [27] in the design of a nanopatterned power splitter application. One advantage of the VAE-based models compared to some other generative networks (i.e., GAN) is that it takes into account a probability distribution of the initial data for the generation of new data (i.e., nanopatterned power splitter geometries). This makes the generated pattern obey the physics (the general light propagation path) while having the variance with detailed pattern parameters.

The VAEs allow us to model a distribution of the nanopatterned power splitters with various splitting ratios. Therefore, VAE can generate novel patterns subject to the same distribution using variational data sampling. When conditions (i.e., optical responses) are supplied in conjunction with the latent variables, a VAE evolves into a CVAE, which enables generation of patterns satisfying given target optical responses. For the arbitrary power splitting application, we have included different hole sizes to define the geometrical structure of the nanopatterned power splitters. Doing so, the generated power splitter can have additional control parameters in wave-guiding an incoming beam by varying phase control and am-

plitude control at each hole position by determining a radius for the hole. We demonstrate that the additional control parameters can yield generated power splitter designs with a flat spectrum response across a wide bandwidth.

However, one major drawback of the CVAE model itself is that the target performance data (i.e., the condition) leaks into the patterns after the training. This degrades the quality of the geometrical design, leading to non-ideal performances. In order to circumvent this issue, we explore adding an adversarial block [28] to isolate the pattern from the device optical performance (transmission and reflection spectra) during the training to further improve the generation performance of the decoder of the machine learning model, as presented herein. We call this the Adversarial CVAE (A-CVAE). Even though the A-CVAE was originally applied to image attribute adjustment [29], the character style transfer problem [28], and brain wave classification problem [30], this paper is the first time that the A-CVAE was successfully applied to a design problem of any physical device, and nanophotonic devices in particular.

The design space of the nanopatterned power splitters has a footprint of $2.25\times 2.25\mu\mathrm{m}^2$ with a 20×20 etch hole positions. We confirm that the optimized device has an overall performance close to 90% across all the bandwidth from C-band to O-band (1250 nm to 1800 nm). To the best of the authors' knowledge, this is the smallest broadband power splitter with arbitrary ratios.

Figure 1a shows an exemplary footprint of a nanopatterned power splitter and Figure 1b shows a Transverse Electric (TE) mode propagating inside the input port and the two output ports. The nanopatterned power splitters are silicon photonic devices, each having a square section with a footprint of $2.25 \times 2.25\ \mu\mathrm{m}^2$. Each device has an input port and two output ports that are all $2\ \mu\mathrm{m}$ long, tapered and connected at one end to optical waveguides with a width of 500 nm and a height of 220 nm and connected to an square section of that device at the other end. The square section includes 20×20 Hole Vector (HV) to define geometrical structures of the device. The silicon photonic device is built on top of a 2 μm-thick silicon dioxide on a silicon substrate, and is covered with silicon oxide cladding. Hole

spacings are 112 nm from center to center of each hole position, and the maximum and minimum hole diameters are 77 nm, and 42 nm respectively, corresponding to the variable value of 1 and 0.3, with hole area proportional to the variable. If the variable is below 0.3, there is no hole. The HV training data only consist of binary numbers initially and is obtained through the direct binary search method. Note that the VAE models the probabilistic distribution for the different HV, each generated value of the HV is a Bernoulli distribution. In order to best reflect the result, continuous variable hole sizes are used to represent the probability of the appearance of etch holes at certain locations.

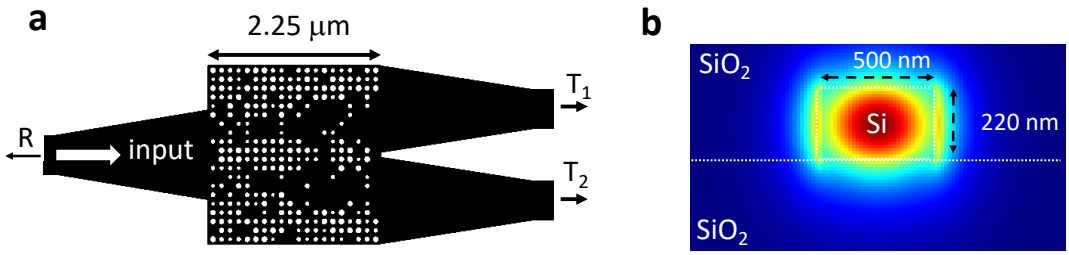

FIGURE 1 Schematic of the power splitter. a) Top view, where T_1 and T_2 denote the modal transmissions of output ports 1 and 2, and R denotes the reflection at input port. b) Cross-section of the input/output waveguide.

The network configuration of the CVAE is shown in Figure 2a. The original HVs are passed to two convolutional layers and reduced to two sets of intermediate parameters of a probability density function (PDF), representing the means $\mu := (\mu_1, \ldots, \mu_J)$ and standard deviations $\sigma := (\sigma_1, \ldots, \sigma_J)$. In order to make the back propagation possible for the network, a reparameterization method is applied, described by the following equation:

$$z_i = \mu_i + \sigma_i \varepsilon_i, \tag{1}$$

where ε_i are independent and identical distributed samples drawn from the standard Gaussian distribution. After the reparameterization process, the latent variable $z := (z_1, \ldots, z_J)$ is concatenated with the encoded condition parameter s (the dimension reduced from the original 63 to 9 through one fully connected layer) to

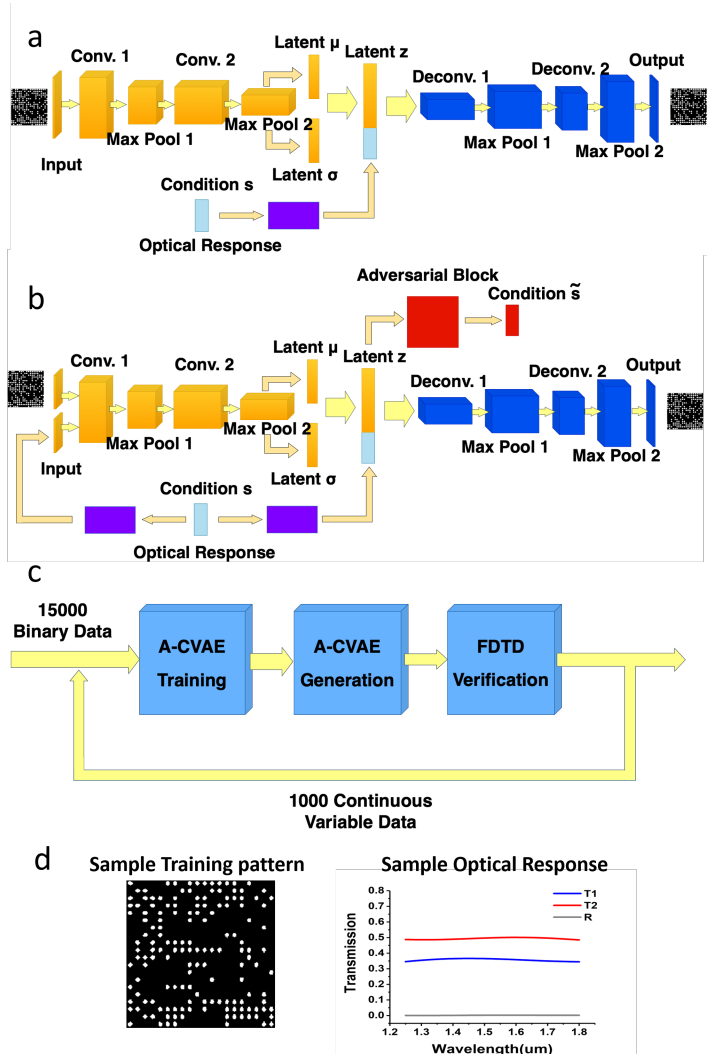

FIGURE 2 a) The CVAE model structure, wherein the input is the 20×20 hole vector. The first convolutional layer has 16 channels with a kernel size of 3. The second convolutional layer has 32 channels with a kernel of 3. The condition s is the transmission and reflection spectra obtained from the FDTD simulation to form a 3×21 matrix, and is fed to the latent variable through a fully connected (FC) layer. b) The A-CVAE model structure, wherein the input right now becomes the two channels of 20×20 hole vector, where the first channel is the hole vector of the device and the second channel is the decoded spectra data. The main difference from a) is that one adversarial block is added, which is composed by two FC layers. c) The active learning method added for the A-CVAE method, where 1000 newly generated continuous variable data are added to the original 15,000 binary training data. d) One sample training data, wherein the left figure is the pattern (hole vector) and the right figure shows the three spectra of the training device (the transmission for the two ports and the reflection). Each spectra response has 21 data points, which will be fed into the network as the Optical response.

deconvolute back to the HV. Here the condition input is the optical response for the device including the transmission spectra of the two ports. and the reflection spectra (shown in Figure 2d). Each spectra has 21 data points which make the total condition input with the dimension of 63. The loss function is constructed by two parts: a) a cross-entropy loss between the original HV $x := (x_1, \ldots, x_n)$ and the decoded HV $y := (y_1, \ldots, y_n)$, and the Kullback–Leibler (KL) divergence between the latent variables and the Gaussian prior. The equation of loss function is shown as follows:

$$\text{Loss} = -\sum_{i=1}^{n}\Big[y_i \log x_i + (1-y_i)\log(1-x_i)\Big] + \frac{1}{2}\sum_{j=1}^{J}\Big[\mu_j^2 + \sigma_j^2 - \log(\sigma_j^2) - 1\Big]. \quad (2)$$

Convolutional neural networks (CNN) have been shown to be effective in handling geometrical input data [20, 21, 23, 31]. In this work, we use two convolutional layers both for the encoder and the decoder. The number of channels for the two layers are 16 and 32, and the max pooling stride is 2, after that there is one fully connected layer to reduce the latent variable to 63. The output is then concatenated with the performance data and feed them into the decoder. The validations are calculated by using the EM simulation to verify a figure of merit (FOM) of generated patterns, where the FOM is calculated by:

$$\text{FOM} = 1 - 10 \times \int_{\lambda_{\min}}^{\lambda_{\max}} \Big[\big|T_1(\lambda) - T_1^{\star}(\lambda)\big|^2 + \big|T_2(\lambda) - T_2^{\star}(\lambda)\big|^2 + \alpha \times R^2(\lambda)\Big] \mathrm{d}\lambda. \quad (3)$$

where $T_1(\lambda)$, $T_2(\lambda)$, $R(\lambda)$, and $[\cdot]^{\star}$ denote transmissions of output ports 1 and 2, reflection at input port at a given wavelength λ, and corresponding target values, respectively. The target values are the desired optical performances. For example, for the 6 : 4 type device, we set $T_1(\lambda) = 60\%$, $T_2(\lambda) = 40\%$ and $R(\lambda) = 0$. α is a weighting factor where $\alpha = 4$ is used to balance between the contributions from transmission and the reflection. We take the average of FOM over the FDTD simulation spectral range. As $T_1(\lambda)$ and $T_2(\lambda)$ approach their target spectra, and as $R(\lambda)$ decreases, the FOM increases. For an ideal power splitter with no excess

loss and no reflection, the FOM should be 1.

Figure 3 shows the training loss and the validation result as a function of the training iterations. The solid black line represents the training loss and the solid blue line represents the trend for validations for the CVAE model. The plot shows that the training result approaches its optimal point when the epoch number is between 10 to 15. The validation results further decreased beyond the epoch number of 30. Here, we used about 15,000 binary training data, including power splitters semi-optimized by DBS with the target splitting ratios of 5 : 5, 6.5 : 3.5, and 1 : 0. The validation result is an FOM calculated over 20 devices of each type.

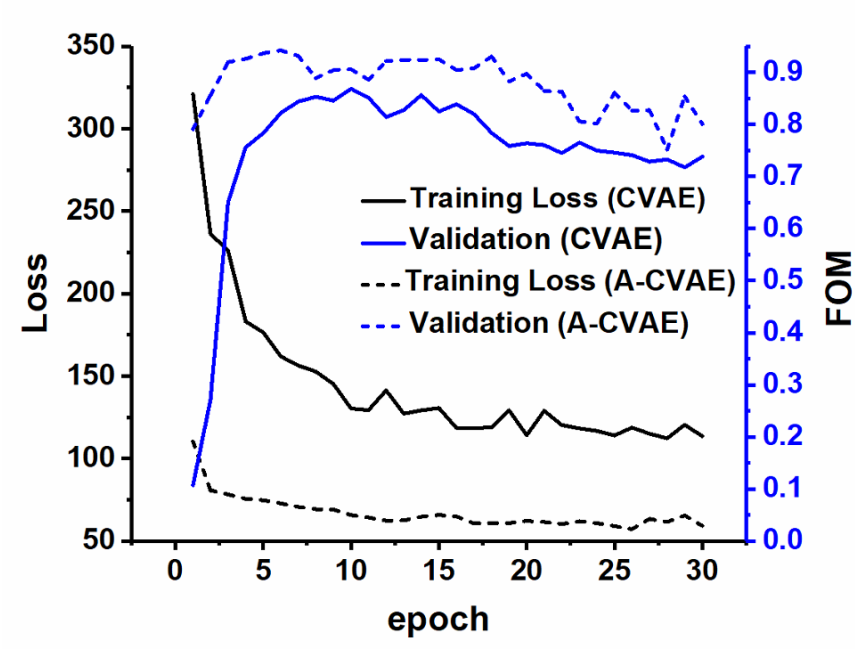

FIGURE 3 The general model performance (training loss and validation result) with different epoch number, for both CVAE and A-CVAE model. Regarding the validation, the FOM is calculated after running FDTD simulation for different generated HV patterns. It shows that the model has generally good performance when the epoch number is between 5 and 15.

The mechanism of applying the trained CVAE model is to feed the trained decoder with the desired condition along with the latent variable following the normal distribution, by which a series of updated designs are generated as the output of the trained decoder. Ideally, the latent variables should obey the normal dis-

tribution $N(0,1)$. When the input pattern information becomes encoded during training of the preliminary model, however, the performance data (transmission spectrum) may also be encoded into the latent variable. This will cause partial clustering to the extracted latent variables (shown in Figure 4 and will result in the degradation of the device performance for the pattern generation because random sampling at latent space can adversely impact the target spectra. To address such a problem of the standard CVAE model, an adversarial block was added as shown in Figure 2b, where a separate branch to an adversary block is used for isolating the latent variable z from the nuisance variations s (the performance data) in order to fit the device distribution better [28, 29, 30]. Figure 2b shows the detailed structure of the A-CVAE network. We use a decoder structure to expand the performance feature s into a 20×20 matrix and then combine with the original 20×20 hole vector to form a 2-channel input, then process it through two convolution layers. One additional step is when the latent z variable is extracted, it will also be fed into an adversarial block to generate $\bar{s} := (\bar{s}_1, \ldots, \bar{s}_n)$. The updated loss function is shown as follows:

$$\begin{aligned} \text{Loss} = &- \sum_{i=1}^{n} \Big[y_i \log x_i + (1 - y_i) \log(1 - x_i) \Big] + \frac{1}{2} \sum_{j=1}^{J} \Big[\mu_{zj}^2 + \sigma_{zj}^2 - \log(\sigma_{zj}^2) - 1 \Big] \\ &- \frac{\beta}{n} \sum_{i=1}^{n} (s_i - \bar{s}_i)^2. \end{aligned} \tag{4}$$

The loss function has three parts. The first is the VAE reconstruction loss, i.e., the Binary Cross-Entropy (BCE) loss and the second is the KL divergence, which are the same as in Equation (2). The last part is a regularization term which is the mean squared error (MSE) loss of the adversarial block. Since the condition information contained in the latent variable z needs to be minimized, the MSE loss between s and $\bar{s}$ needs to be maximized. A complete update of the network generally requires alternating updates in two iteration cycles. The first iteration is used to backpropagate and update the CVAE model based on the loss function stated above. The second iteration is used to update the adversarial block solely based

on the MSE loss between s and $\bar{s}$. In Figure 3, the dashed black line represents the training loss and the dashed blue line represents the trend for validations for the A-CVAE model. The optimum number of epoch is between 5 to 15. During the training, we found when the adversary regularization coefficient β is 5, the generative model gives the best performance. The detailed information about the networks is shown in Table 1.

TABLE 1 The deep learning model parameter comparison

Model	CVAE	A-CVAE	A-CVAE with AL
Encoder	2 Conv, 2 FC	2 Conv, 2 FC	2 Conv, 2 FC
Decoder	2 Conv	2 Conv	2 Conv
Adversarial block	None	2 FC	2 FC
Training samples	approx. 15,000	approx. 15,000	approx 16,000
Input dimension	One channel (20×20)	Two channels (20×20)	Two channels (20×20)
Loss function	Eq. (2): BCE + KLD	Eq. (4): BCE + KLD + MSE	Eq. (5): MSE + KLD + MSE
Ave. FOM	0.771	0.888	**0.901**

AL: Active Learning, FC: Fully Connected Layer
BCE: Binary Cross Entropy, KLD: KL Divergence
MSE: Mean Squared Error

Figure 4a shows the latent variable distribution of the CVAE model for groups of devices with four different types splitting ratios. The original latent variables are in dimension of 63 and the t-distributed Stochastic Neighbor Embedding (t-SNE) method is used to reduce the dimension to 2 for better visualization. This clearly shows that the four groups are clustered, and their centroids are widely distributed. Figure 4b shows the similar plot for the A-CVAE model. The figure clearly shows that with adversarial censoring, all the latent variables are distributed similarly, with centroids almost overlapping. They obey the normal distribution $N(0, 1)$, which is expected. It implies that A-CVAE offers more degrees of freedom to generate different potential devices achieving the target performance by sampling normal latent variables concatenated with desired spectra in the conditional de-

coder. Note that the visualization of multi-dimensional data in a 2D space using the t-SNE is a stochastic process and also depends on the perplexity parameter. We varied the perplexity parameter from 1 to 60, and the trend of much less clustering in the case of A-CVAE is robust and repeatable. We chose the perplexity parameter of 30 in Figure 4.

To further improve the model's performance, we introduce the active learning concept on top of the A-CVAE model. The process is showen in Figure 2c. A preliminary A-CVAE model is first trained using the original binary data. Then, the trained preliminary A-CVAE model is used to generate 1,000 continuous variable hole size patterns with different splitting ratios. The loss function is further modified by changing the Binary Cross-Entropy loss to the MSE loss for the VAE, which is shown as follows:

$$\text{Loss} = \sum_{i=1}^{n}(y_i - x_i)^2 + \frac{1}{2}\sum_{j=1}^{J}\left[\mu_{zj}^2 + \sigma_{zj}^2 - \log(\sigma_{zj}^2) - 1\right] - \frac{\beta}{n}\sum_{i=1}^{n}(s_i - \bar{s}_i)^2. \quad (5)$$

The FDTD results of the 1,000 devices are added as labels, and the data are appended to the training data for the second round. There is a significant boost in terms of performance after we apply the A-CVAE model along with the active learning. The simulation results are shown in the next section.

After training the proposed machine learning model, we test the A-CVAE model by executing it to generate HVs representing the nanopatterned power splitter devices. Since the VAE analyzes the probabilistic distribution for the generated HVs, it makes each generated value of the HV a Bernoulli's distribution. In order to best reflect the result, different hole sizes are used to represent the probability of the appearance of etch holes at certain locations. In order to verify effectiveness of the generative model, we choose four different types of devices with different splitting ratios ($5:5$, $6:4$, $7:3$, $8:2$). Figure 5 shows the comparison of the performance among the devices generated by the CVAE model and the devices generated by the A-CVAE. The FOM is calculated for 20 randomly generated devices from the CVAE models and from the best device in the training data. This figure

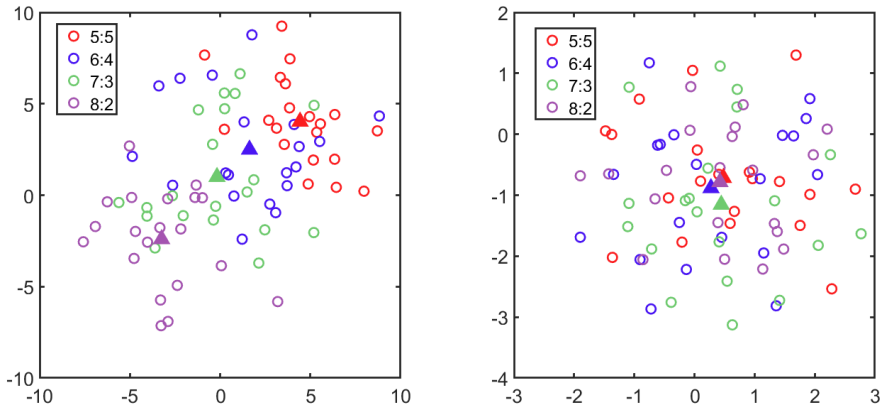

FIGURE 4 t-SNE output of the latent variables. The latent variables are obtained from 4 different types of devices. We use the t-SNE to reduce the dimension to 2 for better visualization. The filled triangle markers are the centroid for each device group. a) Latent variables for the CVAE model, which shows clear clustering. b) Latent variables for the A-CVAE model, where the centroid of the four groups overlap with each other.

shows that the conventional CVAE model can generally learn the distribution of the data, but it cannot beat the training data in terms of performance. With the help of the adversarial censoring, the generated devices generally have a better performance than the training data. The active learning process further improves the performance.

Figure 6 shows the results generated by the A-CVAE model along with the FDTD verification. The solid lines show the transmission and reflection spectra for the four types of devices that are generated by the A-CVAE model. The reflection is lower than -20dB and the achieved transmission is grater than 87% across the broad bandwidth between 1250 nm$-$1800 nm and over 90% transmission between 1500 nm$-$1600 nm. The dashed lines show the transmission and the reflection for the best devices in the training data. As shown in Figure 6, generated devices for $6:4$, $7:3$ and $8:2$ power splitting ratios have significantly improved transmission across a wide range of wavelengths compared to the best training data for those splitting ratios. The generated device for the $5:5$ power splitting ratio is close to the best training data because the training data already include near optimal solution of that specific power splitting ratio.

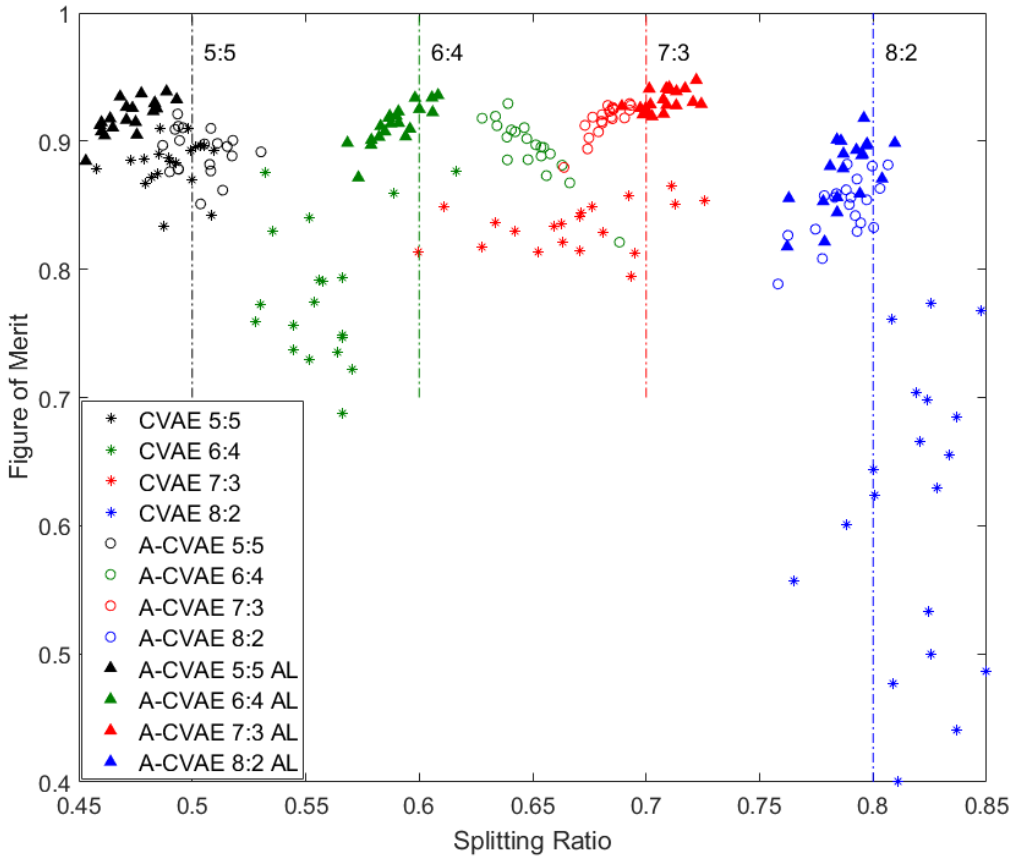

FIGURE 5 FOM comparison for different CVAE models: conventional CVAE (star marker), A-CVAE (round marker) and A-CVAE with active learning (triangle marker). Four different splitting ratios are used as a target value to test the model performance (marked with dashed lines). The devices generated by the active learning assisted CVAE model can fit the target splitting ratio better with excellent total transmission. The average FOM for the three models are: 0.771, 0.888, 0.901, respectively.

The adjoint method is widely used in the inverse design of nanophotonic devices [32, 33, 34]. Given an ideal initial condition for parameters, the optimization process can be done in a small number of iterations (tens of EM simulations). However, the initial condition needs to be carefully chosen in order to get the best optimal result. Our A-CVAE-based method is very different. It is essentially a generative model which is trained from a library (training data) of EM simulation results. They may come from previous imperfect optimization results with multi-

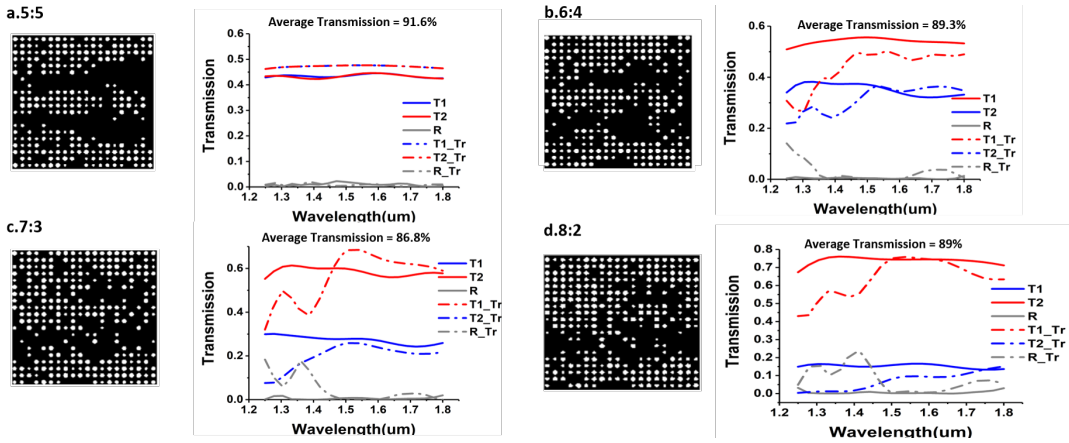

FIGURE 6 FDTD results (transmission and reflection) of the generated patterns via the active learning-assisted A-CVAE model. Since the variable-sized holes are used to express the floating numbers generated by the model, it turns out the generated devices have a much larger bandwidth than the binary training devices (which have the bandwidth of 100nm). After the validation using FDTD simulator, we find the generated devices have the total transmissions above 87% across a 550nm bandwidth also with small reflection (lower than −20dB). The dashed lines show the results of the best training devices around the target splitting ratio.

ple target conditions (splitting ratio and bandwidth, in our case). Then the A-CVAE will try to learn/generalize from the library, and generate a series of improved results for a given condition. Once the model is trained, inverse designs for multiple conditions can be generated in almost no time without any further EM simulations required.

Table 2 provides a comparison of power splitters using different optimization methods, including a more conventional y-junction device optimized by particle swarm optimization (PSO) [35], and nanophotonic splitters designed by the adjoint method [32, 34] and the fast search method [36]. Our work has the most broad bandwidth with very small device footprint and low insertion loss.

Since the network structure is relatively shallow, it significantly reduces the training time for the whole system. The optimizer we used is Adam with the learning rate of 0.001. For all the convolutional layers used, the kernel size is 3 and stride is 1. The batch size that we are using is 128, and optimized epoch number is between 10 to 15 and the total training data is a 15,000 binary hole vector pattern set (further increased to 16,000 for the active learning process). The detailed

TABLE 2 Comparison of simulation results for photonics power splitters using different optimization methods

Split ratio	Insertion loss	Bandwidth	footprint	Reference
1:1	0.09 dB	100 nm	$2 \times 2\ \mu\mathrm{m}^2$	Lalau-Keraly et. al.[32]
1:1	0.32 dB	40 nm	$2.6 \times 2.6\ \mu\mathrm{m}^2$	Wang et. al.[34]
1:1	0.13 dB	80 nm	$1.2 \times 2\ \mu\mathrm{m}^2$	Zhang et. al.[35]
4:6	1 dB (measured)	30 nm	$3.6 \times 3.6\ \mu\mathrm{m}^2$	Xu et. al.[36]
4:6	0.65 dB	550 nm	$2.25 \times 2.25\ \mu\mathrm{m}^2$	this work
3:7	0.51 dB	550 nm	$2.25 \times 2.25\ \mu\mathrm{m}^2$	this work

hyperparameters are included in Table 3. We are using an Nvidia GTX 1080 GPU board and the total training time is around 5 minutes.

TABLE 3 The hyperparameters for the A-CVAE model

Hyperparameter	Value	Description
Input shape	[2,20,20]	Input hole vector shape
Number of Convolution Layers	[2,2]	Convolutional layers for encoder and decoder
Number of Convolution Kernel	[16, 32, 32, 16]	Number of the kernels for Convolutional Layers
Kernel Size	(3,3)	Kernel size for the Convolutional Layers
Number of Max Pooling Layer	[2, 2]	Max polling for encoder and decoder
Kernel Size of Max Pooling Layers	(3,3)	Kernel size of the Max Pooling Layers
Act. function in Convolution Layer	ReLU	Activation function for Convolutional Layer
Fully Connected Layer 1	$[800 \rightarrow 63]$	The fully connected layer in the encoder
Fully Connected Layer 2	$[63 \rightarrow 9]$	Connected to the latent variables
Fully Connected Layer 3	$[63 \rightarrow 400]$	Connected to the input
Fully Connected Layer 4	$[63 \rightarrow 100]$	Fully Connected Layer for the adversarial block
Fully Connected Layer 5	$[100 \rightarrow 63]$	Fully Connected Layer for the adversarial block
Act. function for Fully Connected Layers	ReLU	Activation function for Fully Connected Layer 1 - 5
Optimizer	Adam	The optimizer used for the NN model
Learning rate for CVAE	1e−3	Learning rate for the CVAE model
Learning rate for adversarial block	1e−5	Learning rate for the adversarial block
Batch size	128	Batch size for the training data

To conclude, a state-of-the-art conditional variational autoencoder with adversarial censoring called A-CVAE has been used to design nanopatterned power splitters for photonic integrated circuits. The VAE model takes the binary hole vector as the training sample and can generate patterns with variable hole size. No additional optimization step is needed after the generative model is trained to reproduce well optimized patterns. In addition, with the help of adversarial censoring, the performance of generated patterns was significantly improved (about 5% increase in total transmission). Overall, the devices that are generated through the A-CVAE model described herein show good transmission (with over 90% in total transmission) and a broadband spectral response across the wavelength range of 1250 nm and 1800 nm. To the best of authors' knowledge, this is the first demonstration that an A-CVAE model was applied to a physical design problem. This generative model may be applicable to a wide range of photonic device design problems, including wavelength selective devices.

Conflict of Interest

The authors declare no conflict of interest.

References

[1] D. Mrowca, C. Zhuang, E. Wang, N. Haber, L. F. Fei-Fei, J. Tenenbaum, and D. L. Yamins, in: Advances in Neural Information Processing Systems (NIPS), Montreal, Canada, December 2018, pp. 8799–8810.

[2] E. S. Spelke, K. Breinlinger, J. Macomber, and K. Jacobson, Psychol. Rev. **99**, 605-632 (1992).

[3] J. B. Tenenbaum, C. Kemp, T. L. Griffiths, and N. D. Goodman, Science **331**, 1279–1285 (2011).

[4] M. McCloskey, A. Caramazza, and B.Green, Science **210**, 1139–1141 (1980).

[5] K. A. Smith and E. Vul, Topics in Cognitive Science **5**, 185–199 (2013).

[6] E. S. Spelke, Cognitive Science **14**, 29–56 (1990).

[7] M. B. Chang, T. Ullman, A. Torralba, and J. B. Tenenbaum, arXiv preprint arXiv:1612.00341 (2016).

[8] A. Lerer, S. Gross, and R. Fergus, arXiv, preprint, arXiv:1603.01312 (2016).

[9] P. Battaglia, R. Pascanu, M. Lai, D. J. Rezende, and K. Kavukcuoglu, in: Advances in Neural Information Processing Systems (NIPS), Barcelona, Spain, December 2016, pp. 4502–4510.

[10] A. D. Sendek, E. D. Cubuk, E. R. Antoniuk, G. Cheon, Y. Cui, and E. J. Reed, arXiv, preprint, arXiv:1808.02470 (2018).

[11] R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik, ACS Central Science **4**, 268–276 (2018).

[12] J. Ghaboussi, J. H. Garrett Jr, and X. Wu, J. Eng. Mech **117**, 132–153 (1991).

[13] A. Radovic, M. Williams, D. Rousseau, M. Kagan, D. Bonacorsi, A. Himmel, A. Aurisano, K. Terao, and T. Wongjirad, Nature **560**, 41-48 (2018).

[14] M. H. Tahersima, K. Kojima, T. Koike-Akino, D. Jha, B. Wang, C. Lin, and K. Parsons, Sci. Rep. **9**, 1368 (2019).

[15] D. Liu, Y. Tan, E. Khoram, and Z. Yu, ACS Photonics **5**, 1365-1369 (2018).

[16] W. Ma, F. Cheng, and Y. Liu, ACS Nano **12**, 6326–6334 (2018).

[17] I. Malkiel, M. Mrejen, A. Nagler, U. Arieli, L. Wolf, and H. Suchowski, in: 2018 IEEE International Conference on Computational Photography (ICCP), Pittsburg, USA, May 2018, pp. 1–14.

[18] J. Peurifoy, Y. Shen, L. Jing, Y. Yang, F. Cano-Renteria, B. G. DeLacy, J. D. Joannopoulos, M. Tegmark, and M. Soljačić, Sci. Adv. **4**, eaar4206 (2018).

[19] Y. Sun, Z. Xia, and U. S. Kamilov Opt. Express **26**, 14678–14688 (2018).

[20] Z. Liu, D. Zhu, S. P. Rodrigues, K. -T. Lee, and W. Cai, Nano Lett. **18**, 6570–6576 (2018).

[21] T. Asano and S. Noda, Opt. Express **26**, 32704–32717 (2018).

[22] A. M. Hammond and R. M. Camacho, Opt. Express **27**, 29620–29638 (2019).

[23] W. Ma, F. Cheng, Y. Xu, Q. Wen, and Y. Liu, arXiv preprint arXiv:1901.10819 (2019).

[24] K. Kojima, B. Wang, U. Kamilov, T. Koike-Akino, and K. Parsons, in: Integrated Photonics Research, Silicon and Nanophotonics (IPR), New Orleans, USA, July 2017, paper ITu1A–4.

[25] M. Teng, K. Kojima, T. Koike-Akino, B. Wang, C. Lin, and K. Parsons, in: Optical Fiber Communications Conference and Exposition (OFC), San Diego, USA, March 2018, paper Th2A.8.

[26] S. An, B. Zheng, H. Tang, M. Y. Shalaginov, L. Zhou, H. Li, T. Gu, J. Hu, C. Fowler, and H. Zhang, arXiv preprint arXiv:1908.04851 (2019).

[27] K. Sohn, H. Lee, and X. Yan, in: Advances in Neural Information Processing Systems (NIPS), Montreal, Canada, December 2015, 3483–3491.

[28] Y. Wang, T. Koike-Akino, and D. Erdogmus, arXiv preprint arXiv:1805.08097 (2018).

[29] G. Lample, N. Zeghidour, N. Usunier, A. Bordes, L. Denoyer, and M. Ranzato, in: Advances In Neural Information Processing Systems (NIPS), Long Beach, USA, December 2017, pp. 5967–5976.

[30] O. Özdenizci, Y. Wang, T. Koike-Akino, and D. Erdoğmuş, in: 99th International IEEE/EMBS Conference on Neural Engineering (NER), San Francisco, USA, March 2019, pp. 207–210.

[31] M. H. Tahersima, K. Kojima, T. Koike-Akino, D. Jha, B. Wang, C. Lin, and K. Parsons, in: CLEO:Science and Innovations, San Jose, USA, May 2019, paper SW4J–6.

[32] C. M. Lalau-Keraly, S. Bhargava, O. D. Miller, and E. Yablonovitch, Opt. Express **21**, 21693–21701 (2013).

[33] A. Y. Piggott, J. Lu, K. G. Lagoudakis, J. Petykiewicz, T. M. Babinec, and J. Vučković, Nat. Photonics **6**, 374–377 (2015).

[34] K. Wang, X. Ren, W. Chang, L. Liu, D. Liu, and M. Zhang, Photonics Res. **8**, 528–533 (2020).

[35] Y. Zhang, S. Yang, A. E. -J. Lim, G. -Q. Lo, C. Galland, T. Baehr-Jones, and M.Hochberg, Opt. Express **21**, 1310–1316 (2013).

[36] K. Xu, L. Liu, X. Wen, W. Sun, N. Zhang, N. Yi, S. Sun, S. Xiao, and Q. Song, Opt. Lett. **42**, 855–858 (2017).