Boundary-Only Layout Encoding for Lithography Hotspot Detection

Yufeng Zhang[1*], Zhili Xu[1*], Yuqian Zang[1], Jiale Li[1], Silin Chen[1], Zhaoyang Wang[3], Chihwei Chiang[3†], Ningmu Zou[1,2†]
[1]School of Integrated Circuits, Nanjing University, Suzhou, China
[2]Interdisciplinary Research Center for Future Intelligent Chips (Chip-X), Nanjing University, Suzhou, China
[3]NexChip Co., Ltd., China

***Abstract*—As feature sizes continue to shrink, lithography hotspots have become a critical bottleneck constraining manufacturing yield. Most existing deep learning-based methods rasterize layout fragments into solid-filled images, introducing substantial irrelevant background and redundant internal regions. This weakens boundary/spacing cues, exacerbating the adverse effects of severe category imbalance. To address this issue, we propose a boundary-focused hotspot detection framework for both metal and via layers. Specifically, we propose a boundary-only coding method that transforms polygons into sparse, contour-dominated grids while preserving spacing structures and significantly reducing redundancy. We also further design a ResNet variant tailored to sparse inputs. Extensive experiments on the ICCAD2012 metal-layer benchmark and the more challenging ICCAD2020 via-layer benchmark demonstrate that our detector, built upon boundary-only encoding and a boundary-preserving ResNet variant, achieves competitive hotspot detection performance, validating the effectiveness of the proposed framework.**

***Index Terms*—lithography hotspot detection, EDA, deep learning, layout encoding, boundary-only design**

I. Introduction

As feature sizes continue to shrink and layout densities increase, circuit patterns become more complex, which makes it harder to keep the design masks consistent with the printed wafer. At advanced technology nodes, designers can unintentionally create geometries that are highly sensitive to lithography, causing critical defects after wafer imaging. These failure-prone layout patterns, commonly referred to as lithography hotspots, must be reliably detected and removed during the design stage to ensure correct chip function and high manufacturing yield.

To identify and reduce hotspot risks during the design phase, the industry typically employs three mainstream approaches: lithography simulation, pattern matching, and machine learning/deep learning. Lithography simulation is more commonly used in the industry, where the simulator employs analytical formulas to approximate the optical effects of the lithography system. While lithography simulation achieves high accuracy, it consumes substantial time and computational resources.

*Yufeng Zhang and Zhili Xu contributed equally to this work.
†Corresponding Authors: nzou@nju.edu.cn, benchiang@nexchip.com.cn.
This work was supported in part by the National Natural Science Foundation of China (62341408).

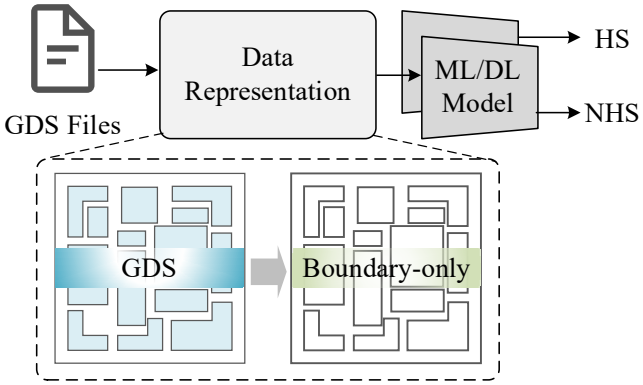

Fig. 1. Hotspot detection pipeline with boundary-only layout encoding.

Pattern matching methods, in contrast, enable fast screening by performing similarity retrieval against a predefined hotspot library. For example, prior work proposed a density-based layout encoding with PCA to distinguish hotspots from non-hotspots [1], while a tangent-space distance metric was adopted for pattern classification [2]. Overall, pattern matching is faster, but its detection capability is limited by the coverage of the hotspot library, and it generalizes poorly to unknown or novel hotspot patterns.

In contrast, machine learning approaches [3]–[7] achieve higher detection accuracy while controlling computational resources, and can handle known and unknown hotspot patterns to a certain extent. Furthermore, deep learning-based detectors [8]–[11], particularly convolutional neural networks, have demonstrated superior representational capabilities and end-to-end learning advantages in hotspot detection. Most current deep learning research rasterizes layouts into images and formulates hotspot detection as an image classification problem, which, to some extent, shortens detection time and improves overall usability. However, traditional "solid-fill" rasterization methods often introduce substantial internal area information weakly correlated with lithography failure mechanisms. This leads to feature redundancy and computational resource waste while diluting high-value clues such as boundaries and critical spacings, ultimately causing excessive false alarms and other issues.

To further reduce computational costs and detection time

without compromising detection capabilities while minimizing false alarms, this paper proposes a boundary-focused lithography hotspot detection method. Specifically, we introduce a boundary-only layout encoding that transforms solid polygons into sparse, contour-dominated representations. Furthermore, we combine a ResNet variant tailored for sparse inputs with a perceptually imbalanced training strategy. This approach steadily improves accuracy while reducing overall computational overhead, significantly decreasing false alarms.

Fig. 1 provides an overview of our boundary-only preprocessing and the resulting hotspot detection pipeline. Starting from GDSII/OASIS layout files, we convert each clip into a 512×512 boundary-only raster grid that retains polygon contours while leaving interiors empty. This contour-dominant representation highlights boundary geometry and spacing patterns that are most relevant to lithography hotspots, and serves as the input to a supervised detector for hotspot/non-hotspot classification. The main contributions of this work are summarized as follows:

- We propose a boundary-focused lithography hotspot detection framework applicable to both metal layers and via layers. This framework aims to emphasize boundary geometries and critical spacing clues most relevant to hotspot formation while maintaining compatibility with standard layout formats and practical grid-based processes.
- We further design a novel boundary-only preprocessing strategy that removes polygon interior fillings and preserves only boundaries and spacing structures, thereby emphasizing geometric cues that are highly correlated with hotspot formation mechanisms.
- The proposed method is evaluated on the ICCAD2012 metal benchmark and the more challenging ICCAD2020 via benchmark. It consistently improves detection accuracy while reducing false alarms and runtime overhead, indicating good generalization across both layer types.

II. BACKGROUND

A. Problem Formulation

In this section, we present preliminary knowledge about hotspot detection and then review the background of ResNet.

Definition 1 (Accuracy). The ratio of the hotspot clips that are correctly detected to the whole number of real hotspot clips [12].

$$Accuracy = \frac{TP}{TP + FN} \tag{1}$$

TP denotes the number of clips that are truly positive predicted. FN denotes the number of actual hotspot clips missed by the detector.

Definition 2 (False Alarm). The number of non-hotspot clips that are incorrectly judged as hotspots [12].

$$FalseAlarm = FP \tag{2}$$

Problem 1 (Hotspot Detection). Given a collection of hotspot clips and non-hotspot clips, the objective of hotspot detection is to train a detector that maximizes accuracy and minimizes false alarms over all layout clips.

B. ResNet

ResNets [16] address the degradation issue in deep neural networks by introducing identity mapping connections. Unlike traditional stacked networks designed to learn latent mappings $H(x)$, ResNets explicitly instruct stacked layers to fit residual mappings.

Formally, consider a building block with input x. Its output y is defined as:

$$y = \sigma\left(\mathcal{F}(x, \{W_i\}) + x\right), \tag{3}$$

where $\mathcal{F}(x, \{W_i\})$ represents the residual mapping to be learned (e.g., a stack of two or three convolutions), and σ denotes the activation function. The operation $x + \mathcal{F}(x)$ is implemented via shortcut connections and element-wise addition.

This formulation enables direct gradient propagation through an identity path, effectively stabilizing training for extremely deep architectures. For lithography hotspot detection, this property is crucial as it allows the network to learn hierarchical representations from low-level boundaries to high-level geometric conflict patterns without gradient vanishing. Thus, the residual mechanism described in Eq. (3) provides a foundation for extracting discriminative features from our proposed boundary-only layout encoding.

III. PROPOSED FRAMEWORK

In this section, we present the overall architecture of the proposed boundary-focused lithography hotspot detection framework. First, we introduce the boundary-only layout encoding strategy, which transforms solid polygons into sparse, contour-dominant representations to effectively reduce redundancy and highlight critical spacing cues concerning traditional solid-filled rasterization. Secondly, the architecture of the customized ResNet backbone is detailed, featuring specific boundary-preserving adjustments to better accommodate sparse inputs. Finally, the imbalance-aware training protocol is elaborated, where class-balanced sampling and Focal Loss are incorporated to ensure stable optimization against severe data imbalance.

A. Boundary-Only Preprocessing

Standard layout rasterization methods typically convert polygons into solid-filled density maps. This is not ideal for hotspot detection because interior pixels provide little useful signal, while failures such as pinching and bridging mainly depend on boundary shapes and small gaps. Solid filling also adds many redundant interior pixels, which can hide the important boundary cues and make training harder under severe class imbalance.

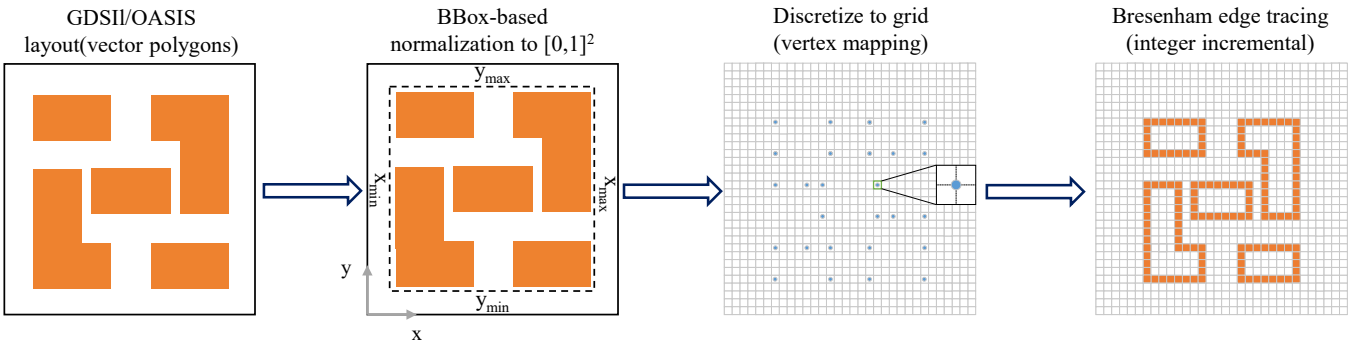

Fig. 2. Boundary-only via normalization, discretization, and boundary tracing.

a) Boundary-only encoding: We take a layout clip and create a fixed sparse grid that only shows the boundaries step by step. First, we adjust the clip to fit a square with sides of length 1, then we map the corner points of the polygon shapes onto a 512-by-512 grid, and finally, we draw just the outer boundaries of these polygons without filling in their inner areas.

Let $\mathcal{P}$ denote all polygons in the clip, and let the clip bounding box be $\mathbf{b} = (x_{\min}, x_{\max}, y_{\min}, y_{\max})$. Each vertex (x, y) is normalized to $[0, 1]^2$ by

$$\tilde{x} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}, \qquad \tilde{y} = \frac{y - y_{\min}}{y_{\max} - y_{\min}}. \tag{4}$$

We then map $(\tilde{x}, \tilde{y})$ to integer grid coordinates on a fixed-resolution canvas of size $G \times G$ via

$$u = \lfloor \tilde{x}\,(G-1) \rfloor, \qquad v = \lfloor \tilde{y}\,(G-1) \rfloor, \tag{5}$$

where $(u, v) \in \{0, 1, \ldots, G-1\}^2$.

Instead of filling polygon interiors, we rasterize only polygon boundaries. We initialize a binary grid $I_{\text{boundary-only}} \in \{0,1\}^{G \times G}$ with zeros, and for each polygon we trace every consecutive vertex pair (including the last-to-first connection) as a connected one-pixel-wide chain, yielding a sparse contour-dominant encoding. Fig. 2 summarizes the boundary-only preprocessing pipeline that converts GDSII/OASIS polygons into sparse contour-dominant grids.

Algorithm 1 Boundary-only contour tracing (Bresenham)

Require: Two consecutive vertices (u_0, v_0) and (u_1, v_1) on a $G \times G$ grid
Ensure: Mark boundary pixels on $I_{\text{boundary-only}}$

$\Delta u \leftarrow |u_1 - u_0|,\ \Delta v \leftarrow |v_1 - v_0|$
$s_u \leftarrow \text{sign}(u_1 - u_0),\ s_v \leftarrow \text{sign}(v_1 - v_0)$
$e \leftarrow \Delta u - \Delta v$
$(u, v) \leftarrow (u_0, v_0)$
while $(u, v) \neq (u_1, v_1)$ **do**
  $I_{\text{boundary-only}}(u, v) \leftarrow 1$
  $e_2 \leftarrow 2e$
  if $e_2 > -\Delta v$ **then**
    $e \leftarrow e - \Delta v$
    $u \leftarrow u + s_u$
  end if
  if $e_2 < \Delta u$ **then**
    $e \leftarrow e + \Delta u$
    $v \leftarrow v + s_v$
  end if
end while
$I_{\text{boundary-only}}(u_1, v_1) \leftarrow 1$

B. Boundary-only contour tracing

This procedure implements a symmetric Bresenham discretization that rasterizes each continuous line segment by incrementally selecting, via integer-only operations (addition, subtraction, and comparison), the grid point whose pixel center is closest to the underlying segment. Consequently, every polygon boundary is mapped to a connected, one-pixel-wide chain that preserves boundary continuity and local topology while avoiding any interior fill. Aggregating all boundaries produces a sparse boundary-only grid $I_{\text{boundary-only}}$, in which interior pixels remain zero and spacing/gap information is implicitly captured by the empty lattice sites between neighboring boundaries. Importantly, the discretized boundary is not formed by heuristically connecting nearby grid-adjacent pixels. Instead, the next pixel is uniquely determined at each step to minimize geometric deviation from the continuous segment. As a result, other lattice connections that may appear plausible but correspond to different continuous paths are deliberately excluded, preventing spurious boundary branches and maintaining geometric fidelity.

C. Backbone Architecture: Boundary-Preserving ResNet Variant

To evaluate the proposed boundary-only representation under a CNN-based baseline, we build a boundary-preserving ResNet variant that follows the ResNet-50 bottleneck template, while modifying the early stem and the final stage to better fit sparse, contour-dominant inputs. Given the boundary-only grid $I_{\text{boundary-only}} \in \{0,1\}^{512 \times 512}$, the network extracts hierarchical features through stacked residual stages and aggregates these features into a global feature vector with one channel dimension via global average pooling, as summarized in Fig. 3.

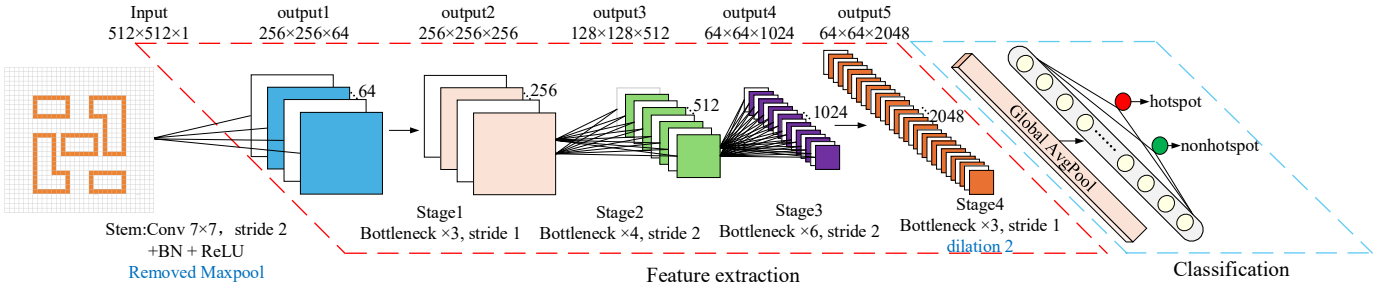

Fig. 3. End-to-end hotspot detector with boundary-only layout encoding and boundary-preserving ResNet backbone

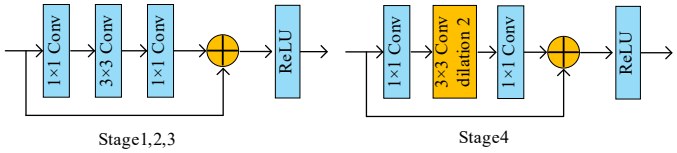

Fig. 4. Standard and modified ResNet bottleneck blocks: Stage1–3 use a conventional 3×3 convolution, while Stage4 adopts a dilated 3×3 convolution (dilation= 2) to enlarge the receptive field without downsampling.

The bottleneck template and our Stage-4 modification are illustrated in Fig. 4.

Compared with the standard ResNet-50 configuration, we introduce two lightweight adjustments: (i) *stem without max-pooling*, where the early max-pooling operation is removed to avoid prematurely suppressing thin, one-pixel-wide contours; and (ii) *a high-resolution final stage with dilation*, where we eliminate stride-based spatial downsampling in the last residual stage and use dilated 3×3 convolutions with dilation rate $r = 2$ to enlarge the receptive field while preserving a higher-resolution feature map for fine boundary and spacing cues. Formally, a 2D dilated convolution with dilation rate r is defined as

$$y(m,n) = \sum_{i=-K}^{K} \sum_{j=-K}^{K} w(i,j), x\left(m + r, i, ; n + r, j\right), \quad (6)$$

where x and y denote the input and output feature maps, w is a $(2K{+}1) \times (2K{+}1)$ convolution kernel, and $r \in \mathbb{N}^{+}$ is the dilation factor. Compared with standard convolution ($r{=}1$), dilation expands the effective receptive field without introducing additional stride-based downsampling; for a $k \times k$ kernel, the effective kernel size becomes $k_{\text{eff}} = k + (k-1)(r-1)$.

These adjustments are practical implementation refinements rather than new architectural designs. They retain the standard residual learning paradigm, and the resulting feature representation is fed to a linear classifier to produce class logits. The logits are converted to the posterior hotspot probability via a softmax function. At inference time, a sample is predicted as a hotspot when this probability exceeds a decision threshold τ.

D. Imbalance-Aware Training

Lithography hotspot detection is characterized by severe class imbalance, where hotspot samples constitute only a small fraction of the dataset. To make training stable and avoid the model being biased toward the majority class, we use class-imbalance techniques that are widely adopted in practice.

Let $\mathcal{D}_{\text{pos}}$ and $\mathcal{D}_{\text{neg}}$ denote the sets of hotspot and non-hotspot samples, respectively. During training, we draw samples using a class-balanced sampling distribution:

$$P(x) = \begin{cases} \frac{1}{2\,|\mathcal{D}_{\text{pos}}|}, & x \in \mathcal{D}_{\text{pos}}, \\ \frac{1}{2\,|\mathcal{D}_{\text{neg}}|}, & x \in \mathcal{D}_{\text{neg}}. \end{cases} \quad (7)$$

This choice makes the expected class composition within each mini-batch approximately balanced, while preserving randomness within each class. In practice, this corresponds to oversampling the hotspot class and subsampling the non-hotspot class, so that gradient updates are not dominated by abundant easy negatives.

E. Loss Function

Due to the extreme imbalance between hotspot and non-hotspot samples, optimization with the standard cross-entropy loss can be dominated by easy negative examples. To mitigate this issue, we employ Focal Loss [17] as the classification objective:

$$\mathcal{L}_{\text{FL}}(p_t) = -\alpha(1-p_t)^{\gamma} \log(p_t), \quad (8)$$

where p_t denotes the predicted probability assigned to the ground-truth class, and α and γ control class re-weighting and hard-example emphasis, respectively. By down-weighting well-classified samples and emphasizing harder ones, Focal Loss encourages the model to focus on the ambiguous boundaries and spacing patterns that are more likely to correspond to lithography hotspots.

IV. EXPERIMENTS

Our framework is implemented in PyTorch, and all experiments are conducted on a Linux workstation equipped with an NVIDIA GeForce RTX 4080 SUPER GPU.

A. Datasets and Benchmarks

To evaluate generalization, we conduct experiments on two benchmarks: the ICCAD2012 benchmark [13] and the ICCAD2020 benchmark [14].

The statistics of the benchmarks are summarized in Table I. "#HS" denotes the total number of hotspots, and "#NHS"

TABLE I
BENCHMARK STATISTICS

Benchmarks	Training Set		Testing Set		Size/Clip (μm^2)
	#HS	#NHS	#HS	#NHS	
ICCAD2012	1204	17096	2524	13503	3.6 × 3.6
Via-1	3430	10290	2267	6878	2.0 × 2.0
Via-2	1029	11319	724	7489	2.0 × 2.0
Via-3	614	19034	432	12614	2.0 × 2.0
Via-4	39	23010	26	15313	2.0 × 2.0
Via-Merge	5112	63653	3449	42294	2.0 × 2.0

denotes the total number of non-hotspots. Note that the ICCAD2020 benchmark contains clips of the layout of vias. It is composed of four small datasets (Via-1 to Via-4) and a big merged one (Via-Merge). Consistent with our analysis in Section III-D, the raw datasets exhibit extreme class imbalance. All layout clips are rasterized into 512×512 sparse grids using our boundary-only strategy before feeding into the detector.

B. Results Comparison

We evaluated the performance of our proposed framework across all benchmark tests, highlighting its effectiveness in identifying defect patterns across various hotspot detection scenarios. To evaluate our framework's performance, we compare it with representative prior methods:

- TCAD'19 [9]: A CNN based on feature tensor generation and deep biased learning using DCT-based frequency features.
- DAC'19 [10]: A binarized residual neural network that accelerates detection by using binary layout images.
- ICCAD'20 [8]: An attention-based CNN that jointly embeds layout features and performs hotspot classification.
- DATE'22 [11]: A GNN-based method using graph feature embedding for hotspot detection.
- ISEDA'25 [15]: A ViT-based method with multi-scale feature modeling.

TABLE II
PERFORMANCE ON THE ICCAD2012 BENCHMARK

Method	Acc(%)	FA	Time(s)
DAC'19 [10]	98.54	3260	561.28
TCAD'19 [9]	98.40	3535	502.70
ICCAD'20 [8]	98.42	2481	143.79
DATE'22 [11]	98.42	1731	3.20
ISEDA'25 [15]	98.48	–	–
Ours	98.73	1500	2.78

The comparison results are reported in Table II and Table III. Table II and Table III summarize the performance of the proposed method against representative methods on the ICCAD2012 and ICCAD2020 benchmarks, respectively. However, it should be noted that ISEDA'25 was evaluated only on the ICCAD2012 benchmark and reported only partial results for the ICCAD2012 test set. Therefore, we took the arithmetic mean of its test results as the overall test performance.

Compared with all methods except DATE'22, our approach achieves the best performance across all metrics, including Acc, FA, and Time, while reducing the runtime by tens of times, thereby substantially lowering computational resource consumption. In addition, on average, our method outperforms DATE'22 in detection accuracy, while significantly reducing false alarms and achieving comparable runtime. Our hotspot detection framework takes the proposed boundary-only layout encoding as input. Its main advantages include low computational overhead, fast inference, and the ability to achieve excellent accuracy with very low false alarms. Compared with existing mainstream solutions, our method not only steadily improves accuracy but also substantially reduces false alarms and runtime. The detailed comparison results are presented below.

On the ICCAD2012 benchmark, as shown in Table II, our method is more than $20\times$ faster than DAC'19, TCAD'19, and ICCAD'20. And it also runs 13% faster than DATE'22. In addition, it achieves the highest accuracy of 98.73% while reducing the number of false alarms to 1500. On the ICCAD2020 benchmark, as shown in Table III, our method attains higher detection accuracy and fewer false alarms on nearly all datasets. Notably, DATE'22 is only slightly better than our method on Via-2 in terms of both accuracy and false alarms. However, it is inferior to our method in terms of average accuracy and average false alarm. Overall, our approach delivers the best performance across all test sets, indicating that the proposed method exhibits strong generalization capability. In summary, we present a hotspot detection framework that enables fast and accurate detection on both the ICCAD2012 and ICCAD2020 benchmarks.

C. Ablation Study

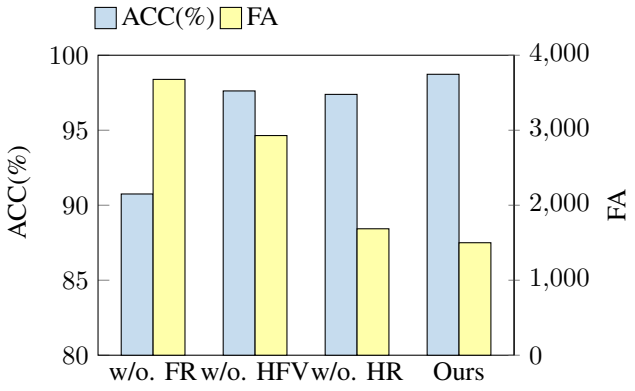

Fig. 5. Ablation studies on ICCAD2012 for different techniques' application.

We conduct an ablation study to quantify the contributions of our key design choices (boundary-only preprocessing and Focal Loss) as well as the impact of different backbone architectures. The results are summarized in Fig. 5. Specifically, "w/o. FR," "w/o. HFV," and "w/o. HR" denote the variants that, relative to the full configuration, use a conventional-filled

TABLE III
PERFORMANCE ON THE ICCAD2020 BENCHMARK. THE BASELINE RESULTS ARE TAKEN FROM [11]

Benchmark	DAC'19 [10]			TCAD'19 [9]			ICCAD'20 [8]			DATE'22 [11]			Ours		
	Acc(%)	FA	Time(s)	Acc(%)	FA	Time(s)	Acc(%)	FA	Time(s)	Acc(%)	FA	Time(s)	Acc(%)	FA	Time(s)
Via-1	89.85	1886	57.76	71.50	773	43.36	93.42	1589	19.83	95.41	1722	1.93	95.59	1696	1.62
Via-2	73.00	1222	21.66	65.06	1290	40.02	86.32	1100	13.22	90.33	1507	1.31	89.23	1536	1.94
Via-3	73.38	3406	43.15	48.15	760	60.23	88.20	2105	20.69	89.81	1928	1.63	90.28	1844	2.37
Via-4	73.08	15288	51.98	76.92	155	67.44	80.77	152	20.70	84.62	400	1.82	84.62	224	2.53
Via-Merge	90.42	9295	105.30	88.01	7633	165.85	92.20	6453	59.74	93.33	5502	4.76	94.09	5449	4.20

preprocessing strategy, adopt a ViT backbone, and disable Focal Loss while keeping other settings unchanged, respectively. Compared with the full configuration, removing the boundary-only preprocessing leads to a pronounced drop in detection accuracy and a substantial increase in false alarms, corroborating our hypothesis that boundary-only encoding amplifies boundary and spacing-related cues that are most predictive of lithography risk. When the boundary-only preprocessing is retained, disabling Focal Loss still consistently degrades performance, yielding lower accuracy and higher false alarms, which indicates that Focal Loss effectively mitigates class imbalance and suppresses spurious hotspot predictions. Overall, under this benchmark and our current implementation settings, combining boundary-only preprocessing, Focal Loss, and a customized ResNet backbone achieves the best trade-off among accuracy, false alarm, and inference efficiency. We further observe that while the ViT backbone achieves comparable hotspot detection performance, it incurs significantly higher inference latency under identical settings and falls short of our approach in terms of accuracy or false alarms.

V. CONCLUSION

To our knowledge, we are the first to propose a boundary-only layout encoding method and a boundary-centric hotspot detection framework applicable to both metal layers and via layers. We convert GDSII/OASIS layouts into boundary-only raster inputs that preserve polygon contours while leaving interiors empty. Unlike conventional solid-filled rasterization methods, which largely duplicate interior regions with little value for hotspot formation, our encoding highlights boundaries and the gaps between them, where lithography-sensitive spacing interactions concentrate. With a lightweight ResNet variant, our method performs well in experiments and shows that boundary-only encoding is a simple and effective choice for lithography hotspot detection.

REFERENCES

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

[2] F. Yang, Z. Li, Y. Lin, B. Yu, and X. Zeng, "Improved tangent space-based distance metric for lithographic hotspot classification," *IEEE TCAD*, vol. 36, no. 9, pp. 1545–1556, 2017.

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

[4] T. Matsunawa, J.-R. Gao, B. Yu, and D. Z. Pan, "A new lithography hotspot detection framework based on AdaBoost classifier and simplified feature extraction," in *Proc. SPIE*, vol. 9427, pp. 201–211, 2015.

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

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

[7] Z. He, Z. Feng, L. He, E. F. Y. Young, and B. Yu, "Bilinear lithography hotspot detection," in *Proc. ISPD*, pp. 7–14, 2017.

[8] H. Geng, H. Yang, L. Zhang, J. Miao, F. Yang, X. Zeng, and B. Yu, "Hotspot detection via attention-based deep layout metric learning," in *Proc. ICCAD*, pp. 1–8, 2020.

[9] H. Yang, J. Su, Y. Zou, Y. Ma, B. Yu, and E. F. Y. Young, "Layout hotspot detection with feature tensor generation and deep biased learning," *IEEE TCAD*, vol. 38, no. 6, pp. 1175–1187, 2019.

[10] Y. Jiang, F. Yang, H. Zhu, B. Yu, D. Zhou, and X. Zeng, "Efficient layout hotspot detection via binarized residual neural network," in *Proc. DAC*, pp. 1–6, 2019.

[11] S. Sun, Y. Jiang, F. Yang, B. Yu, and X. Zeng, "Efficient hotspot detection via graph neural network," in *Proc. DATE*, pp. 1233–1238, 2022.

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

[13] CAD Contest at ICCAD, "ICCAD 2012 CAD contest: Fuzzy pattern matching for physical verification and benchmark suite," 2012. [Online]. Available: https://www.iccad-contest.org/2012/

[14] CAD Contest at ICCAD, "2020 CAD contest @ ICCAD," 2020. [Online]. Available: https://www.iccad-contest.org/2020/

[15] Z. Huang, Z. Li, M. Zhu, and P. Chen, "MF-ViT: lithography hotspot detection based on multi-scale feature and vision transformer," in *Proc. ISEDA*, 2025.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, pp. 770–778, 2016.

[17] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. CVPR*, pp. 2980–2988, 2017.