Data Efficient Lithography Modeling With Transfer Learning and Active Data Selection

Yibo Lin, *Member, IEEE*, Meng Li, *Student Member, IEEE*, Yuki Watanabe, Taiki Kimura, Tetsuaki Matsunawa, Shigeki Nojima, and David Z. Pan, *Fellow, IEEE*

***Abstract*—Lithography simulation is one of the key steps in physical verification, enabled by the substantial optical and resist models. A resist model bridges the aerial image simulation to printed patterns. While the effectiveness of learning-based solutions for resist modeling has been demonstrated, they are considerably data-demanding. Meanwhile, a set of manufactured data for a specific lithography configuration is only valid for the training of one single model, indicating low data efficiency. Due to the complexity of the manufacturing process, obtaining enough data for acceptable accuracy becomes very expensive in terms of both time and cost, especially during the evolution of technology generations when the design space is intensively explored. In this paper, we propose a new resist modeling framework for contact layers, utilizing existing data from old technology nodes and active selection of data in a target technology node, to reduce the amount of data required from the target lithography configuration. Our framework based on transfer learning and active learning techniques is effective within a competitive range of accuracy, i.e., 3×–10× reduction on the amount of training data with comparable accuracy to the state-of-the-art learning approach.**

***Index Terms*—Active learning, convolutional neural networks (CNNs), lithography modeling, machine learning, residual neural networks (ResNet), transfer learning.**

I. Introduction

DUE TO the continuous semiconductor scaling from 10-nm technology node (N10) to 7-nm node (N7) [1], [2], the prediction of printed pattern sizes is becoming increasingly difficult and complicated due to the complexity of manufacturing process and variations. However, complex designs demand accurate simulations to guarantee functionality and yield. Resist modeling, as a key component in lithography simulation, is critical to bridge the aerial image simulation to manufactured wafer data. Rigorous simulations that perform physics-level modeling suffer from large computational overhead, which are not suitable when used extensively. Thus, compact resist models are widely used in practice.

Fig. 1(a) shows the process of lithography simulations where the aerial image is computed from the input mask patterns and the optical model, and the output pattern is computed from the aerial image and the resist model. As the aerial image contains the light intensity map, the resist model needs to determine the slicing thresholds for the output patterns as shown in Fig. 1(b). With the thresholds, the critical dimensions (CDs) of printed patterns can be computed, which need to match CDs measured from manufactured patterns. In practice, various factors may impact a resist model such as the physical properties of photoresist, design rules of patterns, process variations. CD usually refers to the smallest dimension on a lithography level that must be accurately controlled when fabricating a device. Here, CDs refer to the sizes of printed patterns.

Accurate lithography simulation like rigorous physics-based simulation is notorious for its long computational time, while simulation with compact models suffers from accuracy issues [3], [4]. On the other hand, machine learning techniques are able to construct accurate models and then make efficient predictions. These approaches first take training data to calibrate a model and then use this model to make predictions on testing data for validation. The effectiveness of learning-based solutions has been studied in various lithography related areas including aerial image simulation [5], hotspot detection [6]–[16], optical proximity correction (OPC) [17]–[20], subresolution assist features (SRAF) [21], [22], resist modeling [3], [4], etc. In resist modeling, a convolutional neural network (CNN) that predicts slicing thresholds in aerial images is proposed [4]. The neural network consists of three convolution layers and two fully connected layers. Since the slicing threshold is a continuous value, learning a resist model is a regression task rather than a classification task. Around 70% improvement in accuracy is reported compared with calibrated compact models from Mentor Calibre [23]. Shim *et al.* [3] proposed an artificial neural network with five hidden layers to predict the height of resist after exposure. Significant speedup is reported with high accuracy compared with a rigorous simulation.

Although the learning-based approaches are able to achieve high accuracy, they are generally data-demanding in model training. In other words, big data is assumed to guarantee accuracy and generality. Furthermore, one data sample can only be used to train the corresponding model under the

Manuscript received February 23, 2018; revised May 24, 2018; accepted July 20, 2018. Date of publication August 7, 2018; date of current version September 18, 2019. This work was supported in part by the National Science Foundation under Project CCF-1718570, and in part by Toshiba Memory Corporation. This paper was recommended by Associate Editor I. H.-R. Jiang. *(Corresponding author: Yibo Lin.)*

Y. Lin, M. Li, and D. Z. Pan are with the Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX 78741 USA (e-mail: yibolin@utexas.edu).

Y. Watanabe, T. Kimura, T. Matsunawa, and S. Nojima are with the Memory Lithography Group, Toshiba Memory Corporation, Yokohama 247-8585, Japan.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCAD.2018.2864251

0278-0070 © 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

Authorized licensed use limited to: University of Texas at Austin. Downloaded on September 16,2020 at 02:47:43 UTC from IEEE Xplore. Restrictions apply.

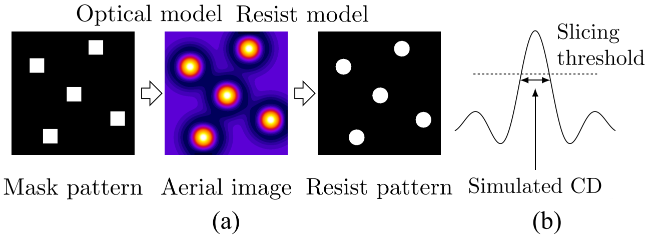


Fig. 1. (a) Process of lithography simulation with optical and resist models. (b) Thresholds for aerial image determine simulated CD, which should match manufactured CD.

same lithography configuration, indicating a low data efficiency. Here data efficiency evaluates the accuracy a model can achieve given a specific amount of data, or the amount of data samples are required to achieve target accuracy. Nevertheless, obtaining a large amount of data is often expensive and time-consuming, especially when the technology node switches from one to another and the design space is under active exploration, e.g., from N10 to N7. The lithography configurations including optical sources, resist materials, etc., are frequently changed for experiments. Therefore, a fast preparation of models with high accuracy is urgently desired. In addition, it remains to be a question that what are the best designs for building a model. Typical practice of regular array patterns or random patterns may not be representative enough to calibrate accurate and generic models. Thus, effective techniques to recognize representative designs will also be beneficial to improving data efficiency.

Different from the previous approaches, in this paper, we assume the availability of large amounts of data from the previous technology generation with old lithography configurations and small amounts of data from a target lithography configuration. We focus on increasing the data efficiency by: 1) reusing those from other lithography configurations and transfer the knowledge between different configurations and 2) active selection of data samples in the target configuration, also known as active learning. The objective is to achieve accurate resist models with significantly fewer data to a target configuration. The major contributions are summarized as follows.

1) We propose a high performance resist modeling technique based on the residual neural network (ResNet).
2) We propose a transfer learning scheme for ResNet that can reduce the amount of data with a target accuracy by utilizing the data from other configurations.
3) We propose an active learning scheme based on K-Medoids algorithm with theoretical insights for both CNN and ResNet.
4) The experimental results demonstrate 3×–10× reduction in the amount of training data to achieve accuracy comparable to the state-of-the-art learning approach [4].

The rest of this paper is organized as follows. Section II illustrates the problem formulation. Section III explains the details of our approach. The effectiveness of our approach is verified in Section IV and the conclusion is drawn in Section V.

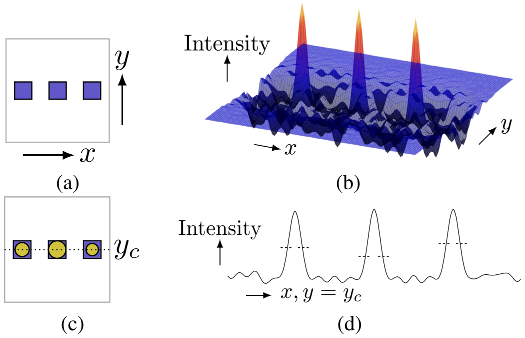


Fig. 2. (a) Design target of three contacts and (b) light intensity plot of aerial image. Assume that RETs such as SRAF and OPC have been already applied to the contacts before optical simulation. (c) Dotted line horizontally crosses the centers at $y = y_c$ and the circles denote the contours of printed patterns. (d) Light intensity profiling along the dotted line at $y = y_c$ extracted from the aerial image and different slicing thresholds for each contact.

II. Preliminaries

In this section, we will briefly introduce the background knowledge on lithography simulation and resist modeling. Then the problem formulation is explained. We mainly focus on contact layers in this paper, but our methodology shall be applicable to other layers. For simplicity, we use the word *label* to represent the target value for prediction, e.g., threshold, given a data sample; we also use the phrase *unlabeled* data to denote data samples whose labels are unknown.

A. Lithography Simulation

Lithography simulation is generally composed of two stages, i.e., optical simulation and resist simulation, where optical and resist models are required, respectively. In the optical simulation, an optical model, characterized by the illumination tool, takes mask patterns to compute aerial images, i.e., light intensity maps. Then in the resist simulation, a resist model finalizes the resist patterns with the aerial images from the optical simulation. Generally, there are two types of resist models. One is a variable threshold resist model in which the thresholds vary according to aerial images, and the other is a constant threshold resist model in which the light intensity is modulated in an aerial image. We adopt the former since it is suitable to learning-based approaches [4].

Fig. 2 shows an example of lithography simulation for a clip with three contacts. We assume that proper resolution enhancement techniques (RETs) such as OPC and SRAF have been applied before the computation of the aerial image [24]. The optical simulation generates the aerial image, as shown in Fig. 2(b). Resist simulation then computes the thresholds in the aerial image to predict printed patterns. If we want to measure the widths of contacts along the dotted line in Fig. 2(c), the light intensity profiling can be extracted from the aerial image along the line and calculates the CDs for each contact with the thresholds.

B. Historical Data and Transfer Learning

Since the lithography configurations evolve from one generation to another with the advancement of technology nodes,

Authorized licensed use limited to: University of Texas at Austin. Downloaded on September 16,2020 at 02:47:43 UTC from IEEE Xplore. Restrictions apply.

TABLE I
LITHOGRAPHY CONFIGURATIONS FOR N10 AND N7

	N10	N7	
		N7$_a$	N7$_b$
Design Rule	A	B	B
Optical Source	A	B	B
Resist Material	A	A	B

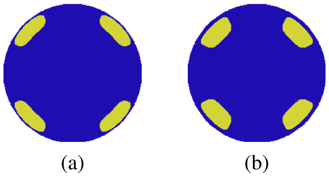


Fig. 3. Optical sources (yellow) for (a) N10 and (b) N7.

there are plenty of historical data available for the old generation. As mentioned in Section I, accurate models require a large amount of data for training or calibration, which are expensive to obtain during the exploration of a new generation. If the lithography configurations have no fundamental changes, the knowledge learned from the historical data may still be applicable to the new configuration, which can eventually help to reduce the amount of new data required.

Transfer learning represents a set of techniques to transfer the knowledge from one or multiple source domains to a target domain, utilizing the underlying similarity between the data from these domains. Various studies have explored the effectiveness of knowledge transfer in image recognition and robotics [25]–[27], while it is not clear whether the knowledge between different resist models is transferable or not.

In this paper, we consider the evolution of the contact layer from the cutting edge technology node N10 to N7 [1], [2]. A large amount of available N10 data are assumed. During the evolution to N7, different design rules for mask patterns, optical sources, and resist materials for lithography are explored. Table I shows the lithography configurations considered for N10 and N7. Differences in letters A, B represent different configurations of design rules, optical sources, or resist materials. One configuration for N10 is considered, while two configurations are considered for N7, i.e., N7$_a$, N7$_b$, with two kinds of resist materials (about 20% difference in the slopes of dissolution curves). From N10 to N7, both the design rules and optical sources are changed. For N10, we consider a pitch of 64-nm with double patterning lithography, while for N7, the pitch is set to 45 nm with triple patterning lithography [1]. The width of each contact is set to half pitch. The lithography target of each contact is set to 60 nm for both N10 and N7. Optical sources calibrated with industrial strength for N10 and N7 are shown in Fig. 3, with the same type of illumination shapes.

Various combinations of knowledge transfer can be explored from Table I, such as N10→N7, N7$_i$→N7$_j$, and N10+N7$_i$→N7$_j$, where $i \neq j, i, j \in \{a, b\}$.

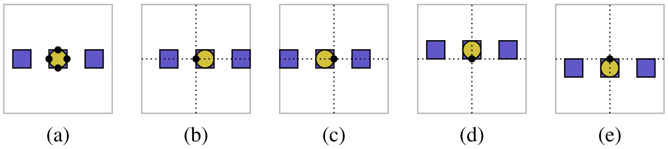


Fig. 4. (a) Thresholds for the middle of the four edges of the center contact are predicted. (b)–(e) Clip window is shifted such that the target position lies in the center of the clip.

C. Active Learning for Regression

Active learning assumes unlabeled data samples exist in a pool or can be generated. Querying for data labels is very expensive and the amount of queries should be minimized. Thus, selecting the proper and limited portion of data samples for querying is essential to modeling accuracy.

We define the problem of pool-based active learning as: given a pool of unlabeled data samples, select k samples to query for labels and train a model to maximize the accuracy across the entire dataset. Be aware that the selection of data samples should not depend on data labels since labels are unknown before querying. Hence, active learning is very effective in improving data efficiency without the requirement of any additional labeled data.

There are extensive studies for active learning in classification for CNN and SVM [28]–[31]. A few studies have explored active learning for support vector regression and multilayer perception [32], [33]. Most techniques are categorized into confidence level or clustering approaches. Confidence level approaches tend to choose data samples with low prediction confidence, and clustering approaches choose representative subset of data samples among an entire dataset. There are also successful applications of active learning in VLSI CAD related areas [34]–[36].

However, practical studies on active learning techniques for regression tasks with CNN or ResNet are lacking, and its performance when combined with transfer learning is unclear. It is often difficult to evaluate the confidence level with large and complicated models like CNN or ResNet, while clustering approaches only rely on the general properties of data samples and models. Therefore, we explore effective clustering strategies for active selection of data samples, which are suitable to regression tasks with CNN and ResNet.

D. Learning-Based Resist Modeling

The thresholds of positions near the contacts are of significant importance since they usually determine the boundaries of printed contacts. Hence we consider the middle of the left, right, bottom, and top edges for each contact, as shown in Fig. 4(a), where the positions for prediction are highlighted with black dots. As the threshold is mainly influenced by the surrounding mask patterns, resist models typically compute the threshold using a clip of mask patterns centered by a target position. To measure the thresholds in Fig. 4(a), we select a clip where the target position lies in its center, as shown in Fig. 4(b)–(e). The task of a resist model is to compute the thresholds for these positions of each contact [4]. For each

Authorized licensed use limited to: University of Texas at Austin. Downloaded on September 16,2020 at 02:47:43 UTC from IEEE Xplore. Restrictions apply.

clip with its aerial image and threshold, Fourier interpolation can be used to recover the printed patterns.

Learning-based resist modeling consists of two phases, i.e., training and testing. In the training phase, training dataset with both aerial images and thresholds are used to calibrate the model, while in the testing phase, the model predicts thresholds for the aerial images from the testing dataset.

E. Problem Formulation

The accuracy[1] of a model is evaluated with root mean square (RMS) error defined as follows:

$$\epsilon = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y} - y)^2} \quad (1)$$

where N denotes the amount of samples, y denotes the golden values, and $\hat{y}$ denotes the predicted values. We further define relative RMS error

$$\epsilon_r = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\frac{\hat{y} - y}{y}\right)^2} \quad (2)$$

where a relative ratio of error from the golden values can be represented. Both metrics can refer to errors in either CD or threshold. Although during model training, the RMS error of threshold is generally minimized due to easier computation, the eventual model is often evaluated with the RMS error of CD for its physical meaning to the patterns. The RMS errors in threshold and CD essentially have almost the same fidelity, and usually yield consistent comparison. For convenience, we report relative RMS error in threshold (ϵ_r^{th}) for comparison of different models since it removes the dependency to the scale of thresholds, and use RMS error in CD (ϵ^{CD}) for data efficiency related comparison.

Definition 1 (Data Efficiency): The amount of target domain data required to learn a model with a given accuracy.

Given a specific amount of data from a target domain, if one can learn a model with a higher accuracy than another, it also indicates higher data efficiency. Thus, improving model accuracy benefits data efficiency as well.

The resist modeling problem is defined as follows.

Problem 1 (Learning-Based Resist Modeling): Given a dataset containing information of aerial images and thresholds at their centers, train a resist model that can maximize the accuracy for the prediction of thresholds.

In practice, accuracy is not the only objective. The amount of training data should be minimized as well due to the high cost of data preparation. Therefore, we propose the problem of data efficient resist modeling as follows.

Problem 2 (Data Efficient Resist Modeling): Given a labeled N10 dataset containing aerial images and thresholds, and an unlabeled N7 dataset containing aerial images only, train a resist model for target dataset N7_i that can achieve high accuracy and meanwhile query labels for as few N7_i data samples as possible, where $i \in \{a, b\}$.

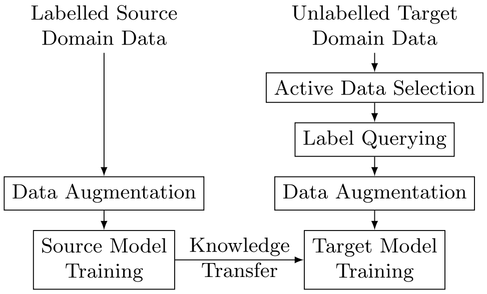


Fig. 5. Training flow with transfer learning and active learning.

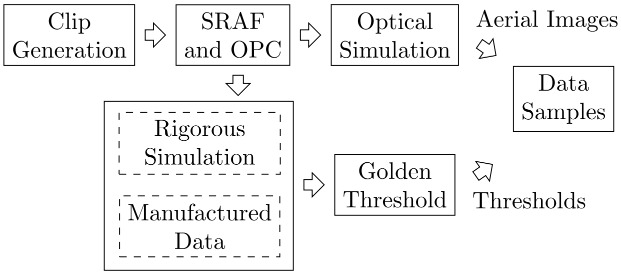


Fig. 6. Flow of data preparation.

Minimizing the times of label querying is equivalent to minimizing the cost of data preparation, since the most expensive part is to obtain the labels, i.e., thresholds, through either manufactured wafer data or rigorous simulation.

III. Algorithms

In this section, we will explain the structure of our models and then the details regarding the transfer learning and active learning schemes. Fig. 5 shows the overall training flow. We first leverage labeled source domain data to train a source domain model. Then before training the target domain model, active learning is applied for active selection of data samples for label querying. The target domain model is eventually trained with selected data samples and knowledge transferred from the source domain model. Data augmentation in Section III-A2 is applied before training of both source and target models.

A. Data Preparation

Fig. 6 gives the flow of data preparation. We first generate clips and perform SRAF insertion and OPC. The aerial images are then computed from the optical simulation, and at the same time, the golden thresholds need to be computed from either the rigorous simulation or the manufactured data. Each data sample consists of an aerial image and the threshold at its center.

1) Clip Generation: Following the design rules such as minimum pitch of contacts, we generate three types of 2 × 2 μm clips. It is necessary to ensure that there is a contact in the center of each clip since that is the target contact for threshold computation.

Contact Array: All possible $m \times n$ arrays of contacts within the dimensions of clips are enumerated. The steps of the arrays can be multiple times of the minimum pitch p, i.e.,

[1] Note that the accuracy we talk about in this paper refers to the accuracy at end of lithography flow including all RETs.

Authorized licensed use limited to: University of Texas at Austin. Downloaded on September 16,2020 at 02:47:43 UTC from IEEE Xplore. Restrictions apply.

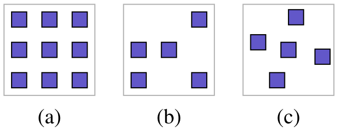


Fig. 7. (a) Clip of 3 × 3 contact array. (b) Clip of 3 × 3 randomized contact array. (c) Clip of contacts with random positions.

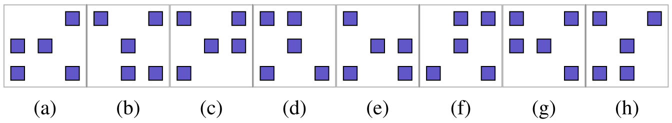


Fig. 8. Combinations of rotation and flipping. (a) Original. (b) Rotate 90°. (c) Rotate 180°. (d) Rotate 270°. (e) Flip. (f) Flip and rotate 90°. (g) Flip and rotate 180°. (h) Flip and rotate 270°.

$p, 2p, 3p, \ldots,$ in horizontal or vertical directions. An example of 3×3 contact array with a certain pitch is shown in Fig. 7(a). It needs to mention that the same 3×3 contact array with different steps should be regarded as different clips due to discrepant spacing.

Randomized Contact Array: The aforementioned contact arrays essentially distribute contacts on grids and fill all the slots in the grid maps. The randomization of contact arrays is implemented by a random distribution of contacts in those grid maps. Fig. 7(b) shows an example of randomized contact array from the 3×3 contact array in Fig. 7(a). Various distribution of contacts can be generated even from the same grid maps.

Contacts With Random Positions: Contacts in this type of clips do not necessarily align to any grid map, as their positions are randomly generated, while the design rules are still guaranteed. An example is shown in Fig. 7(c). No matter how the surrounding contacts change, the contact in the center of the clip should remain the same.

2) Data Augmentation: Due to the symmetry of optical sources in Fig. 3, data can be augmented with rotation and flipping, improving the data efficiency [37]. Eight combinations of rotation and flipping are shown in Fig. 8, where new data samples are obtained without new thresholds. Data augmentation inflates datasets to obtain models with better generalization.

B. Convolutional Neural Networks

CNNs have demonstrated impressive performance on mask related applications in lithography such as hotspot detection and resist modeling [4], [12]. The structure of CNN mainly includes convolution layers and fully connected layers. Features are extracted from convolution layers and then classification or regression is performed by fully connected layers. Fig. 11(a) illustrates a CNN structure with three convolution layers and two fully connected layers [4]. The first convolution layer has 64 filters with dimensions of 7×7. Although not explicitly shown most of the time, a rectified linear unit (ReLU) layer for activation is applied immediately after the convolution layer, where the ReLU function is defined as

$$f(x^{l-1}) = \begin{cases} x^{l-1}, & \text{if } x^{l-1} \geq 0 \\ 0, & \text{otherwise.} \end{cases} \quad (3)$$

Then the max-pooling layer performs down-sampling with a factor of 2 to reduce the feature dimensions and improve the invariance to translation [37]. After three convolution layers, two fully connected layers are applied where the first one has 256 hidden units followed with a ReLU layer and a 50% dropout layer, and second one connects to the output.

C. Residual Neural Networks

One way to improve the performance of CNN is to increase the depth for a larger capacity of the neural networks. However, the counterintuitive degradation of training accuracy in CNN is observed when stacking more layers, preventing the neural networks from better performance [38]. An example of CNNs with five and ten layers is shown in Fig. 9, where the deeper CNN fails to converge to a smaller training error than the shallow one due to gradient vanishing [39], [40], eventually resulting in the failure to achieve a better testing error either. The study from He *et al.* [38] reveals that the underlying reason comes from the difficulty of identity mapping. In other words, fitting a hypothesis $\mathcal{H}(x) = x$ is considerably difficult for solvers to find optimal solutions. To overcome this issue, ResNet, which utilizes shortcut connections, are adopted to assist the convergence of training accuracy.

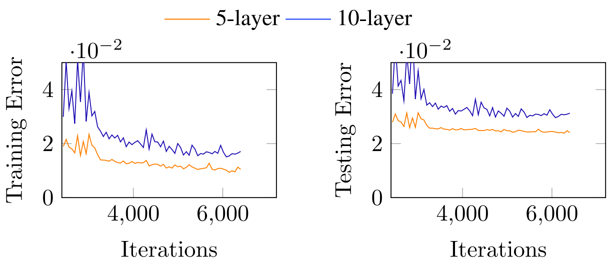
The building block of ResNet is illustrated in Fig. 10, where a shortcut connection is inserted between the input and output of two convolution layers. Let the function $\mathcal{F}(x)$ be the mapping defined by the two convolution layers. Then the entire function for the building block becomes $\mathcal{F}(x) + x$. Suppose the building block targets to fit the hypothesis $\mathcal{H}(x)$. The residual networks train $\mathcal{F}(x) = \mathcal{H}(x) - x$, while the convolution layers without shortcut connections like that in CNN try to directly fit $\mathcal{F}(x) = \mathcal{H}(x)$. Theoretically, if $\mathcal{H}(x)$ can be approximated with $\mathcal{F}(x)$, then it can also be approximated with $\mathcal{F}(x) + x$. Despite the same nature, comprehensive experiments have demonstrated a better convergence of ResNet than that of CNN for deep neural networks [38]. We also observe a better performance of ResNet with the transfer learning schemes than that of CNN in our problem, which has never been explored before.

The ResNet is shown in Fig. 11(b) with eight convolution layers and two fully connected layers. Different from the original setting [38], we add a shortcut connection to the first convolution layer by broadcasting the input tensor of $64 \times 64 \times 1$ to $64 \times 64 \times 64$. This minor change enables better empirical results in our problem. For the rest of the networks, three building blocks for ResNet are utilized.

D. Transfer Learning

Transfer learning aims at adapting the knowledge learned from data in source domains to a target domain. The transferred knowledge will benefit the learning in the target domain with a faster convergence and better generalization [37]. Suppose the data in the source domain has a distribution P_s and that in the target domain has a distribution P_t. The underlying assumption of transfer learning lies in the common factors that need to be captured for learning the variations of P_s and P_t,

Authorized licensed use limited to: University of Texas at Austin. Downloaded on September 16,2020 at 02:47:43 UTC from IEEE Xplore. Restrictions apply.



(a) (b)

Fig. 9. Counterintuitive (a) training and (b) testing errors for different depth of CNN with epochs.

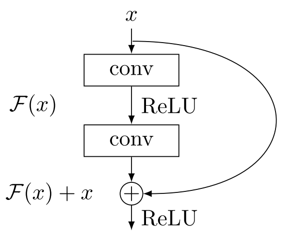


Fig. 10. Building block of ResNet.

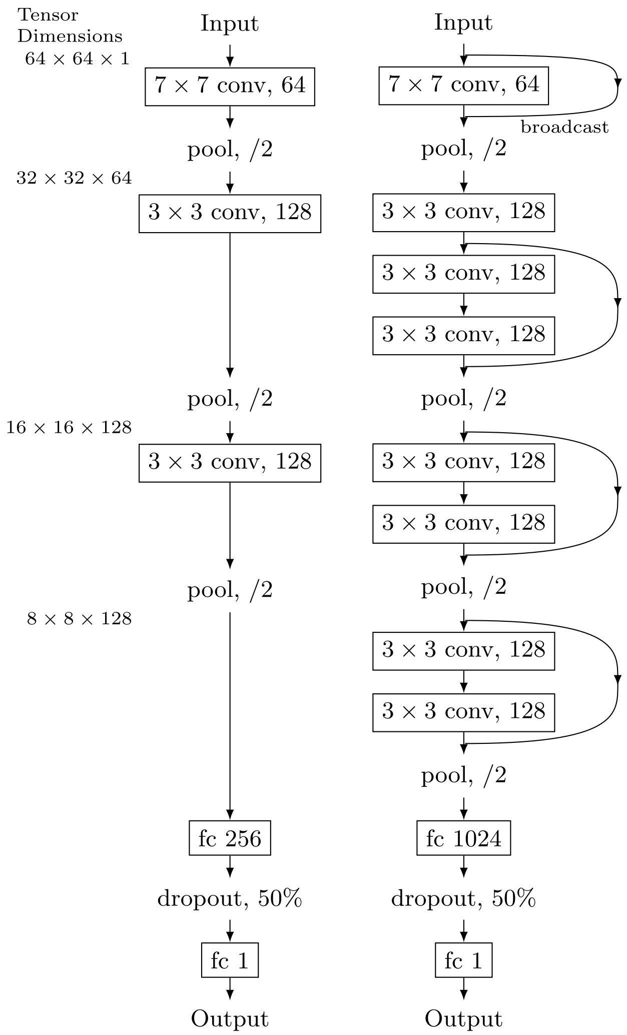
so that the knowledge for P_s is also useful for P_t. An intuitive example is that learning to recognize cats and dogs in the source task helps the recognition of ants and wasps in the target task, especially when the source task has significantly larger dataset than that of the target task. The reason comes from the low-level notions of edges, shapes, etc., shared by many visual categories [37]. In resist modeling, different lithography configurations can be viewed as separate tasks with different distributions.

Typical transfer learning scheme for neural networks fixes the first several layers of the model trained for another domain and finetune the successive layers with data from the target domain. The first several layers usually extract general features, which are considered to be similar between the source and the target domains, while the successive layers are classifiers or regressors that need to be adjusted. Fig. 12 shows an example of the transfer learning scheme. We first train a model with source domain data and then use the source domain model as the starting point for the training of the target domain. During the training for the target domain, the first k layers are fixed, while the rest layers are finetuned. We denote this scheme as TF_k, shortened from "Transfer and Fix," where k is the parameter for the number of fixed layers.

In this paper, we focus on the impacts of transfer learning and do not consider various preprocessing steps like scaling and normalization. In other words, raw aerial images are fed to the neural networks. The benefits of scaling and normalization are left to future work.

E. Active Learning With Clustering

Although transfer learning is potentially able to improve the accuracy of the target dataset using knowledge from a source dataset, selection of representative target data samples may further improve the accuracy. Let D be the unlabeled dataset in the target domain and $\boldsymbol{s}$ be the set of selected data samples for label querying, where $|\boldsymbol{s}| \leq k$ and k is the maximum number of data samples for querying. For any $(\boldsymbol{x}_i, y_i) \in D$, $\boldsymbol{x}_i$ is the feature, e.g., aerial image, and y_i is the label, e.g., threshold,



(a) (b)

Fig. 11. (a) CNN and (b) ResNet structure.

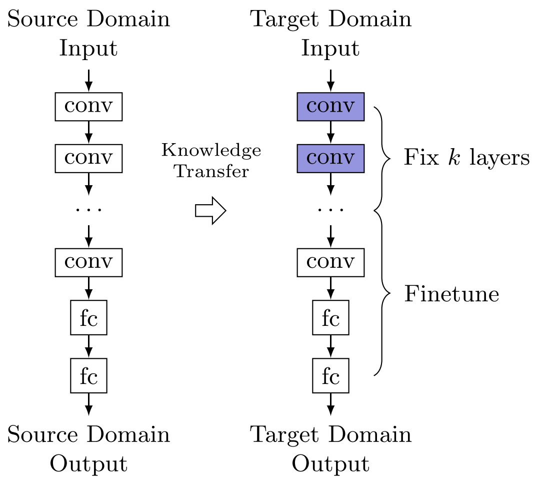


Fig. 12. Transfer learning scheme with the first k layers fixed when training for target domain, denoted as TF_k.

Authorized licensed use limited to: University of Texas at Austin. Downloaded on September 16,2020 at 02:47:43 UTC from IEEE Xplore. Restrictions apply.

where y_i is unknown for D. Consider a loss function $l(\boldsymbol{x}_i, y_i; \boldsymbol{w})$ parameterized over the hypothesis class ($\boldsymbol{w}$), e.g., parameters of a learning algorithm. The objective of active learning is to minimize the average loss of dataset D with a model trained from $\boldsymbol{s}$

$$\min_{s:|s|\le k, s\in D} \frac{1}{n}\sum_{i=1}^{n} l(\boldsymbol{x}_i, y_i; \boldsymbol{w_s}) \tag{4}$$

where $n = |D|$, and $\boldsymbol{w_s}$ represents the parameters of a model trained from $\boldsymbol{s}$.

We present an upper bound of (4) for any Lipschitz loss function and Lipschitz estimator. Then we show that both CNN and ResNet with nonlinear ReLU activations are actually Lipschitz continuous. We also assume the training loss can drop to zero, which is likely to be achieved with large enough models.

Definition 2: Let $g(\cdot;\cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, we say g is L_1-Lipschitz continuous with respect to $g(*;\cdot)$ if

$$|g(\boldsymbol{x}; \boldsymbol{w}) - g(\boldsymbol{x}'; \boldsymbol{w})| \le L_1 \cdot \|\boldsymbol{x} - \boldsymbol{x}'\|.$$

We also write $g(\boldsymbol{x}; \boldsymbol{w})$ as $g_{\boldsymbol{w}}(\boldsymbol{x})$. We use Frobenius norm for norm of a matrix here, i.e., $\|\cdot\|$.

Definition 3: Let $f(\cdot,\cdot;\cdot) : \mathbb{R}^{d_1} \times \mathbb{R} \times \mathbb{R}^{d_2} \to \mathbb{R}_{\ge 0}$, we say f is L_2-Lipschitz continuous with respect to $f(*,*;\cdot)$ if

$$|f(\boldsymbol{x}, y; \boldsymbol{w}) - f(\boldsymbol{x}', y'; \boldsymbol{w})| \le L_2 \cdot (\|\boldsymbol{x} - \boldsymbol{x}'\| + |y - y'|)$$
$$\forall \boldsymbol{x} \in \mathbb{R}^{d_1}, \forall y, y' \in \mathbb{R}, \forall \boldsymbol{w} \in \mathbb{R}^{d_2}.$$

We also write $f(\boldsymbol{x}, y; \boldsymbol{w})$ as $f_{\boldsymbol{w}}(\boldsymbol{x}, y)$.

We state the following theorem.

Theorem 1: Given n independent and identically distributed (independent identically distributed) random samples as $D = \{\boldsymbol{x}_i, y_i\}_{i\in\{1,2,\ldots,n\}}$, and a set of selected points $\boldsymbol{s}$. If the following properties hold.

1) Loss function $l(\boldsymbol{x}, y; \boldsymbol{w})$ is λ^l-Lipschitz continuous with respect to $(\boldsymbol{x}, y)$.
2) The ground truth of label $y = f(\boldsymbol{x}) + \epsilon$ has the property that $f(\cdot)$ is λ^f-Lipschitz continuous and random noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$.
3) $\hat{f}(\cdot)$ in the prediction function $\hat{y} = \hat{f}(\boldsymbol{x})$ is $\lambda^{\hat{f}}$-Lipschitz.
4) $l(\boldsymbol{x}_j, y_j; \boldsymbol{w_s}) = 0, \forall j \in \boldsymbol{s}$, where $\boldsymbol{w_s}$ is the weights of the trained model with samples $\boldsymbol{s}$, then we have the following inequality:

$$\frac{1}{n}\sum_{i\in D} l(\boldsymbol{x}_i, y_i; \boldsymbol{w_s}) \le \frac{\lambda^l\left(\lambda^f + 1\right)}{n}\sum_{j\in s}\sum_{i=1}^{k_j}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j^c\right\| + 2\lambda^l \sum_{i\in D}|\epsilon_i| \tag{5}$$

where k_j is the number of samples whose closest sample in $\boldsymbol{s}$ is $\boldsymbol{x}_j^c$; $|\epsilon_i|$ is a sample from an independent random half-normal distribution with mean $(\sigma\sqrt{2}/\sqrt{\pi})$ and variance $\sigma^2(1 - [2/\pi])$.

The left-hand side of the inequality is the average loss across the entire dataset. The right-hand side, i.e., the upper bound of the average loss, is correlated to the objective of a *K-Medoids Clustering* problem [41], where K is the number of labeled data samples for training ($K = |\boldsymbol{s}|$). K-Medoids clustering problem is required to return K clusters from a set of points as well as K centers for each cluster. Therefore, minimizing $\sum_{j\in s}\sum_{i=1}^{k_j}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|$ helps to bound the left-hand side.

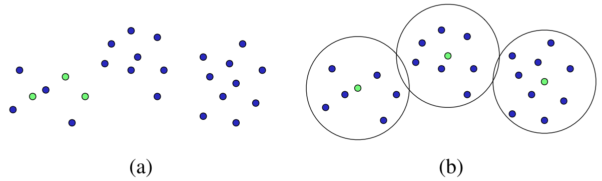


Fig. 13. Example of (a) bad data selection and (b) K-Medoids clustering selection in 2-D space. Three selected points are highlighted. Circles denote three clusters centered by selected points.

Fig. 13 provides an intuition for the K-Medoids clustering in a 2-D space. Random selection may result in biased coverage of the entire dataset, causing significant overfitting of model training. K-Medoids clustering is able to select medoids (data) evenly from the space for training such that most unselected data samples are close to their nearest medoids.

Theorem 1 requires both the loss function and the estimator to be Lipschitz continuous.

Lemma 1: If the following conditions hold:

1) $\forall i \in D$, $(\boldsymbol{x}_i, y_i)$ satisfies $\|\boldsymbol{x}_i\| \le b_1$, $|y_i| \le b_2$;
2) $\hat{f}_{\boldsymbol{w_s}}(\boldsymbol{x})$ is $\lambda^{\hat{f}}$-Lipschitz continuous with respect to $\boldsymbol{x}$;
3) $\exists(\boldsymbol{x}_0, y_0)$ such that $\hat{f}_{\boldsymbol{w_s}}(\boldsymbol{x}_0) = y_0 + \delta$, where δ is a bounded constant, then square loss function $l_{\boldsymbol{w_s}}(\boldsymbol{x}, y) = (y - \hat{f}_{\boldsymbol{w_s}}(\boldsymbol{x}))^2$ is λ^l-Lipschitz continuous, where y is the label and $\hat{f}_{\boldsymbol{w_s}}(*)$ is the learned function with parameter $\boldsymbol{w_s}$ (also denoted as $\hat{f}(*)$ for brevity)

$$\lambda^l = \left(4\lambda^{\hat{f}} b_1 + 4b_2 + 2|\delta|\right) \cdot \max\left(1, \lambda^{\hat{f}}\right). \tag{6}$$

In practice, the three assumptions are not difficult to hold. Consider the physical meaning of $\boldsymbol{x}$ and y, both $\|\boldsymbol{x}\|$ and $|y|$ are numerically small in this paper. Lemma 2 proves that CNN/ResNet is Lipschitz continuous. If the training error for CNN/ResNet is small, which is mostly true, $(\boldsymbol{x}_0, y_0)$ can be selected from the training dataset and then $|\delta|$ is also small.

Lemma 2: A CNN/ResNet for regression with n_c convolution layers (with max-pooling and ReLU) and n_{fc} fully connected layers is $(1 + \alpha\sqrt{N})^{n_c+n_{fc}}$-Lipschitz.

Detailed proofs for Theorem 1, Lemma 1, and Lemma 2 can be found in the Appendix.

K-Medoids clustering is a variation of K-Means clustering. Different from K-Medoids clustering, the centroids of K-Means clustering may not be the data points in the dataset. Despite several K-Medoids clustering algorithms [42], [43], there are stable implementations available for K-Means clustering [44], [45]. To leverage the existing implementation of K-Means clustering and reduce the development overhead, we find the nearest data points to the centroids as the medoids for the K-Medoids clustering. The algorithm is described in Algorithm 1. Empirically we observe comparable clustering costs to the dedicated K-Medoids clustering algorithm [43].

IV. Experimental Results

Our framework is implemented with Tensorflow [46] and validated on a Linux server with 3.4 GHz Intel i7 CPU

Authorized licensed use limited to: University of Texas at Austin. Downloaded on September 16,2020 at 02:47:43 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1 *K*-Medoids Clustering Using *K*-Means Engine

```
Require: A set of points $D$ and an integer $k$.
Ensure: Select a set of medoids $s$ ($|s| = k$) with minimum cost.
Solve K-Means clustering and obtain $k$ centroids denoted as $c_1, c_2, \dots, c_k$;
$s \leftarrow \emptyset$;
for $i = 0, 1, \dots, k$ do
    Find data point $j$ ($j \in D$) with minimum distance to $c_i$;
    $s \leftarrow s \cup \{j\}$;
end for
return $s$
```

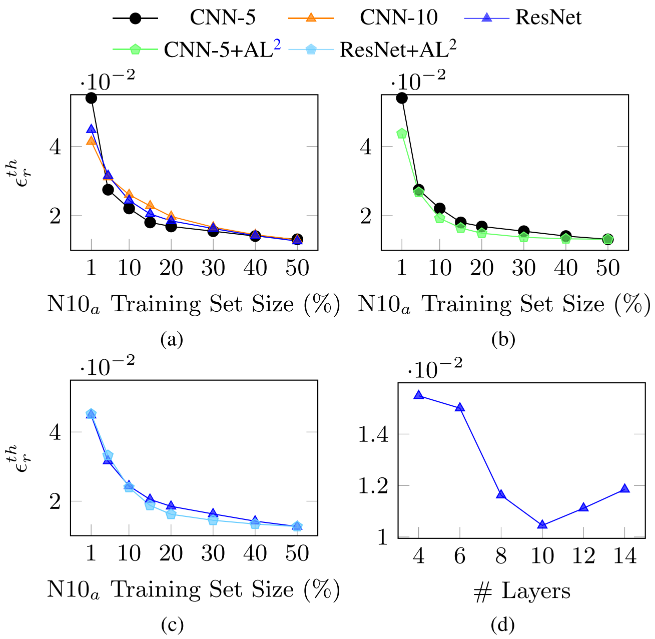
and Nvidia GTX 1080 GPU. The *K*-Medoids clustering algorithm uses the *K*-Means clustering engine in scikit-learn [45]. We observe that this approach provides better and more stable objectives of *K*-Medoids clustering than does dedicated *K*-Medoids clustering solver in PyClust package in our experiments.

Around 980 mask clips are generated according to Section III-A for N10 and N7 separately following the design rules in Section II-B, respectively. N7$_a$ and N7$_b$ use the same set of clips, but different lithography configurations. SRAF, OPC, and aerial image simulation are performed with Mentor Calibre [23]. The golden CD values are obtained from rigorous simulation using Synopsys Sentaurus Lithography models [47] calibrated from manufactured data for N10, N7$_a$, and N7$_b$ according to Table I. Then golden thresholds are extracted. Each clip has four thresholds as shown in Fig. 4. Hence the N10 dataset contains 3928 samples and each N7 dataset contains 3916 samples, respectively. The data augmentation technique in Section III-A2 is applied, so the training set and the testing set will be augmented by a factor of 8 independently. For example, if 50% of the data for N10 are used for training, then there are 3928 × 50% × 8 = 15712 samples. It needs to mention that always the same 50% portions are used during the validation of a dataset for fair comparison of different techniques. The batch size is set to 32 for training accommodating to the large variability in the sizes of training datasets. Adam [48] is used as the stochastic optimizer and maximum epoch is set to 200 for training.

The training time for one model takes 10 to 40 min according to the portions of a dataset used for training, and prediction time for an entire N10 or N7 dataset takes less than 10 s, while the rigorous simulation takes more than 15 h for each N10 or N7 dataset. Thus, we no longer report the prediction time which is negligible compared with that of the rigorous simulation. Each experiment runs ten different random seeds and averages the numbers.

A. CNN and ResNet

We first compare CNN and ResNet in Fig. 14(a). Column "CNN-5" denotes the network with five layers shown in Fig. 11(a). Column "CNN-10" denotes the one with ten layers that has the same structure as that in Fig. 11(b) but without shortcut connections. Column "ResNet" denotes the one with ten layers shown in Fig. 11(b). When using 1% to 20% training data, ResNet shows better average relative RMS error ϵ_r^{th} than CNN-10, but CNN-5 provides the best error. We will show later that ResNet on the contrary outperforms CNN-5 when transfer learning is incorporated.



(a) (b) (c) (d)

Fig. 14. (a) Comparison on testing accuracy of CNN-5, CNN-10, and ResNet on N10. (b) Testing accuracy of CNN with active learning on N10. (c) Testing accuracy of ResNet with active learning on N10. (d) Impact of depth on the testing accuracy of ResNet.

We then show the performance of active learning for CNN and ResNet in Fig. 14(b) and (c), denoted as "CNN-5+AL" and "ResNet+AL," respectively. The beneficial amount of training data for active data selection is from 10% to 40%. For example, for 20% training data, it provides 11.6% accuracy improvement for CNN and 12.5% for ResNet; for 30% training data, it provides 11.4% improvement for CNN and 11.3% for ResNet [49]. The benefit of active learning is not significant for extremely small training dataset, e.g., 1% and 5%. When there are very few training data, it is more likely for randomly selected data samples to distribute quite some distance away than to squeeze as small clusters. Although active selection of data can avoid corner cases of extremely poor sampling, e.g., all data samples squeezing as a small cluster, while it is difficult to demonstrate the benefit of active learning in ordinary cases. On the other hand, when the amount of training data increases, the benefit from active learning drops due to sufficient coverage. The rightmost points take all 50% training data and thus show the same accuracy as that without active learning.

The impacts of depth on the performance of ResNet are further explored in Fig. 14(d), where we gradually stack more building blocks in Fig. 10 before fully connected layers. The *x*-axis denotes total number of convolution and fully connected layers corresponding to different numbers of building blocks. For instance, 0 building block leads to four layers and three building blocks result in ten layers [Fig. 11(b)]. The testing error decreases to lowest value at ten layers and then starts to increase, indicating potential overfitting afterwards [37]. Therefore, we use ten layers for the ResNet in the experiment.

B. Knowledge Transfer From N10 to N7

We then compare the testing accuracy between knowledge transfer from N10 to N7 and directly training from N7 datasets in Fig. 15(a). In this example, the *x*-axis represents the percentage of training dataset for the target domain N7$_a$, while the percentage of data from the source domain N10 is always 50%. Similar trends are also observed for N7$_b$. Curve "CNN" denotes training the CNN of five layers in Fig. 11(a) with data from target domain only, i.e., no transfer learning involved. Curve "CNN TF$_0$" denotes the transfer learning scheme in Section III-D for the same CNN with zero layer fixed. Curve "ResNet TF$_0$" denotes applying the same scheme to ResNet. The most significant benefit of transfer learning comes from small training dataset with a range of 1% to 20%, where there are around 52% to 18% improvement in the accuracy from CNN. Meanwhile, ResNet TF$_0$ can achieve an average of 13% smaller error than CNN TF$_0$.

Fig. 15(b) further compares the results of fixing different numbers of layers during transfer learning. In this case, ResNet TF$_0$ and ResNet TF$_4$ have the best accuracy, while the error increases with more layers fixed. It is indicated that the tasks N10 and N7 are quite different and both feature extraction layers and regression layers need finetuning.

In Fig. 15(c), we enable transfer learning plus active learning, which provides 7% to 11% additional accuracy improvement for 10% to 40% amount of training data from the target domain.

C. Knowledge Transfer Within N7

The transfer learning between different N7 datasets, e.g., from N7$_a$ to N7$_b$, is also explored in Fig. 16. The *x*-axis represents the percentage of training dataset for the target domain N7$_b$, while the percentage of data from the source domain N7$_a$ is always 50%. Compared with the knowledge transfer from N10 to N7, we achieve even higher accuracy between 1% and 20% training datasets in Fig. 16(a). For example, with 1% training dataset, there is around 65% improvement in accuracy from CNN, and with 20% training dataset, the improvement is around 23%. ResNet TF$_0$ keeps having lower errors than that of CNN TF$_0$ as well, with an average benefit around 15%.

The curves in Fig. 16(b) show different insights from that of the knowledge transfer from N10 to N7. The accuracy of ResNet TF$_0$ can be further improved with more layers fixed, e.g., ResNet TF$_8$, by around 28% to 14%. This is reasonable since N7$_a$ and N7$_b$ have the same design rules and illumination shapes, and the only difference lies in the resist materials. Therefore, the feature extraction layers are supposed to remain almost the same. With the sizes of the training dataset increasing to 15% and 20%, the differences in the accuracy become smaller, because there are enough data to find good configurations for the networks. Since knowledge transfer is remarkably effective with ResNet TF$_8$, we do not see the room for further improvement with active learning. Thus we did not plot the curves for that.

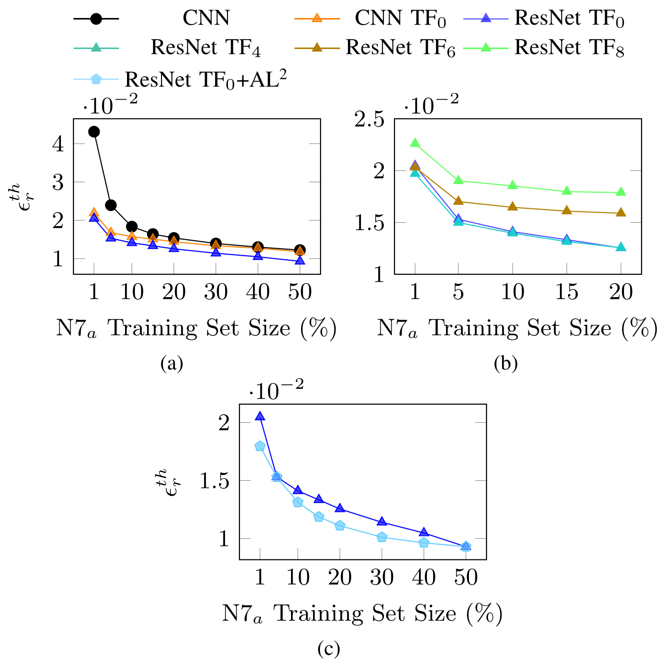


Fig. 15. Testing accuracy of transfer learning from N10 to N7$_a$. (a) Comparison between CNN and transfer learning. (b) Comparison between transfer learning schemes where different numbers of layers are fixed. (c) Comparison between transfer learning only and transfer learning plus active learning for ResNet.

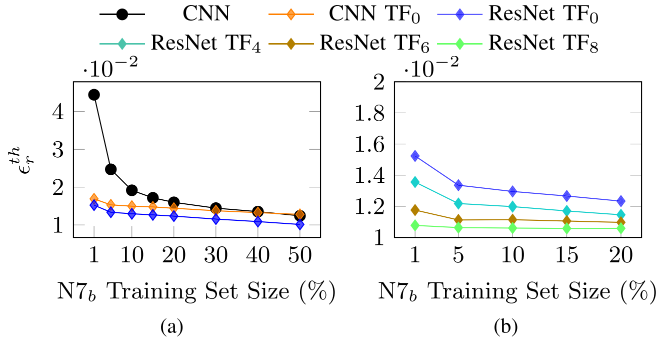


Fig. 16. Testing accuracy of transfer learning from N7$_a$ to N7$_b$. (a) Comparison between CNN and transfer learning. (b) Comparison between transfer learning schemes where different numbers of layers are fixed.

D. Impact of Various Source Domains

In transfer learning, the correlation between the datasets of source and target domains is critical to the effectiveness of knowledge transfer. Thus, we explore the impacts of source domain datasets on the accuracy of modeling for the target domain. Fig. 17 plots the testing errors of learning N7$_b$ using ResNet TF$_0$ with various source domain datasets. Curves "N10$^{50\%}$" and "N7$_a^{50\%}$" indicate that 50% of the N10 or the N7$_a$ dataset is used to train source domain models, respectively. Curve "N10$^{50\%}$ + N7$_a^{1\%}$" describes the situation where we have 50% of the N10 dataset and 1% of the N7$_a$ dataset

[1] Results for active learning extended from [49].

Authorized licensed use limited to: University of Texas at Austin. Downloaded on September 16,2020 at 02:47:43 UTC from IEEE Xplore. Restrictions apply.

TABLE II
RELATIVE THRESHOLD RMS ERROR AND CD RMS ERROR FOR N7$_b$ WITH DIFFERENT SOURCE DOMAIN DATASETS

Source Datasets		$\emptyset$		N10$^{50\%}$						N7$_a^{50\%}$				N10$^{50\%}$ + N7$_a^{5\%}$		N10$^{50\%}$ + N7$_a^{10\%}$	
Neural Networks		CNN		CNN TF$_0$		ResNet TF$_0$		ResNet TF$_0$+AL^2		CNN TF$_0$		ResNet TF$_0$		ResNet TF$_0$		ResNet TF$_0$	
		ϵ_r^{th} (10^{-2})	ϵ^{CD}	ϵ_r^{th} (10^{-2})	ϵ^{CD}	ϵ_r^{th} (10^{-2})	ϵ^{CD}	ϵ_r^{th} (10^{-2})	ϵ^{CD}	ϵ_r^{th} (10^{-2})	ϵ^{CD}	ϵ_r^{th} (10^{-2})	ϵ^{CD}	ϵ_r^{th} (10^{-2})	ϵ^{CD}	ϵ_r^{th} (10^{-2})	ϵ^{CD}
N7$_b$	1%	4.44	4.76	2.34	2.48	2.29	2.39	1.95	2.09	1.69	1.79	1.52	1.60	1.94	2.03	1.82	1.91
	5%	2.78	2.96	1.73	1.86	1.60	1.70	1.58	1.70	1.53	1.64	1.34	1.43	1.67	1.78	1.57	1.67
	10%	1.92	2.04	1.63	1.76	1.47	1.57	1.36	1.48	1.50	1.60	1.30	1.38	1.50	1.60	1.51	1.61
	15%	1.72	1.84	1.56	1.68	1.39	1.47	1.23	1.32	1.48	1.55	1.27	1.35	1.41	1.50	1.43	1.52
	20%	1.60	1.71	1.50	1.61	1.31	1.39	1.16	1.24	1.44	1.55	1.23	1.31	1.32	1.41	1.34	1.43
ratio		1.00	1.00	0.77	0.77	0.70	0.69	0.63	0.64	0.69	0.69	0.60	0.60	0.69	0.69	0.69	0.68

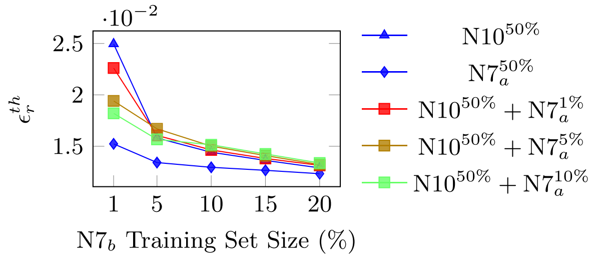


Fig. 17. Testing accuracy of ResNet TF$_0$ for N7$_b$ from different source domain datasets.

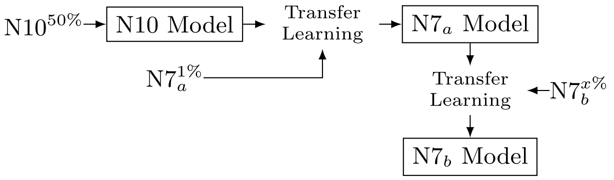
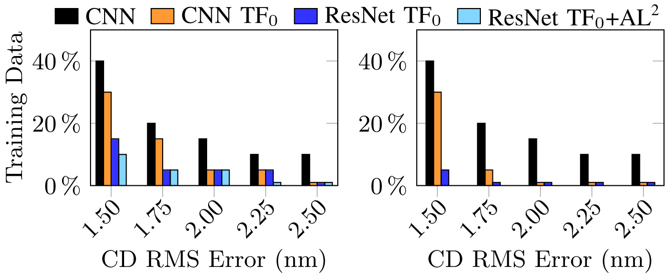


Fig. 18. Transfer learning from 50% of N10 dataset and 1% of N7$_a$ dataset (i.e., N10$^{50\%}$ + N7$_a^{1\%}$) to N7$_b$ with x% of N7$_b$ dataset.



(a) (b)

Fig. 19. Amount of training data required for N7$_b$ given target CD RMS errors when (a) 50% N10 dataset is available or (b) 50% N7$_a$ dataset is available.

for training. In this case, as shown in Fig. 18, we first use the 50% N10 data to train the first source domain model; then train the second source domain model using the first model as the starting point with the 1% N7$_a$ data; in the end, the target domain model for N7$_b$ is trained using the second model as the starting point with N7$_b$ data. Curves "N10$^{50\%}$+N7$_a^{5\%}$" and "N10$^{50\%}$+N7$_a^{10\%}$" are similar, simply with different amounts of N7$_a$ data for training.

The knowledge from N7$_a^{50\%}$ is the most effective for N7$_b$ due to the minor difference in resist materials between two datasets. For the rest curves, the accuracy of N10$^{50\%}$ + N7$_a^{5\%}$ and N10$^{50\%}$ + N7$_a^{10\%}$ is in general better than or at least comparable to that of N10$^{50\%}$. This indicates that having more data from closer datasets to the target dataset, e.g., N7$_a$, is still helpful.

E. Improvement in Data Efficiency

Table II presents the accuracy metrics, i.e., relative threshold RMS error (ϵ_r^{th}) and CD RMS error (ϵ^{CD}), for learning N7$_b$ from various source domain datasets. Since we consider the data efficiency of different learning schemes, we focus on the small training dataset for N7$_b$, from 1% to 20%. Situations such as no source domain data ($\emptyset$), only source domain data from N10 (N10$^{50\%}$), only source domain data from N7$_a$ (N7$_a^{50\%}$), and combined source domain datasets, are examined. As mentioned in Section II, the fidelity between relative threshold RMS error and CD RMS error is very consistent, so they share almost the same trends. Transfer learning with any source domain dataset enables an average improvement of 23% to 40% from that without knowledge transfer. In small training datasets of N7$_b$, ResNet also achieves around 8% better performance on average than CNN in the transfer learning scheme. Enabling active learning together with transfer learning allows additional 5% accuracy improvement on average compared with transfer learning only for ResNet. At 1% of N7$_b$, combined source domain datasets have better performance compared with N10$^{50\%}$ only, but the benefits vanish with the increase of the N7$_b$ dataset.

In real manufacturing, models are usually calibrated to satisfy a target accuracy or target CD RMS error. Fig. 19 demonstrates the amount of training data required in the target domain for learning the N7$_b$ model. Curve "CNN" does not involve any knowledge transfer, while curves "CNN TF$_0$" and "ResNet TF$_0$" utilize transfer learning in CNN and ResNet, respectively. The curves in Fig. 19(a) assume the availability of N10 data. Consider the CD RMS error from 1.5 nm to 2.5 nm, which is around 10% of the half pitch for N7 contacts. This range of accuracy is also comparable to that of the state-of-the-art CNN [4]. ResNet TF$_0$ requires significantly fewer data than both CNN and CNN TF$_0$. For instance, when the target CD error is 1.75 nm, ResNet TF$_0$ demands 5% training data from N7$_b$, while CNN requires 20% and CNN TF$_0$ requires 15%. By enabling active learning, ResNet TF$_0$+AL further reduces data requirement from ResNet TF$_0$, e.g., 1.5× and 4× fewer training data than ResNet TF$_0$ and CNN for 1.5 nm, respectively. Fig. 19(b) considers the transfer from

Authorized licensed use limited to: University of Texas at Austin. Downloaded on September 16,2020 at 02:47:43 UTC from IEEE Xplore. Restrictions apply.

N7$_a$ to N7$_b$. Both ResNet TF$_0$ and CNN TF$_0$ only require 1% training data from N7$_b$ for most target CD RMS errors, where CNN TF$_0$ cannot achieve the accuracy unless given 30% data. Overall, ResNet TF$_0$ can achieve 3×–10× reduction of training data within this range compared with CNN. It needs to mention that 1% of dataset only correspond to fewer than 40 samples owing to the data augmentation, indicating only thresholds of 40 clips are required.

V. Conclusion

A transfer learning framework with a clustering-based active data selection on ResNets is proposed for resist modeling. The combination of transfer learning and active learning for ResNet is able to achieve high accuracy with very few data from the target domain, under various situations for knowledge transfer, indicating high data efficiency. Extensive experiments demonstrate that the proposed techniques can achieve 3×–10× reduction according to various requirements of accuracy comparable to the state-of-the-art learning approach. It is shown that the performance of transfer learning differs from dataset to dataset and is worth exploring to see the correlation between datasets. Active selection of data samples is also useful to guide the generation of mask designs for model calibration in manufacturing. Examining the quantitative relation between the correlation of datasets and performance of transfer learning is valuable in the future. There is still room to improve the effectiveness of knowledge transfer from N10 to N7 datasets. Therefore, in the future, we will actively explore other learning techniques to further improve the accuracy, such as preprocessing steps like scaling and normalization, various regularization techniques, and semi-supervised learning.

Appendix

Proof of Lemma 1: We first bound $\left|\hat{f}(\boldsymbol{x}_i)\right|$

$$\begin{aligned}\left|\hat{f}(\boldsymbol{x}_i)\right| &\le \left|\hat{f}(\boldsymbol{x}_0)\right| + \left|\hat{f}(\boldsymbol{x}_i) - \hat{f}(\boldsymbol{x}_0)\right| \\ &\le |y_0| + |\delta| + \lambda^{\hat{f}}\|\boldsymbol{x}_i - \boldsymbol{x}_0\| \\ &\le b_2 + |\delta| + \lambda^{\hat{f}}(\|\boldsymbol{x}_i\| + \|\boldsymbol{x}_0\|) \\ &\le b_2 + |\delta| + 2\lambda^{\hat{f}} b_1. \end{aligned} \tag{7}$$

Then prove the Lipschitz-continuity of square loss function

$$\begin{aligned}&\left|l_{\boldsymbol{w}_s}(\boldsymbol{x}_i, y_i) - l_{\boldsymbol{w}_s}\left(\boldsymbol{x}_j, y_j\right)\right| \\ &= \left|\left(y_i - \hat{f}(\boldsymbol{x}_i)\right)^2 - \left(y_j - \hat{f}\left(\boldsymbol{x}_j\right)\right)^2\right| \\ &\le \left|y_i - y_j - \hat{f}(\boldsymbol{x}_i) + \hat{f}\left(\boldsymbol{x}_j\right)\right|\left|y_i + y_j - \hat{f}(\boldsymbol{x}_i) - \hat{f}\left(\boldsymbol{x}_j\right)\right| \\ &\le \left(\left|y_i - y_j\right| + \left|\hat{f}(\boldsymbol{x}_i) - \hat{f}\left(\boldsymbol{x}_j\right)\right|\right) \\ &\quad \times \left(|y_i| + \left|y_j\right| + \left|\hat{f}(\boldsymbol{x}_i)\right| + \left|\hat{f}\left(\boldsymbol{x}_j\right)\right|\right) \\ &\le \left(\left|y_i - y_j\right| + \lambda^{\hat{f}}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|\right)\left(4b_2 + 2|\delta| + 4\lambda^{\hat{f}} b_1\right) \\ &\le \left(4\lambda^{\hat{f}} b_1 + 4b_2 + 2|\delta|\right) \cdot \max\left(1, \lambda^{\hat{f}}\right) \\ &\quad \times \left(\left|y_i - y_j\right| + \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|\right). \end{aligned} \tag{8}$$

■

Proof of Lemma 2: We assume the output after a series of convolution and fully connected layers is the prediction of the CNN for regression. Consider two inputs $\boldsymbol{x}$ and $\boldsymbol{x}'$, with their representation $\boldsymbol{x}^{(d)}$ and $\boldsymbol{x}'^{(d)}$. We first show the Lipschitz property of convolution layers and fully connected layers. Any convolution or fully connected layer can be denoted as $\boldsymbol{x}_j^{(d)} = \sum_i w_{i,j}^{(d)} \boldsymbol{x}_i^{(d-1)}$. By assuming $\sum_i |w_{i,j}^{(d)}| \le \alpha, \forall i, j, d$, we can state

$$\left\|\boldsymbol{x}_j^{(d)} - \boldsymbol{x}_j'^{(d)}\right\| \le \alpha\left\|\boldsymbol{x}^{(d-1)} - \boldsymbol{x}'^{(d-1)}\right\|. \tag{9}$$

Let $n^{(d)}$ be the dimension of $\boldsymbol{x}^{(d)}$, which is bounded by N

$$\begin{aligned}\left\|\boldsymbol{x}^{(d)} - \boldsymbol{x}'^{(d)}\right\| &= \sqrt{\sum_j \left\|\boldsymbol{x}_j^{(d)} - \boldsymbol{x}_j'^{(d)}\right\|^2} \\ &\le \sqrt{n^{(d)}\alpha^2\left\|\boldsymbol{x}^{(d-1)} - \boldsymbol{x}'^{(d-1)}\right\|^2} \\ &\le \alpha\sqrt{n^{(d)}}\left\|\boldsymbol{x}^{(d-1)} - \boldsymbol{x}'^{(d-1)}\right\| \\ &\le \alpha\sqrt{N}\left\|\boldsymbol{x}^{(d-1)} - \boldsymbol{x}'^{(d-1)}\right\|. \end{aligned} \tag{10}$$

We then consider ReLU and max-pooling layers. For any ReLU layer, it is straightforward to verify the following inequality:

$$|\max(0, a) - \max(0, b)| \le |a - b|. \tag{11}$$

Any max-pooling layer can be viewed as a convolution layer in which only one weight is 1 and others are 0. Thus, we can state for ReLU and max-pooling layers

$$\left\|\boldsymbol{x}^{(d)} - \boldsymbol{x}'^{(d)}\right\| \le \left\|\boldsymbol{x}^{(d-1)} - \boldsymbol{x}'^{(d-1)}\right\|. \tag{12}$$

Combining the Lipschitz property of all layers of CNN

$$\left\|\mathrm{CNN}(\boldsymbol{x}; \boldsymbol{w}) - \mathrm{CNN}(\boldsymbol{x}'; \boldsymbol{w})\right\| \le (\alpha\sqrt{N})^{n_c + n_{fc}}\left\|\boldsymbol{x} - \boldsymbol{x}'\right\| \tag{13}$$

where $\boldsymbol{w}$ is the weights for CNN.

For ResNet, a shortcut connection can be viewed as a layer d which takes input from layer $d - 1$ and layer d', i.e., $\boldsymbol{x}^{(d)} = \boldsymbol{x}^{(d-1)} + \boldsymbol{x}^{(d')}$, where $d - 1 > d'$. Then we can state

$$\begin{aligned}\left\|\boldsymbol{x}^{(d)} - \boldsymbol{x}^{(d)}\right\| &= \left\|\boldsymbol{x}^{(d-1)} + \boldsymbol{x}^{(d')} - \boldsymbol{x}'^{(d-1)} - \boldsymbol{x}'^{(d')}\right\| \\ &\le \left\|\boldsymbol{x}^{(d-1)} - \boldsymbol{x}'^{(d-1)}\right\| + \left\|\boldsymbol{x}^{(d')} - \boldsymbol{x}'^{(d')}\right\| \\ &\le \left((\alpha\sqrt{N})^{d-d'-1} + 1\right)\left\|\boldsymbol{x}^{(d')} - \boldsymbol{x}'^{(d')}\right\| \\ &\le (1 + \alpha\sqrt{N})^{d-d'-1}\left\|\boldsymbol{x}^{(d')} - \boldsymbol{x}'^{(d')}\right\|. \end{aligned} \tag{14}$$

Therefore, combining all layers of ResNet

$$\begin{aligned}&\left\|\mathrm{ResNet}(\boldsymbol{x}; \boldsymbol{w}) - \mathrm{ResNet}(\boldsymbol{x}'; \boldsymbol{w})\right\| \\ &\qquad \le (1 + \alpha\sqrt{N})^{n_c + n_{fc}}\left\|\boldsymbol{x} - \boldsymbol{x}'\right\|. \end{aligned} \tag{15}$$

We combine (13) and (15) for generalization to both CNN and ResNet. ■

Proof of Theorem 1:

$$\left|l_{\boldsymbol{w}_s}(\boldsymbol{x}_i, y_i) - l_{\boldsymbol{w}_s}\left(\boldsymbol{x}_j, y_j\right)\right| \overset{(a)}{\le} \lambda^l\left(\left|y_i - y_j\right| + \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|\right). \tag{16}$$

Authorized licensed use limited to: University of Texas at Austin. Downloaded on September 16,2020 at 02:47:43 UTC from IEEE Xplore. Restrictions apply.

Inequality (a) uses the Lipschitz property of the loss function

$$\begin{aligned}\left|y_i - y_j\right| &= \left|f(\boldsymbol{x}_i) + \epsilon_i - f(\boldsymbol{x}_j) - \epsilon_j\right| \\ &\leq \left|f(\boldsymbol{x}_i) - f(\boldsymbol{x}_j)\right| + |\epsilon_i| + \left|\epsilon_j\right| \\ &\overset{(b)}{\leq} \lambda^f \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\| + |\epsilon_i| + \left|\epsilon_j\right|. \qquad (17)\end{aligned}$$

Inequality (b) uses the Lipschitz property of the ground truth function f.

Combine previous two inequalities (16) and (17), we have

$$\begin{aligned}&\left|l_{\boldsymbol{w}_s}(\boldsymbol{x}_i, y_i) - l_{\boldsymbol{w}_s}(\boldsymbol{x}_j, y_j)\right| \\ &\leq \lambda^l\Big(\lambda^f \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\| + |\epsilon_i| + \left|\epsilon_j\right| + \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|\Big) \\ &= \lambda^l\Big(\lambda^f + 1\Big)\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\| + \lambda^l\big(|\epsilon_i| + \left|\epsilon_j\right|\big). \qquad (18)\end{aligned}$$

Denote the selected data samples as $(\boldsymbol{x}_j^c, y_j^c), \forall j \in \boldsymbol{s}$. Then assign each point i of the entire dataset to a cluster centered by its nearest selected data sample j, and suppose that there are k_j points within the cluster. Then the average loss of all data points is bounded as follows:

$$\begin{aligned}&\frac{1}{n}\sum_{i\in D} l_{\boldsymbol{w}_s}(\boldsymbol{x}_i, y_i) \\ &= \frac{1}{n}\sum_{j\in \boldsymbol{s}}\sum_{i=1}^{k_j} l_{\boldsymbol{w}_s}(\boldsymbol{x}_i, y_i) \\ &\overset{(c)}{\leq} \frac{1}{n}\sum_{j\in \boldsymbol{s}}\sum_{i=1}^{k_j}\Big(l_{\boldsymbol{w}_s}\Big(\boldsymbol{x}_j^c, y_j^c\Big) + \Big|l_{\boldsymbol{w}_s}(\boldsymbol{x}_i, y_i) - l_{\boldsymbol{w}_s}\Big(\boldsymbol{x}_j^c, y_j^c\Big)\Big|\Big) \\ &\overset{(d)}{\leq} \frac{1}{n}\sum_{j\in \boldsymbol{s}}\sum_{i=1}^{k_j}\Big(0 + \Big|l_{\boldsymbol{w}_s}(\boldsymbol{x}_i, y_i) - l_{\boldsymbol{w}_s}\Big(\boldsymbol{x}_j^c, y_j^c\Big)\Big|\Big) \\ &\overset{(e)}{\leq} \frac{1}{n}\sum_{j\in \boldsymbol{s}}\sum_{i=1}^{k_j}\Big(\lambda^l\Big(\lambda^f + 1\Big)\left\|\boldsymbol{x}_i - \boldsymbol{x}_j^c\right\| + \lambda^l\Big(|\epsilon_i| + \left|\epsilon_j^c\right|\Big)\Big) \\ &= \frac{\lambda^l\big(\lambda^f + 1\big)}{n}\sum_{j\in \boldsymbol{s}}\sum_{i=1}^{k_j}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j^c\right\| + \frac{\lambda^l}{n}\sum_{j\in \boldsymbol{s}}\sum_{i=1}^{k_j}\Big(|\epsilon_i| + \left|\epsilon_j^c\right|\Big) \\ &= \frac{\lambda^l\big(\lambda^f + 1\big)}{n}\sum_{j\in \boldsymbol{s}}\sum_{i=1}^{k_j}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j^c\right\| \\ &\quad + \frac{\lambda^l}{n}\sum_{i\in D}|\epsilon_i| + \frac{\lambda^l}{n}\sum_{i\in \boldsymbol{s}} k_i\left|\epsilon_i^c\right| \\ &= \frac{\lambda^l\big(\lambda^f + 1\big)}{n}\sum_{j\in \boldsymbol{s}}\sum_{i=1}^{k_j}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j^c\right\| + \frac{\lambda^l}{n}\sum_{i\in D}\alpha_i|\epsilon_i| \\ &\quad \text{where } \alpha_i = \begin{cases} 1, & i \in D \setminus \boldsymbol{s} \\ k_i + 1, & i \in \boldsymbol{s} \end{cases} \\ &\overset{(f)}{\leq} \frac{\lambda^l\big(\lambda^f + 1\big)}{n}\sum_{j\in \boldsymbol{s}}\sum_{i=1}^{k_j}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j^c\right\| + \frac{\lambda^l}{n}\sum_{i\in D}\alpha_i\sum_{i\in D}|\epsilon_i| \\ &\overset{(g)}{\leq} \frac{\lambda^l\big(\lambda^f + 1\big)}{n}\sum_{j\in \boldsymbol{s}}\sum_{i=1}^{k_j}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j^c\right\| + 2\lambda^l\sum_{i\in D}|\epsilon_i| \qquad (19)\end{aligned}$$

Inequality (c) utilizes the fact that $a - b \leq \|a - b\|$. Inequality (d) uses the zero loss assumption. Inequality (e) embeds (18). We assume ϵ_i and ϵ_j^c follow the same normal distribution, because they come from the same dataset. Inequality (f) leverages the fact that $\sum_i a_i b_i \leq \sum_i a_i \sum_i b_i, \forall a_i, b_i \geq 0$. Inequality (g) cancels out n by $\sum_{i\in D}\alpha_i = 2n$, where $|\epsilon_i|$ is a sample from an independent random half-normal distribution with mean $\sigma\sqrt{(2/\pi)}$ and variance $\sigma^2(1 - [2/\pi])$.

Acknowledgment

The authors would like to thank Memory Lithography Group from Toshiba Memory Corporation and Dr. K. Zhong from the Computer Science Department of UT Austin for helpful discussions and feedback.

References

[1] L. Liebmann, A. Chu, and P. Gutwin, "The daunting complexity of scaling to 7NM without EUV: Pushing DTCO to the extreme," in *Proc. SPIE*, vol. 9427. San Jose, CA, USA, 2015, Art. no. 942702.

[2] L. Liebmann *et al.*, "Overcoming scaling barriers through design technology cooptimization," in *Proc. IEEE Symp. VLSI Technol.*, 2016, pp. 1–2.

[3] S. Shim, S. Choi, and Y. Shin, "Machine learning-based resist 3D model," in *Proc. SPIE*, vol. 10147. San Jose, CA, USA, 2017, Art. no. 101471D.

[4] Y. Watanabe, T. Kimura, T. Matsunawa, and S. Nojima, "Accurate lithography simulation model based on convolutional neural networks," in *Proc. SPIE Adv. Lithography*, San Jose, CA, USA, 2017, Art. no. 101470K.

[5] X. Ma *et al.*, "Fast lithography aerial image calculation method based on machine learning," *Appl. Opt.*, vol. 56, no. 23, pp. 6485–6495, 2017.

[6] J.-Y. Wuu, F. G. Pikus, and M. Marek-Sadowska, "Efficient approach to early detection of lithographic hotspots using machine learning systems and pattern matching," in *Proc. SPIE Adv. Lithography*, San Jose, CA, USA, 2011, Art. no. 79740U.

[7] T. Matsunawa, S. Nojima, and T. Kotani, "Automatic layout feature extraction for lithography hotspot detection based on deep neural network," in *Proc. SPIE*, vol. 9781. San Jose, CA, USA, 2016, Art. no. 97810H.

[8] T. Matsunawa, B. Yu, and D. Z. Pan, "Laplacian eigenmaps-and Bayesian clustering-based layout pattern sampling and its applications to hotspot detection and optical proximity correction," *J. Micro Nanolithography MEMS MOEMS*, vol. 15, no. 4, 2016, Art. no. 043504.

[9] H. Zhang, B. Yu, and E. F. Young, "Enabling online learning in lithography hotspot detection with information-theoretic feature optimization," in *Proc. 35th Int. Conf. Comput.-Aided Design*, 2016, p. 47.

[10] M. Shin and J.-H. Lee, "Accurate lithography hotspot detection using deep convolutional neural networks," *J. Micro Nanolithography MEMS MOEMS (JM3)*, vol. 15, no. 4, 2016, Art. no. 043507.

[11] H. Yang, L. Luo, J. Su, C. Lin, and B. Yu, "Imbalance aware lithography hotspot detection: A deep learning approach," *J. Micro Nanolithography MEMS MOEMS*, vol. 16, no. 3, 2017, Art. no. 033504.

[12] H. Yang *et al.*, "Layout hotspot detection with feature tensor generation and deep biased learning," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, to be published, doi: 10.1109/TCAD.2018.2837078.

[13] H. Yang, Y. Lin, B. Yu, and F. E. Young, "Lithography hotspot detection: From shallow to deep learning," in *Proc. IEEE Int. Syst. Chip Conf. (SOCC)*, 2017, pp. 233–238.

[14] H. Zhang, F. Zhu, H. Li, E. F. Young, and B. Yu, "Bilinear lithography hotspot detection," in *Proc. ACM Int. Symp. Phys. Design*, 2017, pp. 7–14.

[15] Y. Lin, X. Xu, J. Ou, and D. Z. Pan, "Machine learning for mask/wafer hotspot detection and mask synthesis," in *Proc. Photomask Technol.*, vol. 10451, 2017, Art. no. 104510A.

[16] H. Yang, S. Li, C. Tabery, B. Lin, and B. Yu, "Bridging the gap between layout pattern sampling and hotspot detection via batch active learning," *ArXiv e-prints*, Jul. 2018. [Online]. Available: http://adsabs.harvard.edu/abs/2018arXiv180706446Y

[17] A. Gu and A. Zakhor, "Optical proximity correction with linear regression," *IEEE Trans. Semicond. Manuf.*, vol. 21, no. 2, pp. 263–271, May 2008.

Authorized licensed use limited to: University of Texas at Austin. Downloaded on September 16,2020 at 02:47:43 UTC from IEEE Xplore. Restrictions apply.

[18] N. Jia and E. Y. Lam, "Machine learning for inverse lithography: Using stochastic gradient descent for robust photomask synthesis," *J. Opt.*, vol. 12, no. 4, 2010, Art. no. 045601.

[19] R. Luo, "Optical proximity correction using a multilayer perceptron neural network," *J. Opt.*, vol. 15, no. 7, 2013, Art. no. 075708.

[20] T. Matsunawa, B. Yu, and D. Z. Pan, "Optical proximity correction with hierarchical Bayes model," *J. Micro Nanolithography MEMS MOEMS*, vol. 15, no. 2, 2016, Art. no. 021009.

[21] X. Xu *et al.*, "Subresolution assist feature generation with supervised data learning," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 6, pp. 1225–1236, Jun. 2018, doi: 10.1109/TCAD.2017.2748029.

[22] C. B. Tan, K. K. Koh, D. Zhang, and Y. M. Foong, "Sub-resolution assist feature (SRAF) printing prediction using logistic regression," in *Proc. SPIE*, 2015, Art. no. 94261Y.

[23] *Calibre Verification User's Manual*, Mentor Graphics, Wilsonville, OR, USA, 2008.

[24] L. W. Liebmann *et al.*, "TCAD development for lithography resolution enhancement," *IBM J. Res. Develop.*, vol. 45, no. 5, pp. 651–665, 2001.

[25] J. P. Hanna and P. Stone, "Grounded action transformation for robot learning in simulation," in *Proc. AAAI*, 2017, pp. 3834–3840.

[26] A. A. Rusu *et al.*, "Progressive neural networks," *ArXiv e-prints*, Jun. 2016. [Online]. Available: http://adsabs.harvard.edu/abs/2016arXiv160604671R

[27] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[28] O. Sener and S. Savarese, "Active learning for convolutional neural networks: A core-set approach," in *Proc. Int. Conf. Learn. Represent.*, 2018. [Online]. Available: https://openreview.net/forum?id=H1aIuk-RW

[29] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *J. Mach. Learn. Res.*, vol. 2, pp. 45–66, Nov. 2001.

[30] C. Berlind and R. Urner, "Active nearest neighbors in changing environments," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1870–1879.

[31] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," *CoRR*, vol. abs/1605.09782, 2016. [Online]. Available: http://arxiv.org/abs/1605.09782

[32] B. Demir and L. Bruzzone, "A multiple criteria active learning method for support vector regression," *Pattern Recognit.*, vol. 47, no. 7, pp. 2558–2567, 2014.

[33] K. Fukumizu, "Statistical active learning in multilayer perceptrons," *IEEE Trans. Neural Netw.*, vol. 11, no. 1, pp. 17–26, Jan. 2000.

[34] C. Zhuo, K. Agarwal, D. Blaauw, and D. Sylvester, "Active learning framework for post-silicon variation extraction and test cost reduction," in *Proc. Int. Conf. Comput.-Aided Design*, 2010, pp. 508–515.

[35] H. Lin and P. Li, "Classifying circuit performance using active-learning guided support vector machines," in *Proc. Int. Conf. Comput.-Aided Design*, 2012, pp. 187–194.

[36] M. Li *et al.*, "Provably secure camouflaging strategy for IC protection," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, to be published, doi: 10.1109/TCAD.2017.2750088.

[37] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[39] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[40] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Stat.*, 2010, pp. 249–256.

[41] X. Jin and J. Han, "K-medoids clustering," in *Encyclopedia of Machine Learning*, C. Sammut and G. I. Webb, Eds. Boston, MA, USA: Springer, 2010, pp. 564–565, doi: 10.1007/978-0-387-30164-8_426.

[42] L. Kaufman and P. J. Rousseeuw, "Partitioning around medoids (program PAM)," in *Finding Groups in Data: An Introduction to Cluster Analysis*. New York, NY, USA: Wiley, 1990, pp. 68–125.

[43] H.-S. Park and C.-H. Jun, "A simple and fast algorithm for K-medoids clustering," *Expert Syst. Appl.*, vol. 36, no. 2, pp. 3336–3341, 2009.

[44] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proc. 18th Annu. ACM SIAM Symp. Discr. Algorithms*, 2007, pp. 1027–1035.

[45] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Feb. 2011.

[46] M. Abadi *et al.* (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. [Online]. Available: https://www.tensorflow.org

[47] *Sentaurus Lithography*, Synopsys, Mountain View, CA, USA, 2016. [Online]. Available: https://www.synopsys.com/silicon/mask-synthesis/sentaurus-lithography.html.

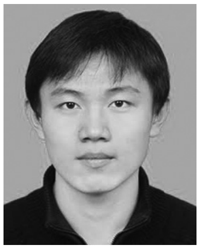
[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[49] Y. Lin *et al.*, "Data efficient lithography modeling with residual neural networks and transfer learning," in *Proc. ACM Int. Symp. Phys. Design (ISPD)*, Monterey, CA, USA, Mar. 2018, pp. 82–89.

Yibo Lin (S'16–M'18) received the B.S. degree in microelectronics from Shanghai Jiao Tong University, Shanghai, China, in 2013 and the Ph.D. degree from the Electrical and Computer Engineering Department, University of Texas at Austin, Austin, TX, USA, in 2018.

He is currently a Post-Doctoral Researcher with the University of Texas at Austin. He has interned with Toshiba, Yokohama, Japan, IMEC, Leuven, Belgium, Cadence, San Jose, CA, USA, and Oracle, Redwood City, CA, USA. His current research interests include physical design, machine learning applications, emerging technology in very large scale integration CAD, and hardware security.

Dr. Lin was a recipient of the Best Paper Award at Integration, the *VLSI Journal* in 2018, the Franco Cerrina Memorial Best Student Paper Award at SPIE Advanced Lithography Conference in 2016, and the University Continuing Fellowship at the University of Texas at Austin in 2017.



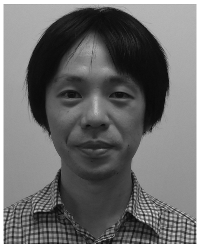
Meng Li (S'15) received the B.S. degree in microelectronics from Peking University, Beijing, China, in 2013. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the University of Texas at Austin, Austin, TX, USA, under the supervision of Prof. D. Z. Pan.

His current research interests include hardware-oriented security, reliability, power grid simulation acceleration, and deep learning.

Prof. Li was a recipient of the First Place in the Grand Final of ACM Student Research Competition in 2018, the Best Poster (Presentation) Award in ASPDAC Ph.D. Forum in 2018, the Gold Medal in ACM ICCAD Student Research Competition in 2017, the University Graduate Fellowship from UT Austin in 2013, the Best Paper Award in HOST'17, and the Best Paper Candidate in GLSVLSI'18.

Yuki Watanabe received the B.S. and M.S. degrees in mechanical engineering from the Tokyo Institute of Technology, Tokyo, Japan, in 2005 and 2007, respectively.

Since 2007, he has been with Toshiba Corporation, Yokohama, Japan, where he has been researching in the area of optical lithography, inspection, and metrology. His current research interests include computational lithography, machine learning algorithms, and image processing.



Taiki Kimura received the M.S. degree in electronic information and energy engineering from the University of Osaka, Osaka, Japan, in 2004.

He joined Toshiba Corporation, Yokohama, Japan, in 2004, where he has been researching in the area of optical lithography. He visited the University of Texas at Austin, Austin, TX, USA, as a Visiting Scholar from 2015 to 2016. His current research interests include design for manufacturability and machine learning algorithms with applications in computational lithography.

Authorized licensed use limited to: University of Texas at Austin. Downloaded on September 16,2020 at 02:47:43 UTC from IEEE Xplore. Restrictions apply.

Tetsuaki Matsunawa received the Ph.D. degree in computer science from the University of Tsukuba, Tsukuba, Japan, in 2008.

He joined Toshiba Corporation, Yokohama, Japan, in 2008, where he has been researching in the area of optical lithography. He visited the University of Texas at Austin, Austin, TX, USA, as a Visiting Scholar from 2013 to 2015. His current research interests include design for manufacturability and machine learning algorithms with applications in computational lithography.

Shigeki Nojima received the B.S. degree in geoscience and the M.S. degree in mineralogy from the University of Tokyo, Tokyo, Japan, in 1992 and 1994, respectively.

Since 1994, he has been with Toshiba Corporation, Yokohama, Japan, where he was engaged in the development of lithography and Technology CAD, especially process simulation, optical proximity correction, design for manufacturability, and CAD for emerging lithography. He has published over 30 technical papers in international conference proceedings and journals and holds over 30 U.S. patents.

Mr. Nojima was a recipient of the Best Paper Award in the Ninth and Tenth Symposium on Photomask and Next-Generation Lithography Mask Technology in 2002 and 2003, respectively. He was a Technical Program Committee Member of Design for Manufacturability and Reliability in IEEE/ACM Asia and South Pacific Design Automation Conference from 2013 to 2015. He has been a program committee of Design-Process-Technology Co-optimization for Manufacturability in International Society for Optics and Photonics (SPIE) Advanced Lithography since 2014. He is a member of the SPIE.

David Z. Pan (S'97–M'00–SM'06–F'14) received the B.S. degree from Peking University, Beijing, China, and the M.S. and Ph.D. degrees from the University of California at Los Angeles (UCLA), Los Angeles, CA, USA.

From 2000 to 2003, he was a Research Staff Member with IBM T. J. Watson Research Center, Yorktown Heights, NY, USA. He is currently the Engineering Foundation Professor with the Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX, USA. He has published over 300 technical papers and holds eight U.S. patents. He has graduated 23 Ph.D. students who are now holding key academic and industry positions. His current research interests include cross-layer nanometer IC design for manufacturability, reliability, security, physical design, analog design automation, and CAD for emerging technologies, such as 3-D-IC and nanophotonics.

Dr. Pan was a recipient of number of prestigious awards for his research contributions, including the SRC Technical Excellence Award, the DAC Top ten Author in Fifth Decade, the DAC Prolific Author Award, the ASP-DAC Frequently Cited Author Award, 16 Best Paper Awards at premier venues, such as GLSVLSI 2018, VLSI Integration 2018, HOST 2017, SPIE 2016, ISPD 2014, ICCAD 2013, ASPDAC 2012, ISPD 2011, IBM Research 2010 Pat Goldberg Memorial Best Paper Award, ASPDAC 2010, DATE 2009, ICICDT 2009, and SRC Techcon in 1998, 2007, 2012, and 2015 plus 11 additional Best Paper Award nominations/finalists at DAC/ICCAD/ASPDAC/ISPD, Communications of the ACM Research Highlights in 2014, ACM/SIGDA Outstanding New Faculty Award in 2005, the NSF CAREER Award in 2007, the SRC Inventor Recognition Award three times, the IBM Faculty Award four times, the UCLA Engineering Distinguished Young Alumnus Award in 2009, the UT Austin RAISE Faculty Excellence Award in 2014, and many international CAD contest awards, among others. His students have also won many awards, including the First Place of ACM Student Research Competition Grand Finals in 2018, ACM/SIGDA Student Research Competition Gold Medal (twice), and ACM Outstanding Ph.D. Dissertation in EDA (twice). He has served as a Senior Associate Editor for *ACM Transactions on Design Automation of Electronic Systems*, an Associate Editor for the IEEE TRANSACTIONS ON COMPUTER AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART I: REGULAR PAPERS, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART II: EXPRESS BRIEFS, IEEE DESIGN & TEST, *Science China Information Sciences*, the *Journal of Computer Science and Technology*, and *IEEE CAS Society Newsletter*. He has served in the executive and program committees of many major conferences, including DAC, ICCAD, ASPDAC, and ISPD. He is the Program Chair of ASPDAC 2017 and ICCAD 2018, the Tutorial Chair of DAC 2014, and the General Chair of ISPD 2008. He is an SPIE Fellow.

Authorized licensed use limited to: University of Texas at Austin. Downloaded on September 16,2020 at 02:47:43 UTC from IEEE Xplore. Restrictions apply.