LithoExp: Explainable Two-stage CNN-based Lithographic Hotspot Detection with Layout Defect Localization

CONG JIANG, Huazhong University of Science and Technology, Wuhan, China
HAOYANG SUN, Huazhong University of Science and Technology, Wuhan, China
DAN FENG, Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, Wuhan, China
ZHIYAO XIE, The Hong Kong University of Science and Technology, Hong Kong, Hong Kong
BENJAMIN TAN, Electrical and Software Engineering, University of Calgary, Calgary, Canada
KANG LIU, Huazhong University of Science and Technology, Wuhan, China

Convolutional neural networks (CNNs) successfully detect lithographic hotspots by learning from hand-designed features of layout patterns or entire layouts, as images, in an end-to-end fashion. However, compared to lithography simulation, CNN-based solutions demonstrate inferior hotspot detection accuracy and a high false-alarm rate. Moreover, the interpretability of the hotspot prediction process has yet to be considered due to the "black-box" nature of CNNs. In this work, inspired by conventional lithography simulation where defect regions are simulated as direct evidence for hotspot identification, we propose an explainable two-stage CNN-based hotspot detector that considers both the accuracy and interpretability of hotspot detection. Our architecture learns to locate the defect areas in the first stage as extracted hotspot features. In the second stage, we combine the strength of feature engineering and end-to-end learning, incorporating the original layout input, the learned defect location map from the first stage, and a fixed auxiliary region of interest (ROI) map for final hotspot detection. Experimental results for our technique exhibit the highest hotspot accuracy (98.1%) and the lowest false-alarm rate (4.0%) thus far compared to all prior CNN solutions. We also demonstrate the best overall qualitative and quantitative interpretability results with the highest increase in confidence (IC) and the lowest average drop (AD) in scores when CNN interpretation methods such as Grad-CAM-based approaches are applied. We further demonstrate use cases of our technique for successfully justifying and pinpointing hotspot mispredictions by examining the prediction evidence from our learned defect locations.

CCS Concepts: • **Hardware → Physical design (EDA)**; **Best practices for EDA**; • **Computing methodologies → Neural networks;**

Kang Liu is partly supported by National Natural Science Foundation of China No. 62202190, Hubei Natural Science Foundation No. 2023AFB237, and the Knowledge Innovation Program of Wuhan-Shuguang. Zhiyao Xie is supported in part by Hong Kong Research Grants Council ECS Grant No. 26208723.

Authors' Contact Information: Cong Jiang, Huazhong University of Science and Technology, Wuhan, China; e-mail: jiangconghust@hust.edu.cn; Haoyang Sun, Huazhong University of Science and Technology, Wuhan, China; e-mail: hsun2023@hust.edu.cn; Dan Feng, Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, Wuhan, Hubei, China; e-mail: dfeng@hust.edu.cn; Zhiyao Xie, The Hong Kong University of Science and Technology, Hong Kong, Hong Kong; e-mail: eezhiyao@ust.hk; Benjamin Tan, Electrical and Software Engineering, University of Calgary, Calgary, Alberta, Canada; e-mail: benjamin.tan1@ucalgary.ca; Kang Liu (Corresponding author), Huazhong University of Science and Technology, Wuhan, China; e-mail: kangliu@hust.edu.cn.

ACM 1084-4309/2025/03-ART40
https://doi.org/10.1145/3721129

Additional Key Words and Phrases: Machine learning for CAD, lithographic hotspot detection, explainable AI

ACM Reference Format:
Cong Jiang, Haoyang Sun, Dan Feng, Zhiyao Xie, Benjamin Tan, and Kang Liu. 2025. LithoExp: Explainable Two-stage CNN-based Lithographic Hotspot Detection with Layout Defect Localization. *ACM Trans. Des. Autom. Electron. Syst.* 30, 3, Article 40 (March 2025), 25 pages. https://doi.org/10.1145/3721129

1 Introduction

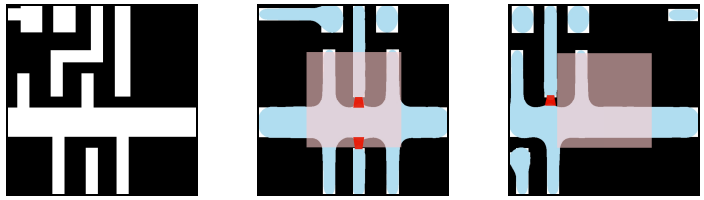
Optical lithography is the process in which a chip design is transferred from a photomask to a photoresist layer applied to a wafer. In advanced technology nodes, complex interactions between light patterns in lithography have made printed patterns sensitive to process variations, resulting in printing defects known as *lithographic hotspots*. To avoid yield loss, designers must identify these potential design defects as early as possible. This process, known as lithographic hotspot detection, is a critical step in the physical design of a **computer-aided design (CAD)** flow.

Over the years, various methods have been proposed to detect lithographic hotspots, including lithography simulation [12, 26], **pattern matching (PM)** [16, 42, 47], and **machine learning (ML)/deep learning (DL)**-based approaches [31, 44, 46, 48]. Among these, lithography simulation is deemed the golden solution due to its high accuracy, as shown in Figure 1. It applies mathematical and physical modeling of the lithography process on the layout patterns and simulates the defect regions, i.e., error markers, between problematic metals. However, it is also time-consuming. PM-based approaches, on the other hand, speed up the detection process by analyzing the feature characteristics of layout patterns against a library of known hotspots. However, a new hotspot that has not been seen before and is not included in the library may go undetected. Recent studies use ML/DL models for hotspot detection by learning the correlations between hotspot and non-hotspot features and their ground-truth labels from many simulated layout clips. These trained ML/DL models can achieve a much faster turnaround time than lithography simulation and improved accuracy and generalization over PM-based approaches, promising new directions for achieving both speed and accuracy in lithographic hotspot detection.

Early ML/DL-based solutions involved feature engineering of layout clips followed by ML models or **convolutional neural networks (CNNs)** [10, 11, 28, 42, 46, 48, 50]—for instance, bounded rectangle-based representation [48], fragmentation-based signature extraction [10, 11], concentric-circle-based sampling [50], and density transforms [28, 42] were used as layout feature inputs to ML models for layout classification. DCT coefficients were used in [46] to denote layout features and further analyzed by CNNs for hotspot detection. However, handcrafted feature engineering has limitations as it can exclude other important layout information crucial for hotspot detection, such as overall structure and relative positions between polygons, resulting in less optimal detection accuracy. Given the compelling feature extraction and expressive capability of CNNs, end-to-end lithographic hotspot detection has become the standard practice for all DL methods. It achieves **state-of-the-art (SoTA)** accuracy compared to all previous PM and ML methods thus far, where layout patterns are represented as images as direct input to a CNN without any loss of information. Many CNN architectures have been explored for hotspot detection, including binarized neural networks [19], Inception networks with attention modules [15], automated searched architectures [9], and more.

However, despite the promising performance of end-to-end CNN hotspot detection, pressing hurdles hold back the widespread adoption of CNN-based solutions. These include the following:

(1) **Inferior detection accuracy**. Ample and balanced training data are essential to a neural network's performance. However, in lithographic hotspot detection, hotspot samples are



(a) Layout clip (b) HS simulation (c) NHS simulation

Fig. 1. (a) Layout clip and lithography simulation outputs of (b) hotspot and (c) non-hotspot clips. Error markers are in red, and regions of interest (ROI) are in pink.

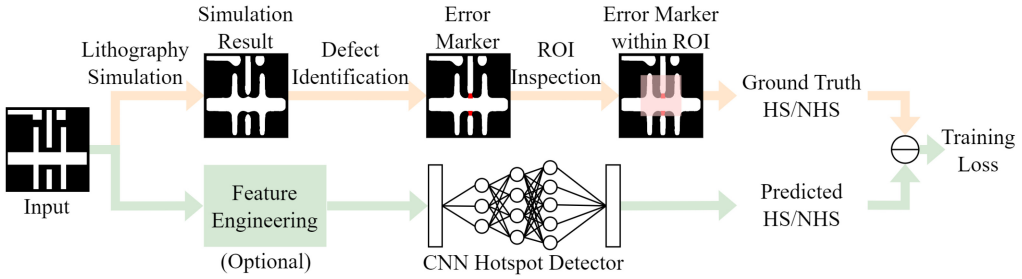


Fig. 2. Illustration of the training process (green arrow) of a CNN-based hotspot detector, where lithography simulation (orange arrow) provides the ground-truth labels for layout inputs.

generally far fewer than non-hotspots for training. Designers pursue overall accuracy but prioritize hotspot accuracy over non-hotspot by using higher class weight for hotspots [23] or biased learning [46] in training optimization; such tradeoffs usually result in a high false-alarm rate where non-hotspots are mispredicted. All prior works using CNNs maintain a tradeoff between hotspot and non-hotspot accuracy, and both are less accurate than the simulation-based approach, leaving room for more accurate layout feature extraction and classification within the DL framework.

(2) **Lack of interpretability**. In contrast to conventional hotspot detection based on lithography simulation tools whose mathematical modeling forms the basis for explanation, CNN-based models rely on neural networks, and the exact contribution of each convolutional layer is not explicitly known. These include their computing paradigms and weights of neurons that all remain obscure to humans for hotspot identification. The rationale for the prediction results of CNN-based hotspot detectors has yet to be probed in all prior work. Thus, we aim to address this problem by improving the interpretability of CNNs used for hotspot detection.

It is worth noting that CNN training uses lithography simulation for layout ground-truth labeling, and error markers denoting printing defects are simulated for hotspot determination within a predefined **region of interest (ROI)**, as shown in Figure 2. Upon obtaining these hotspot/non-hotspot labels, CNN-based hotspot detectors learn to classify layout features or entire layouts directly into binary categories. However, these error marker results, where hotspots occur, present direct evidence for identifying a hotspot but are entirely discarded and excluded from the learning process of DL solutions. As far as we know, all the prior work on CNN-based hotspot detection ignores such essential information and only relies on the resulting layout labels for training.

In this work, we seek to explain the final classification results of our end-to-end CNN hotspot detector instead of the internal function of each filter/module. In natural image classification tasks,

the interpretation results via interpretation methods should align with human interpretations of what the classified object typically looks like and can be visualized as specific patterns of an object. However, in the context of clip-based hotspot detection, there is no single polygon pattern alone that leads to the prediction of a hotspot clip but complex interactions during lithography of proximate polygons within a certain region of the clip. We consider this region the root cause and the critical area for a CNN to make a hotspot prediction. As a result, instead of identifying specific patterns, we explain the end results of our classification CNN using its focused region within the layout clip and inspect whether this region actually contains a defect.

Inspired by lithography simulation, in this work, we propose an explainable two-stage CNN-based hotspot detector to combine the strengths of feature engineering and end-to-end learning. In the first stage, we abstract the lithography simulation process and learn its resulted defect locations as extracted hotspot features of layout clips, which we further integrate into the second stage with the original layout clip and the predefined ROI map as additional feature channels, producing a novel three-channel feature for hotspot classification. Compared to prior work, experimental results demonstrate that our proposed architecture produces the highest hotspot accuracy, the lowest false-alarm rate, and the most accurate interpretation when commonly used CNN interpretation methods are applied.

Our contributions include:

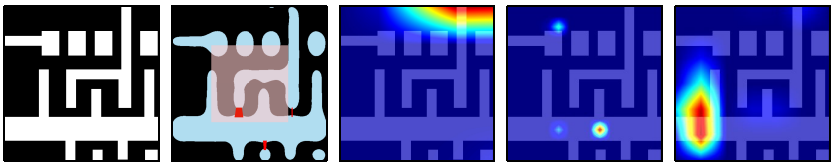
— A CNN-based lithographic hotspot detector where layout defect locations are explicitly learned for improved prediction accuracy and interpretability
— A two-stage CNN-based lithographic hotspot detector using combined feature engineering and end-to-end learning, yielding more accurate feature extraction and classification
— Extensive exploration and insights on the accuracy and interpretation of CNN-based hotspot detectors across a range of architectures and interpretation methods
— Successful identification of hotspot mispredictions based on learned defect locations of our proposed architecture

The rest of this article is organized as follows. In Section 2, we examine the interpretability of existing CNN-based hotspot detectors and explore a case study using error markers as hotspot features for classification. This motivates our design of an explainable two-stage CNN-based hotspot detector. To provide the technical preliminaries of our design, we present in Section 3 the basics of CNN, CNN interpretation methods, and the concept of lithographic hotspot detection. Following this, we describe our two-stage CNN-based hotspot detector in Section 4, detailing the proposed CNN architecture with training data augmentation and preprocessing methods. We then describe our experimental setups with datasets, baseline models for comparison, and CNN interpretation methods for evaluation in Section 5. We demonstrate, via our experimental results in Section 6, that our model achieves the highest accuracy and interpretability results compared to all prior work, which we further discuss in Section 7. We contextualize our work with related work in Section 8 and conclude in Section 9.

2 Motivation

2.1 Interpretability of Existing CNN-based Hotspot Detectors

Model interpretation aims to answer the question: "Why does the model make this decision?" An interpretable CNN model is assumed to be able to reasonably explain its predictions, thus significantly enhancing user trust and providing valuable insights for further model improvement. Although significant achievements have been made in explaining CNNs for image classification tasks, they have yet to receive attention in CNN-based lithographic hotspot detection.



(a) Layout clip (b) Simulation (c) TCAD'19 (d) TCAD'20 (e) TCAD'22

Fig. 3. (a) A layout clip, (b) its lithography simulation result with error markers (in red), and its Grad-CAM [34] interpretations of the prediction results made by various CNN-based hotspot detectors, including (c) TCAD'19 [46], (d) TCAD'20 [19], and (e) TCAD'22 [15].

To motivate our research on explainable CNN-based hotspot detectors, we examined the interpretability of existing CNN-based hotspot detectors used in prior work, including those from TCAD'19 [46], TCAD'20 [19], and TCAD'22 [15]. We applied a commonly used CNN interpretation method, Grad-CAM [34], to obtain corresponding explanatory heatmaps for each CNN given an input layout clip, as shown in Figure 3. In the heatmap, red areas indicate areas a CNN primarily focuses on and substantially influence the model's predictions. In contrast, blue regions are relatively less influential in the model's predictions. The visual interpretation results in Figure 3 reflect the crucial areas that different CNN hotspot detectors rely on for their hotspot determination. Without exception, all these highlighted areas either are on the boundary or corners of the layout or have no causal relationship with the "hotspot" defect areas. These highlighted areas are far from the ground-truth error markers (Figure 3(b) in red) and their surrounding regions, which are the actual areas that cause and enclose the hotspot. These results suggest the poor interpretability of these hotspot detection CNNs despite their SoTA detection accuracy, raising severe concerns about the reliability and trustworthiness of their predictions of hotspots. Therefore, it's imperative to develop an explainable CNN model that can accurately focus on the root-cause areas of a layout clip for hotspot identification that adheres to lithography principles and provides high detection accuracy at the same time.

2.2 Case Study: Classifying Error Markers into Hotspots

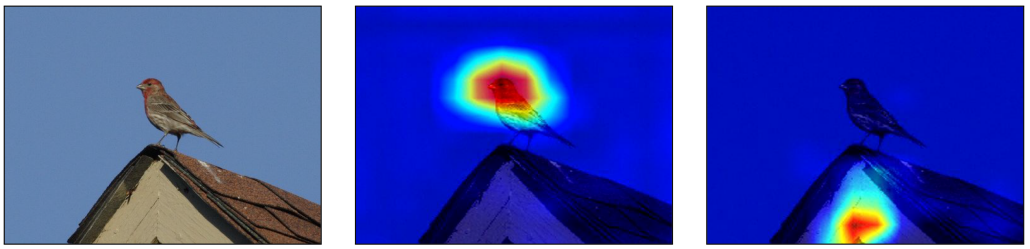
Enlightened by the fact that the highly interpretable lithography simulation relies on defect information for hotspot identification, a natural idea is to use defect locations as the extracted layout features for a CNN-based hotspot detector.

To verify this idea, we conduct a case study that uses simulated error marker maps from lithography simulation as the input for a simple four-layer neural network with three convolutional layers and one fully connected layer. We classify the error marker maps and, thus, the underlying layout clips into hotspot and non-hotspot. We use the dataset and experimental setups described in Section 5.1. Our experimental results in Table 1 show that it achieves 96.3% hotspot accuracy and a 5.5% false-alarm rate, comparable to the SoTA detection accuracy of prior hotspot detectors as we later show in Table 2. For a fair comparison, we train the same four-layer neural network to directly classify the layout clips instead of the error markers into binary classification, which obtains 81.9% hotspot accuracy and a 20.7% false-alarm rate, far inferior to the detection accuracy obtained using error marker inputs.

Our experimental results based on several CNN architectures in Table 1 suggest that using error markers as hotspot features is far more effective than end-to-end learning using layout inputs for hotspot detection. A relatively simple CNN can achieve such high accuracy, demonstrating its advantage in accuracy by identifying genuine hotspot features for classification.

Table 1. Comparison of Hotspot Detection Accuracy (%) between Using Inputs of Layout Clips and Error Markers

Model Input		Layout Clip	Error Marker
Four-layer CNN	Hotspot Accuracy	81.9	**96.3**
	False-alarm Rate	20.7	**5.5**
ResNet50	Hotspot Accuracy	91.0	**95.1**
	False-alarm Rate	0.9	**0.1**
ResNet152	Hotspot Accuracy	89.4	**99.8**
	False-alarm Rate	1.1	**0.1**



(a) (b) (c)

Fig. 4. Grad-CAM explanations of VGG-16 for (a) an input image when predicting (b) the *house finch* class and (c) the *house* class.

Motivated by our case study, our development in the next step of an explainable CNN-based hotspot detector involves the localization of layout defects and the following classification network that classifies these learned defects into hotspot/non-hotspot.

3 Preliminary and Problem Formulation

To appreciate the potential of explainable CNN-based lithographic hotspot detection, we present relevant technical preliminaries for CNN interpretation methods and formulate our problem of explainable CNN-based hotspot detection with evaluation metrics.

3.1 CNN Interpretation

CNNs have achieved great success in classification tasks, and they are also the most commonly used architecture in DL-based lithographic hotspot detection. However, their unique computing paradigm, equipped with specific network parameters, makes them extremely difficult for humans to understand and interpret. One line of CNN interpretation methods [5, 18, 30, 32, 34, 41] has been proposed to rationalize CNNs' prediction by attempting to identify discriminative regions of an input that positively influence the prediction of a specific class. An *explanation mask* highlights these regions with pixel-wise weights, as illustrated in Figure 4. Among these methods, **Class Activation Map (CAM)**-based techniques [5, 18, 34, 41] are now SoTA in CNN interpretation. In CAM, the explanation mask is obtained by linearly combining a weighted sum of each feature map in the output of the last convolutional layer, which is known for capturing high-level semantics of the target class for prediction, as shown in Equation (1):

$$M_{CAM} = Upsample\left(\text{ReLU}\left(\sum_k \alpha_k A_k\right)\right). \tag{1}$$

Here α_k represents the "importance" of the kth feature map A_k for predicting a target class. Various CAM-series CNN interpretations differ, in large part, in the way they assign α_k.

3.2 Lithographic Hotspot Detection

Lithography simulation applies optical and resist models on layouts, producing contours that resemble actual printed images. The simulation results are then compared against predefined design rules to identify hotspots, such as line width and spacing between metals. To avoid high computational costs and excessive runtime for full-layout lithography simulation, designers usually partition layouts into clips using a sliding window. Each clip undergoes a separate lithography simulation. Since only the central area within the clip has access to all its proximal information for lithography, we refer to this central region as the ROI for each clip, as shown in pink in Figure 1. In all the clips, only ROIs that are predefined as a fixed area are examined for hotspots. A clip is classified as a "hotspot" if the simulated defect is inside the ROI or its overlap with the ROI is above a predefined threshold (e.g., 30%) of its own area. Otherwise, it is classified as a "non-hotspot."

3.3 Problem Formulation

Lithographic hotspot detection aims to (1) identify as many actual hotspot clips as possible to increase yield and (2) avoid misprediction on layout clips that are non-hotspots, which can lead to wasted design effort. When a CNN model is used for detection, the objectives also include reasonable interpretation of its predictions through qualitative and quantitative analysis of the explanation mask generated by a commonly used CNN interpretation method.

We define the following metrics to evaluate the classification performance of a CNN-based hotspot detector:

*Definition 1 (**Hotspot Accuracy (HA)**).* The ratio of actual hotspot clips that are truthfully predicted as hotspots, which indicates how well the hotspot detector can identify real hotspots among defective designs.

*Definition 2 (**False-alarm Rate (FAR)**).* The ratio of actual non-hotspot clips that are mispredicted as hotspots, which measures the undesired misdiagnosis that causes wasted effort in resimulation for non-hotspot verification.

The explanation mask displays the specific input regions that a CNN uses to predict a particular class. In an *ideal* explainable CNN, these input regions should be intuitive to humans or are consistent with the laws of physics, thus providing reasonable *visual interpretation*. Moreover, these highlighted regions should fulfill two criteria: (1) they should be complete, meaning that they include all the essential information pertaining to the target class, and (2) they should exclude any irrelevant information that is not useful or could even negatively impact the target prediction. Therefore, we quantitatively evaluate the interpretability of a CNN-based hotspot detector using the following two metrics, as in prior literature [5, 41].

*Definition 3 (**Increase in Confidence (IC)**).* The ratio of T real hotspots that have increased hotspot prediction score p' than their original prediction score p, when the explanation mask is applied to the layout clip as input to the CNN hotspot detector, as shown in Equation (2):

$$IC = \frac{1}{T}\sum_{i=1}^{T}\{p_i' <= p_i : 0, 1\}. \tag{2}$$

The IC score measures how much irrelevant information is removed from the original layout clip when an explanation mask is applied to the input. This reduction of irrelevant information can

potentially improve the accuracy of the target prediction. Therefore, **more explainable CNNs tend to have higher IC scores**.

*Definition 4 (**Average Drop (AD)**).* The drop ratio of hotspot prediction score p' when the explanation mask is applied to the layout clip as input to the CNN hotspot detector, compared to their original prediction score p. When the masked clip increases hotspot prediction, the drop is 0. The AD is averaged over all T actual hotspots and calculated as in Equation (3):

$$AD = \frac{1}{T}\sum_{i=1}^{T}\frac{\max(0, p_i - p_i')}{p_i}. \tag{3}$$

In contrast to the IC score, the AD score measures the integrity of the critical information related to the target class that remains within the input after explanation masking. Incomplete target information can undermine its prediction. Therefore, **more explainable CNNs usually result in lower AD scores**.

With the above evaluation metrics, we formulate the CNN-based lithographic hotspot detection problem as follows:

Problem 1: Explainable CNN-based Hotspot Detection—given layout clips with ground-truth hotspot/non-hotspot labels and corresponding error markers from lithography simulation, we want to train a CNN-based classifier that (1) maximizes the HA and minimizes the FAR and (2) maximizes the IC and minimizes the AD when applying CNN interpretation methods.

4 Proposed Method

4.1 Overview

Complex interactions between light and layout patterns during lithography result in printing defects where open or short circuits occur. Lithography simulation proactively identifies these suspected problematic metals and denotes them with error markers. The locations of these possible printing defects directly determine a hotspot if it overlaps with the ROI by a predefined amount, and we deem such defect locations the *root cause* for a hotspot layout. Our case study using error markers for hotspot classification in Section 2.2 has demonstrated the potential of CNN hotspot detection based on defect locations. Intuitively, we can learn to localize these defect regions as input features for a hotspot detector instead of aimlessly learning hard-to-interpret hotspot features end to end as in prior work.

Learning the locations of defect regions has dual benefits: (1) the guided learning of defect locations extracts genuine features for hotspot detection, which presumably increases detection accuracy, and (2) as in lithography simulation for hotspot identification, hotspot root causes are taken as inputs to the hotspot detector, which enables interpretability and transparency of a CNN-based hotspot detector. Therefore, we propose a two-stage CNN-based hotspot detector, as shown in Figure 5, that learns to localize defect locations at the first stage and incorporates them as hotspot features for classification at the second stage.

4.2 Preprocessing of Error Markers

In all prior work on CNN-based lithographic hotspot detection, simulated error markers are solely used for layout ground-truth labeling but excluded from the training optimization loop. In this work, we augment the training dataset with simulated error markers and use them to learn the defect maps, i.e., blank layouts with defects only, as the critical layout features. In this case, simulated error markers are used as ground truth for these learned defects. Specifically, we train a feature extractor using simulated error markers as the learning targets. The extractor inputs the

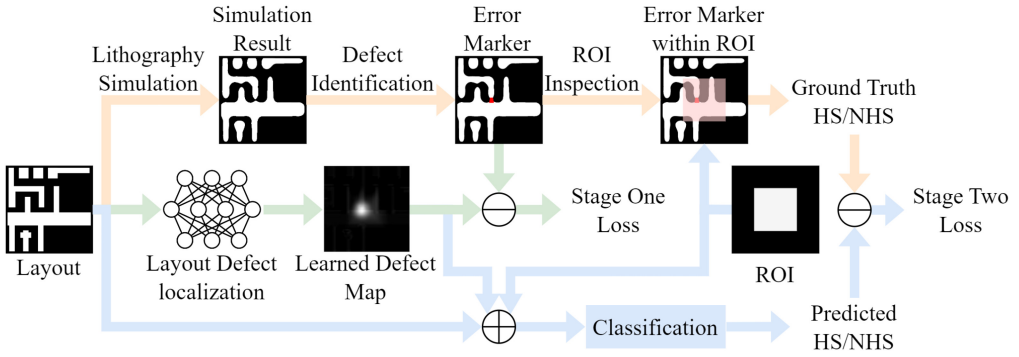


Fig. 5. Illustration of our proposed explainable two-stage CNN-based hotspot detection.

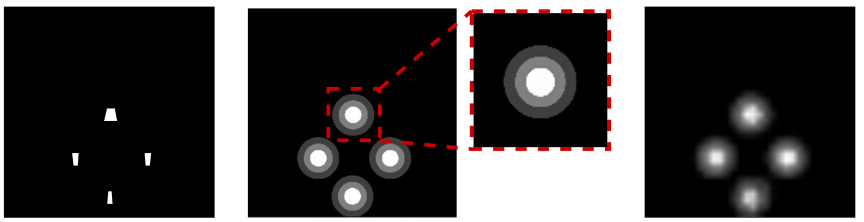


Fig. 6. (a) Original error marker, (b) preprocessed error marker, and (c) learned defect location map.

layout clip and outputs a learned defect map of the same size, similar to the process in lithography simulation. Such a feature extractor is expected to extract as many defects as possible with minimum omissions, and the defect occupations should be restrained enough without mistakenly identifying any background areas as defects.

However, precise defect learning presents several difficulties. On the one hand, as error markers occupy a minuscule area in the defect map, accurately finding their coordinates poses a significant challenge. On the other hand, the error marker shapes vary, adding more obstacles to the feature extractor. In fact, instead of attributing defects with precise coordinates and shapes, we are primarily interested in their locations within the defect map; this indicates that employing simple-to-learn shapes to indicate the defect locations is appropriate in our task. Thus, we propose a preprocessing method for the error markers to facilitate defect localization.

As shown in Figure 6, we substitute the original error markers with round shapes proven to ease the learning process [2]. Specifically, we define circumcircles to locate different defect regions, each with a minimum radius r_i covering the entire ith original error marker. To avoid overlap between error marker locations, we restrict the radius of the round shape to be no larger than 1/10 of the layout width s. This constraint is mainly for cases of large error markers, which usually reside on the margins of the clip. We place the origin of the round shape in the center of the error marker and constrain the radius of the processed defect location as shown in Equation (4):

$$r_i \leftarrow min\left(r_i, \frac{s}{10}\right). \tag{4}$$

We assign different weights to different parts of a round shape to characterize their varying importance as hotspot features and introduce attenuating importance from the center to the margins of the round shapes. We represent the defect locations with varying pixel intensities—the outside

areas are denoted with smaller pixel values, and the centers have larger pixel values. Specifically, we use three-level pixel intensities to represent each pixel in the round shapes, which are equally divided over the radius, and from the center to the outside are valued by 1, 2/3, and 1/3, respectively.

4.3 Combining Feature Engineering and End-to-end Learning

We learn the locations of hotspot defect areas as extracted features to help improve prediction accuracy when followed by a classification network. However, deficiencies exist in the localization results that lead to inferior detection accuracy when the learned defect maps are directly classified into hotspots and non-hotspots, as we later show in the experimental results in Figure 9(b) and Table 4. On the one hand, the learned defect location maps may not encompass all defect locations, potentially missing critical defects that highly intersect with the ROI, i.e., hotspots, leading to classification errors. On the other hand, the generated defect maps unavoidably include noise, compromising classification accuracy. Furthermore, structural information of the layout clips and the proximity of the defects are not included in the defect location maps but are beneficial for hotspot detection. Therefore, we include both learned defect location maps and original layout clips as inputs to our CNN hotspot detector, which combines the strengths of feature engineering and end-to-end learning by providing a shortcut to the root cause of hotspot prediction and compensating for any deficiency in the extracted hotspot features by including the original layout clip as in end-to-end learning.

We note that not all simulated error markers cause actual printing defects, and only those intersected with the central ROI are deemed to result in hotspots. In prior work, the CNN hotspot detector has to explore this ROI effect in determining a hotspot by learning from massive training samples, which increases the difficulty of effective feature extraction for hotspot detection. In this work, instead of implicitly learning the ROI features from training data, we introduce an ROI map as an additional feature map in addition to the learned defect location map and original layout clip. Such an ROI map uses the same ROI as in lithography simulation for labeling the training clips, and it's predefined and fixed as an internal network parameter. This ROI map explicitly emphasizes the "real" hotspot features for prediction, reducing the learning difficulty of identifying actual defects for the final classification task.

Thus, we combine the learned defect location maps with the original layout clips and a predefined ROI map that include distilled (defect locations), constraining (ROI map), and overall layout information (layout clip) and formulate three-channel features for the following classification network for hotspot detection.

4.4 A Two-stage CNN for Hotspot Detection

To facilitate defect localization trained on preprocessed error markers and the following classification of combined feature maps, we propose a two-stage CNN hotspot detector where each stage has different CNN designs. In the first stage, we aim to locate the layout defect regions and formulate a defect location map as hotspot features for further classification in the second stage. Inspired by image segmentation tasks in computer vision [20, 21], we adopt and modify a commonly used segmentation network, the FCN8s [29], for defect localization. Instead of segmenting each pixel in the layout into specific categories as in image segmentation, we learn the actual pixel values of the preprocessed error markers as described in Section 4.2.

In the second stage, we take as input the three-channel feature map consisting of the original layout clip, the learned defect location map, and the predefined ROI map used in layout labeling, as described in Section 4.3. To extract spatial information in the three-channel feature map, we apply four consecutive convolutional layers followed by one fully connected layer with sigmoid activation for the final hotspot/non-hotspot classification.

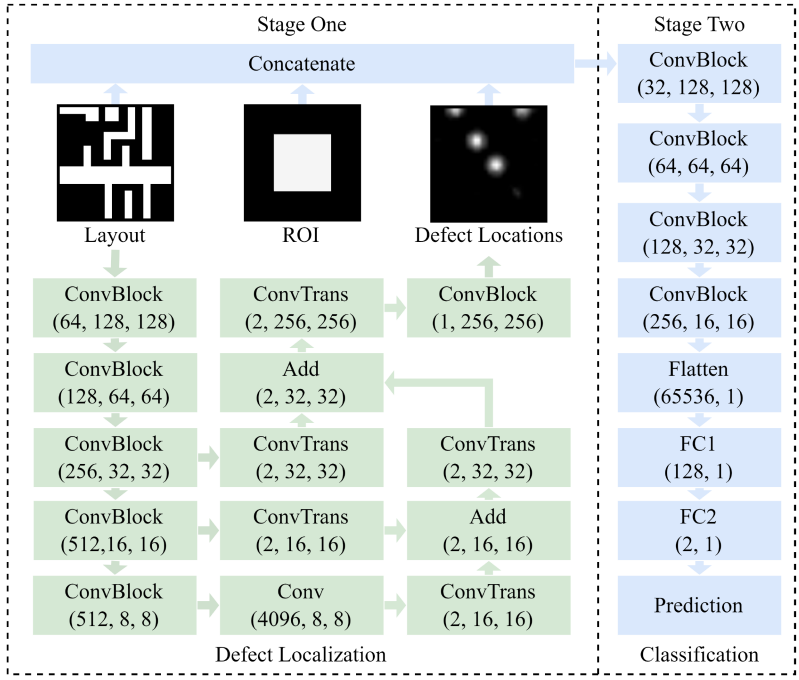


Fig. 7. Architecture of our two-stage CNN hotspot detector.

We show our two-stage CNN architecture in Figure 7, where the ConvBlock consists of two consecutive convolutional layers, each followed by a batch normalization and ReLU activation. We represent the output dimensions of network layers within parentheses.

We train the two-stage networks separately and sequentially. In stage 1, we first train the neural network with learning targets of preprocessed error markers, which generates a learned defect location map for each training layout clip. These learned defect location maps, along with their original layout clips and the fixed ROI map, are used as training inputs for the neural network in the second stage. The training optimization in the two stages uses an MSE and cross-entropy loss, respectively, as shown in Equation (5):

$$\begin{aligned} \mathcal{L}_1 &= \sum_{x \in \mathcal{D}} MSE(M_E, F_1(x)) \\ \mathcal{L}_2 &= \sum_{x \in \mathcal{D}} -y \log \left(F_2(x, F_1(x), M_R)\right). \end{aligned} \tag{5}$$

Here, the training loss $\mathcal{L}_1$ for stage 1 and $\mathcal{L}_2$ for stage 2 are summed up over all training clips x in dataset $\mathcal{D}$ with ground-truth labels $y \in [0, 1]$. We denote the first-stage defect localization network F_1 and the second-stage classification network F_2. M_E is the learning target of the preprocessed error markers map in the first stage, and M_R is the predefined ROI map in the second stage.

5 Experimental Setup

5.1 Dataset

We experiment with the same dataset used in prior work [22, 31], which uses 45 *nm* FreePDK for its design and Mentor Calibre [35] for lithography simulation. We note that other commonly used

datasets, such as ICCAD-2012 [40], have proprietary simulation settings and parameters and thus cannot be used here for lithography simulation and error marker generation. We divide the chip layout into clips of $1,110\ nm \times 1,110\ nm$ in GDSII format, with the ROI set at the center of each clip, measuring $555\ nm \times 555\ nm$. Defect location maps for each layout clip are generated by lithography simulation and saved as GDSII files in the same dimensional size as the layout clip.

To prepare the clips as appropriate input to the neural network, we use the gdspy library [25] to convert each clip and its corresponding defect location map into images and resize them to 256×256. We use binary pixel values; metal areas have a pixel value of 1, and the unpopulated regions are 0. In the preprocessed defect location map, we use three-level pixel intensities 1/3, 2/3, and 1 to represent defect round-shape locations from the outside to the center. Our training dataset comprises 250,509 hotspots and 268,466 non-hotspots, and the test dataset includes 999 hotspots and 19,001 non-hotspots nonoverlapping with the training data.

5.2 Baseline Hotspot Detectors

We compare our proposed hotspot detector with three baselines from prior work using CNNs. One is a CNN using hand-designed features as inputs, i.e., the DCT coefficients of layout clips [46], and the other two are representative end-to-end learning networks [15, 19]. We denote them as TCAD'19, TCAD'20, and TCAD'22, respectively. TCAD'19 consists of 4 convolutional layers and 2 fully connected layers, TCAD'20 consists of 10 binarized convolutional layers grouped in 4 blocks with residual connections, and TCAD'22 uses 5 Inception and attention blocks followed by 3 fully connected layers.

5.3 CNN Interpretation Methods

To interpret the CNN's prediction of a specific class, we use SoTA CNN interpretation methods, e.g., the CAM-based series [18], to highlight the discriminative regions that positively affect a target prediction. We apply Grad-CAM [34], Grad-CAM++ [5], LayerCAM [18], and ScoreCAM [41] to the feature maps of the last convolutional layers in our proposed architecture and layout-clip-classification baselines to obtain the explanation masks for each network, which we superimpose on the layout clip for better visualization. In the quantitative evaluation of the interpretability of each network, we apply the explanation masks on the input layout clips to calculate the IC and AD scores as defined in Section 3. Specifically, in our architecture, we mask our layout clip in the three-channel features to the second-stage classification network.

5.4 Experimental Platform and Training Hyperparameters

We implement our two-stage CNN-based hotspot detector in Python 3.7 with Tensorflow 2.8.0 and test on a server with Xeon W-3335 CPU and Nvidia GeForce RTX 3090 GPU. We train each of the two-stage CNNs separately for 10 epochs with an SGD optimizer, with a batch size of 32 and a learning rate of 0.001.

6 Experimental Results

6.1 Classification Accuracy and Interpretation Results of Different Hotspot Detectors

Table 2 presents the classification accuracy and inference time of our proposed two-stage CNN hotspot detector against three baselines and their interpretability results for hotspot clips under various interpretation methods.

Accuracy. Our proposed architecture achieves the highest HA of 98.1% and the lowest FAR of 4.0%. TCAD'19 using DCT coefficients as layout features exhibits the lowest detection accuracy, suggesting higher efficiency of end-to-end learning (TCAD'20, TCAD'22) and our combined architecture than feature engineering-based methods.

Table 2. Accuracy (%), Interpretability Results (%), and Inference Time (ms) of Our Proposed Architecture Compared with Baselines

Net	Accuracy		Grad-CAM		Grad-CAM++		LayerCAM		ScoreCAM		I.T.
	HA	FAR	IC	AD	IC	AD	IC	AD	IC	AD	
TCAD'19	94.3	9.8	3.3	98.3	3.1	145.6	1.8	67.0	**3.1**	96.1	4.03
TCAD'20	97.2	7.2	0.0	364.2	0.0	340.1	0.0	336.8	0.0	338.0	1.34
TCAD'22	97.3	6.9	0.1	172.8	**6.8**	83.7	6.6	58.7	0.4	119.8	2.53
Ours	**98.1**	**4.0**	**7.8**	**14.4**	5.4	**28.0**	**13.8**	**9.4**	2.7	**16.4**	6.52

I.T.: inference time.

Runtime. TCAD'20 requires the lowest inference time of 1.34 ms per layout clip due to its binarized weights and reduced computational complexity, whereas the DCT computations of TCAD'19 result in even longer inference time than the complex attention and inception modules used in TCAD'22. Our two-stage CNN hotspot detector requires the longest 6.52 ms per inference run for a layout clip incorporating an encoder–decoder architecture; in contrast, all prior CNNs use only the encoder for compressed latent learning rather than our defect map generation that has a feature map size the same as the input. Despite the additional inference cost, our architecture achieves the highest detection accuracy among all CNN-based solutions. It is worth noting that lithography simulation on average requires 3,779 ms to classify one layout clip, and all CNN-based hotspot detectors, including ours, are much faster by three orders of magnitude.

Interpretability Analysis. In interpretability analysis, we favor a higher IC and a lower AD. Our architecture outperforms all three baselines in the AD among all interpretation methods. It obtains the highest IC (Grad-CAM of 7.8%, LayerCAM of 13.8%) in two of four interpretation analyses and the second highest in the other two cases (Grad-CAM++ of 5.4%, ScoreCAM of 2.7%) against three baselines. TCAD'22 using Grad-CAM++ and TCAD'19 using ScoreCAM achieve slightly better performance on the IC metric than our architecture; however, they demonstrate inferior interpretability in all the other cases. TCAD'20 employing binary neural networks exhibits the worst interpretability performance against all other architectures examined by any of the four interpretation methods.

Visual Interpretation. We visualize the interpretation results in Figure 8 for a hotspot clip with their lithography simulation results and learned defect location map from our architecture. Regions marked in red have a larger influence than blue regions in predicting a hotspot. Our architecture best rationalizes its classification results by locating the most precise areas where actual defects occur, as evidenced by error markers from lithography simulation. In contrast, the visual interpretations of three baselines are mostly less accurate and either occupy a large area of the layout clip (e.g., Grad-CAM++ and ScoreCAM for TCAD'20 and TCAD'22) or mark the regions on the layout borders that are entirely irrelevant for hotspot prediction (e.g., Grad-CAM for TCAD'20 and TCAD'22, and all cases for TCAD'19). TCAD'22 occasionally presents slightly better visual interpretation than the other two baselines by covering the defect area but also including large non-essential parts. Our architecture demonstrates the most stable results across all interpretation methods.

6.2 First-stage Layout Defect Localization with Different Network Architectures and Error Marker Preprocessing Methods

To explore the efficacy of error marker preprocessing for first-stage defect localization, we compare the accuracy of defect localization in cases with or without error marker preprocessing as the

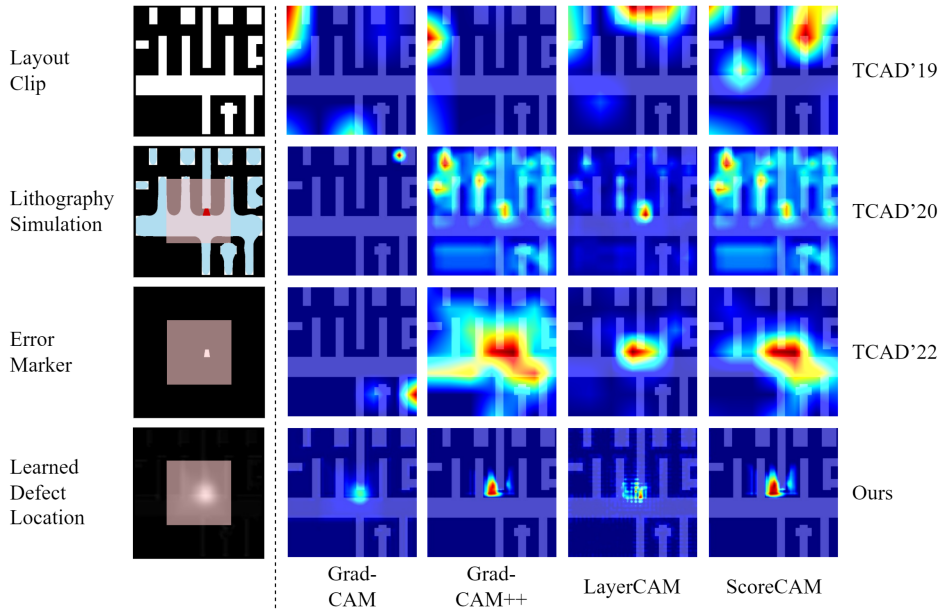


Fig. 8. Visual interpretation of our proposed CNN hotspot detector and three baselines with various interpretation methods.

learning target. We also evaluate how the segmentation network affects defect localization, of which we experiment with networks including FCN32s, FCN8s [24], U-Net [33], and DeepLabv3+ [6], all commonly used in segmentation tasks [29]. We show the visualization of the learned defect location maps for each combination of error marker preprocessing methods and segmentation networks in Figure 9. Quantitatively, we measure defect localization accuracy using the **Defect Match Rate (DMR)** and **False Defect Rate (FDR)**, where the DMR measures the ratio of ground-truth error markers that are successfully learned in a defect location map, and the FDR calculates the area percentage of learned locations at which actual defects do not exist. We provide their definitions as follows.

Definition 5 (Defect Match Rate (DMR)). The ratio of ground-truth defects, as verified by lithography simulation, that have been successfully located in the learned defect location maps. We denote the number of actual defects e_t in the tth layout clip with d_t and the number of learned defects e'_t with d'_t. The DMR is calculated with the average ratio of all T layout clips, as described in Equation (6):

$$DMR = \frac{1}{T}\sum_{t=1}^{T}\frac{\sum_{i=1}^{d_t}\sum_{j=1}^{d'_t} I_{IoU(e'_{tj},e_{ti})>0.5}}{d_t}. \tag{6}$$

Here, $I_{condition}$ is the indicator function that returns 1 if the condition is met and otherwise 0. $IoU(a,b)$ is the Intersection over Union function that calculates the overlap ratio between a and b.

Definition 6 (False Defect Rate (FDR)). The ratio of non-defect background area b_t in the tth ground-truth defect map that is actually segmented as defects e'_t in its corresponding learned defect location map. The FDR is averaged over all T clips, as described in Equation (7):

$$FDR = \frac{1}{T}\sum_{t=1}^{T}\frac{e'_t \wedge b_t}{b_t}. \tag{7}$$

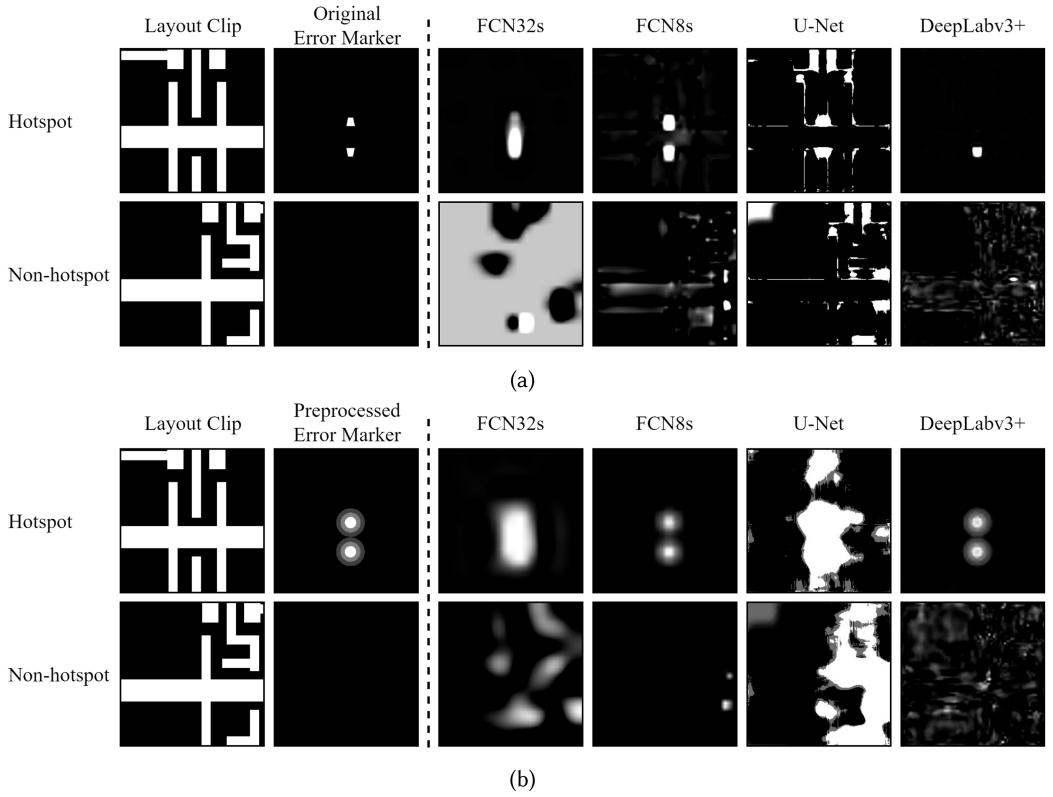


Fig. 9. Visualization of the learned defect location maps with different error marker preprocessing methods and segmentation networks. (a) Learned defect location maps with learning targets of original error markers and (b) learned defect location maps with learning targets of preprocessed error markers.

In practice, to identify and regularize each individual defect in the learned defect locations, we use the `findContours` function in the OpenCV library to represent these learned defect locations with circumcircles, each covering a cluster of pixels with positive pixel values. Specifically, we consider a defect "matched" if its learned location intersects with over 50% of the area of the corresponding error marker or preprocessed round shape in the simulated defect maps, as shown in Equation (6). To mitigate noise generated in the learned defect location maps, we filter out learned defects with areas smaller than five pixels.

We report the DMR and FDR results in Table 3 for different cases of error marker preprocessing and segmentation networks. We find that learning the original shapes of error markers results in an unsatisfactory DMR and FDR. FCN32s achieves the highest DMR of 76.2% but with an FDR of 23.9%, which suggests almost one-quarter of the background area of the ground-truth defect map is predicted as non-existing defects. DeepLabv3+ has a lower FDR of 7.2%, indicating fewer segmentation errors of non-defect areas. However, it only detects 57.5% of true defect locations. The other two segmentation networks, FCN8s with a DMR of 55.1% and an FDR of 26.0% and U-Net with a DMR of 43.8% and an FDR of 10.5%, also demonstrates inadequate performance for reliable defect localization.

In comparison, learning preprocessed error markers as round shapes significantly improves the DMR for all segmentation networks and reduces the FDR for FCN8s. Specifically, the DMR for

Table 3. Defect Match Rate (DMR) and False Defect Rate (FDR) of Defect Localization with or without Error Marker Preprocessing Using Different Encoder–Decoder Networks

Error Marker	Net	DMR (%)			FDR (%)		
		NHS	HS	Avg	NHS	HS	Avg
Original	FCN32s	**87.2**	**65.3**	**76.2**	41.6	8.5	23.9
	FCN8s	52.1	61.2	55.1	23.5	29.0	26.0
	U-Net	53.0	36.7	43.8	**9.7**	11.2	10.5
	DeepLabv3+	58.4	54.7	57.5	11.9	**3.0**	**7.2**
Preprocessed	FCN32s	**100.0**	**100.0**	**100.0**	27.0	34.8	30.4
	FCN8s	94.7	**100.0**	97.5	**5.3**	12.7	**8.9**
	U-Net	**100.0**	**100.0**	**100.0**	37.6	54.0	45.4
	DeepLabv3+	95.4	**100.0**	97.8	18.2	**11.8**	14.5

FCN32s and U-Net increases to 100% by successfully detecting all defects in the layouts. However, their increase in the FDR indicates a rise in falsely identifying the layout background area as defects. The FDR of FCN32s increases from 23.9% to 30.4%, and for U-Net it largely increases from 10.5% to 45.4%. DeepLabv3+ also achieves a significant improvement of the DMR in locating actual defects, rising from 57.5% to 97.8%, but at the cost of an increase in segmentation errors, with the FDR increasing from 7.2% to 14.5%. FCN8s achieves the most significant improvement by learning preprocessed error markers instead of the original ones, with its DMR increasing from 55.1% to 97.5% and the FDR decreasing from 26.0% to 8.9%, suggesting enhanced defect localization accuracy and reduced noise. Therefore, we use FCN8s for defect localization in our first-stage CNN hotspot detector.

DMR and FDR results for learning original and preprocessed error markers can also be examined by the visualization of their learned defect location maps in Figure 9. We see that FCN8s more accurately identifies the actual defects and largely reduces background noise when learning from preprocessed error markers, whereas the rest of the three segmentation networks discover more defect areas but also with increased noise background.

6.3 Second-stage Classification Accuracy with Different Feature Combinations

We further explore how the feature inputs of the second-stage classification network affect prediction accuracy, as shown in Table 4. We compare cases using features of the learned defect location map only, its combination with the predefined ROI map or the layout clip, and our final feature design of a three-channel feature map, which includes the learned defect location map, the layout clip, and the ROI map. We also evaluate the basic end-to-end learning case where the layout clip is directly classified into hotspots/non-hotspots and the case in which its combination with the ROI is used as input.

Experimental results demonstrate that instead of using the original layout clips as inputs as in end-to-end learning, which results in a huge FAR of 20.7%, learning from extracted layout features—the defect locations—drastically improves the HA to 95.8% and the FAR to 6.6%. When the ROI constraint is directly provided in the input instead of implicitly learned from training data, its combination with the original layout clip or the learned defect map increases accuracy. Again, the learned defect map shows advantages over the original layout clip in classification accuracy when coupled with the ROI as input. We also find that concatenating the learned defect map with

Table 4. Accuracy (%) of Various Feature Combinations between Layout Clip, Learned Defect Location Map, and the Predefined ROI Map for Second-stage Classification

Feature Combinations	HA	FAR
Layout	81.9	20.7
Layout and ROI	96.4	12.1
Learned Defect Locations	95.8	6.6
Learned Defect Locations and ROI	97.6	5.5
Learned Defect Locations and Layout	96.6	**4.0**
Learned Defect Locations and ROI and Layout	**98.1**	**4.0**

the original layout clip achieves better overall accuracy than all prior combinations. When the ROI constraint is directly provided in the input by coupling the leaned defect locations with the predefined ROI map, the HA increases to 97.6%, and the FAR reduces to 5.5%, compared to cases that use learned defect locations only and implicitly learn such ROI constraint from training data. We achieve the highest HA of 98.1% and FAR of 4.0% by combining feature engineering and end-to-end learning, which joins the extracted features of defect locations, the original layout clip, and an auxiliary ROI map as an integrated feature map to the classification network, which encompasses all the essential and auxiliary information for hotspot identification.

6.4 Identifying Mispredictions by Verifying Learned Defect Locations

Lithographic hotspot detection is expected to identify as many hotspots as possible. However, despite the SoTA hotspot accuracy of our proposed hotspot detector, a 100% detection rate is more than challenging to accomplish, as either stage of the defect localization and classification can induce deviations from their ground truth that impact final detection accuracy. However, even though we cannot achieve perfect precision, we can still make more use of our fairly accurate defect localization ability, as indicated in Table 3 and Figure 9, by using error marker preprocessing and FCN8s segmentation. We illustrate how we can rectify possible flaws in the second-stage classification by identifying mispredicted hotspots by verifying their learned defect location maps. We show in Figure 10 one example of misprediction that classifies a hotspot as non-hotspot, where we find in the learned defect location map that our hotspot detector locates the defect region within the ROI. According to lithography simulation, it should be a hotspot, but it has a contradictory non-hotspot prediction. In this way, we can identify real hotspots that escape detection.

We carefully inspected all 40 hotspot clips that our CNN hotspot detector misclassifies as non-hotspot in our dataset and successfully found that 26 of them (a ratio of 65%) have classification results that contradict their learned defect locations. These hotspot clips will apply lithography simulation for final verification.

7 Discussion

7.1 Detection Accuracy of Various Architectures

As shown in Table 2, our architecture surpasses all three prior works in accuracy, and we owe this to the simulated error markers we used in training. These error markers guide the network to learn actual defect locations as genuine hotspot features for further hotspot detection, instead of solely considering feature distinction between hotspots and non-hotspots as in all prior studies. In addition to the essential defect information, we also use layout clips and predefined ROI maps for classification that compensate for any information loss that's not included in the learned defect

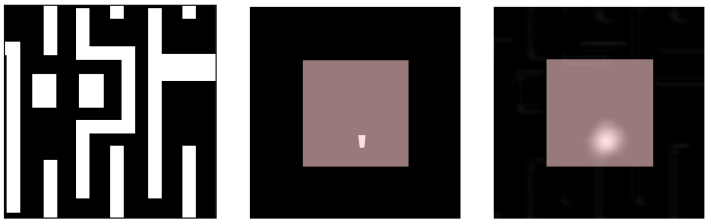


Fig. 10. Identifying hotspot clips mispredicted as non-hotspot from learned defect locations. We show (a) a hotspot clip, (b) its simulated error marker with ROI, and (c) its learned defect location map with ROI.

location maps. Among the three models, the improved accuracy of end-to-end learning architectures (TCAD'20 and TCAD'22) demonstrates the compelling feature extraction capability of CNNs compared to hand-designed feature engineering (TCAD'19). TCAD'22 exhibits slightly better detection performance than the other two. Its enhanced accuracy is mainly due to its use of spatial- and channel-wise attention modules, which contribute significantly to the feature extraction that allows the model to extract more distinctive features between hotspots and non-hotspots. In contrast, TCAD'19 shows a relatively lower detection accuracy, which can be partially attributed to the loss of crucial information during the DCT conversion of the layouts, after which the layout structural information no longer exists.

7.2 Detection Accuracy of Various Feature Combinations

In Table 4, we compare the hotspot detection accuracy of the classification network using different feature combinations. We find increased HA and FAR of hotspot detection using learned defect locations than using the original layout clips as classification input features. Detection accuracy increases when the original clip or the defect map is coupled with the ROI map. Besides, we see further enhancement in accuracy by complementing the feature input to include the original layout clips and an ROI map alongside the learned defect locations. We see the largest accuracy enhancement when we include the original layout clips and the ROI map alongside the learned defect locations. The structure information embedded in the original clips, such as metal distance and relative positions, is also essential for hotspot determination, which is not reflected in the defect location maps. Implementing an ROI map to the features helps regulate the network's attention to the central regions of the layout clips and defect location maps for more directional feature extraction. The integrated information embedded in the layout clips, defect locations, and ROI maps combines to provide the most comprehensive yet critical information that yields the highest accuracy.

7.3 Can We Directly Identify Hotspots from Learned Defect Location Maps?

In light of the accurate defect localization of our proposed architecture, as shown in Table 3 and Figure 9, a simple and straightforward approach to hotspot detection is to inspect whether there exist defects within the ROI of our learned defect location maps after stage 1. This method directly identifies a hotspot clip without needing a second-stage classification network. Specifically, we define a threshold parameter to identify a learned defect as an actual hotspot based on its intersection with the ROI. Such a threshold is defined as the *minimum* ratio of the intersection between the learned defect and the ROI to the defect area, which varies between 0 and 1. We examine special cases of threshold = 0, where a layout clip is considered a hotspot as long as a learned defect overlaps with the ROI, and threshold = 1, where a learned defect has to entirely reside within the ROI

Table 5. Accuracy (%) of Directly Identifying Hotspots from Learned Defect Location Maps with Various Threshold Settings That Denote the Minimum Overlap Ratio between Learned Defect Locations and the ROI

Threshold	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
HA	100	100	100	100	99.7	99.3	98.2	94.2	90.2	85.7	81.0
FAR	69.3	62.7	58.4	55.2	52.5	49.5	46.9	44.1	41.4	38.7	34.9

A layout clip is considered a hotspot as long as the intersection ratio is above the threshold. Otherwise, it's a non-hotspot.

to be determined as a hotspot. All learned defects will be preprocessed to become circle contours as described in Section 6.2 before overlap calculation.

We show hotspot detection accuracy with the HA and FAR in Table 5. When we use a lower threshold on the intersection ratio of learned defect and ROI, we obtain a higher HA. For example, when threshold = 0 to 0.3, we achieve 100% HA but also with a high FAR up to 69.3%. This occurs due to the surrounding noise areas generated in the learned defect locations that are mistakenly identified as part of the defects. As we increase the threshold, we can partially mitigate the noise effect with a lower FAR but also come out with lower HA. On examining all the thresholds varying between 0 and 1, it's hard to find an optimum intersection ratio between the learned defect location and ROI for actual defect identification that can balance the HA and FAR. In fact, learned defects are imperfect compared to ground-truth error markers. Other useful information embeds in the learned defects such as pixel value intensities but are not considered in hotspot detection. An oversimplified binary representation of the learned defects and layout background for hotspot determination inevitably results in unsatisfactory hotspot detection accuracy.

7.4 Interpretability of Various Architectures

As direct evidence for hotspot determination, our use of the defect locations in the architecture and training process tremendously enhances the transparency and interpretability of our CNN hotspot detector, as indicated by the improved IC and AD scores, compared to all baseline architectures, as shown in Table 2. Our architecture significantly reduces AD scores, suggesting that more accurate "hotspot" regions are used for detection. Among the baseline architectures, TCAD'22 achieves the highest IC score using Grad-CAM++; its use of spatial- and channel-wise attention improves the feature extraction process between hotspots and non-hotspots. However, such features, though distinctive, can be non-essential for hotspots, as reflected by its large AD scores. TCAD'19 achieves the highest IC score using ScoreCAM. Its information transformation from the spatial domain to the frequency domain of layouts discards rich information, which potentially includes both hotspot-relevant and hotspot-irrelevant information, thus sometimes yielding a high IC but also a high AD.

7.5 Visual Interpretation of Various Architectures

As depicted by the visual interpretation results in Figure 8, TCAD'20 mostly fails to concentrate on a specific layout area for hotspot classification, primarily because its binarized calculation makes it challenging to process layout structural information. TCAD'19 shows a significant deviation from the ground-truth defects in its interpretation, as it uses the DCT coefficients to represent the layout and discards spatial information, which is essential for hotspot determination. TCAD'22 shows more accurate interpretation than TCAD'19 and TCAD'20, owing to its more efficient attention modules, which enrich the diversity of extracted features, thereby increasing the likelihood of extracting critical features. However, this diversity also includes features that are irrelevant to

hotspots, as reflected by the large highlighted areas in its visual interpretation. In contrast, our architecture provides the most accurate and clean interpretation consistent with the actual defects located by lithography simulation, and we attribute this to the use of defect locations in training the CNN to classify a hotspot.

7.6 Our Training Data Augmentation Requires No Additional Effort

Prior works use data augmentation mostly to expand dataset size and enrich data information to improve classification accuracy [31]. Though effective, they require delicate data augmentation techniques and extensive lithography simulation for augmented layouts. The additional effort for data augmentation linearly rises with the amount of augmented data. Our approach, on the contrary, extends the training dataset with intermediate results of lithography simulation when labeling the training data, i.e., the error marker information denoting actual defect locations in the layouts. These error marker maps are entirely neglected in prior work but contain significant information for hotspot detection. They are immediately accessible after training data labeling with no additional cost.

7.7 Effects of Error Marker Preprocessing and Segmentation Networks on Defect Localization

As shown in Table 3 and Figure 9, the first-stage defect feature extractor that learns preprocessed error markers significantly improves defect localization, compared to learning the original error markers. We attribute this improvement to the fact that, as found in prior work [2], it is easier for a neural network to learn a round shape than a complex shape. Since our focus in hotspot detection lies on the location of defects represented by error markers, and their shape is of less importance, preprocessing them with round shapes retains the defect locations while reducing task complexity. This change leads to improved performance in defect localization. Furthermore, the original error markers constitute only a tiny portion of the entire layout clip, which further adds difficulties to the task of learning their accurate locations.

In terms of the segmentation networks used in the feature extractor, we notice a similar DMR achieved by various segmentation networks, and FCN8s obtains the smallest FDR. We attribute this to its structural design. The convolution flow and integration of feature maps from multiple layers in FCN8s, compared to FCN32s and DeepLabv3+, enable better learning of defect location features with less noise. U-Net has the most output noise, potentially from its connection to the coarse features obtained at the input level.

7.8 Wider Implications for ML-based CAD Flows

The insufficient interpretability of existing CNN lithographic hotspot detectors raises important questions about the reliability and trustworthiness of using ML in CAD. With increasing layout complexity, it becomes more challenging for neural networks to learn effective feature representations through end-to-end learning from the layout. These neural networks are not guaranteed to learn essential and genuine features of a hotspot. In addition, as neural networks become increasingly sophisticated, their interpretability concerns compound.

Since ML and DL techniques are becoming more and more involved in the many CAD flow steps, interpretability considerations in ML-CAD are paramount. Given the existing end-to-end learning paradigm of CNN architectures used in the CAD flows, we surmise that any current ML-CAD solutions using neural networks are facing interpretability issues. Given that the key insight we provide in this study is using actual hotspot features, i.e., the defect locations, to assist hotspot detection in the lithographic context, we posit that similar ground-truth features exist in other

CAD domains that are beneficial to the interpretability and accuracy of a neural network solution, and future work is needed to discover these.

7.9 Differences between Our CNN Classifier-based Hotspot Detector for Layout Clips and Object Detection-based Hotspot Detector for a Full Layout

Our explainable CNN-based hotspot detection relies on layout clip defect localization at the first stage and learns a defect map followed by a second-stage classification network. We note that some works use object-detection-based methods for hotspot detection, such as [7, 8], which are quite different from our method. First, their methods identify hotspot regions for a full/large layout, whereas ours classifies partitioned clips into hotspots and non-hotspots. They work on an object detection problem, and ours solves a binary classification problem. We operate in a more fine-grained manner in learning the printing defects within a layout clip rather than framing out all the potential hotspot clips as in [7, 8]. Our defect localization step is more like an image-to-image generative process than object detection within the original input with bounding-box generation as in [7, 8]. It is worth noting that our layout clip defect localization only requires the placement information of defects, which is sufficient for layout clip classification, rather than locating their precise coordinates as in [7, 8]. This makes complex object detection architectures for bounding-box generation unnecessary. In addition, unlike the classifiers in [7, 8] that perform prediction for each proposed hotspot region, our classifier predicts the input layout clip based on the entire defect feature map and operates only once. Lastly, the hotspot prediction for the proposed clip regions in [7, 8] relies on the CNN classifiers for automatic layout clip feature extraction (as in all prior CNN hotspot detection works); the interpretability of these classifiers remains unknown.

7.10 Detection Accuracy Comparison between Our Two-stage CNN-based Hotspot Detector and GNN-based Hotspot Detector

In addition to CNN-based architectures for lithographic hotspot detection, other neural networks have been explored in prior works, such as GNN, as used in [38]. We reproduce the same GNN architecture using the exact node and edge feature representations and evaluate our layout clip dataset, as shown in Table 6. We obtain an HA of 97.5% and an FA of 5.3%. This is competitive with our HA of 98.1% and FA of 4.0%, and it is the best performing compared to all three baseline CNN architectures in Table 2. Layout clips in the form of metal polygons are structured data and can be expressed using graph representation. With proper node and edge feature representation of the layout graph, high accuracy of hotspot detection is feasible. Of particular note is the fastest inference time of the GNN architecture, which requires only 0.06 ms per layout clip and is 100 times faster than our architecture. This efficiency results from the fact that GNN typically has only a small number of layers, and its architecture takes input from extracted graph features. In contrast, all CNN-based methods take the entire layout clip image as input; complex feature extraction using deep convolutional layers consumes large interference time. However, as interpretation methods on GNNs are still being actively explored [1, 13, 17], their explanations are on the sub-graph level presented as nodes and edges, which are less informative than the visual explanations of CNNs and sometimes hard to be directly compared with hotspot root causes, i.e., error makers.

7.11 ROC Curves and AUC Scores of Various Architectures

In addition to the commonly used HA and FAR metrics as in all prior works, we present the ROC curves and calculate the corresponding AUC scores for our proposed explainable CNN-based hotspot detector and our baselines, as shown in Figure 11. We have similar findings in the ROC curves and AUC scores that operate among various classification thresholds. Our proposed architecture achieves the highest AUC score above all baselines. TCAD'20 and TCAD'22 perform

Table 6. Accuracy (%) and Inference Time (ms) of GNN-based Hotspot Detector Compared with Our Explainable Two-stage CNN-based Hotspot Detector

	HA	FAR	I.T.
DATE'22	97.5	5.3	**0.06**
Ours	**98.1**	**4.0**	6.52

I.T.: inference time.

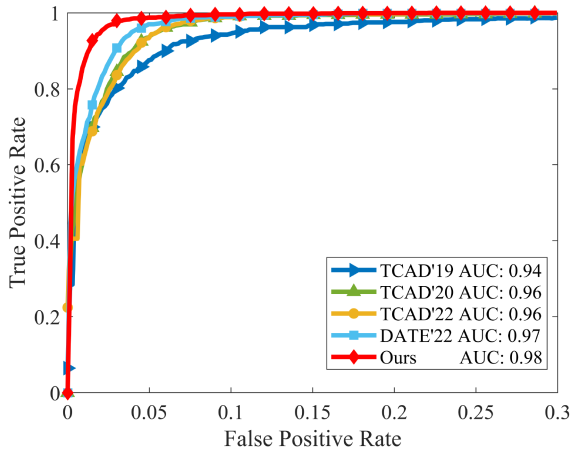


Fig. 11. ROC curves and AUC scores of our proposed CNN-based hotspot detector and baseline architectures.

similarly, and TCAD'19 is the least accurate. DATE'22, based on a GNN architecture for hotspot detection, surpasses all prior CNN-based solutions except ours.

8 Related Work

8.1 Lithographic Hotspot Detection

Lithography simulation [12, 26] provides the most accurate hotspot detection but is also time-consuming. With the increasing scale and complexity of chip layouts, it is hard to meet the requirements of fast turnaround time for trial designs. PM-based methods [16, 42, 47] facilitate faster hotspot detection but usually fail to generalize to unseen patterns. Recently, ML, especially CNN-based, methods [45] are promising in lithographic hotspot detection by improving the detection accuracy and generalizability compared to PM-based methods. Various research has been conducted from the perspectives of feature extraction and model architectures. For example, hand-designed feature engineering of layout patterns, such as DCT coefficients, is proposed [46], followed by a CNN. End-to-end CNN learning for hotspots includes explorations of binary neural networks [19], sophisticated Inception blocks with attention modules [15], or networks designed by **neural architecture search (NAS)** [9]. Our proposed two-stage CNN architecture joins the forces of feature engineering and end-to-end learning by combining learned layout features and original layout clips for hotspot detection. Other architectures for hotspot detection include graph neural networks [38] that use graph representations of layouts for classification.

Other works [7, 8, 14, 51] detect hotspots from an entire layout for faster detection speed; we perform hotspot detection in fine-grained layout clips partitioned from a full layout. Other topics, such as data sampling for training efficiency [43], have also been studied.

However, in all prior work for lithographic hotspot detection, the interpretability analysis of existing CNN models has yet to be considered, and explaining NAS [9] or GNN-based architectures [38] still calls for more efficient solutions. We explore and enhance the interpretability of CNN-based lithographic hotspot detectors by learning actual defect locations from simulated error markers, which are direct evidence for a hotspot as in lithography simulation.

8.2 CNN Interpretation

CNN interpretation methods have been extensively studied to explain why CNN models predict what they predict in the computer vision domain [4]. Early work [3, 36, 37, 39] based on the gradient information of the CNN generates saliency maps by backpropagating gradients of a specific class to the input. However, these methods tend to produce noisy results and are susceptible to the "gradient saturation" effect of the model. Post methods [27, 49] use models with good interpretability, such as linear regression and decision trees, to fit partial data predictions of the CNN. They then use the explanation of the agent model as a substitute for the original CNN. The downside is their inability to fit complex decision boundaries of sophisticated tasks. Perturbation-based methods [30, 32] explore the importance of different input regions on CNN predictions by controlling local variations with methods such as masking, but they usually suffer from high computational costs as the number of perturbation combinations to the inputs is enormous. In our work, we use SoTA CAM-based methods [5, 18, 34, 41] to explain CNN-based hotspot detectors, where a weighted combination of the feature maps of convolutional layers highlights the specific input areas that the CNN relies on for prediction.

9 Conclusion

In this article, we proposed a two-stage CNN-based lithographic hotspot detector that demonstrates the highest hotspot accuracy, the lowest false-alarm rate, and the most accurate qualitative and quantitative interpretation results compared with prior work under a series of CNN interpretation methods. It sheds light on the enhancement of the interpretability of ML-CAD solutions and calls for more explainable DL architectures and learning paradigms in developing future ML-CAD tools.

References

[1] Chirag Agarwal, Marinka Zitnik, and Himabindu Lakkaraju. 2022. Probing GNN explainers: A rigorous theoretical and empirical analysis of GNN explanation methods. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 8969–8996.

[2] Mohamed Baker Alawieh, Yibo Lin, Zaiwei Zhang, Meng Li, Qixing Huang, and David Z. Pan. 2020. GAN-SRAF: Subresolution assist feature generation using generative adversarial networks. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems* 40, 2 (2020), 373–385.

[3] Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, Klaus-Robert Müller, and Wojciech Samek. 2016. Layer-wise relevance propagation for neural networks with local renormalization layers. In *Proceedings of the 25th International Conference on Artificial Neural Networks (ICANN'16), Part II 25*. Springer, 63–71.

[4] Diogo V. Carvalho, Eduardo M. Pereira, and Jaime S. Cardoso. 2019. Machine learning interpretability: A survey on methods and metrics. *Electronics* 8, 8 (2019), 832.

[5] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N. Balasubramanian. 2018. Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV'18)*. IEEE, 839–847.

[6] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. 2018. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV'18)*. Springer, 801–818.

[7] Ran Chen, Wei Zhong, Haoyu Yang, Hao Geng, Fan Yang, Xuan Zeng, and Bei Yu. 2022. Faster region-based hotspot detection. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems* 41, 3 (2022), 669–680.

[8] Ran Chen, Wei Zhong, Haoyu Yang, Hao Geng, Xuan Zeng, and Bei Yu. 2019. Faster region-based hotspot detection. In *Proceedings of the 56th Annual Design Automation Conference 2019*. 1–6.

[9] Zihao Chen, Fan Yang, Li Shang, and Xuan Zeng. 2023. Automated and agile design of layout Hotspot detector via neural architecture search. In *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE'23)*. IEEE, 1–6.
[10] Duo Ding, Jhih-Rong Gao, Kun Yuan, and David Z. Pan. 2011. AENEID: A generic lithography-friendly detailed router based on post-RET data learning and hotspot detection. In *Proceedings of the 48th Design Automation Conference*. IEEE, 795–800.
[11] Duo Ding, Andres J. Torres, Fedor G. Pikus, and David Z. Pan. 2011. High performance lithographic hotspot detection using hierarchically refined machine learning. In *16th Asia and South Pacific Design Automation Conference (ASP-DAC'11)*. IEEE, 775–780.
[12] Andreas Erdmann, Tim Fühner, Feng Shao, and Peter Evanschitzky. 2009. Lithography simulation: Modeling techniques and selected applications. In *Modeling Aspects in Optical Metrology II*, Vol. 7390. SPIE, 13–29.
[13] Lukas Faber, Amin K. Moghaddam, and Roger Wattenhofer. 2021. When comparing to ground truth is wrong: On evaluating GNN explanation methods. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 332–341.
[14] Tianyang Gai, Tong Qu, Shuhan Wang, Xiaojing Su, Renren Xu, Yun Wang, Jing Xue, Yajuan Su, Yayi Wei, and Tianchun Ye. 2022. Flexible hotspot detection based on fully convolutional network with transfer learning. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems* 41, 11 (2022), 4626–4638.
[15] Hao Geng, Haoyu Yang, Lu Zhang, Fan Yang, Xuan Zeng, and Bei Yu. 2022. Hotspot detection via attention-based deep layout metric learning. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems* 41, 8 (2022), 2685–2698.
[16] Xu He, Yao Wang, Zhiyong Fu, Yipei Wang, and Yang Guo. 2023. A general layout pattern clustering using geometric matching-based clip relocation and lower-bound aided optimization. *ACM Transactions on Design Automation of Electronic Systems* 28, 6 (2023), 1–23.
[17] Qiang Huang, Makoto Yamada, Yuan Tian, Dinesh Singh, and Yi Chang. 2022. Graphlime: Local interpretable model explanations for graph neural networks. *IEEE Transactions on Knowledge and Data Engineering* 35, 7 (2022), 6968–6972.
[18] Peng-Tao Jiang, Chang-Bin Zhang, Qibin Hou, Ming-Ming Cheng, and Yunchao Wei. 2021. Layercam: Exploring hierarchical class activation maps for localization. *IEEE Transactions on Image Processing* 30 (2021), 5875–5888.
[19] Yiyang Jiang, Fan Yang, Bei Yu, Dian Zhou, and Xuan Zeng. 2021. Efficient layout hotspot detection via binarized residual neural network ensemble. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems* 40, 7 (2021), 1476–1488.
[20] L'ubor Ladickỳ, Paul Sturgess, Karteek Alahari, Chris Russell, and Philip H. S. Torr. 2010. What, where and how many? Combining object detectors and CRFs. In *Proceedings of the 11th European Conference on Computer Vision (ECCV'10)Computer Vision, Part IV 11*. Springer, 424–437.
[21] Biao Li, Yong Shi, Zhiquan Qi, and Zhensong Chen. 2018. A survey on semantic segmentation. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW'18)*. IEEE, 1233–1240.
[22] Kang Liu, Benjamin Tan, Gaurav Rajavendra Reddy, Siddharth Garg, Yiorgos Makris, and Ramesh Karri. 2021. Bias busters: Robustifying DL-based lithographic hotspot detectors against backdooring attacks. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems* 40, 10 (2021), 2077–2089.
[23] Kang Liu, Haoyu Yang, Yuzhe Ma, Benjamin Tan, Bei Yu, Evangeline F. Y. Young, Ramesh Karri, and Siddharth Garg. 2020. Adversarial perturbation attacks on ML-based CAD: A case study on CNN-based lithographic hotspot detection. *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 25, 5 (2020), 1–31.
[24] Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 3431–3440.
[25] Adam McCaughan and Lucas H. Gabrielli. 2023. gdspy. (April 2023). https://gdspy.readthedocs.io/en/stable/
[26] Chris A. Mack. 2005. Thirty years of lithography simulation. In *Optical Microlithography XVIII*, Vol. 5754. SPIE, 1–12.
[27] Juan Mata. 2011. Interpretation of concrete dam behaviour with artificial neural network and multiple linear regression models. *Engineering Structures* 33, 3 (2011), 903–910.
[28] Tetsuaki Matsunawa, Jhih-Rong Gao, Bei Yu, and David Z. Pan. 2015. 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, 201–211.
[29] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. 2021. Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 7 (2021), 3523–3542.
[30] Vitali Petsiuk, Abir Das, and Kate Saenko. 2018. Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421* (2018).
[31] Gaurav Rajavendra Reddy, Constantinos Xanthopoulos, and Yiorgos Makris. 2018. Enhanced hotspot detection through synthetic pattern generation and design of experiments. In *2018 IEEE 36th VLSI Test Symposium (VTS'18)*. IEEE, 1–6.

[32] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why should I trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. IEEE 1135–1144.

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Proceedings of the 18th International Conference on Medical Image Computing and Computer-assisted Intervention (MICCAI'15), Part III 18*. Springer, 234–241.

[34] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 618–626.

[35] Siemens. 2019. Calibre-LFD. (October 2019). https://www.mentor.com/products/ic_nanometer_design/design-for-manufacturing/calibre-lfd

[36] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034* (2013).

[37] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. 2017. Smoothgrad: Removing noise by adding noise. *arXiv preprint arXiv:1706.03825* (2017).

[38] Shuyuan Sun, Yiyang Jiang, Fan Yang, Bei Yu, and Xuan Zeng. 2022. Efficient hotspot detection via graph neural network. In *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE'22)*. IEEE, 1233–1238.

[39] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2016. Gradients of counterfactuals. *arXiv preprint arXiv:1611.02639* (2016).

[40] J. Andres Torres. 2012. ICCAD-2012 CAD contest in fuzzy pattern matching for physical verification and benchmark suite. In *Proceedings of the International Conference on Computer-aided Design*. IEEE, 349–350.

[41] Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Sirui Ding, Piotr Mardziel, and Xia Hu. 2020. Score-CAM: Score-weighted visual explanations for convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 24–25.

[42] Wan-Yu Wen, Jin-Cheng Li, Sheng-Yuan Lin, Jing-Yi Chen, and Shih-Chieh Chang. 2014. A fuzzy-matching model with grid reduction for lithography hotspot detection. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems* 33, 11 (2014), 1671–1680.

[43] Yifeng Xiao, Miaodi Su, Haoyu Yang, Jianli Chen, Jun Yu, and Bei Yu. 2021. Low-cost lithography hotspot detection with active entropy sampling and model calibration. In *2021 58th ACM/IEEE Design Automation Conference (DAC'21)*. IEEE, 907–912.

[44] Haoyu Yang, Luyang Luo, Jing Su, Chenxi Lin, and Bei Yu. 2017. Imbalance aware lithography hotspot detection: A deep learning approach. *Journal of Micro/Nanolithography, MEMS, and MOEMS* 16, 3 (2017), 033504–033504.

[45] Haoyu Yang, Piyush Pathak, Frank Gennari, Ya-Chieh Lai, and Bei Yu. 2019. Detecting multi-layer layout hotspots with adaptive squish patterns. In *Proceedings of the 24th Asia and South Pacific Design Automation Conference*. IEEE, 299–304.

[46] Haoyu Yang, Jing Su, Yi Zou, Yuzhe Ma, Bei Yu, and Evangeline F. Y. Young. 2019. Layout hotspot detection with feature tensor generation and deep biased learning. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems* 38, 6 (2019), 1175.

[47] Yen-Ting Yu, Ya-Chung Chan, Subarna Sinha, Iris Hui-Ru Jiang, and Charles Chiang. 2012. Accurate process-hotspot detection using critical design rule extraction. In *Proceedings of the 49th Annual Design Automation Conference*. IEEE, 1167–1172.

[48] Yen-Ting Yu, Geng-He Lin, Iris Hui-Ru Jiang, and Charles Chiang. 2013. Machine-learning-based hotspot detection using topological classification and critical feature extraction. In *Proceedings of the 50th Annual Design Automation Conference*. IEEE, 1–6.

[49] Timothy Zee, Geeta Gali, and Ifeoma Nwogu. 2019. Enhancing human face recognition with an interpretable neural network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*. IEEE.

[50] Hang Zhang, Bei Yu, and Evangeline F. Y. Young. 2016. Enabling online learning in lithography hotspot detection with information-theoretic feature optimization. In *2016 IEEE/ACM International Conference on Computer-aided Design (ICCAD'16)*. IEEE, 1–8.

[51] Binwu Zhu, Ran Chen, Xinyun Zhang, Fan Yang, Xuan Zeng, Bei Yu, and Martin D. F. Wong. 2021. Hotspot detection via multi-task learning and transformer encoder. In *2021 IEEE/ACM International Conference on Computer-aided Design (ICCAD'21)*. IEEE, 1–8.

Received 27 April 2024; revised 26 January 2025; accepted 21 February 2025