Lithography Hotspot Detection: From Shallow to Deep Learning
(Invited Paper)

Haoyu Yang , Yajun Lin , Bei Yu , and Evangeline F. Y. Young

Department of Computer Science and Engineering
The Chinese University of Hong Kong, NT, Hong Kong
{hyyang,byu,fyyoung}@cse.cuhk.edu.hk

***Abstract*—As VLSI technology nodes continue, the gap between lithography system manufacturing ability and transistor feature size induces serious problems, thus lithography hotspot detection is of importance in physical verification flow. Existing hotspot detection approaches can be categorized into pattern matching-based and machine learning-based. With extreme scaling of transistor feature size and the growing complexity of layout patterns, the traditional methods may suffer from performance degradation. For example, pattern matching-based methods have lower hotspot detection rates for unseen patterns, while machine learning-based methods may lose information in manual feature extraction for ultra-large-scale integrated circuit masks. To overcome the drawbacks derived from existing methods, in this paper, we survey very recent deep learning techniques and argue that the pooling layers in ordinary deep learning architecture are not necessary. We further propose a novel pooling-free neural network architecture, whose effectiveness is verified by industrial benchmark suites.**

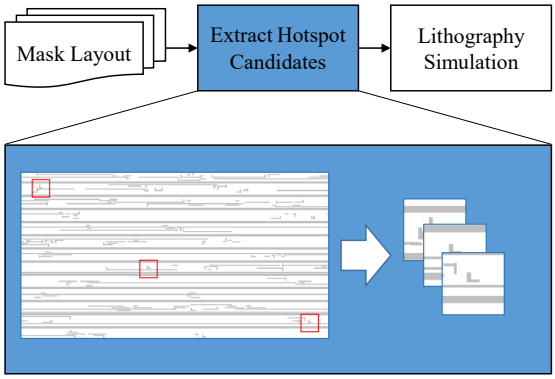

Fig. 1: VLSI physical verification flow.

I. Introduction

With the VLSI technology node continuously shrinking down, there is a large gap between the mask pattern feature size and the 193nm lithography system wavelength [1], [2]. The printability of the mask layout is seriously affected by light diffraction and circuit failures (open or short circuit) are more likely to occur for some patterns. Therefore it is necessary to detect and correct the problematic patterns (i.e. hotspots) before the manufacturing process. Varies resolution enhancement technologies (RETs) have been proposed and developed to provide yielding-friendly patterns under the sub-wavelength lithography condition. Existing RETs include optical proximity correction (OPC) [3], [4], phase shift masks (PSMs) [5] and sub-resolution assist features (SRAFs) [6]. However, the effectiveness of RETs is affected by many factors including the patterns' compatibility with OPC algorithms and the pitch differences, therefore simply adopting them cannot guarantee to print ideal shapes and additional layout refinements are required to increase the quality of the printed patterns.

The objective of hotspot detection task is identifying the problematic patterns or layout regions that need further refinements. Lithography simulation is currently the most accurate approach to recognize lithography hotspots and estimate the mask layout process windows [7], yet the progress is time consuming to obtain the full chip characteristics. To speed up the physical verification flow (see Fig. 1), state-of-the-art approaches usually apply one or more rounds of hotspot classification or prediction, where hotspot candidates are extracted from the designed mask layout via efficient classification procedure and only hotspot candidates are required to feed into the lithography simulation engine. As a result, the verification flow is significantly facilitated.

A major challenge of the physical verification flow is how to extract reliable hotspot candidates (i.e. predict hotspot regions) that can cover as much the real problematic regions as possible. Pattern matching and machine learning are two main methodologies to extract hotspot candidates. On one hand, pattern matching based methods discover the problematic regions through comparing the topologies of the designed layout patterns with the topologies of the patterns in hotspot libraries [8]–[10]. Because the quality of the hotspot candidates rely highly on the hotspot library, pattern matching based methods have weak generality especially for unique topologies under advanced technology nodes. On the other hand, in machine learning flow, layout patterns are usually converted into a low dimensional representation known as feature extraction. Then efficient machine learning models are able to learn the beneath relationships between the layout features and their attributes [11]–[18]. However, as the layout feature size shrinks down to 22nm and beyond, manually crafted layout features cannot effectively grasp the pattern properties. Very recently, deep neural networks have shown great success in pattern recognition tasks because of the existence of convolution layers that are able to automatically extract pattern features [19]–[21]. Several attempts are made to apply convolutional neural networks for hotspot detection and achieve promising hotspot prediction results [22]–[25]. Although [23]–[25] include customized training strategies

978-1-5386-4034-0/17/$31.00 ©2017 IEEE

according to the properties of layout datasets, there are still several aspects that have not been taken into account when designing the neural networks.

To address the issues in state-of-the-art hotspot detectors, in this paper we study the functionality of the current convolutional neural network architecture and analyze the necessity of different layers, based on which, an updated network architecture is built and trained to classify hotspot candidates accurately and efficiently. Our contributions are listed as follows.

- We elaborate the state-of-the-art hotspot detector solutions based on machine learning techniques, including both shallow and deep learning models, and analyze the advantages and drawbacks.
- We analyze the functionality of different layer types within the deep neural networks and develop an improved network architecture that can offer better detection results.
- Experimental results show that our neural network model achieves satisfactory results on both pre-OPC and post-OPC layout datasets.

The rest of the paper is organized as follows. Section II introduces some terminologies of the hotspot detection problem. Section III surveys and analyzes traditional machine learning based hotspot detectors. Section IV elaborates the deep learning solutions for hotspot detection and presents an improved architecture to achieve better detection results. Section VI lists the experimental results, followed by conclusion in Section VI.

II. Preliminaries

In this section, we will introduce some terminologies related to the hotspot detection problem. As mentioned in previous section, a good hotspot detector should be able to predict as much real hotspot regions as possible at the minimum cost of false positives. Therefore, we adopt the evaluation metrics that are defined in the ICCAD Contest 2012 [26].

Definition 1 (Accuracy)**.** *The ratio between the number of correctly predicted hotspot clips and the number of all real hotspot clips.*

Definition 2 (False Alarm)**.** *The number of non-hotspot clips that are predicted as hotspots by the classifier.*

Problem 1 (Hotspot Detection)**.** *Given a set of clips consisting of hotspot and non-hotspot patterns, the object of hotspot detection is training a classifier that can maximize the accuracy and minimize the false alarm.*

Because hotspot detectors aim to speed up the physical verification flow, the model test runtime, which though is less important than the accuracy and the false alarm, is also taken into consideration.

III. Shallow Learning Models

Machine learning methods have been applied in the design automation field extensively [27], e.g., adder synthesis [28], statistical path selection [29], sensor placement [30], parametric yield estimation and improvement [31]–[35], hardware security enhancement [36], and post-silicon tuning buffer allocation [37]–[39]. Artificial neural networks (ANN) [12], [40], support vector machine (SVM) [11]–[13] and Boosting [14], [15] have been widely explored specifically for hotspot detection problems. These solutions share a similar process called feature extraction, which obtains low dimensional representations of layout clips. In this section, we will elaborate both the feature extraction methods and learning models for hotspot detection.

A. Feature Extraction

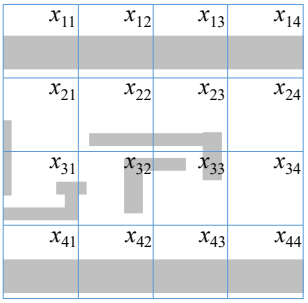

Fig. 2: Density-based feature representation. The local density information is converted to a feature vector $\{x_{11}, x_{12}, ..., x_{44}\}$.

1) Density-based Feature: Usually mask layouts with high pattern density show a higher risk of suffering defects, therefore it is reasonable to measure the mask printability via its local pattern density [9], [14]. As shown in Fig. 2, each layout clip is first divided into square grids and each grid $G(i, j)$ corresponds to a value $x_{i,j}$ reflects the density information that is calculated through the Equation (1).

$$x_{i,j} = \frac{A_M(i,j)}{A_G(i,j)}, \tag{1}$$

where $A_M(i, j)$ denotes the mask pattern area within $G(i, j)$ and $A_G(i, j)$ is the area of $G(i, j)$. Finally, the density information is flattened into a vector $\mathbf{x} = \{x_{11}, x_{12}, ..., x_{44}\}$ that is applicable for varies machine learning models.

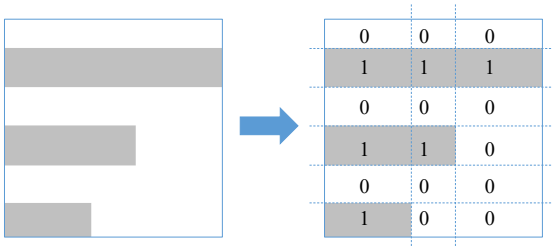

Fig. 3: Topology-based layout representation. Polygon edges divide the clip into grids. The grid filled with geometry is labeled 1 and the grid filled with space is labeled 0.

2) Layout Topology: The topological representation of a layout clip contains the geometry relationships of the patterns (rectangles) within it [13], [41]. As shown in Fig. 3, all edges

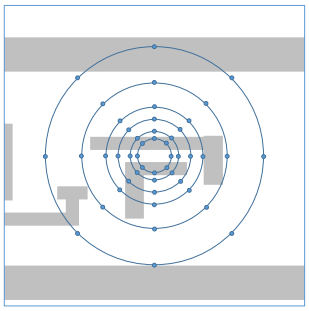

Fig. 4: Concentric circle area sampling.

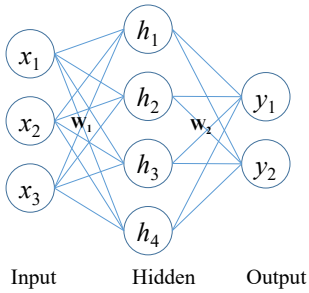

Fig. 5: Artificial neural network architecture.

are extended to the border of the clip and cut the clip into grids, which are filled with either geometry or space. If we label each grid with 1 (for geometry) and 0 (for space), the original layout clip can then be converted into a binary matrix representation. The topology matrix itself has already been a good representation for pattern matching [41] and constructing the space of design layout configurations and sub-configurations [42]. Besides, by incorporating with design rules, the topological representation can be a effective layout feature for accurate hotspot detection [13].

3) Concentric Circle Area Sampling: It has been believed that layout hotspots are caused by optical proximity (diffraction) effect in the lithography process. Whether a pattern is problematic or not is determined by all the patterns within a square [26] or circular ambit [43]. In the $28nm$ technology node, the ambit area is approximately $1\mu m^2$ which corresponds to an $\mathbb{R}^{1000000}$ space with $1nm$ precision that makes it difficult to do quantitative analysis for OPC and hotspot detection. Concentric circle area sampling (CCAS) [3], developed from concentric square sampling (CSS) [44], has become an efficient feature extraction method because of being coherent with the fact that diffracted light propagates in a circular concentric scheme. As illustrated in Fig. 4, all the circles are concentrated on the center of the clip (or ambit). Eight points are evenly sampled from each circle and each point is labeled 0 or 1 based on whether it is located on the geometry or space. Although CCAS is originally designed for machine learning based OPC [3], it is also effective in hotspot detection [17].

B. Learning Models

1) Boosting: Boosting is a family of ensemble machine learning methods which are able to build a strong classifier from a set of weak classifiers. [14] and [15] are two representative works for hotspot detection, where decision tree is chosen as the weak classifier. Decision tree works in the form of a flow chart where each non-leaf node splits the data by examining one attribute and leaf nodes predict the label. Information gain (IG) is utilized to determine the feature of each node, as shown in Equation (2) [45].

$$\text{IG}(P, f) = H(P) - H(P|f), \tag{2}$$

where $H(P)$ is the entropy related to feature f of the parent node and $H(P|f)$ is the weighted entropy of all children. Final prediction are made by a weighted combination of the results from all decision trees (weak classifiers).

2) SVM: SVM is deemed as one of the most powerful machine learning models with the problem formulation as follows.

$$\max \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j k(\mathbf{x}_i, \mathbf{x}_j), \quad (\textbf{??}\text{eq:svm})$$

$$\text{s.t.} \sum_{i=1}^{N} \alpha_i d_i = 0, \tag{3a}$$

$$0 \leq \alpha_i \leq C, \quad \forall i = 1, 2, \ldots, N, \tag{3b}$$

where α_i and d_i are the Lagrangian coefficient and the label that correspond to the instance $\mathbf{x}_i$. C is a manually chosen constant that controls the soft margin for non-separable datasets. After solving the problem, we can form the following classifier for hotspot detection.

$$d = \sum_{i=i}^{N_S} \alpha_i d_i k(\mathbf{x}_i, \mathbf{x}), \tag{4}$$

where $\mathbf{x}_i$s are support vectors from the training set that have non-zero Lagrangian coefficients. Because designers are free to choose different kernels, SVM provides more robust solutions for hotspot detection problems [12], [13].

3) ANN: ANN is a standard multilayer perceptron model that is able to fit highly nonlinear functions. An example of ANN is presented in Fig. 5 with one input layer, one hidden layer and one output layer. The output layer generates the prediction scores or regression values. The value of each node corresponds to the weighted sum of the nodes in previous layer, as shown in the following equations.

$$\mathbf{h} = f(\mathbf{W_1 x}), \tag{5}$$

$$\mathbf{y} = g(\mathbf{W_2 h}), \tag{6}$$

where f and g are activation functions that perform element wise operation on each neuron. If we map layout features to $\mathbf{x} = \{x_1, x_2, x_3, \ldots\}$, the ANN can do either regression tasks (e.g. OPC [40]) or classification tasks (e.g. hotspot detection [12]).

IV. Deep Learning Models

Traditional machine learning methods rely highly on manually designed layout features. Such crafted features cannot

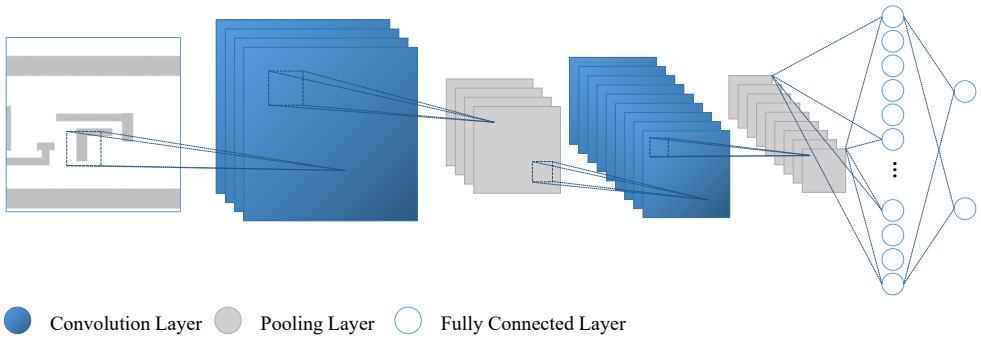

Fig. 6: Ordinary convolutional neural network architecture with two convolution layers (blue), two pooling layers (grey), and two fully connected layers (circle).

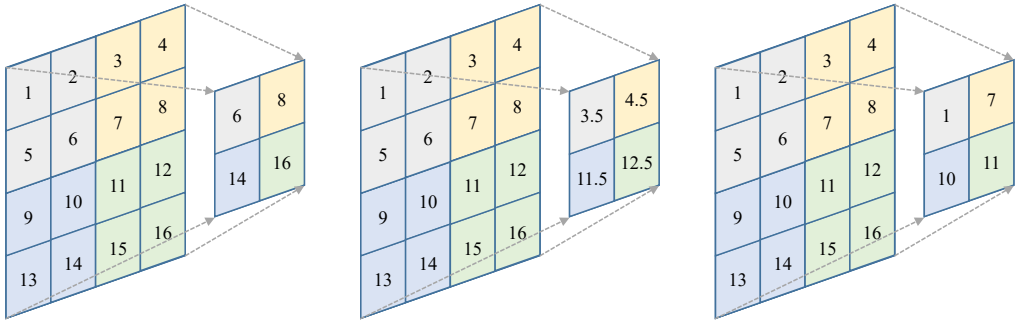

Fig. 7: Example of different pooling operations on 2×2 regions with a stride of 2 pixels between pooling regions. (a) Maximum pooling. (b) Average pooling. (c) Stochastic pooling.

always grasp the dominant information even with prior knowledge. Recently, convolutional neural network (CNN) starts to show strong feature learning ability and brings promising hotspot detection results [22]–[25]. In this section, we will introduce the basis of CNN and the functionalities of different layer types, based on which a customized architecture is designed for complex layout hotspot detection.

A. CNN Basis

Convolution layers, pooling layers and fully connected layers are three main components of normal CNN architecture, as shown in Fig. 6. We detail the functionalities of each layer type as follows.

1) Convolution Layers: Convolution layers are the key elements of deep neural networks, and they work similarly with traditional ANN except some edges share same weights which enables the automatic extraction of common features. The operations are also changed from simple inner products (i.e., Equations (5) and (6)) to convolution, as shown in Equation (7).

$$\mathbf{I} \otimes \mathbf{K}(x, y) = \sum_{i=1}^{c} \sum_{j=1}^{m} \sum_{k=1}^{m} \mathbf{I}(i, x-j, y-k)\mathbf{K}(i, j, k), \quad (7)$$

where $\mathbf{I}$ is input image or feature maps, while $\mathbf{K} \in \mathbb{R}^{c \times m \times m}$ is the convolution kernel (i.e. shared weights). Within the convolution layer, the kernel scans the input image or feature maps from the upper-right to the bottom-left corner. The output pixel values are obtained from the inner product of each scanning step.

2) Pooling Layers: Pooling layers extract the statistical summary of the local regions of the previous layer which can reduce the feature map dimension as well as make the neural network not sensitive to small changes. Maximum pooling, average pooling and stochastic pooling are three main pooling methods which output the maximum, average or random value of a local region. We visualize the pooling operation in Fig. 7. When building the CNN architecture, pooling can be applied on any local region in any dimension.

3) Fully Connected Layers: In the design of CNN architecture, multiple convolution layers and pooling layers can stack together to form deeper networks. Fully connected layers are used to flatten the feature maps, which are extracted from multiple convolution and pooling operations into a one dimensional vector to predict the final results. The operations inside the fully connected layers are pure inner product which is the same as ANN. For the hotspot detection problems, node number in the last fully connected layer is two that correspond to the score of the instance being hotspot and non-hotspot, respectively.

4) Rectified Linear Unit: Unlike ANN, modern CNNs usually take the rectified linear unit (ReLU, as in Equation (8)) as activation functions instead of traditional sigmoid function and tanh. ReLUs are applied after convolution layers and fully connected layers to perform element-wise operations on each intermediate node and introduce nonlinearity to the neural

TABLE I: Proposed Neural Network Configuration.

Layer	Kernel Size	Stride	Padding	Output Vertexes
Conv1-1	$2 \times 2 \times 4$	2	0	$512 \times 512 \times 4$
Conv1-2	$3 \times 3 \times 4$	2	0	$256 \times 256 \times 4$
Conv2-1	$3 \times 3 \times 8$	1	1	$256 \times 256 \times 8$
Conv2-2	$3 \times 3 \times 8$	1	1	$256 \times 256 \times 8$
Conv2-3	$3 \times 3 \times 8$	1	1	$256 \times 256 \times 8$
Conv2-4	$3 \times 3 \times 8$	2	0	$128 \times 128 \times 8$
Conv3-1	$3 \times 3 \times 16$	1	1	$128 \times 128 \times 16$
Conv3-2	$3 \times 3 \times 16$	1	1	$128 \times 128 \times 16$
Conv3-3	$3 \times 3 \times 16$	1	1	$128 \times 128 \times 16$
Conv3-4	$3 \times 3 \times 16$	2	0	$64 \times 64 \times 16$
Conv4-1	$3 \times 3 \times 32$	1	1	$64 \times 64 \times 32$
Conv4-2	$3 \times 3 \times 32$	1	1	$64 \times 64 \times 32$
Conv4-3	$3 \times 3 \times 32$	1	1	$64 \times 64 \times 32$
Conv4-4	$3 \times 3 \times 32$	2	0	$32 \times 32 \times 32$
Conv5-1	$3 \times 3 \times 32$	1	1	$32 \times 32 \times 32$
Conv5-2	$3 \times 3 \times 32$	1	1	$32 \times 32 \times 32$
Conv5-3	$3 \times 3 \times 32$	1	1	$32 \times 32 \times 32$
Conv5-4	$3 \times 3 \times 32$	2	0	$16 \times 16 \times 32$
FC1	–	–	–	2048
FC2	–	–	–	512
FC3	–	–	–	2

TABLE II: Benchmark Statistics

Benchmarks	Training Set		Testing Set	
	HS#	NHS#	HS#	NHS#
ICCAD	1204	17096	2524	13503
Industry1	34281	15635	17157	7801
Industry2	15197	48758	7520	24457
Industry3	24776	49315	12228	24817

network.

$$\text{ReLU}(x) = \begin{cases} x, & \text{if } x > 0, \\ 0, & \text{if } x \leq 0. \end{cases} \tag{8}$$

B. Pooling Is Not Necessary

In ordinary object recognition tasks, the existence of pooling layer makes the neural network have more generality because small variations are diminished after the pooling operation. However, this characteristic will not benefit the hotspot detection problem when predicting post-OPC patterns. That is, a portion of the edge displacements that turns a hotspot pattern into a non-hotspot pattern may be ignored by the neural network due to the pooling layer. For such reason, we propose removing the pooling layers to avoid information loss in hotspot detection. In this work, we extend the CNN architecture developed in [24], and replace all the pooling layers with 3×3 convolution layers. The strides are the same as original pooling layer to make sure the modification is minimized. More details on the polished architecture are listed in TABLE I, where our modifications are highlighted in bold.

V. Experimental Results

The proposed deep learning model is implemented using Python with TensorFlow library [46]. We conduct the experiment on four benchmarks, as shown in TABLE II, where `ICCAD` contains all the $28nm$ clips from ICCAD-2012 CAD contest [26] and `Industry1-Industry3` are three more complicated industrial benchmarks. Columns "HS#" and "NHS#" denote the number of hotspot and non-hotspot clips in each benchmark respectively. To show the effectiveness of our model, we compare our results with four recently proposed hotspot detectors as detailed in TABLE III, where [14] and [17] are two hotspot detectors with boosting models. Besides, [24] is an ordinary CNN classifier that takes raw layout clips as input, while [25] applies small CNN model on frequency representations of layout patterns. Here columns "Accu (%)" and "FA#" correspond to the hotspot detection accuracy and the number of false alarm respectively.

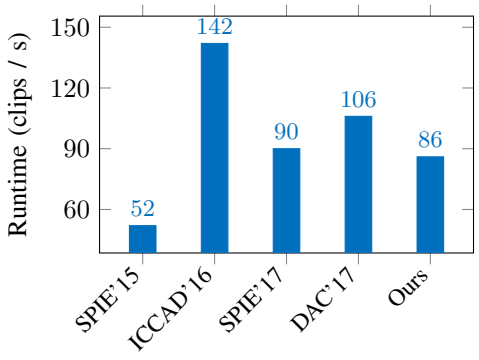

Fig. 8: Runtime Comparison with State-of-the-art Hotspot Detectors.

Our model is developed based on [24] by replacing all the pooling layers with convolution layers. Compared to [24], the average hotspot detection accuracy increases from 90.55% to 92.49% and the false alarm drops 337, both of which demonstrate the effectiveness of removing pooling layers. Particularly, the improvements are prominent on post-OPC patterns. On average, CNN solutions work better that shallow learning models with 2.91% advantage on accuracy and 1247 less false alarm penalties. It should be noted that since [25] optimizes the detector with bias, it achieves better detection accuracy with some cost of false alarm penalties.

VI. Conclusion

Lithography hotspot detection is a key step in VLSI physical verification flow. In this paper, we survey state-of-the-art machine learning models and varies layout representations for the hotspot detection problem. To address the drawbacks of the shallow learning models and the manually crafted features, we motivate the deep neural network as an alternate hotspot detector. We further propose a pooling-free CNN architecture that fits well with post-OPC mask images. The experimental results show that our model achieves the best false alarm among several recently proposed hotspot detectors. Moreover, through this paper we hope to demonstrate the importance of application-specific CNN design.

Acknowledgment

The authors would like to thank Jing Su, Chenxi Lin, and Yi Zou from ASML for helpful comments.

References

[1] "ITRS," http://www.itrs.net.
[2] D. Z. Pan, B. Yu, and J.-R. Gao, "Design for manufacturing with emerging nanolithography," *IEEE TCAD*, vol. 32, no. 10, pp. 1453–1472, 2013.

TABLE III: Performance Comparison with State-of-the-art Hotspot Detectors.

Benchmarks	SPIE'15 [14]		ICCAD'16 [17]		SPIE'17 [24]		DAC'17 [25]		Ours	
	Accu (%)	FA#	Accu (%)	FA#	Accu (%)	FA#	Accu (%)	FA#	Accu (%)	FA#
ICCAD	84.20	2919	97.70	4497	97.70	2703	**98.20**	3413	97.36	**1776**
Industry1	93.20	2204	89.90	1136	97.74	519	**98.90**	680	98.41	**307**
Industry2	44.80	1320	88.40	7402	89.62	**760**	**93.60**	2165	90.56	793
Industry3	44.00	3144	82.30	8609	77.14	1966	**91.30**	4196	83.63	**1723**
Average	66.55	2397	89.58	5411	90.55	1487	**95.50**	2614	92.49	**1150**
Ratio	0.720	2.084	0.968	4.705	0.979	1.293	**1.033**	2.273	1.000	**1.000**

[3] T. Matsunawa, B. Yu, and D. Z. Pan, "Optical proximity correction with hierarchical bayes model," in *Proc. SPIE*, vol. 9426, 2015.

[4] S. Banerjee, K. B. Agarwal, and M. Orshansky, "Simultaneous OPC and decomposition for double exposure lithography," in *Proc. SPIE*, vol. 7973, 2011.

[5] C. A. Mack, *Field Guide to Optical Lithography*. SPIE Press Bellingham, 2006, vol. 6.

[6] R. Viswanathan, J. T. Azpiroz, and P. Selvam, "Process optimization through model based SRAF printing prediction," in *SPIE Advanced Lithography*, vol. 8326, 2012.

[7] J. Kim and M. Fan, "Hotspot detection on Post-OPC layout using full chip simulation based verification tool: A case study with aerial image simulation," in *Proc. SPIE*, vol. 5256, 2003.

[8] Y.-T. Yu, Y.-C. Chan, S. Sinha, I. H.-R. Jiang, and C. Chiang, "Accurate process-hotspot detection using critical design rule extraction," in *Proc. DAC*, 2012, pp. 1167–1172.

[9] W.-Y. Wen, J.-C. Li, S.-Y. Lin, J.-Y. Chen, and S.-C. Chang, "A fuzzy-matching model with grid reduction for lithography hotspot detection," *IEEE TCAD*, vol. 33, no. 11, pp. 1671–1680, 2014.

[10] F. Yang, S. Sinha, C. C. Chiang, X. Zeng, and D. Zhou, "Improved tangent space based distance metric for lithographic hotspot classification," *IEEE TCAD*, 2017.

[11] D. G. Drmanac, F. Liu, and L.-C. Wang, "Predicting variability in nanoscale lithography processes," in *Proc. DAC*, 2009, pp. 545–550.

[12] D. Ding, B. Yu, J. Ghosh, and D. Z. Pan, "EPIC: Efficient prediction of IC manufacturing hotspots with a unified meta-classification formulation," in *Proc. ASPDAC*, 2012, pp. 263–270.

[13] Y.-T. Yu, G.-H. Lin, I. H.-R. Jiang, and C. Chiang, "Machine-learning-based hotspot detection using topological classification and critical feature extraction," *IEEE TCAD*, vol. 34, no. 3, pp. 460–470, 2015.

[14] T. Matsunawa, J.-R. Gao, B. Yu, and D. Z. Pan, "A new lithography hotspot detection framework based on AdaBoost classifier and simplified feature extraction," in *Proc. SPIE*, vol. 9427, 2015.

[15] Z. Xiao, Y. Du, H. Tian, M. D. F. Wong, H. Yi, H.-S. P. Wong, and H. Zhang, "Directed self-assembly (DSA) template pattern verification," in *Proc. DAC*, 2014, pp. 55:1–55:6.

[16] B. Yu, J.-R. Gao, D. Ding, X. Zeng, and D. Z. Pan, "Accurate lithography hotspot detection based on principal component analysis-support vector machine classifier with hierarchical data clustering," *JM3*, vol. 14, no. 1, p. 011003, 2015.

[17] H. Zhang, B. Yu, and E. F. Y. Young, "Enabling online learning in lithography hotspot detection with information-theoretic feature optimization," in *Proc. ICCAD*, 2016, pp. 47:1–47:8.

[18] H. Zhang, F. Zhu, H. Li, E. F. Y. Young, and B. Yu, "Bilinear lithography hotspot detection," in *Proc. ISPD*, 2017, pp. 7–14.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint*, 2014.

[21] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. ECCV*, 2014, pp. 818–833.

[22] T. Matsunawa, S. Nojima, and T. Kotani, "Automatic layout feature extraction for lithography hotspot detection based on deep neural network," in *SPIE Advanced Lithography*, vol. 9781, 2016.

[23] M. Shin and J.-H. Lee, "Accurate lithography hotspot detection using deep convolutional neural networks," *JM3*, vol. 15, no. 4, p. 043507, 2016.

[24] H. Yang, L. Luo, J. Su, C. Lin, and B. Yu, "Imbalance aware lithography hotspot detection: A deep learning approach," in *SPIE Advanced Lithography*, vol. 10148, 2017.

[25] H. Yang, J. Su, Y. Zou, B. Yu, and E. F. Y. Young, "Layout hotspot detection with feature tensor generation and deep biased learning," in *Proc. DAC*, 2017, pp. 62:1–62:6.

[26] A. J. Torres, "ICCAD-2012 CAD contest in fuzzy pattern matching for physical verification and benchmark suite," in *Proc. ICCAD*, 2012, pp. 349–350.

[27] B. Yu, D. Z. Pan, T. Matsunawa, and X. Zeng, "Machine learning and pattern matching in physical design," in *Proc. ASPDAC*, 2015, pp. 286–293.

[28] S. Roy, Y. Ma, J. Miao, and B. Yu, "A learning bridge from architectural synthesis to physical design for exploring power efficient high-performance adders," in *Proc. ISLPED*, 2017.

[29] J. Xiong, Y. Shi, V. Zolotov, and C. Visweswariah, "Statistical multilayer process space coverage for at-speed test," in *Proc. DAC*, 2009, pp. 340–345.

[30] T. Wang, C. Zhang, J. Xiong, and Y. Shi, "Eagle-eye: a near-optimal statistical framework for noise sensor placement," in *Proc. ICCAD*, 2013, pp. 437–443.

[31] C. Zhuo, D. Sylvester, and D. Blaauw, "Design time body bias selection for parametric yield improvement," in *Proc. ASPDAC*, 2010, pp. 681–688.

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

[33] C. Zhuo, K. Chopra, D. Sylvester, and D. Blaauw, "Process variation and temperature-aware full chip oxide breakdown reliability analysis," *IEEE TCAD*, vol. 30, no. 9, pp. 1321–1334, 2011.

[34] C. Zhuo, D. Sylvester, and D. Blaauw, "A statistical framework for post-fabrication oxide breakdown reliability prediction and management," *IEEE TCAD*, vol. 32, no. 4, pp. 630–643, 2013.

[35] F. Gong, Y. Shi, H. Yu, and L. He, "Variability-aware parametric yield estimation for analog/mixed-signal circuits: concepts, algorithms, and challenges," *IEEE MDAT*, vol. 31, no. 4, pp. 6–15, 2014.

[36] J. Miao, M. Li, S. Roy, and B. Yu, "LRR-DPUF: Learning resilient and reliable digital physical unclonable function," in *Proc. ICCAD*, 2016, pp. 46:1–46:8.

[37] G. L. Zhang, B. Li, J. Liu, Y. Shi, and U. Schlichtmann, "Design-phase buffer allocation for post-silicon clock binning by iterative learning," *IEEE TCAD*, 2017.

[38] G. L. Zhang, B. Li, and U. Schlichtmann, "Effitest: Efficient delay test and statistical prediction for configuring post-silicon tunable buffers," in *Proc. DAC*, 2016, pp. 60:1–60:6.

[39] ——, "Sampling-based buffer insertion for post-silicon yield improvement under process variability," in *Proc. DATE*, 2016, pp. 1457–1460.

[40] R. Luo, "Optical proximity correction using a multilayer perceptron neural network," *Journal of Optics*, vol. 15, no. 7, p. 075708, 2013.

[41] J. P. Cain, Y.-C. Lai, F. Gennari, and J. Sweis, "Methodology for analyzing and quantifying design style changes and complexity using topological patterns," in *SPIE Advanced Lithography*. International Society for Optics and Photonics, 2016, pp. 978 108–978 108.

[42] V. Dai, E. K. C. Teoh, J. Xu, and B. Rangarajan, "Optimization of complex high-dimensional layout configurations for IC physical designs using graph search, data analytics, and machine learning," in *SPIE Advanced Lithography*. International Society for Optics and Photonics, 2017, pp. 1 014 808–1 014 808.

[43] C. Tabery, Y. Zou, V. Arnoux, P. Raghavan, R.-h. Kim, M. Côté, L. Mattii, Y.-C. Lai, and P. Hurat, "In-design and signoff lithography physical analysis for 7/5nm," in *SPIE Advanced Lithography*, 2017, pp. 1 014 705–1 014 705.

[44] A. Gu and A. Zakhor, "Optical proximity correction with linear regression," *IEEE TSM*, vol. 21, no. 2, pp. 263–271, 2008.

[45] Z. John Lu, "The elements of statistical learning: data mining, inference, and prediction," *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, vol. 173, no. 3, pp. 693–694, 2010.

[46] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean *et al.*, "TensorFlow: A system for large-scale machine learning," in *Proc. OSDI*, 2016, pp. 265–283.