Research Article | Vol. 26, No. 25 | 10 Dec 2018 | OPTICS EXPRESS 32565
Optics EXPRESS

Model-driven convolution neural network for inverse lithography

XU MA,[1,3] QILE ZHAO,[1] HAO ZHANG,[1] ZHIQIANG WANG,[1] AND GONZALO R. ARCE[2,4]

[1]*Key Laboratory of Photoelectronic Imaging Technology and System of Ministry of Education of China, School of Optics and Photonics, Beijing Institute of Technology, Beijing, 100081, China*

[2]*Department of Electrical and Computer Engineering, University of Delaware, Newark, DE, 19716, USA*
[3]*maxu@bit.edu.cn*
[4]*grmaple2015@gmail.com*

Abstract: Optical lithography is a fundamental process to fabricate integrated circuits, which are the basic fabric of the information age. Due to image distortions inherent in optical lithography, inverse lithography techniques (ILT) are extensively used by the semiconductor industry to improve lithography image resolution and fidelity in semiconductor fabrication. As the density of integrated circuits increases, computational complexity has become a central challenge in ILT methods. This paper develops a new and powerful framework of a model-driven convolution neural network (MCNN) to obtain the approximate guess of the ILT solutions, which can be used as the input of the following ILT optimization with much fewer iterations as compared with conventional ILT algorithms. The combined approach to use the proposed MCNN together with the gradient-based method can improve the speed of ILT optimization algorithms up to an order of magnitude and further improve the imaging performance of coherent optical lithography systems. This paper, to the best of our knowledge, is the first to exploit a state-of-the-art MCNN to solve the ILT problem and provide considerable performance advantages. The imaging model of optical lithography is utilized to establish a neural network architecture and an unsupervised training strategy. The neural network architecture and the initial network parameters are derived by unfolding and truncating the model-based iterative ILT optimization procedure. Then, a model-based decoder is proposed to enable the unsupervised training of the neural network, which averts the time-consuming labelling process in training data. This work opens a new window for MCNN techniques to effectively improve the computational efficiency and imaging performance of conventional ILT algorithms. Some impressive good simulation results are provided to verify the superiority of the proposed MCNN approach.

1. Introduction

Optical lithography is a fundamental process in the fabrication of very-large scale integrated circuits. According to Moore's law, the number of transistors included in a dense integrated circuit doubles approximately every two years [1]. Most of the very-large scale integrated circuits are fabricated by the optical lithography systems. Figure 1(a) depicts a sketch of a typical optical lithography system. The light source illuminates deep ultraviolet light rays, which are transmitted through the mask and projector, to then replicate the layout pattern from the mask onto the wafer. The wafer is coated with light-sensitive photoresist. After the development of photoresist, the print image appears on the surface of wafer.

However, the printed patterns are invariably distorted in optical lithography due to diffraction limits, mutual interference and so on. Especially, the image distortion becomes increasingly pronounced as the critical dimension of integrated circuits continuously shrinks, such that the dense layout patterns in critical layers cannot even be resolved. To date, the semiconductor industry has heavily relied on computational lithography to resolve the densely-packed layout features

#342103 https://doi.org/10.1364/OE.26.032565
Journal © 2018 Received 13 Aug 2018; revised 14 Nov 2018; accepted 15 Nov 2018; published 28 Nov 2018

Optics EXPRESS

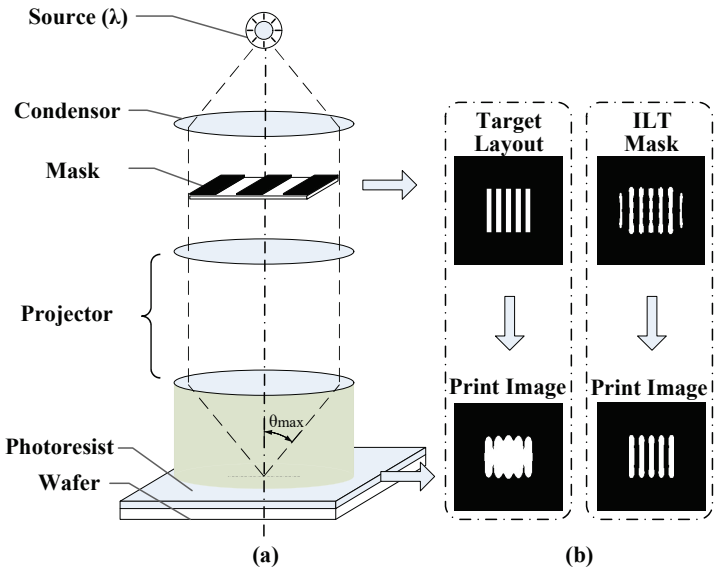

Fig. 1. (a) Sketch of an optical lithography system and (b) the ILT method where the mask is predistorted to compensate for the optical distortion of the system (revised from Fig. 1 in [11]).

and improve the yield of semiconductor devices [2, 3]. Computational lithography encompasses many mathematical and algorithmic methods to improve the lithography imaging performance by individually or jointly optimizing the mask, lithography tools and process parameters. Inverse lithography techniques (ILT) are extensively used in computational lithography methods to improve the resolution and image fidelity of lithography systems. As illustrated in Fig. 1(b), ILT pre-warps the mask pattern by inverting the lithography imaging model to compensate for the image distortions attributed to the optical proximity effect [4–9]. In particular, ILT grids the mask pattern into pixels, and optimizes the transmittances of all mask pixels by solving for the inverse lithography problems based on the imaging model of lithography systems [10–16]. Due to the high degrees of optimization freedom, these methods can flexibly modulate the electric field, and effectively improve the imaging performance. However, ILTs need to handle a large volume of data during optimization, which poses a big challenge on its computational efficiency. In the past, a set of efficient ILT algorithms have been proposed. Some of these approaches use gradient-based numerical optimization algorithms [17–21]. Recently, traditional machine learning techniques, such as neural network [22], support vector machine (SVM) [23], and nonparametric kernel regression [24, 25], have been used to improve the speed of ILTs. Conventional machine learning techniques, however, have their own inherent limitations. Neural networks consist of simple unbranched feedforward structures with only one or a few hidden layers. However, ILT is a nonlinear inverse optimization problem with numerous local minima. Based on limited training samples, conventional neural networks sometimes are inadequate to precisely synthesize the complex ILT process. The same problem happens with SVMs, which attempt to divide the solution space by a hyperplane and reduce the ILT problem to a classification problem. On the other hand, nonparametric kernel regression predicts the behavior of ILT by the kernel-based average on a set of neighboring training samples. Nonparametric kernel regression, however, needs a large volume of training data to support the model estimate. Thus, this approach suffers from a time-consuming training stage and requires large memory to store the training data library.

Optics EXPRESS

In the past few years, deep learning has emerged as a new research area in the machine learning realm [26–28]. As a significant branch of deep neural networks, convolution neural networks (CNN) have gained considerable interest in a number of applications [29–32]. CNNs typically consist of one or more convolution layers that are customized to process data sets. CNNs have significant advantages in reducing the complexity of network structure and training procedure by sharing the convolution weights over the entire data sets. The weight sharing strategy is also beneficial to alleviate the overfitting problem compared to the fully connected networks. Notably, deep learning has been introduced to efficiently solve the inverse sparse coding and compressive sensing problems, where the trained deep networks are used to emulate the unfolded iterative optimization process [33–37].

Inspired by the works mentioned above, this paper develops a model-driven CNN (MCNN) to solve for the ILT problem in lithography systems. To the best of our knowledge, this is the first paper focusing on model-driven CNN to address computational lithography. This paper focuses on coherent optical lithography systems, however, the proposed approaches can be generalized to the partially coherent lithography systems in the future. Typical CNN used to date includes three kinds of basic layers, i.e., the convolutional layer, a pooling layer and a fully connected layer. The structure and the parameters of the CNN could be initialized in heuristic and empirical manners. In a broad sense, the MCNN proposed in this paper belongs to a special category of CNN. The common characteristic of the MCNN and the conventional CNN is that they both use convolution operations, linear connections and nonlinear activations to extract the information of the data in a sequence of network layers, to then transform the input data to the expected output data. However, one of the merits of the MCNN approach is that the network structure and parameters can be derived and initialized from the underlying data model, which may result in networks falling well outside the realm of known design forms. The initialization method for the structure and parameters of the MCNN framework are model-adaptive, which are more systematic compared to random or heuristic initialization methods. The contribution of this paper includes two aspects. The first is to build up the architecture of MCNN based on the lithography imaging model, while the other is to develop an unsupervised training strategy for the proposed MCNN framework. This is, the lithography imaging model is exploited twice, for constructing the network and for the unsupervised training.

The launching point of the proposed MCNN is that the existing gradient-based ILT methods inversely optimize the mask patterns based on the lithography imaging model and iterative optimization algorithms. The inverse optimization can be unfolded into a series of iterations, which resembles a deep learning process with each iteration analogous to one layer in the deep network. In general, gradient-based ILT algorithms need many iterations to obtain the optimal mask patterns. To control the complexity of CNN, this paper truncates the optimization process after a few iterations. Then, the convolution kernels in the network are jointly trained using the back-propagation algorithm based on a set of representative layout patterns [38]. Since the proposed method is derived from the model-based ILT algorithm, we refer to the proposed method as model-driven CNN. Benefiting from the guidance of the imaging model and the training process, the MCNN can emulate the outputs of many iterations using only a few convolution layers. This can be explained from the perspective of optimization. Most convex optimization algorithms find out the search direction in each iteration based on the gradient of cost function. But, the gradient of the cost function varies with the observation point (i.e., the point in space represented by the optimization variables at the current iteration). Thus, gradient-based algorithms need to iterate many times to gradually approximate the observation point towards the optimal solution. On the other hand, the MCNN approach learns the optimal convolution kernels from training data, so that each convolution layer provides an optimal search direction regardless of the observation point. Thus, the effect of one convolution layer in MCNN is somehow equivalent to many iterations in gradient-based algorithms. After the MCNN is well trained, the output of the MCNN

Optics EXPRESS

is used as the approximate guess of the solution to the ILT problem. Then, the gradient-based ILT algorithm with much fewer iterations than the conventional ILT algorithms can be used to further optimize the output of MCNN to obtain the final optimized mask pattern. In this paper, the simulations prove that the proposed MCNN method can effectively improve on both the computational efficiency of the inverse algorithm, and the lithography imaging performance. It is noted that the number of convolution layers used should depend on the amount of training data and the complexity and dimension of the layout pattern to be optimized. This paper presents some examples to prove the validity of the proposed MCNN approach using a small set of training data and some representative mask features. However, the depth and complexity of the MCNN should be scaled with the training data volume and the dimension of the mask being constructed.

In addition, this paper develops an unsupervised training strategy for the proposed MCNN that avoids the need of labelling in the training samples. The basic idea is to regard the proposed MCNN as an encoder to translate training layouts into ILT solutions. Thus, if the outputs of MCNN are used as the mask patterns, the print images of lithography systems should be very close to the corresponding training layouts. That means the MCNN can be treated as the reversal of imaging process. In this paper, we add a decoder after the MCNN to calculate the print images of the network outputs based on the lithography imaging model. Then, the objective of the training stage is to minimize the distance between the input of the encoder and the output of the decoder. Based on this method, we can avoid labelling the training samples, which need to pre-calculate the ILT solutions for all training layouts. Another advantage of this training strategy is to prevent inappropriate labeled data for the training samples. It is known that the inverse lithography problem is ill-posed, such that different mask patterns may lead to the same print image. Therefore, it is difficult to label a unique optimal ILT solution for each training layout beforehand. The proposed unsupervised training method is blind to the intermediate outputs between the encoder and decoder. That means we do not need to solve the ill-posed ILT problem for every training layout, but to optimize the network parameters to obtain end-to-end matching between the encoder and decoder.

The remainder of this paper is organized as follows. The fundamentals of ILT modelling in coherent lithography systems are described in Section 2. The proposed MCNN approaches is proposed in Section 3 to solve for the ILT problem. Simulations and analysis are provided in Section 4. Section 5 concludes the paper.

2. ILT modelling in coherent lithography system

The imaging process of coherent lithography systems is illustrated in Fig. 2. Assume $\mathbf{M} \in \mathbb{R}^{N\times N}$ is an $N \times N$ binary matrix representing the mask pattern, where the zero-valued and one-valued pixels in $\mathbf{M}$ indicate the opaque and transparent mask regions. According to the Hopkins diffraction model, the aerial image of coherent lithography systems is formulated as [3, 39, 40]

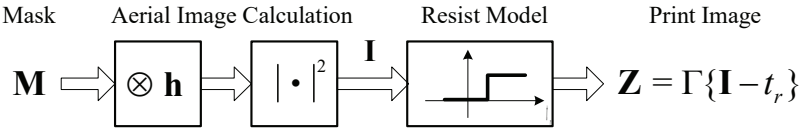

Fig. 2. The imaging process of coherent optical lithography systems.

$$\mathbf{I} = |\mathbf{h} \otimes \mathbf{M}|^2, \tag{1}$$

Optics EXPRESS

where $\mathbf{I}$ is the aerial image, $\otimes$ is the convolution operation, and $\mathbf{h}$ is the point spread function given by

$$\mathbf{h} = \frac{J_1(2\pi r \mathrm{NA}/\lambda)}{2\pi r \mathrm{NA}/\lambda}, \tag{2}$$

where NA is the numerical aperture, λ is the wavelength of DUV light, r is the distance from the center of $\mathbf{h}$, and $J_1(\cdot)$ is the Bessel function of first kind. In this paper, we normalize $\mathbf{h}$ to have unit energy. The aerial image is the light intensity distribution exposed on the photoresist. After the development of photoresist, the layout image will be printed on the wafer. According to the constant threshold resist model, the print image $\mathbf{Z}$ can be calculated as [3]

$$\mathbf{Z} = \Gamma\{\mathbf{I} - t_r\} = \Gamma\{|\mathbf{h} \otimes \mathbf{M}|^2 - t_r\}, \tag{3}$$

where $\Gamma\{x\} = 1$ if $x > 0$, otherwise $\Gamma\{x\} = 0$. t_r is the threshold of photoresist. So, if the exposed intensity exceeds t_r, the image will be printed out. Otherwise, the image will disappear from the wafer. In order to use the gradient-based algorithm to solve for the ILT problem, we use the sigmoid function to approximate the hard threshold function in Eq. (3), since the sigmoid function is differentiable [8, 11]. Then, Eq. (3) is adjusted as follows

$$\mathbf{Z} = \mathrm{sig_r}(\mathbf{I}, t_r) = \frac{1}{1 + \exp[-a_r(\mathbf{I} - t_r)]}, \tag{4}$$

where a_r is the parameter to control the steepness of sigmoid function. The sigmoid function with larger a is more close to the hard threshold function.

Assume $\tilde{\mathbf{Z}}$ is the target layout. The goal of the ILT approaches is to find the optimal mask pattern to minimize the difference between the actual print image $\mathbf{Z}$ and the target layout $\tilde{\mathbf{Z}}$. Thus, the cost function of the ILT problem can be formulated as

$$F = \left\|\tilde{\mathbf{Z}} - \mathbf{Z}\right\|_2^2 = \left\|\tilde{\mathbf{Z}} - \mathrm{sig_r}(\mathbf{I}, t_r)\right\|_2^2, \tag{5}$$

where $\|\cdot\|_2^2$ is the l_2-norm. Thus, the ILT optimization problem is formulated as

$$\hat{\mathbf{M}} = \arg\min_{\mathbf{M}} F, \tag{6}$$

where $\hat{\mathbf{M}}$ represents the optimal mask pattern, and F is given by Eq. (5). We can use the gradient-based algorithm to solve for the problem in Eq. (6), where the mask pattern is iteratively updated as following [7, 8]:

$$\mathbf{M}^{n+1} = \mathbf{M}^n - \mathrm{step} \cdot \nabla F, \tag{7}$$

where $\mathbf{M}^{n+1}$ and $\mathbf{M}^n$ are the masks in the $(n+1)$th and nth iterations, and ∇F is the gradient of cost function with respect to mask. According to Eqs. (4) and (5), we have

$$\nabla F = -2a_r \cdot \left\{\mathbf{h}^\circ \otimes \left[(\tilde{\mathbf{Z}} - \mathbf{Z}) \odot \mathbf{Z} \odot (\mathbf{1} - \mathbf{Z}) \odot (\mathbf{h} \otimes \mathbf{M})\right]\right\}, \tag{8}$$

where $\odot$ is the element-by-element multiplication, $\mathbf{h}^\circ$ means to rotate matrix $\mathbf{h}$ by 180° in both horizontal and vertical directions, $\mathbf{1} \in \mathbb{R}^{N \times N}$ is the matrix of one, and $\mathbf{Z}$ is described in Eq. (4). The flowchart of the gradient-based ILT algorithm is shown in Fig. 3. According to Eq. (8), we can define the operations in Fig. 3 as following:

$$\mathbf{S} = \delta(x, y), \quad \mathbf{D} = \mathbf{h}, \quad \mathbf{T} = (\tilde{\mathbf{Z}} - \mathbf{Z}) \odot \mathbf{Z} \odot (\mathbf{1} - \mathbf{Z}), \quad \mathbf{W} = 2a \cdot \mathrm{step} \cdot \mathbf{h}^\circ, \tag{9}$$

where $\delta(x, y)$ is the Dirac delta function.

Optics EXPRESS

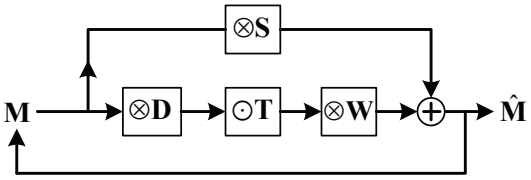

Fig. 3. The flowchart of the gradient-based ILT algorithm.

3. MCNN for the ILT problem

This section first derives the architecture of MCNN from the ILT model, and then develops an unsupervised training method for the proposed MCNN.

3.1. Architecture of MCNN

According to Fig. 3, we can unfold the gradient-based ILT algorithm into a series of iterations. As shown in Fig. 4(a), we truncate the unfolded loop after K iterations. Then, the unfolded gradient-based ILT algorithm can be modelled as an MCNN, where each iteration serves as one layer of the MCNN. The output of one layer is used as the input of the next layer. The initial input of MCNN is denoted by $\mathbf{M}_1^b \in \mathbb{R}^{N\times N}$, which is equal to the target layout $\tilde{\mathbf{Z}}$. The final output of MCNN is the optimized mask pattern denoted by $\hat{\mathbf{M}}$. That means if we input a target layout into the MCNN, the output will be the corresponding optimized mask pattern.

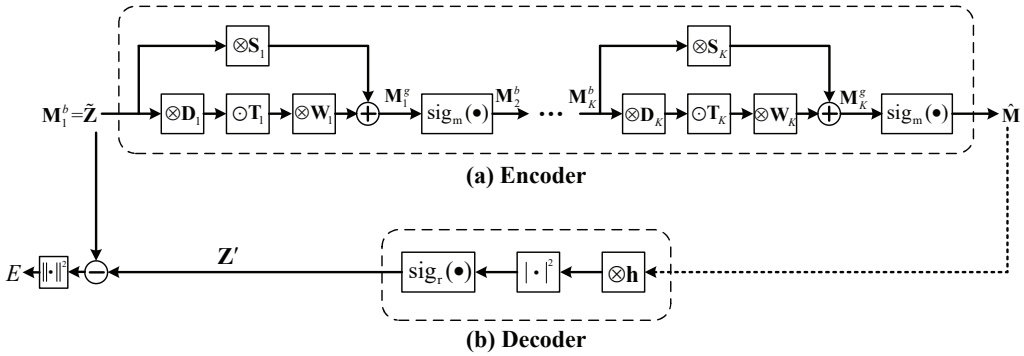

Fig. 4. (a) The architecture of MCNN, and (b) the decoder used to train the MCNN. The architecture of MCNN is generated by truncating the unfolded loop of the gradient-based ILT algorithm, where the input is the target layout, and the output is the optimized mask pattern. The decoder is set up according to the lithography imaging model and photoresist model, where the output of the decoder is expected to be close to the target layout.

Now, let's consider the kth ($k = 1, 2, \ldots, K$) layer of the MCNN. The input data is an approximate binary mask $\mathbf{M}_k^b \in \mathbb{R}^{N\times N}$ with its elements close to 0 or 1. $\mathbf{M}_k^b$ is the intermediate result during the learning process. Then, the output of the kth layer is calculated as

$$\mathbf{M}_k^g = \mathbf{S}_k \otimes \mathbf{M}_k^b + \mathbf{W}_k \otimes \left[\mathbf{T}_k \odot (\mathbf{D}_k \otimes \mathbf{M}_k^b)\right], \tag{10}$$

where $\mathbf{S}_k \in \mathbb{R}^{N_f\times N_f}$, $\mathbf{W}_k \in \mathbb{R}^{N_f\times N_f}$ and $\mathbf{D}_k \in \mathbb{R}^{N_f\times N_f}$ are the convolution kernels that can be optimized based on the training samples. N_f is the lateral dimension of the convolution kernels. $\mathbf{T}_k \in \mathbb{R}^{N\times N}$ is the transmission matrix defined as

$$\mathbf{T}_k = \mathbf{T} = (\tilde{\mathbf{Z}} - \mathbf{Z}) \odot \mathbf{Z} \odot (\mathbf{1} - \mathbf{Z}), \tag{11}$$

Optics EXPRESS

where $\mathbf{T}$ is described in Eq. (9). Next, we will interpret the meaning of $\mathbf{T}_k$. In Eq. (11), the term $\tilde{\mathbf{Z}} - \mathbf{Z}$ is referred to as the difference pattern, which represents the regions where the actual print image $\mathbf{Z}$ is different from the target layout $\tilde{\mathbf{Z}}$. In fact, one goal of ILT is to minimize or even eliminate the difference pattern between $\tilde{\mathbf{Z}}$ and $\mathbf{Z}$. The term $\mathbf{Z} \odot (\mathbf{1} - \mathbf{Z})$ extracts the contours of the print image. Thus, the matrix $\mathbf{T}$ represents the overlapped regions between the difference pattern and the contour of print image. In order words, $\mathbf{T}_k$ identifies the boundary regions of $\mathbf{Z}$ that are not on target. It is natural for ILT algorithms to iteratively modify the mask pixels supported by $\mathbf{T}_k$, since these mask pixels have important influence on the pattern error.

It is noted that the pixel values on the binary mask should be equal to 0 or 1. However, the output $\mathbf{M}_k^g$ in Eq. (10) is a grey-scaled mask pattern. Thus, at the end of each layer we use the sigmoid function to transform $\mathbf{M}_k^g$ to the approximate binary mask $\mathbf{M}_{k+1}^b$:

$$\mathbf{M}_{k+1}^b = \mathrm{sig_m}(\mathbf{M}_k^g, t_m) = \frac{1}{1 + \exp[-a_m(\mathbf{M}_k^g - t_m)]}, \tag{12}$$

where a_m is the steepness index, and the threshold $t_m = 0.5$ for the binary mask. The output of the sigmoid function is then used as the input data for the $(k+1)$th layer. It is noted that the sigmoid function serves as the activation function in the MCNN. Only the mask pixels larger than the threshold can activate the neural node, and transmit the signals to the next layer. The final output of MCNN is $\hat{\mathbf{M}}$, which is calculated by the sigmoid function. Thus, $\hat{\mathbf{M}}$ is still an approximate binary mask. When the MCNN is used to predict the ILT solutions, we need to threshold $\hat{\mathbf{M}}$ to get the binary mask pattern.

It is worth noting that the proposed MCNN framework can be generalized to lithography system with partially coherent illuminations. The partially coherent illumination has finite physical size, and can be discretized into a set of point sources. According to the Abbe's imaging model, the aerial image of the partially coherent lithography system can be formulated as [11, 15]

$$\mathbf{I} = \sum_{x_s, y_s} \mathbf{J}(x_s, y_s) \, |\mathbf{h}^{x_s y_s} \otimes \mathbf{M}|^2, \tag{13}$$

where $\mathbf{J}(x_s, y_s)$ is the light intensity of the point source at coordinate (x_s, y_s), and $\mathbf{h}^{x_s y_s}$ is the point spread function corresponding to the source point (x_s, y_s). Compared to Eq. (1), the aerial image of the partially coherent lithography system is equal to the summation of the aerial images contributed to a set of coherent lithography systems. Accordingly, the gradient of the cost function in Eq. (8) is modified as

$$\nabla F = -2a_r \cdot \sum_{x_s, y_s} \mathbf{J}(x_s, y_s) \left\{ (\mathbf{h}^{x_s y_s})^\circ \otimes \left[(\tilde{\mathbf{Z}} - \mathbf{Z}) \odot \mathbf{Z} \odot (\mathbf{1} - \mathbf{Z}) \odot (\mathbf{h}^{x_s y_s} \otimes \mathbf{M}) \right] \right\}. \tag{14}$$

Similar to the coherent lithography system, we can use the MCNN approach to emulate the gradient-based optimization under partially coherent illuminations. However, Eq. (14) is the summation of several convolution terms. Thus, each layer of the MCNN for partially coherent lithography system should include several parallel branches with different convolution kernels. Using partially coherent illuminations will increase the complexity of the network structure and the training stage. The MCNN approaches for the partially coherent lithography systems will be studied in future work.

3.2. Unsupervised training method for MCNN

The training process usually has a significant impact on the performance of deep networks. This section develops an unsupervised training method for the proposed MCNN. In the conventional supervised training manner, we need to select a number of representative layout patterns as the training samples, and then pre-calculate their ILT solutions as the labeled output data. The

labelling process will cost an amount of computation resources, and may lead to inappropriate labeled data. That is because ILT is an ill-posed problem. Given a target layout, we may find out several different mask patterns leading to the same print image. Thus, it is hard to determine which optimized mask is the most suitable labeled data for that sample layout. In order to overcome this limitation, this paper develops an unsupervised training strategy for the proposed MCNN to get rid of the labelling process.

As shown in Fig. 4, the proposed MCNN is regarded as an encoder to translate the training layout $\mathbf{M}_1^b$ into the ILT solution $\hat{\mathbf{M}}$. The print image corresponding to $\hat{\mathbf{M}}$ should be approximately equal to the target layout $\mathbf{M}_1^b$. Therefore, the MCNN synthesizes the inverse process of the imaging formation, and vice verse. Based on this idea, we add a decoder after the MCNN as shown in Fig. 4(b). The input of decoder is the output of MCNN. The decoder calculates the print image of $\hat{\mathbf{M}}$ based on Eqs. (1) and (4). The output of decoder is given by

$$\mathbf{Z}' = \mathrm{sig}_{\mathrm{r}}\left(|\mathbf{h} \otimes \hat{\mathbf{M}}|^2, t_r\right) = \frac{1}{1 + \exp\left[-a_r\left(|\mathbf{h} \otimes \hat{\mathbf{M}}|^2 - t_r\right)\right]}. \tag{15}$$

Then, the goal of the training process is to find out the optimal convolution kernels $\mathbf{S}_k$, $\mathbf{D}_k$ and $\mathbf{W}_k$ $(k = 1, 2, \ldots, K)$ to minimize the difference between the encoder input and decoder output, i.e.,

$$\left\{\hat{\mathbf{S}}_k, \hat{\mathbf{D}}_k, \hat{\mathbf{W}}_k\right\} = \arg \min_{\mathbf{S}_k, \mathbf{D}_k, \mathbf{W}_k} E = \arg \min_{\mathbf{S}_k, \mathbf{D}_k, \mathbf{W}_k} \|\mathbf{M}_1^b - \mathbf{Z}'\|_2^2. \tag{16}$$

On the other hand, we would like to constrain the MCNN output to be as close to a binary mask as possible. In order to achieve this goal, we add a quadratic penalty term to the cost function [11]. The quadratic penalty is defined as following

$$R_Q = \mathbf{1}_{N\times 1}^T \cdot 4\hat{\mathbf{M}} \odot (\mathbf{1} - \hat{\mathbf{M}}) \cdot \mathbf{1}_{N\times 1}, \tag{17}$$

where $\hat{\mathbf{M}}$ is the output of MCNN, and $\mathbf{1}_{N\times 1}$ is the one-valued vector with dimension of $N \times 1$. Thus, Eq. (16) is modified as

$$\left\{\hat{\mathbf{S}}_k, \hat{\mathbf{D}}_k, \hat{\mathbf{W}}_k\right\} = \arg \min_{\mathbf{S}_k, \mathbf{D}_k, \mathbf{W}_k} E' = \arg \min_{\mathbf{S}_k, \mathbf{D}_k, \mathbf{W}_k} \left\{\|\mathbf{M}_1^b - \mathbf{Z}'\|_2^2 + \gamma_Q \cdot R_Q\right\}, \tag{18}$$

where γ_Q is the weight of quadratic penalty. According to Eq. (18), we only need to drop a set of training layouts into the MCNN, and the MCNN will be automatically trained in an unsupervised manner.

This paper applies the back-propagation algorithm to solve for the training problem in Eq. (18) [38]. First, the convolution kernels are initialized as in Eq. (9). Different from conventional CNN methods, the proposed MCNN provides a systematic initialization approach. The initial values of the convolution kernels are determined by the model-based ILT framework, rather than random or heuristic manners. Other initialization methods of the network convolution kernels could be developed and is a topic of future research. Then, we calculate the gradients of the cost function with respect to the convolution kernels. According to the Chain Rule, we can calculate the partial derivatives of the cost function to the elements of $\mathbf{S}_k$, $\mathbf{W}_k$ and $\mathbf{D}_k$, respectively. For simplicity, assume the matrix $\mathbf{A}_k = \mathbf{S}_k$, $\mathbf{W}_k$ or $\mathbf{D}_k$, then the partial derivatives are formulated as following

$$\begin{aligned}\frac{\partial E'}{\partial \mathbf{A}_k(x,y)} = \sum_{\hat{x},\hat{y}} \Bigg(\frac{\partial E'}{\partial \hat{\mathbf{M}}(\hat{x},\hat{y})} \cdot \sum_{x_K, y_K} \Bigg(\frac{\partial \hat{\mathbf{M}}(\hat{x},\hat{y})}{\partial \mathbf{M}_K^g(x_K, y_K)} \cdots \\ \sum_{x_k, y_k} \left(\frac{\partial \mathbf{M}_{k+1}^g(x_{k+1}, y_{k+1})}{\partial \mathbf{M}_k^g(x_k, y_k)} \cdot \frac{\partial \mathbf{M}_k^g(x_k, y_k)}{\partial \mathbf{A}_k(x,y)} \right) \cdots \Bigg) \Bigg),\end{aligned} \tag{19}$$

Optics EXPRESS

where (x, y), $(\hat{x}, \hat{y})$, (x_K, y_K), (x_{k+1}, y_{k+1}) and (x_k, y_k) are the coordinates. Appendix A provides the detailed method to calculate the derivatives in Eq. (19). After that, the back-propagation algorithm iteratively updates the convolution kernels as follows:

$$\mathbf{A}_k^{n+1} = \mathbf{A}_k^n - \text{step}_A \cdot \nabla E'|_{\mathbf{A}}, \tag{20}$$

where $\nabla E|_{\mathbf{A}}$ is the gradient of cost function to $\mathbf{A}$, and step_A is the step size. The update in Eq. (20) may results in asymmetric and unnormalized convolution kernels. Thus, after each iteration we enforce the convolution kernels to be symmetric along the x-axis and y-axis. Then, we normalize the convolution kernels by its l_2-norm such that $\mathbf{A}_k^{n+1} = \mathbf{A}_k^{n+1}/\|\mathbf{A}_k^{n+1}\|_2$. After the training process, $\hat{\mathbf{S}}_k$, $\hat{\mathbf{D}}_k$, and $\hat{\mathbf{W}}_k$ are used as the convolution kernels, and the decoder is removed from the MCNN. In order to obtain the binary optimized mask patterns, the last sigmoid function in MCNN will be replaced by the hard threshold function.

4. Simulation and analysis

This section provides the simulations to verify the advantage in proposed MCNN approach. In Section 4.1, we train and test the MCNN based on a set of layout patterns at 90nm and 45nm technology nodes. In Section 4.2, the MCNN method is compared to the traditional gradient-based ILT algorithm. It is noted that this paper focuses on the coherent lithography system, where the imaging resolution is limited by the on-axis illumination. Currently, the semiconductor industry uses the partially coherent illuminations to further enhance the lithography resolution up to 14-7nm and beyond. Nevertheless, the principles and methodologies proposed in this paper can be generalized into partially coherent lithography systems and used to resolve finer patterns with more critical sizes.

4.1. Training and test of MCNN

This section provides the simulations of the proposed MCNN approach at 90nm and 45nm technology nodes. In these simulations, we use a deep ultra-violet coherent lithography system with the wavelength of $\lambda = 193$nm. The numerical aperture is set to be 1.35. The lateral size of $\mathbf{M}$ is $N = 51$. The lateral size of the convolution kernels $\mathbf{S}_k$, $\mathbf{W}_k$ and $\mathbf{D}_k$ is $N_f = 11$. The threshold of photoresist in Eq. (3) is $t_r = 0.25$. The steepness indexes in Eqs. (4) and (12) are $a_r = a_m = 0.01$. The weight of the quadratic penalty in Eq. (18) is $\gamma_Q = 0.05$. Figure 5 shows 9 layout patterns used as the training samples. Figures 6(a)-6(c) show 3 test layouts to evaluate the performance of MCNN. From left to right, the test layouts are called "Target 1", "Target 2" and "Target 3", respectively. The second and third rows of Fig. 6 show the print images of the test layouts at 90nm and 45nm technology nodes, respectively. The pattern errors (PE) of the print images are presented in the figure. In this paper, PE is used as the metric to evaluate the image fidelity of lithography systems. The PE is defined as the square of the Euclidean distance between print image and target layout, that is $\text{PE} = \|\tilde{\mathbf{Z}} - \Gamma\{\mathbf{I} - t_r\}\|_2^2$.

Figure 7 illustrates the simulations of MCNN at 90nm and 45nm technology nodes, where the number of layers is $K = 2$. The top half of Fig. 7 illustrates the simulations of MCNN at 90nm technology node, where the critical dimension of the layout features is 90nm. During the training process, we use each of the training layouts to update the convolution kernels for 5 iterations based on Eq. (20). The step sizes to update the convolution kernels are $\text{step}_S = \text{step}_W = \text{step}_D = 7 \times 10^{-3}$. Similar to conventional neural network methods, the step size to update the convolution kernels is selected empirically. The step size should be set up according to the range of convolution kernels and the range of gradients of cost function. If the step size is too large, the training process will not converge smoothly. On the other hand, if the step size is too small, the training process will converge very slowly. In general, we could gradually increase the step size from a small value, and use a line search method to obtain an

Optics EXPRESS

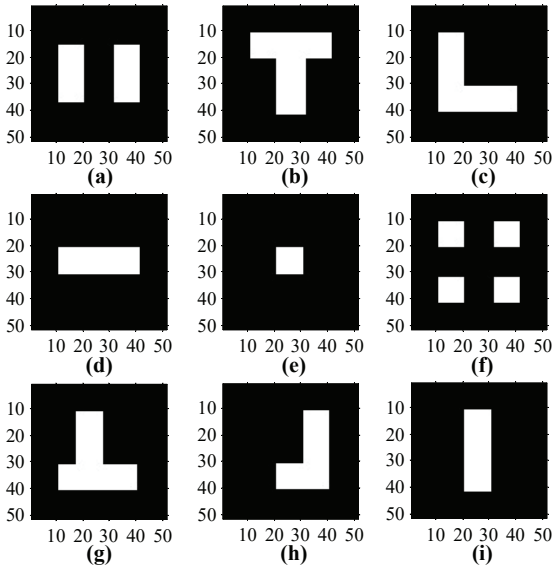

Fig. 5. The training layouts for MCNN.

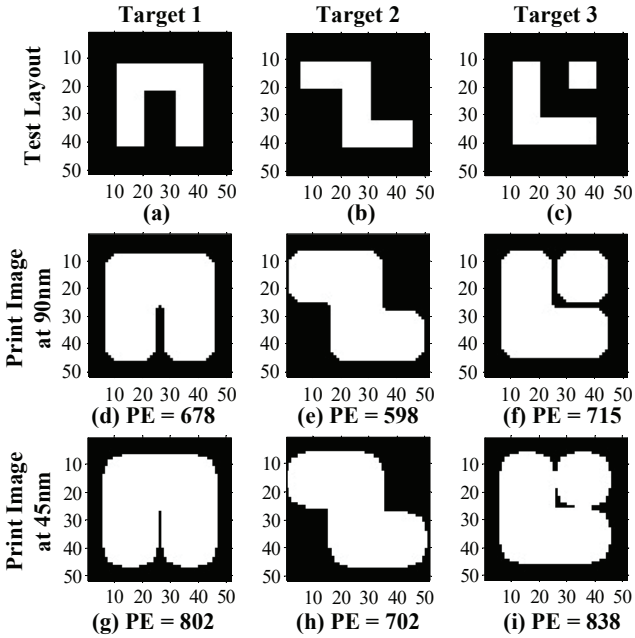

Fig. 6. The test layouts for MCNN and their corresponding print images.

approximately optimal step size. During the test stage, we input the test layout into the trained MCNN, and the output of MCNN is used as the initial guess of ILT solution. The first row of Fig. 7 illustrates the learned mask patterns obtained by the MCNN. The second row shows the corresponding print images and the pattern errors. The bottom half of Fig. 7 illustrates the simulations of MCNN at 45nm technology node. The training method is the same as mentioned

Optics EXPRESS

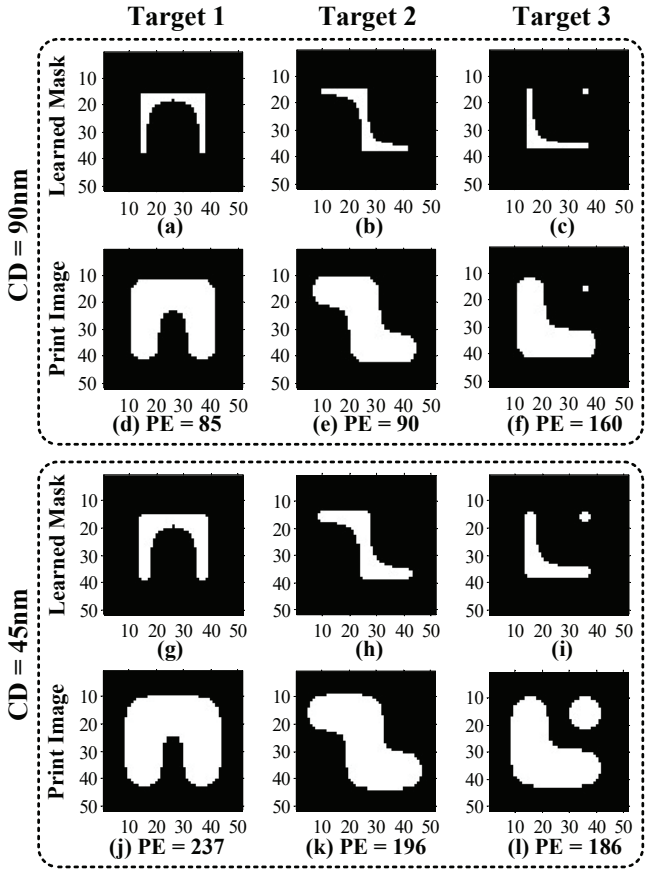

Fig. 7. The simulations of MCNN with two layers at 90nm and 45nm technology nodes.

above. The step sizes to update the convolution kernels are $\text{step}_S = \text{step}_W = \text{step}_D = 4 \times 10^{-3}$. The third and fourth rows in Fig. 7 illustrate the learned mask patterns and their print images, respectively.

Figure 8 shows the obtained pattern errors of the of MCNN with different number of layers. Figures 8(a) and 8(b) illustrate the simulation results of 90nm and 45nm technology nodes, respectively. The solid curve, dashed curve and dash-dotted curve correspond to different target layouts, respectively. During the training process, the simulation parameters are the same as those used in Fig. 7. It is observed that the two-layer MCNN leads to the best image fidelity among all of the three test layouts. In addition, less layer means less computational complexity in the training and test stages. Thus, we choose the two-layer MCNN in the following simulations. Although the two-layer MCNN results in the best accuracy in the simulations provided above, the number of convolution layers should be adjusted according to the training data and the test layout patterns in other application scenarios.

Comparing Fig. 6 to Fig. 7, we find that the proposed MCNN can properly modify the mask patterns and effectively improve the imaging performance by using only 2 convolution layers. The third row in Table 1 provides the average runtimes of MCNN ($K = 2$) to calculate the three learned mask patterns. It is noted that the computational complexity of the MCNN is equal to only two iterations of the gradient-based algorithm. However, the proposed MCNN can wisely learn how to optimize the mask patterns from the training layouts.

Optics EXPRESS

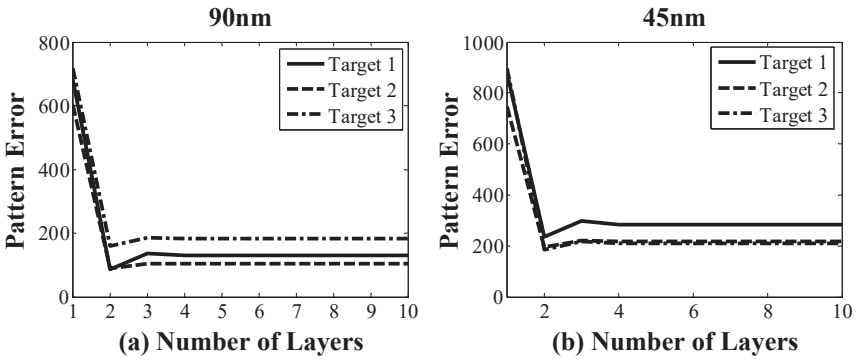

Fig. 8. The resulting pattern errors corresponding to different number of convolution layers at (a) 90nm and (b) 45nm technology nodes.

Table 1. Average runtimes of MCNN ($K = 2$) and gradient-based approaches over the three test layouts.

CD	90nm		45nm	
Algorithm	MCNN	Gradient-based	MCNN	Gradient-based
Learning	5.0×10^{-3} s	–	7.1×10^{-3} s	–
Iteration	0.29 s	1.09 s	0.30 s	1.10 s
Speedup	×3.7	–	×3.6	–

4.2. *Comparison between MCNN and gradient-based ILT*

This section compares the proposed MCNN approach to the steepest descent (SD) ILT algorithm in [7, 8]. SD is a conventional gradient-based algorithm extensively used in computational lithography. In the following simulations, the MCNN is used to produce the initial guess of the ILT solutions as shown in Fig. 7. Then, the output of MCNN is used as the starting point of the following iterative optimization. In particular, we carry out the SD algorithm for several iterations to refine the learned masks and further improve the imaging performance. The top half of Fig. 9 shows the simulation results of MCNN approach at 90nm technology node. In this simulation, we use the learned masks in Figs. 7(a), 7(b) and 7(c) as the initial mask patterns. Then, we carry out SD algorithm for 305 iterations to further optimize the mask patterns. In the SD algorithm, step size is set to be 0.5, and the weight of quadratic penalty is $\gamma_Q = 0.05$. The first and second rows in Fig. 9 illustrate the final optimized mask patterns and their corresponding print images. The third row shows the error patterns obtained by subtracting the target layouts from the print images. The white, grey and black colors represent 1, 0 and -1, respectively.

The bottom half of Fig. 9 illustrates the simulation results of the traditional gradient-based ILT algorithm, where the initial masks are the same as the target layouts. Hereafter, we refer to the traditional gradient-based ILT algorithm as the gradient-based approach for short. In the simulations, we carry out the SD algorithm for 1000 times to optimize the mask patterns. Other parameters are the same as MCNN approach. The convergence curves of PEs for the MCNN and gradient-based approaches are provided in the first row of Fig. 10. It is observed that the MCNN leads to much faster convergence than the gradient-based algorithm. That is because the MCNN

Optics EXPRESS

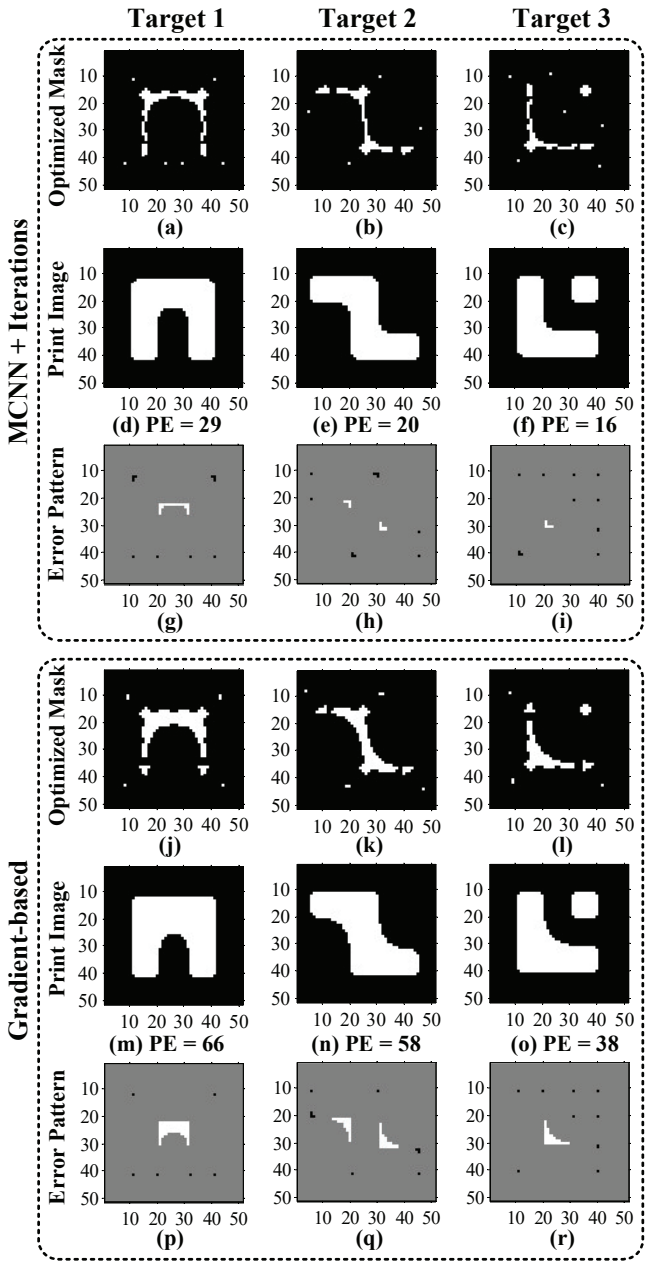

Fig. 9. Comparison between MCNN and gradient-based approaches at 90nm technology node.

can provide very good initial mask patterns that are very close to the optimal solutions. Thus, compared to the gradient-based algorithm, the MCNN approach uses much less iterations but results in higher image fidelity. The average runtimes of MCNN and gradient-based approaches are summarized in the second and third columns in Table 1. In contrast to the gradient-based algorithm, the proposed MCNN method can achieve more than 3-fold speedup, and further reduce the PEs by 60% on average.

Optics EXPRESS

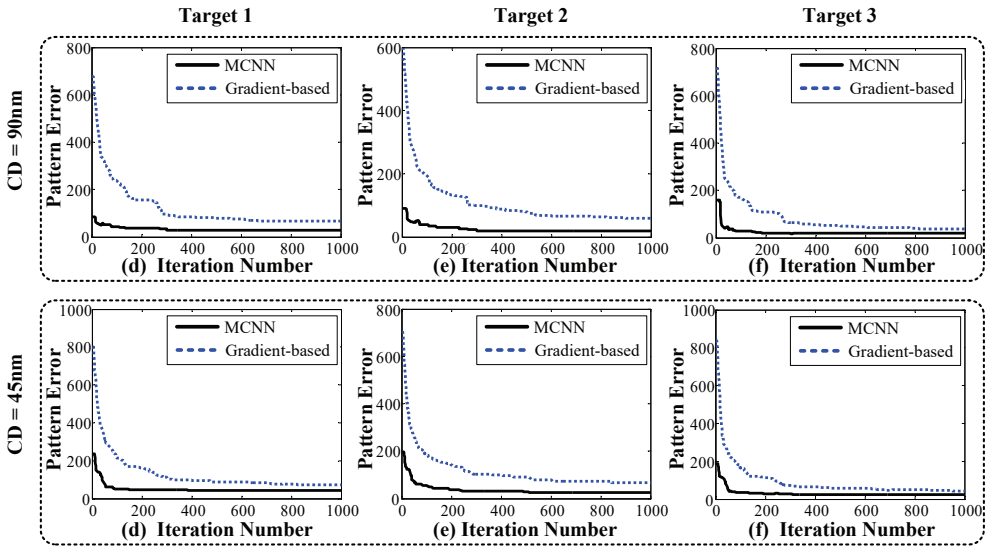

Fig. 10. The convergence curves of pattern errors for MCNN and gradient-based approaches.

Figure 11 shows the simulations of MCNN and gradient-based approaches at 45nm technology node. The simulation parameters are the same as those used in Fig. 9. The bottom row of Fig. 10 compares the convergence between the MCNN and gradient-based approaches. In addition, the average runtimes of these two methods are summarized in the fourth and fifth columns in Table 1. Compared to the gradient-based algorithm, the proposed MCNN method can improve the computational efficiency by 3.6 times, while further reduce the PEs by 42% on average.

In Fig. 12, another complex layout pattern is used to assess the performance of the MCNN and gradient-based approaches. The top half and bottom half in Fig. 12 show the simulation results at 90nm and 45nm technology nodes, respectively. From left to right, the figure shows the optimized mask patterns, the print images and the corresponding error patterns. In the MCNN approach, we first use the MCNN to obtain the learned masks and then carry out the SD algorithm for 400 iterations to refine the mask patterns. As a comparison, the gradient-based ILT approach conducts 3000 iterations to obtain the optimized mask patterns. Other simulation parameters are the same as those used in Figs. 7 and 9. Table 2 provides the runtimes of MCNN ($K = 2$) and gradient-based approaches for the complex test layouts. The average runtimes of the MCNN and gradient-based approaches are 1.11s and 11.44s, respectively. Compared to the gradient-based algorithm, the proposed MCNN method can improve the computational efficiency by an order of magnitude, while further reduce the PEs by 60% on average. Based on the aforementioned simulations and analysis, we conclude that the proposed MCNN method can effectively improve the speed of the inverse lithography technique, and simultaneously improve the imaging performance of lithography systems.

At the end of this section, we conduct a simulation to study the impact of the number of training samples on the lithography image fidelity. As mentioned above, the MCNN used in the previous simulations is trained by 9 sample layouts as shown in Fig. 5. In the following simulations, we extend the amount of training data to 12 by adding the 3 training samples in Fig. 13. Figure 14 compares the simulations of the MCNN using different number of training samples. The first row illustrates the simulation results using the 9 training samples in Fig. 5. The bottom row shows the results using the 12 training samples in Figs. 5 and 13. During the training process using 12 samples, we use each of the training layouts to update the convolution kernels

Optics EXPRESS

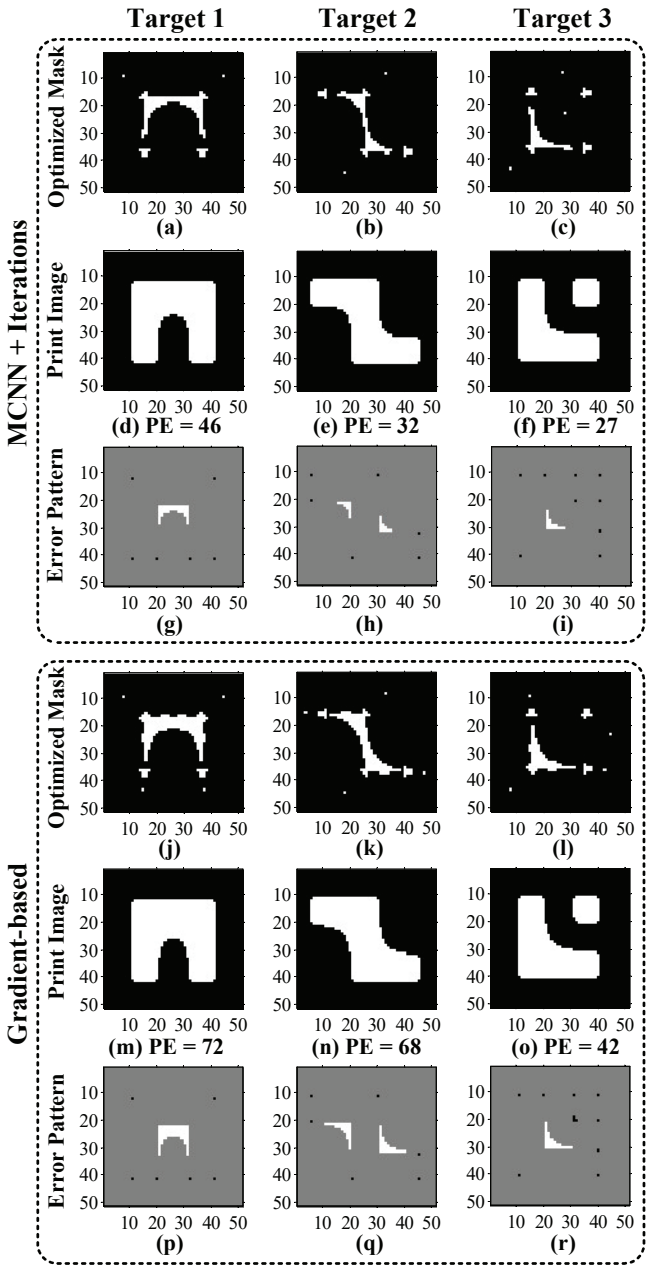

Fig. 11. Comparison between MCNN and gradient-based approaches at 45nm technology node.

for 4 iterations based on Eq. (20). The first column and the second column show the learned mask patterns obtained by the MCNN and the corresponding print images. The third column shows the optimized mask patterns refined by the SD algorithm for several iterations, and the fourth column shows the corresponding print images. It is observed that increasing the amount of training data may further improve the image fidelity obtained by the MCNN approach.

Please note that the learning problem in ILT is different from the general image processing

Optics EXPRESS

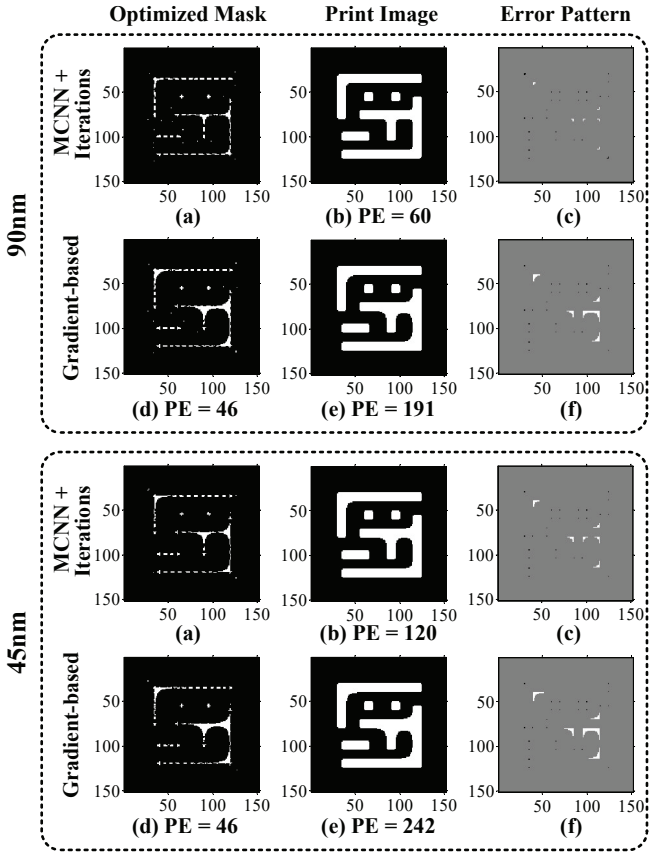

Fig. 12. Comparison between MCNN and gradient-based approaches based on the complex target layout.

Table 2. Runtimes of MCNN ($K = 2$) and gradient-based approaches for the complex test layouts.

CD	90nm		45nm	
Algorithm	MCNN	Gradient-based	MCNN	Gradient-based
Learning	7.3×10^{-3} s	–	7.7×10^{-3} s	–
Iteration	1.10 s	11.46 s	1.11 s	11.42 s
Speedup	×10.3	–	×10.2	–

problem. While the natural images are flexible to include arbitrary features, most of layout patterns in integrated circuits are regularized to have Manhattan geometries. It is known that any Manhattan geometries can be decomposed into three kinds of basic features: convex corners, concave corners, and straight edges [24]. Although the simulations mentioned above use only a small set of training layouts, they include enough common convex corners, concave corners and edges that exist in the test layout patterns. It is also noted that this paper provides a primary idea

Optics EXPRESS

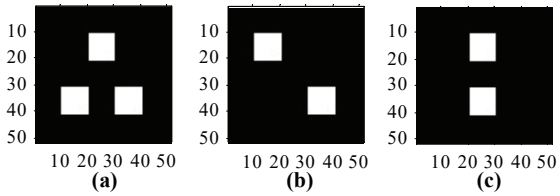

Fig. 13. The additional three training samples.

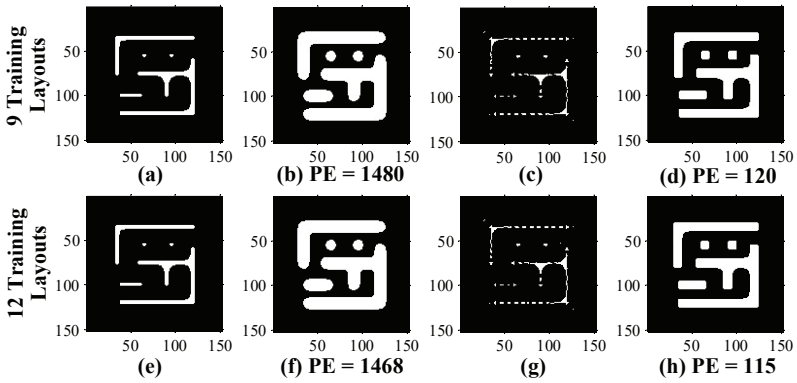

Fig. 14. The simulations of MCNN using different number of training samples.

to use the MCNN approach to solve for the ILT problem. In order to apply the MCNN framework to deal with full-chip layout in semiconductor fabrication, a large amount of training data should be collected from the real integrated circuit patterns. This work will be carried out in the future.

5. Conclusion

This paper develops an MCNN method to obtain the approximate guess of the ILT solution, which can effectively improve the computational efficiency of the conventional gradient-based ILT algorithms, and further improve the imaging performance of coherent lithography systems. The imaging model of lithography systems was used to build up the architecture of the MCNN. The network structure was designed by unfolding and truncating the inverse lithography optimization framework. After that, an unsupervised training strategy was developed for the proposed MCNN to get rid of the time-consuming labelling process. By introducing the decoder into the training loop, the MCNN can automatically learn the ILT solutions from the pre-defined training layouts. Given the output of MCNN, the gradient-based ILT algorithm is run for several iterations to obtain the final optimized mask patterns. The merits of the proposed MCNN approach was verified by a set of simulations using different test layouts. In future work, we will generalize the proposed MCNN into partially coherent lithography systems, and study the relationship between the layer number of the MCNN with respect to the training data and test patterns. More research is also needed to study different convolution networks to further improve the computational efficiency, imaging performance and mask manufacturability.

A. Appendix

According to Eqs. (8) and (18), the gradient of cost function E' with respect to $\hat{\mathbf{M}}$ is

$$\nabla E'|_{\hat{\mathbf{M}}} = \nabla E|_{\hat{\mathbf{M}}} + \nabla R_Q|_{\hat{\mathbf{M}}}. \tag{21}$$

Optics EXPRESS

where $\nabla E|_{\hat{\mathbf{M}}}$ can be calculated as follows

$$\nabla E|_{\hat{\mathbf{M}}} = -2a_r \cdot \left\{\mathbf{h}^{\circ} \otimes \left[(\mathbf{M}_1^b - \mathbf{Z}') \odot \mathbf{Z}' \odot (\mathbf{1} - \mathbf{Z}') \odot (\mathbf{h} \otimes \hat{\mathbf{M}})\right]\right\}, \tag{22}$$

where the notation $\circ$ means to rotate the matrix by 180° in both horizontal and vertical directions. According to Eq. (17), the gradient of quadratic penalty is formulated as

$$\nabla R_Q|_{\hat{\mathbf{M}}} = -8\hat{\mathbf{M}} + 4. \tag{23}$$

According to Eq. (12), the gradient of $\mathbf{M}_{k+1}^b$ to $\mathbf{M}_k^g$ is

$$\nabla \mathbf{M}_{k+1}^b|_{\mathbf{M}_k^g} = a_m \cdot \mathrm{sig_m}(\mathbf{M}_k^g, t_m) \cdot \left[\mathbf{1} - \mathrm{sig_m}(\mathbf{M}_k^g, t_m)\right]. \tag{24}$$

According to Eq. (10), the gradient of $\mathbf{M}_k^g$ to $\mathbf{M}_k^b$ is

$$\nabla \mathbf{M}_k^g|_{\mathbf{M}_k^b} = \mathbf{S}_k^{\circ} \otimes \mathbf{M}_k^g + \mathbf{D}_k^{\circ} \otimes \left[\mathbf{T}_k \odot (\mathbf{W}_k^{\circ} \otimes \mathbf{M}_k^g)\right]. \tag{25}$$

The gradient of $\mathbf{M}_k^g$ to $\mathbf{S}_k$ is

$$\nabla \mathbf{M}_k^g|_{\mathbf{S}_k} = \Pi\left\{\mathbf{M}_k^b \otimes \mathbf{M}_k^g\right\}, \tag{26}$$

where $\Pi\{\cdot\}$ is the window function to truncate the $N_f \times N_f$ central part of the matrix in the argument. The window function $\Pi\{\cdot\}$ is used to keep the dimension of $\mathbf{S}_k$ unchanged during the training process. The gradient of $\mathbf{M}_k^g$ to $\mathbf{W}_k$ is

$$\nabla \mathbf{M}_k^g|_{\mathbf{W}_k} = \Pi\left\{\mathbf{B}^{\circ} \otimes \mathbf{M}_k^g\right\}, \text{ where } \mathbf{B} = \mathbf{T} \odot (\mathbf{D}_k \otimes \mathbf{M}_k^b). \tag{27}$$

The gradient of $\mathbf{M}_k^g$ to $\mathbf{D}_k$ is

$$\nabla \mathbf{M}_k^g|_{\mathbf{D}_k} = \Pi\left\{(\mathbf{M}_k^b)^{\circ} \otimes \left[\mathbf{T} \odot (\mathbf{W}_k^{\circ} \otimes \mathbf{M}_k^g)\right]\right\}. \tag{28}$$

According to the Chain Rule, we can calculate the derivatives of cost function with respect to the elements of $\mathbf{S}_k$, $\mathbf{W}_k$ and $\mathbf{D}_k$, respectively. For simplicity, assume the matrix $\mathbf{A}_k = \mathbf{S}_k$, $\mathbf{W}_k$ or $\mathbf{D}_k$. Then, the derivatives are formulated as following

$$\begin{aligned}\frac{\partial E'}{\partial \mathbf{A}_k(x,y)} = \sum_{\hat{x},\hat{y}} \Bigg(\frac{\partial E'}{\partial \hat{\mathbf{M}}(\hat{x},\hat{y})} \cdot \sum_{x_K,y_K} \Bigg(\frac{\partial \hat{\mathbf{M}}(\hat{x},\hat{y})}{\partial \mathbf{M}_K^g(x_K,y_K)} \cdots \\ \sum_{x_k,y_k} \left(\frac{\partial \mathbf{M}_{k+1}^g(x_{k+1},y_{k+1})}{\partial \mathbf{M}_k^g(x_k,y_k)} \cdot \frac{\partial \mathbf{M}_k^g(x_k,y_k)}{\partial \mathbf{A}_k(x,y)} \right) \cdots \Bigg) \Bigg),\end{aligned} \tag{29}$$

where (x, y), $(\hat{x}, \hat{y})$, (x_K, y_K), (x_{k+1}, y_{k+1}) and (x_k, y_k) are the coordinates. Based on Eqs. (21)-(28), we can calculate the derivative in Eq. (29).

Funding

National Natural Science Foundation of China (NSFC) (61675021); Natural Science Foundation of Beijing Municipality (4173078); Fundamental Research Funds for the Central Universities (2018CX01025); China Scholarship Council (201706035012).

Optics EXPRESS

References

1. G. E. Moore, "Cramming more components onto integrated circuits," Electronics **38**, 114ff (1965).
2. A. K. Wong, *Resolution Enhancement Techniques in Optical Lithography* (SPIE, 2001).
3. X. Ma and G. R. Arce, *Computational Lithography*, Wiley Series in Pure and Applied Optics, 1st ed. (John Wiley and Sons, 2010).
4. Y. Liu and A. Zakhor, "Binary and phase shifting mask design for optical lithography," IEEE Trans. on Semicond. Manuf. **5**, 138–152 (1992).
5. N. B. Cobb and Y. Granik, "Dense OPC for 65nm and below," Proc. SPIE **5992**, 599259 (2005).
6. P. M. Martin, C. J. Progler, G. Xiao, R. Gray, L. Pang, and Y. Liu, "Manufacturability study of masks created by inverse lithography technology (ILT)," Proc. SPIE **5992**, 599235 (2005).
7. A. Poonawala and P. Milanfar, "OPC and PSM design using inverse lithography: a non-linear optimization approach," Proc. SPIE **6154**, 61543H (2006).
8. X. Ma and G. R. Arce, "Generalized inverse lithography methods for phase-shifting mask design," Opt. Express **15**, 15066–15079 (2007).
9. A. Poonawala, B. Painter, and C. Kerchner , "Model-based assist feature placement for 32nm and 22nm technology nodes using inverse mask technology," Proc. SPIE **7488**, 748814 (2009).
10. Y. Granik, "Fast pixel-based mask optimization for inverse lithography," J. Microlith. Microfab. Microsyst. **5**, 043002 (2006).
11. X. Ma and G. R. Arce, "Binary mask optimization for inverse lithography with partially coherent illumination," J. Opt. Soc. Am. A **25**, 2960–2970 (2008).
12. Y. Shen, N. Wong, and E. Y. Lam, "Level-set-based inverse lithography for photomask synthesis," Opt. Express **17**, 23690–23701 (2009).
13. J. Yu and P. Yu, "Impacts of cost functions on inverse lithography patterning," Opt. Express **18**, 23331–23342 (2010).
14. Y. Shen, N. Jia, N. Wong, and E. Y. Lam, "Robust level-set-based inverse lithography," Opt. Express **19**, 5511–5521 (2011).
15. X. Ma, Y. Li, and L. Dong, "Mask optimization approaches in optical lithography based on a vector imaging model," J. Opt. Soc. Am. A **29**, 1300–1312 (2012).
16. X. Ma, Z. Song, Y. Li, and G. R. Arce, "Block-based mask optimization for optical lithography," Appl. Opt. **52**, 3351–3363 (2013).
17. N. Jia and E. Y. Lam, "Machine learning for inverse lithography: using stochastic gradient descent for robust photomask synthesis," J. Opt. **12**, 045601 (2010).
18. X. Ma and G. R. Arce, "Pixel-based OPC optimization based on conjugate gradients," Opt. Express **19**, 2165–2180 (2011).
19. X. Ma, G. R. Arce, and Y. Li, "Optimal 3D phase-shifting masks in partially coherent illumination," Appl. Opt. **50**, 5567–5576 (2011).
20. W. Lv, S. Liu, Q. Xia, X. Wu, Y. Shen, and E. Y. Lam, "Level-set-based inverse lithography for mask synthesis using the conjugate gradient and an optimal time step," J. Vac. Sci. Technol. B **31**, 041605 (2013).
21. X. Wu, S. Liu, W. Lv, and E. Y. Lam, "Robust and efficient inverse mask synthesis with basis function representation," J. Opt. Soc. Am. A **31**, B1–B9 (2014).
22. R. Luo, "Optical proximity correction using a multilayer perceptron neural network," J. Opt. **15**, 075708 (2013).
23. K. Luo, Z. Shi, X. Yan, and Z. Geng, "SVM based layout retargeting for fast and regularized inverse lithography," J. Zhejiang Uni-Sci C (Comput & Electron) **15**, 390–400 (2014).
24. X. Ma, B. Wu, Z. Song, S. Jiang, and Y. Li, "Fast pixel-based optical proximity correction based on nonparametric kernel regression," J. Micro/Nanolith. MEMS MOEMS **13**, 043007 (2014).
25. X. Ma, S. Jiang, J. Wang, B. Wu, Z. Song, and Y. Li, "A fast and manufacture-friendly optical proximity correction based on machine learning," Microelectron. Eng. **168**, 15-26 (2016).
26. G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," Science **313**, 504-507 (2006).
27. Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature **521**, 436-444 (2015).
28. I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, Adaptive Computation and Machine Learning Series (The MIT Press, 2016).
29. Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," in the handbook of brain theory and neural networks, 255-258 (The MIT Press, 1998).
30. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM **60**, 84-90 (2017).
31. O. A. Hamid, A. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," IEEE/ACM Trans. Audio, Speech, and Language Process. **22**, 1533-1545 (2014).
32. S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: a convolutional neural-network approach," IEEE Trans. Neural Network **8**, 98-113 (1997).
33. Z. Wang, Q. Ling, and T. S. Huang, "Learning deep l_0 encoders," Proc. Thirtieth AAAI Conference on Artificial Intelligence, 2194-2200 (2016).
34. K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," Proc. 27th International Conference on Machine Learning, 399-406 (2010).

Optics EXPRESS

35. P. Sprechmann, A. M. Bronstein, and G. Sapiro, "Learning efficient sparse and low rank models," IEEE Trans. Pattern Anal. **37**, 1821-1833 (2015).
36. J. Hershey, J. Roux, and F. Weninger, "Deep unfolding: model-based inspiration of novel deep architectures," https://arXiv:1409.2574 (2014).
37. B. Xin, Y. Wang, W. Gao, and D. Wipf, "Maximal sparsity with deep networks?," https://arXiv:1605.01636 (2016).
38. E. Alpaydin, *Introduction to Machine Learning*, 2nd ed. (China Machine Press, 2014).
39. H. H. Hopkins, "On the diffraction theory of optical images," Proc. R. Soc. Lond. A **217**, 408–432 (1953).
40. H. H. Hopkins, "The concept of partial coherence in optics," Proc. R. Soc. Lond. A **208**, 263–277 (1951).