Optical-System-Aware Feature Extraction for Lithography Hotspot Detection

Masahiro Yamamoto, Masato Inagi, Shinobu Nagayama
Graduate School of Information Sciences
Hiroshima City University
Hiroshima, Japan
e-mail: m_yamamoto@lcs.info.hiroshima-cu.ac.jp, {inagi | s_naga}@hiroshima-cu.ac.jp

***Abstract*—This paper proposes a new feature vector for machine learning-based hotspot detection in lithography for Large-Scale Integration (LSI) fabrication, which incorporates the optical characteristics of exposure systems. Unlike existing features that focus only on local layout sub-patterns, the proposed feature takes into account optical behavior essential to accurate pattern transfer. In LSI fabrication, a hotspot is a region in the layout where an undesired open or short circuit may occur, even if the design rules are satisfied. Hotspots can significantly reduce manufacturing yield, and the cost of reworking after fabrication begins is substantial. Therefore, it is crucial to detect and remove hotspots at the pre-fabrication stage. Although several feature vectors have been developed for hotspot detection, most of them ignore the optical system's influence, which is critical in the lithography process. By incorporating optical characteristics, our proposed feature aims to improve detection accuracy and reduce the need for time-consuming lithography simulations.**

Keywords-lithography; hotspot; feature vector; optical system.

I. INTRODUCTION

In the lithography process, which is one of the key steps in semiconductor manufacturing, laser light from the exposure system is projected onto a photomask, which serves as the master template of circuit patterns, and the pattern is transferred onto a semiconductor wafer coated with a photosensitive material. In this process, due to optical diffraction, some areas may fail to be correctly transferred even if they comply with the design rules. Such regions are referred to as hotspots. Since photomask fabrication is highly expensive, it is necessary to detect these hotspots prior to manufacturing and revise the layout patterns accordingly.

Lithography simulation, which computes phenomena, such as light diffraction and the behavior of the photosensitive material on the wafer, is a common method used to detect hotspots before mask or product fabrication. However, applying this simulation across the entire layout is extremely time-consuming. If hotspots can be rapidly detected through methods other than simulation, allowing prompt initiation of pattern revision, the overall cost in terms of simulation coverage, frequency, and runtime can be significantly reduced.

Therefore, several studies have explored hotspot detection using machine learning techniques [1]–[5]. These methods train classifiers using known hotspot and non-hotspot layout patterns and detect unseen hotspot patterns based on learned features. However, false detections still occur, and higher detection accuracy is desired. In machine learning-based approaches, the design of features that effectively capture characteristics strongly related to hotspots is crucial. A widely used pixel-based feature is Density Based Layout Feature (DBLF) [1][2], which represents the local wiring density in layout patterns. Other proposed pixel-based features include Histogram of Oriented Light Propagation (HOLP) [3], which approximates optical diffraction by smoothing layout images, and Line Width and Separation (LiWS) [6][7], which considers wire widths and the spacing between wires in the layout.

While several features have been proposed for machine learning-based hotspot detection, actual hotspots vary depending on the behavior of light on the wafer surface, which in turn is influenced by the optical characteristics of the exposure system. Since hotspots are caused by light diffraction from the exposure source to the wafer, it is important to consider the optical characteristics (i.e., source characteristics) of the exposure system. These source characteristics are indispensable in lithography simulation and are already known to those who perform hotspot detection. Thus, this information is potentially applicable outside simulation-based approaches.

Some studies do make use of source characteristics for hotspot determination [4][8], but such approaches remain close to machine learning-based lithography simulation. For example, the method from [4] is also regarded as considering optical characteristics, but it is based on the idea of training a model using intensity images generated by lithography simulation. Therefore, it does not directly incorporate the optical parameters of the exposure system into the learning process.

In this study, we propose a new feature that considers the optical characteristics of the exposure system, which have not been taken into account in existing features. This work is an extended and revised version of our previously published technical report [9]. Because hotspots are induced by diffraction of light as it travels from the light source to the wafer, incorporating source characteristics is essential. By leveraging information already available from lithography simulators, our method enables effective hotspot detection in a machine learning framework. Through comparative experiments with existing features, we confirmed that our proposed feature improves detection accuracy. Note that the effectiveness of the proposed feature may depend on the availability and precision of source characteristics provided by the exposure system.

The remainder of this paper is organized as follows. Section II provides an overview of lithography, hotspots, and

machine learning. Section III defines the hotspot detection problem and describes machine learning-based hotspot detection methods and existing features. Section IV explains the proposed feature incorporating source characteristics. Section V presents the experimental results, and Section VI concludes the paper.

II. Preliminaries

In this section, we first explain the mechanism of lithography and the concept of lithography hotspots. We then describe the machine learning framework used for hotspot detection, focusing on supervised learning for classification, and the process of feature extraction required when applying machine learning.

A. Mechanism of Lithography

Lithography is the pattern transfer process in semiconductor fabrication [10]. In lithography, ultraviolet (UV) light is projected onto a photomask, which serves as the master template for semiconductor chips, and the circuit pattern (layout pattern) is transferred onto a silicon wafer through the photomask, as illustrated in Figure 1.

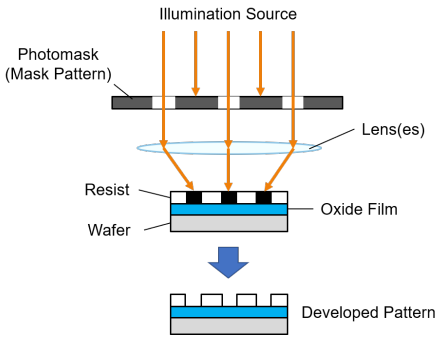

Figure 1. Lithography

B. Lithography Hotspots

With the continued scaling of semiconductor devices, it has become increasingly difficult to accurately transfer designed layout patterns. Examples of degraded pattern fidelity include corner rounding, necking, and line-end shortening, which are caused by Optical Proximity Effects (OPE). To improve the fidelity of pattern transfer, technologies, such as Optical Proximity Correction (OPC) and Sub-Resolution Assist Features (SRAF), have been developed [11]. However, patterns that cannot be accurately transferred still emerge, even with these technologies. Such patterns are referred to as hotspots, and they are one of the factors that degrade yield and reliability of semiconductor products. Figure 2 shows an example of short and open circuits caused by lithography.

Because photomasks, which serve as the masters for layout patterns, are extremely expensive, it is essential to eliminate hotspots at the design stage to avoid costly rework.

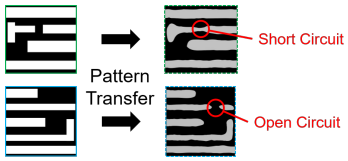

Figure 2. Short and open circuits caused by lithography

C. Optical Simulation for Hotspot Detection

In lithography, UV light is projected onto a photomask, which acts as the master template for semiconductor chips, and the pattern is transferred onto the silicon wafer through the photomask.

In optical simulation for lithography, the exposure process is simulated by calculating the light intensity distribution as the light emitted from the source passes through the photomask and projection lens system and reaches the photoresist. In this simulation, the optical characteristics of the exposure system (i.e., source characteristics) are represented by a matrix called the Sum of Coherent Systems (SOCS) kernel [12]. The post-exposure light intensity distribution is computed as the squared magnitude of the convolution between the SOCS kernel in the spatial domain and the layout pattern.

Let ϕ_j be the j-th kernel matrix and $\mathbf{M}$ the layout pattern matrix. The resulting light intensity distribution $\mathbf{I}$ is given by:

$$I(x,y) = \sum_{j=1}^{n} \sigma_j |(\phi_j * \mathbf{M})(x,y)|^2, \tag{1}$$

where the symbol $*$ denotes convolution and σ_j is a constant.

D. Machine Learning and Feature Extraction

It is time-consuming and labor-intensive for humans to analyze large amounts of data and derive rules or conditions related to specific phenomena. To address this, machine learning [13] has attracted attention as a technique that enables computers to learn from large-scale data and automatically construct models or algorithms for tasks, such as classification and prediction.

Machine learning can be categorized into supervised and unsupervised learning. Supervised learning involves estimating a mapping function based on given input data and corresponding labeled outputs. Among supervised learning tasks, classification aims to predict the class label of a given instance.

In classification problems, raw data alone often fails to achieve sufficient prediction accuracy. To address this, relevant elements are extracted from raw data in a process known as feature extraction. If two such elements are denoted as x and y, then the vector $\mathbf{f} = (x, y)$ is referred to as a feature vector. In this paper, we refer to both the feature vector itself and its components as features.

 ISBN: 978-1-68558-308-8

III. Hotspot Detection Using Machine Learning

In this section, we define the hotspot detection problem and describe a method for detecting hotspots using machine learning. We also introduce two widely used features: DBLF, which considers layout density, and HOLP, which approximates optical diffraction effects.

A. Hotspot Detection Problem

A hotspot refers to a pattern in lithography that poses a risk of causing an open or short circuit. Whether a given pattern is a hotspot can only be determined through lithography simulation or after actual semiconductor fabrication. The hotspot detection problem is to identify such hotspot patterns from a layout, based on known hotspot and non-hotspot examples.

B. Detection Method Using Machine Learning

Hotspot detection using machine learning consists of two main phases: the training phase and the testing phase. In this study, layout patterns are assumed to be represented as bitmap images, where wiring areas are white (pixel value: 1) and empty areas are black (pixel value: 0). The flow of machine learning-based hotspot detection is illustrated in Figure 3.

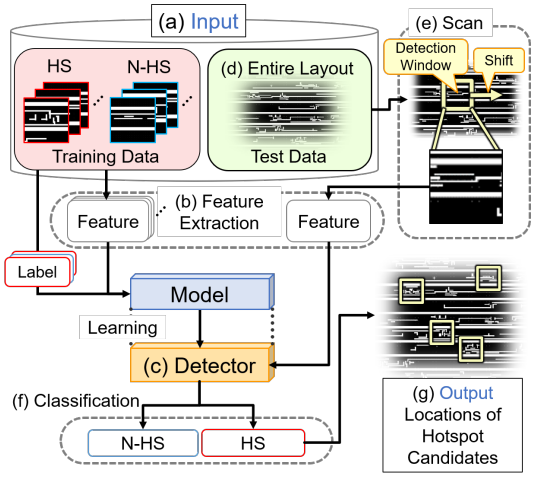

Figure 3. Flow of hotspot detection using machine learning [7]

In the training phase, as shown in Figure 3(a), a set of known hotspot and non-hotspot pattern images are provided as training data. Feature extraction is performed on each image to obtain features (Figure 3(b)). These features, along with the corresponding class labels indicating whether the image is a Hotspot (HS) or Non-Hotspot (N-HS), are input to a machine learning algorithm. The model is then trained to construct a hotspot classifier (Figure 3(c)).

In the testing phase, as shown in Figure 3(d), an image of the entire layout is given as test data. A region of interest used to determine whether a hotspot is present is referred to as a detection window. The detection window is scanned over the layout image (as in Figure 3(c)), and for each region corresponding to the detection window, feature extraction is performed and the extracted features are input into the trained classifier. The classifier outputs predicted labels (Figure 3(f)), enabling hotspot detection.

C. Existing Feature: DBLF

The feature DBLF considers the density of wiring in the layout, i.e., the proportion of area occupied by wires. The procedure to compute DBLF is as follows. The image corresponding to a detection window of $i \times i$ pixels is divided into $N \times N$ subregions, each consisting of $k \times k$ pixels (Figure 4). These subregions are referred to as local regions.

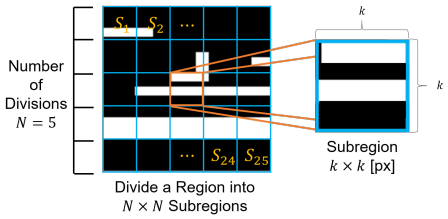

Figure 4. Division of image and local regions

For each local region s_l, the wire area ratio d_l is calculated. The DBLF feature is then represented as the vector of these values, as shown in (2):

$$\mathbf{F}_{\mathrm{DBLF}} = (d_1, d_2, \ldots, d_{N^2}). \tag{2}$$

D. Existing Feature: HOLP

The feature HOLP approximates the effect of optical diffraction in lithography by smoothing layout images.

First, a Gaussian filter is applied to the image $\mathbf{M}$ corresponding to a detection window to produce a smoothed image $\mathbf{M}_\mathrm{S}$ as shown in Figure 5(a). For each local region of $\mathbf{M}_\mathrm{s}$, intensity gradients are computed, as illustrated in Figure 5(b), and a histogram is constructed using the intensity gradients. Gradient

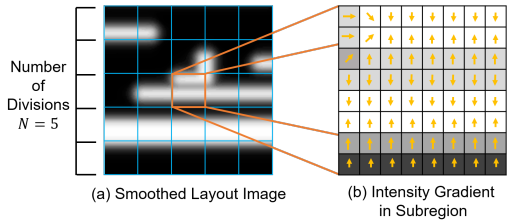

Figure 5. Gradient computation from smoothed image [3]

angles are quantized into bins as shown in Figure 6(a). In the example with 8 bins, the illustrated gradient falls into bin 4, and weighted voting is applied using gradient magnitude as the weight, as shown in Figure 6(b). Next, each local histogram is normalized so that the sum of all bin values equals 1. For a local region s_l, the histogram with B bins is represented as a vector $(g_1^l, g_2^l, \ldots, g_B^l)$. The overall HOLP feature is obtained by concatenating the histograms from all local regions, as shown in (3):

$$\mathbf{F}_{\mathrm{HOLP}} = (g_1^1, g_2^1, \ldots, g_B^1, \ldots, g_1^{N^2}, \ldots, g_B^{N^2}). \tag{3}$$

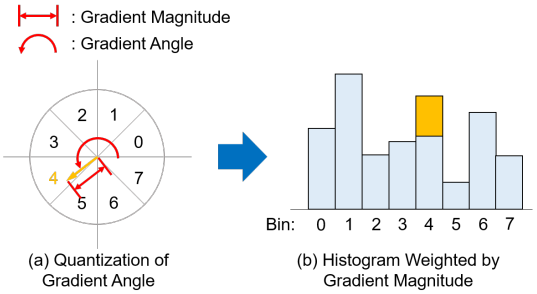

Figure 6. Construction of gradient histograms [3]

IV. Proposed Feature

Since hotspots are caused by optical effects as light travels from the exposure system's light source to the wafer, it is important to consider information about the optical system. Therefore, we propose a novel feature vector, named Optical System-Aware Mapping (OSAM), which incorporates the optical characteristics of the exposure system (hereafter, source characteristics).

A. Feature Considering Optical System Characteristics

As the source characteristics can be represented using SOCS kernels, we propose a feature based on these kernels. The proposed feature vector is derived from a simplified light intensity distribution calculated using reduced versions of both the mask pattern and the kernels, as well as by truncating the number of kernel components. By reducing the mask and kernel sizes in advance and limiting the kernel order, the computation time becomes significantly shorter compared to full lithography simulations.

B. Computation Procedure of the Proposed Feature

The computation procedure of the proposed feature is described below. We assume that the pattern image and the kernels have the same size.

First, the mask pattern image $\mathbf{M}$ is divided into $N \times N$ regions, where N is a user-defined constant. Next, for each $k \times k$-pixel local region s_l in the pattern image, the proportion of area occupied by wires is calculated as d_l, and a new $N \times N$ matrix $\mathbf{M}'$ is formed using these values in the same way as in DBLF, as illustrated in Figure 7.

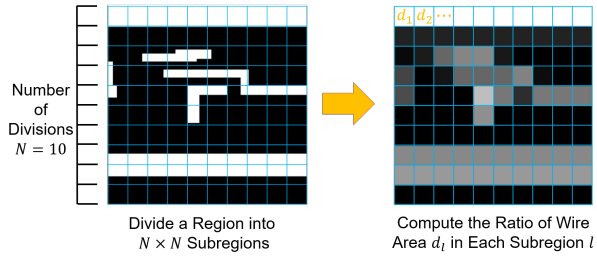

Figure 7. Downsampling of pattern image

Then, the kernels ϕ_j $(j = 1, 2, \ldots n')$ in the spatial domain are divided into $k \times k$-pixel local regions such that the center of the central local region aligns with the center of the kernels, where $n' (< n)$ is a user-defined integer constant and n is the original number of SOCS kernels used in the full optical simulation.

That is, to avoid splitting the central part of the kernels, each kernel is divided into $(N-1) \times (N-1)$ local regions when N is even, and into $N \times N$ regions when N is odd. When N is even, the peripheral areas of the kernels are ignored.

For each kernel, the average value e_l of the pixels in each local region s_l is calculated, forming a matrix ϕ'_j of size $(N-1) \times (N-1)$ or $N \times N$ depending on whether N is even or odd, as shown in Figure 8.

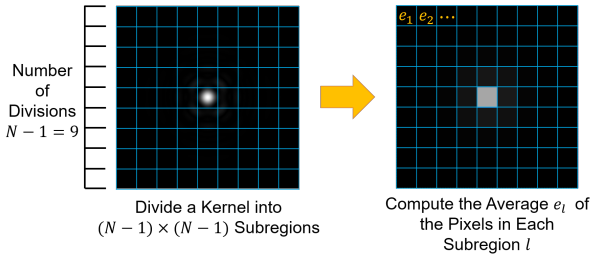

Figure 8. Downsampling of kernel

The simplified light intensity distribution $\mathbf{I}'$ is calculated using the following equation:

$$I'(x,y) = \sum_{j=1}^{n'} \sigma_j |(\phi'_j * \mathbf{M}')(x,y)|^2. \quad (4)$$

The central $N \times N$ submatrix $\mathbf{C}$ from $\mathbf{I}'$ is then flattened to form the proposed feature vector $\mathbf{F}_{\text{OSAM}}$, as shown in (5):

$$\mathbf{F}_{\text{OSAM}} = (C_{1,1}, C_{1,2}, \ldots, C_{N,N-1}, C_{N,N}). \quad (5)$$

Note that $\mathbf{C}$ can be obtained without computing the full convolution results, by restricting the computation to the region of interest.

The overall flow of computing the proposed feature is illustrated in Figure 9.

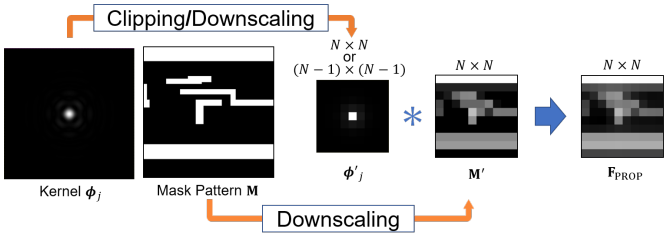

Figure 9. Flow of proposed feature computation

In our experiments, the proposed feature OSAM is used in combination with DBLF to enhance detection performance.

V. Experimental Results

In this section, we compare the hotspot detection accuracy between the existing feature DBLF and the proposed feature that considers optical system characteristics. Experiments

were conducted on a Linux server equipped with an Intel Xeon E5-2620 v4 2.2GHz processor and 128GB of memory. The experimental programs were implemented in Python 3.6.10, using OpenCV, Cython, scikit-learn, and TensorFlow as libraries.

To evaluate the proposed method, a dataset with specified optical conditions is required. We used data relabeled using an optical simulator [14] based on the ICCAD 2012 CAD contest dataset [10], as shown in Table I.

TABLE I. RELABELED ICCAD 2012 CONTEST DATASET

Circuit	#Samples	Process	#HS	#N-HS
data1	545	32nm	246	299
data2	4644	28nm	1417	3227
data3	5349	28nm	2930	2419
data4	3563	28nm	1100	2463
data5	2152	28nm	625	1527

A. Comparison Using AdaBoost

To compare the proposed feature with DBLF, we used AdaBoost [15], a commonly used machine learning algorithm in feature-based hotspot detection. The implementation used scikit-learn, and decision trees were used as weak learners in the ensemble model.

To focus on the fundamental performance of each feature, we evaluated hotspot classification (not full detection). In the experiments, 70% of the HS and N-HS regions in each dataset were used for training, and the remaining 30% were used for testing. The classification results were categorized into four types, as shown in Table II.

TABLE II. CLASSIFICATION ACCURACY CATEGORIES

	Hotspot	Non-hotspot
Predicted as Hotspot	True Positive (TP)	False Positive (FP)
Predicted as Non-hotspot	False Negative (FN)	True Negative (TN)

As an evaluation metric, we used the F1 score, which is the harmonic mean of Precision and Recall, defined as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{6}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{7}$$

$$\text{F1\_score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{8}$$

For DBLF, the pattern image was divided into $N = 10$, resulting in a 100-dimensional feature vector. For the proposed feature, the pattern was also divided with $N = 10$, while the kernel was divided with $N - 1 = 9$ to align the region center. To match region sizes, the outer region of the kernel was excluded. Although the optical simulator's kernel order n was 24, we set $n' = 1$ for the proposed feature. Thus, the proposed feature had 100 dimensions, and since it was concatenated with DBLF, the total dimensionality became $100 + 100 = 200$.

To ensure a fair comparison, we explored all 225 combinations of the following hyperparameters for each feature, selecting those that yielded the highest F1 scores:

- Number of weak learners: 2, 4, 6, ..., 1000
- Maximum tree depth: 2, 3, 4
- Learning rate: 0.95, 0.96, 0.97, 0.98, 0.99

These hyperparameters follow prior studies [7]. Each dataset was tested 5 times, and the average F1 score was computed. The best and average F1 scores across all combinations are shown in Table III. In all tables presented in this section, PROP denotes our proposed feature, OSAM.

TABLE III. F1 SCORES WITH ALL PARAMETER COMBINATIONS (DBLF 10×10)

	Best			Average		
Dataset	DBLF	PROP	(Ratio)	DBLF	PROP	(Ratio)
data1	0.8640	0.8769	(1.0149)	0.8111	0.8271	(1.0197)
data2	0.6199	0.6090	(0.9824)	0.5332	0.5370	(1.0071)
data3	0.8399	0.8456	(1.0067)	0.7907	0.8171	(1.0334)
data4	0.8281	0.8240	(0.9950)	0.6672	0.7475	(1.1204)
data5	0.7993	0.8260	(1.0334)	0.6544	0.7515	(1.1423)
Average	0.7902	0.7963	(1.0077)	0.6913	0.7360	(1.0647)

To compare under more similar dimensionality, we also tested DBLF with $14 \times 14 = 196$ dimensions. Results are shown in Table IV.

TABLE IV. F1 SCORES WITH ALL PARAMETER COMBINATIONS (DBLF 14×14)

	Best			Average		
Dataset	DBLF	PROP	(Ratio)	DBLF	PROP	(Ratio)
data1	0.8711	0.8769	(1.0067)	0.8111	0.8271	(1.0197)
data2	0.6247	0.6090	(0.9749)	0.5332	0.5370	(1.0071)
data3	0.8279	0.8456	(1.0214)	0.7907	0.8171	(1.0334)
data4	0.8306	0.8240	(0.9921)	0.6672	0.7475	(1.1204)
data5	0.8191	0.8260	(1.0084)	0.6543	0.7515	(1.1423)
Average	0.7947	0.7963	(1.0020)	0.6913	0.7360	(1.0647)

As shown in Table III, the proposed feature performed better than DBLF in both best and average F1 scores. Computation times were comparable. Table IV further shows that even with nearly equal dimensionality, the proposed feature still performed better than DBLF. These results indicate the effectiveness of incorporating source characteristics in hotspot detection.

We also investigated whether the proposed feature can be further improved by maximizing the optical detail, ignoring computation time. We used simulation images directly as feature vectors, resizing them to control dimensionality. These were concatenated with DBLF as in the proposed method. Results are shown in Table V.

TABLE V. F1 SCORES USING SIMULATION IMAGES AS FEATURES

Dataset	10×10	15×15	20×20	30×30	50×50	100×100
data1	0.8839	0.8885	0.8849	0.8848	0.8772	0.8771
data2	0.7164	0.7284	0.7176	0.7003	0.6797	0.6768
data3	0.8735	0.8823	0.8726	0.8688	0.8665	0.8617
data4	0.8611	0.8987	0.8864	0.8823	0.8811	0.8714
data5	0.8441	0.8613	0.8329	0.8372	0.8267	0.8225
Average	0.8358	0.8518	0.8389	0.8347	0.8263	0.8219

These results suggest that while the proposed feature can be further improved by adjusting kernel order or partition size, its performance is already strong at 10×10. Unexpectedly, the best score occurred at 15×15, not at higher resolutions,

indicating that the relabeled dataset is challenging, and that AdaBoost may perform better with lower-dimensional input. Similar findings have been reported in [16].

B. Comparison Using CNN

Based on the previous findings, we also performed experiments using Convolutional Neural Networks (CNNs) instead of AdaBoost. We used TensorFlow for implementation.

We compared DBLF and the proposed feature using the same parameter ($N = 10$). The CNN consisted of five layers: conv1-pool1-conv2-pool2-dense. Each convolutional layer used ReLU activation, with filters of size 3×3 and stride 1. The number of filters was 16 in conv1 and 32 in conv2. Each pooling layer performed max pooling with a 2×2 filter. We experimented with all combinations of epochs {10, 20, 30, 40, 50} and batch sizes {16, 32, 64, 128, 256}. The best F1 scores are shown in Table VI.

TABLE VI. BEST F1 SCORES USING CNN (FEATURE SIZE 10×10)

Dataset	DBLF	PROP	(Ratio)
data1	0.8774	0.8662	(0.9872)
data2	0.6552	0.6714	(1.0247)
data3	0.8531	0.8697	(1.0195)
data4	0.8471	0.8405	(0.9922)
data5	0.7880	0.8126	(1.0312)
Average	0.8042	0.8120	(1.0098)

From Tables III and VI, both features improved in F1 score using CNN. From Table VI, the proposed feature performed better than DBLF by approximately 1%, suggesting its potential effectiveness. We further experimented using simulation images as features in CNN, comparing 10×10 and 20×20 sizes. Results are shown in Table VII.

TABLE VII. BEST F1 SCORES USING CNN WITH DIFFERENT FEATURE SIZES

	Feature Size 10×10			Feature Size 20×20		
Dataset	DBLF	Sim.	(Ratio)	DBLF	Sim.	(Ratio)
data1	0.8774	0.8774	(1.0000)	0.8533	0.8701	(1.0197)
data2	0.6552	0.6583	(1.0047)	0.6910	0.7477	(1.0821)
data3	0.8531	0.8659	(1.0150)	0.8755	0.8972	(1.0248)
data4	0.8471	0.8513	(1.0050)	0.8649	0.9356	(1.0817)
data5	0.7880	0.8486	(1.0769)	0.8563	0.9065	(1.0586)
Average	0.8042	0.8203	(1.0200)	0.8282	0.8714	(1.0522)

These results indicate that the proposed feature has room for improvement, but already performs well at 10×10, supporting the practicality and effectiveness of our approach.

VI. CONCLUSION AND FUTURE WORK

In this study, we proposed a novel feature for machine learning-based hotspot detection that incorporates the optical characteristics of the exposure system, which are typically overlooked in existing approaches. Experimental comparisons with existing features showed that the proposed feature consistently improved detection performance.

As future work, we aim to improve the runtime efficiency of the proposed approach, for example, by performing convolutions in the frequency domain.

REFERENCES

[1] 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 *Design-Process-Technology Co-optimization for Manufacturability IX*, vol. 9427, SPIE, Mar. 2015, pp. 94270S1–11.

[2] 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 Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 3, pp. 460–470, Jan. 2015.

[3] Y. Tomioka, T. Matsunawa, C. Kodama, and S. Nojima, "Lithography hotspot detection by two-stage cascade classifier using histogram of oriented light propagation," in *2017 22nd Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jan. 2017, pp. 81–86.

[4] J. W. Park, A. Torres, and X. Song, "Litho-aware machine learning for hotspot detection," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 7, pp. 1510–1514, Jul. 2018.

[5] T. Zhou *et al.*, "Mining lithography hotspots from massive sem images using machine learning model," in *2021 China Semiconductor Technology Int. Conf. (CSTIC)*, Mar. 2021, pp. 1–3.

[6] G. Kataoka, M. Inagi, S. Nagayama, and S. Wakabayashi, "Novel feature vectors considering distances between wires for lithography hotspot detection," in *2018 21st Euromicro Conference on Digital System Design (DSD)*, Aug. 2018, pp. 85–90.

[7] G. Kataoka, M. Yamamoto, M. Inagi, S. Nagayama, and S. Wakabayashi, "Feature vectors based on wire width and distance for lithography hotspot detection," *IPSJ Trans. System and LSI Design Methodology*, vol. 16, pp. 2–11, Feb. 2023.

[8] T. Matsunawa, T. Kimura, and S. Nojima, "Lithography hotspot candidate detection using coherence map," in *Design-Process-Technology Co-optimization for Manufacturability XIII*, J. P. Cain, Ed., vol. 10962, SPIE, Mar. 2019, pp. 109620Q1–8.

[9] M. Yamamoto, M. Inagi, and S. Nagayama, "A feature vector considering characteristics of optical system for lithography hotspot detection," in *IEICE Technical Report (VLD2022-81)*, in Japanese, vol. 122, Mar. 2023, pp. 49–54.

[10] J. A. Torres, "ICCAD-2012 CAD contest in fuzzy pattern matching for physical verification and benchmark suite," in *2012 IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 2012, pp. 349–350.

[11] T. Matsunawa, B. Yu, and D. Z. Pan, "Optical proximity correction with hierarchical Bayes model," *J. Micro/Nanolithography, MEMS, and MOEMS*, vol. 15, no. 2, pp. 021009-1–8, Mar. 2016.

[12] N. Cobb, "Sum of coherent systems decomposition by SVD," University of California, Berkeley, Technical Report, Sep. 1995, pp. 1–7.

[13] S. Raschka, Y. H. Liu, and V. Mirjalili, *Machine Learning with PyTorch and Scikit-Learn: Develop machine learning and deep learning models with Python*. Packt Publishing Ltd., Feb. 2022.

[14] S. Banerjee, Z. Li, and S. R. Nassif, "ICCAD-2013 CAD contest in mask optimization and benchmark suite," in *2013 IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 2013, pp. 271–274.

[15] D. Solomatine and D. Shrestha, "AdaBoost.RT: A boosting algorithm for regression problems," in *2004 IEEE Int. Joint Conf. Neural Networks*, vol. 2, Jul. 2004, pp. 1163–1168.

[16] X. Liu, Y. Dai, Y. Zhang, Q. Yuan, and L. Zhao, "A preprocessing method of adaboost for mislabeled data classification," in *2017 29th Chinese Control And Decision Conf. (CCDC)*, May 2017, pp. 2738–2742.