Delving into Topology Representation for Layout Pattern: A Novel Contrastive Learning Framework for Hotspot Detection

Silin Chen[1], Kangjian Di[1], Guohao Wang[4], Wenzheng Zhao[4], Li Du[2], Ningmu Zou[1,3]*,
[1]School of Integrated Circuits, Nanjing University, Suzhou, China
[2]School of Electronic Science and Engineering, Nanjing University, Nanjing, China
[3]Interdisciplinary Research Center for Future Intelligent Chips (Chip-X), Nanjing University, Suzhou, China
[4]ZetaTech Co.,Ltd.

***Abstract*—Recently, machine learning-based techniques have been applied for layout hotspot detection. However, existing methods encounter challenges in capturing the decision boundary across the entire dataset and ignore the geometric properties and topology of the polygons. In this paper, we introduce CLI-HD, a novel contrastive learning framework on layout sequences and images for hotspot detection. Our framework improves the ability to distinguish between hotspots and non-hotspots by similarity computations instead of a single decision boundary. To effectively incorporate geometric information into the model training process, we propose Layout2Seq, which encodes polygon shapes as vectors within sequences that are subsequently fed into the CLI-HD. Furthermore, to better represent topology information, we develop an absolute position embedding, replacing the standard position encoders used in Transformer architectures. Extensive evaluations on various benchmarks demonstrate that CLI-HD outperforms current state-of-the-art methods, with an accuracy improvement ranging from 0.82% to 4.77% and a reduction in false alarm rates by 4.9% to 23.18%.**

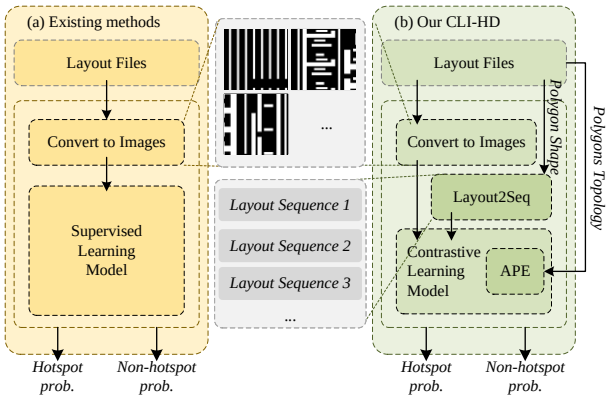

Fig. 1. Illustration of the hotspot detection flow. (a) The pipeline of existing supervised learning-based methods. (b) The pipeline of our proposed CLI-HD.

I. Introduction

With the rapid development of semiconductor techniques, the shrinking of transistor feature sizes and the increasing size of dies present significant challenges to Design for Manufacturability (DFM). Hotspot detection is introduced as a critical procedure in DFM to identify potential defect patterns early. Lithography simulation techniques can accurately detect hotspot patterns, but they consume a large amount of time for simulation and are not applicable to chip fabrication. Other works were based on pattern-matching methods to identify hotspots with similar patterns by retrieval or clustering. Such methods cannot cover hotspots outside the pattern or layout libraries [1], [2].

Recently, machine learning-based methods have been applied in DFM [3], [4] and achieved impressive progress in hotspot detection [5], [6]. As shown in Fig. 1(a), existing methods convert the layout files into binary images and employ supervised learning to classify HotSpot (HS) and Non-HotSpot (NHS). However, these methods utilizing convolutional neural networks (CNNs) to distinguish hotspots are limited by the receptive field in acquiring the global topology information of the layout images. To represent the global topology, others utilize polygons to construct a layout graph and perform pattern graph classification with graph neural networks (GNNs) for hotspot detection [7], [8]. Nevertheless, they decompose the polygon into multiple rectangular nodes, which introduces additional complexity and results in the shape information loss of the polygon. Generally, existing CNN-based and GNN-based methods rely on supervised learning, which tends to suffer from a high false-alarm rate when trained to construct a single decision boundary across the entire dataset with very few hotspot samples [6].

Beyond image and graph-based approaches, alternative layout representations have been explored in other DFM applications. A sequence pattern representation [9] is proposed for layout pattern generation that attempts to learn the shape and spatial information of patterns by the Transformer. However, there is a representation gap when converting coordinates and edge values to language token embedding. In addition, it ignores the position of polygonal patterns. Other methods represent a layout clip by using a topological matrix and two vectors [3], [4]. This representation is not flexible for existing deep neural networks [9]. To the best of our knowledge, the

*Corresponding Author: nzou@nju.edu.cn.
This work was supported by the National Natural Science Foundation of China (62341408).

shape (e.g., height, width, orientation) of individual polygons, along with the topological relationships (e.g., edge2edge, side2side) among multiple polygons, are critical factors for identifying hotspots [4]. Thus, the aforementioned supervised learning methods and layout representation struggle to construct robust statistical models for hotspot detection.

Considering the existing issues, we design a novel contrastive learning framework on layout sequences and images for hotpot detection, CLI-HD, which effectively distinguishes between hotspots and non-hotspots through similarity computations while incorporating geometric representations and topological information. The main contributions of this paper are summarized as follows:

- We propose a novel contrastive learning framework for hotspot detection, that leverages similarity computations, thereby eliminating the need to establish a unified decision boundary across the entire dataset.
- We present Layout2Seq, a novel sequence representation for layouts that efficiently converts polygons into vector formats, facilitating the extraction of geometric features. This approach can also provide the possibility for other machine learning-based DFM applications.
- We develop the Absolute Position Embedding (APE) method for Transformer-based sequence models, specifically designed to efficiently encode the topology relationships of multiple polygons based on their position context within the layout.
- Experimental results demonstrate that CLI-HD significantly improves the discrimination of hotspots through contrastive methods, achieving an accuracy improvement of 0.82% to 4.77% and a reduction in false alarm rates ranging from 4.9% to 23.18%. Furthermore, CLI-HD can be effectively fine-tuned with a single linear layer, yielding remarkable performance in hotspot detection.

II. Preliminaries

The lithographic process utilizes a mask to transfer a precisely designed layout pattern onto the wafer. Sensitive patterns are particularly susceptible to yield reduction caused by variations in the manufacturing process. These patterns are defined as hotspots within the layout [5]. In our paper, hotspot detection is considered as a binary classification task for the layout clips. Different from [6], [8], [10], we are dealing with layout clips where the hotspot locations are random and may include multiple hotspots, rather than just a single centered marker. It will test the model's ability to learn non-fixed hotspot patterns making it more relevant to real-world applications relevant to real-world application scenarios. The following definitions and metrics are used to evaluate the performance of a hotspot detector.

Definition 1 (Accuracy). The ratio of correctly identified hotspots to the total number of ground truth hotspots.

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

To match the evaluation metrics above, we formulate the hotspot detection problem as follows.

Problem 1 (Hotspot Detection). Given a collection of layout clips containing hotspot and non-hotspot patterns, our goal is to learn a feature vector from building image representations and sequence representations. Based on this vector we can use contrastive learning to distinguish hotspots or fine-tune a classifier for detecting hotspots.

III. Methodology

A. Overview

As shown in Fig. 1(b), we initially obtain the image representation of the layout, consistent with previous work. To construct a geometric representation of polygons, Layout2Seq is proposed to transform the shape of each polygon into a vector and combine multiple polygon vectors into a sequence corresponding to a layout clip. Leveraging images and layout sequences, a contrastive learning framework is designed to generate a unified representation aligning visual features with the geometric properties of polygons. In the contrastive network, we discard the wrong position encoder in the Transformer and develop the APE for introducing the centroid location of polygons.

B. Contrastive learning framework in CLI-HD

As illustrated in Fig. 2, our proposed framework consists mainly of two key components, specifically image encoder $\mathcal{I}(\cdot)$ and sequence encoder $\mathcal{S}(\cdot)$. Motivated by the [11], our objective is to jointly train $\mathcal{I}(\cdot)$ and $\mathcal{S}(\cdot)$ using pairs of binary layout images and corresponding sequence representations, ensuring that the learned image embedding $\mathbf{F}^I = [f_1^I, f_2^I ..., f_N^I] \in \mathbb{R}^{N \times D}$ are effectively aligned with the sequence embedding $\mathbf{F}^S \in \mathbb{R}^{N \times D}$, where N denotes the batch size and D represents the embedding dimension. Basically, a ViT-based model [12] is utilized to encode binary layout images, generating their corresponding image latent representations. In our proposed, the sequence encoder $\mathcal{S}(\cdot)$ is implemented using a transformer architecture, which generates the sequence embedding $\mathbf{F}^S$ from the output of Layo ut2Seq. Specifically, for the given image-sequence pairs $(\mathbf{X}_i^I, \mathbf{X}_i^S)$, the embedding f_i^I and f_i^S can be computed as follows:

$$f_i^I, f_i^S = \mathrm{Proj}(\mathcal{I}(\mathbf{X}_i^I)), \mathrm{Proj}(\mathcal{S}(\mathbf{X}_i^S)) \tag{1}$$

where $\mathrm{Proj}(\cdot)$ denotes the projection layer. During training process, we compute the loss function of image-to-text $L_{I \to S}$ as:

$$L_{I \to S} = -\frac{1}{N} \sum_{i=1}^{N} \log \frac{\exp(\mathrm{sim}(f_i^I, f_i^S)/\tau)}{\sum_{j=1}^{N} \exp(\mathrm{sim}(f_i^I, f_j^S)/\tau)} \tag{2}$$

and the text-to-image contrastive loss $L_{S \to I}$:

$$L_{S \to I} = -\frac{1}{N} \sum_{i=1}^{N} \log \frac{\exp(\mathrm{sim}(f_i^S, f_i^I)/\tau)}{\sum_{j=1}^{N} \exp(\mathrm{sim}(f_i^S, f_j^I)/\tau)} \tag{3}$$

where the τ is the temperature parameter and $\mathrm{sim}(\cdot)$ denotes the similary function which is calculated as:

$$\mathrm{sim}(f_i^I, f_i^S) = \frac{f_i^I \cdot \left(f_i^S\right)^{\mathrm{T}}}{\|f_i^I\| \cdot \|f_i^S\|} \tag{4}$$

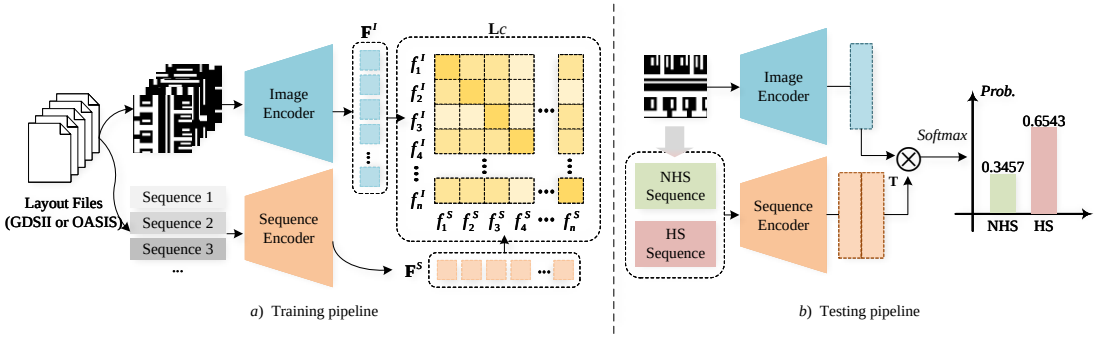

Fig. 2. The framework of contrastive learning. a) The training process of the constrastive learning for the hotspot detection. b) The simple testing process of our proposed framework.

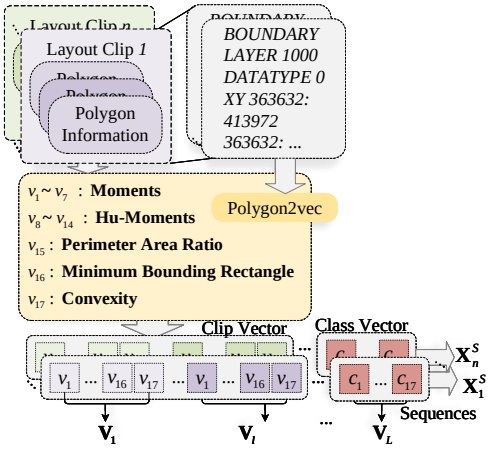

Fig. 3. Illustration of Layout2Seq.

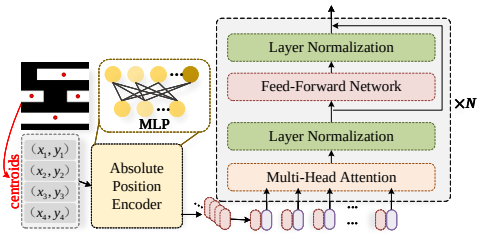

Fig. 4. Illustration of absolute position embedding in sequence encoder.

The total contrastive loss L_{con} in Fig 2(a) is calculated as:

$$L_c = \frac{L_{I\to S} + L_{S\to I}}{2} \tag{5}$$

This loss function is designed to optimize the model by increasing the similarity between matched pairs of representations, (f_i^I, f_i^S), while simultaneously decreasing the similarity between mismatched pairs, (f_i^I, f_j^S), for $i \neq j$.

As shown in Fig 2(b), given a layout image $\mathbf{X}^I$, the Layout2Seq is employed to extract both the hotspot sequence $\mathbf{X}_{hs}^S$ and the non-hotspot sequence $\mathbf{X}_{nhs}^S$ during the inference phase. These sequences, along with the image itself, are subsequently encoded via separate image and sequence encoders to generate their corresponding latent embedding, denoted as $(f^I, (f_{hs}^S, f_{nhs}^S))$. The similarity scores between the image embedding and each sequence embedding are then computed to facilitate category assignment.

Algorithm 1 Layout2Seq

Require: a set of polygons for the layout clip $\{P^k\}_{k=0}^{l}$, the polygon number of the layout clip l, the sequence context length L, the label of the layout $\hat{Y}$.
Ensure: the sequence for the i-th layout clip $\mathbf{X}_i^S$.

Initialize empty sequences $\mathbf{X}_i^S \leftarrow ()$.
for $k = 1$ to l **do**
  Initialize empty sequences $\mathbf{V}_i^k \leftarrow ()$;
  Rendering a binary image I^k by polygon P^k;
  Compute $\tilde{\mathbf{M}}_{pq}^k, \mathbf{Hu}_i^k$ by Equation(6)(7)(8)(9) and I^k;
  $\{v_1 \sim v_7\} = \{\tilde{\mathbf{M}}_{20}^k, \tilde{\mathbf{M}}_{11}^k, \tilde{\mathbf{M}}_{02}^k, \tilde{\mathbf{M}}_{30}^k, \tilde{\mathbf{M}}_{21}^k, \tilde{\mathbf{M}}_{12}^k, \tilde{\mathbf{M}}_{03}^k\}$;
  $\mathbf{V}_i^k \leftarrow \{v_1 \sim v_7\}$;
  $\{v_8 \sim v_{14}\} = \{\mathbf{Hu}_1, \mathbf{Hu}_2, \mathbf{Hu}_3, \mathbf{Hu}_4, \mathbf{Hu}_5, \mathbf{Hu}_6, \mathbf{Hu}_7\}$;
  $\mathbf{V}_i^k \leftarrow \{v_8 \sim v_{14}\}$;
  $a = \text{Area}(P^k), p = \text{Perimeter}(P^k)$;
  $w_{bbox}, h_{bbox} = \text{getBoundingRect}(P^k)$;
  $v_{15} = p/a, v_{16} = a/(w_{bbox} \times h_{bbox})$;
  $v_{17} = \text{isContourConvex}(P^k)$;
  $\mathbf{V}_i^k \leftarrow \{v_{15}, v_{16}, v_{17}\}$;
  $\mathbf{X}_i^S \leftarrow \mathbf{V}_i^k$;
end for
$l_{cur} = len(\mathbf{X}_i^S)$;
for $l = l_{cur}$ to L **do**
  if $\hat{Y} = HS$ **then**
    $\mathbf{X}_i^S \leftarrow \{1\}^{17}$;
  else if $\hat{Y} = NHS$ **then**
    $\mathbf{X}_i^S \leftarrow \{-1\}^{17}$;
  end if
end for
return $\mathbf{X}_i^S$.

C. Layout to Sequence

As shown in prior work [5], [6], converting layout files into their corresponding binary images is a straightforward task. However, to effectively capture geometric information from polygon, we propose a novel approach, called Layout2Seq.

This method transforms layout files into sequences, thereby enhancing the extraction and analysis of geometric features.

As shown in Fig. 3, we extract polygon coordinates from the original layout files and systematically parse each polygon using a vectorized polygon method. Image moments are extensively employed in computer vision and image processing to characterize object shapes [13]. Therefore, for each polygon P^k, where k is the index of the polygon, we generate a binary image I^k using the polygon's coordinates. Subsequently, the geometric moment $\mathbf{M}^k$ of the I^k is computed by:

$$\mathbf{M}^k_{pq} = \sum_{x=0}^{W}\sum_{y=0}^{H} x^p \cdot y^q \cdot I^k(x,y) \tag{6}$$

where W and H represent the width and height of I^k, p and q denote the order of the moment. We further compute the central moments $\mathbf{C^k}$ as:

$$\mathbf{C}^k_{pq} = \sum_{x=0}^{W}\sum_{y=0}^{H} (x - \frac{\mathbf{M}^k_{10}}{\mathbf{M}^k_{00}})^p \cdot (y - \frac{\mathbf{M}^k_{01}}{\mathbf{M}^k_{00}})^q \cdot I^k(x,y) \tag{7}$$

where $\mathbf{M}_{00}$, $\mathbf{M}_{10}$ and $\mathbf{M}_{01}$ can be used to compute the x-coordinate and y-coordinate of the centroid. For scale-invariant moments which are useful for comparing shapes regardless of their size, we compute the normalized moments $\tilde{\mathbf{M}}^k$ as follows:

$$\tilde{\mathbf{M}}^k_{pq} = \frac{\mathbf{C}^k_{pq}}{(\mathbf{M}^k_{00})^{(p+q)/2+1}} \tag{8}$$

The $\{\tilde{\mathbf{M}}^k_{20}, \tilde{\mathbf{M}}^k_{11}, \tilde{\mathbf{M}}^k_{02}, \tilde{\mathbf{M}}^k_{30}, \tilde{\mathbf{M}}^k_{21}, \tilde{\mathbf{M}}^k_{12}, \tilde{\mathbf{M}}^k_{03}\}$ are incorporated into the polygon vector $\mathbf{V}^k$ as components $\{v_1 \sim v_7\}$.

It is well known that polygons in hotspot patterns exhibit geometric invariants, such as rotational invariance and other transformation properties. In our work with Layout2Seq, we utilize Hu's seven moments [13] to encapsulate the shape invariants of polygons, thereby enhancing robustness against geometric transformations. The Hu's moments can be calculated as:

$$\{\mathbf{Hu}_i\}^7_{i=1} = \{\mathrm{f}_i(\tilde{\mathbf{M}}_{pq})\}^7_{i=1} \tag{9}$$

where f_i is specific function or formula (non-linear combination) of the normalized moments $\tilde{\mathbf{M}}^k_{pq}$. The detailed calculations of the seven Hu moments can be found in [13]. Additionally, we have incorporated area-perimeter ratio v_{15}, minimum bounding rectangle v_{16}, and convexity v_{17} to more accurately characterize the geometric properties of polygons. Consequently, for each polygon, we derive a comprehensive vector of geometric information, denoted as $\mathbf{V}^k_i$, which includes a total of 17 values.

For the final sequence $\mathbf{X}^S \in \mathbb{R}^{L\times 17}$, where L denotes the length of the sequence, we insert an additional class vector to facilitate the similarity calculation in the contrastive learning. We assign a value of $+1$ to the category vectors for clips corresponding to the hotspot, while for all other cases, we assign a value of -1. Ultimately, l polygon vectors, along with their corresponding category vectors, collectively constitute the sequence $\mathbf{X}^S$. The complete Layout2Seq pipeline is detailed in Algorithm 1.

Algorithm 2 Training and testing for our proposed

Training process:

Require: Training pair set of layout representations $\{\mathbf{X}^I_i, \mathbf{X}^S_i\}^N_{i=0}$, where $\mathbf{X}^S_i$ is from Alogorithm 1.

Parameters: An image encoder $\mathcal{L}$, an sequence model $\mathcal{S}$.

Initialize $\mathcal{L}$, $\mathcal{S}$ randomly.
for $i = 1$ to N **do**
  $\mathbf{X}^{pos}_i \leftarrow$ Compute centroid by Equation (11);
  $\mathbf{X}^S_i = \mathbf{X}^{pos}_i + \mathbf{X}^S_i$, Absolute Position Embedding;
  Compute (f^I_i, f^S_i) by Equation (1);
  Optimize $\mathcal{L}, \mathcal{S}$ by Equation (5);
end for

Testing process:

Require: A layout clip image $\mathbf{X}^I$, $\mathcal{L}$ and $\mathcal{S}$, as returned by training process.

Ensure: the prediction result Y.

Assuming $\hat{Y} = HS$, compute the $\mathbf{X^S_{hs}}$ by Alogorithm 1;
Assuming $\hat{Y} = NHS$, compute the $\mathbf{X^S_{nhs}}$ by Alogorithm 1;
$f^I = \mathrm{Proj}(\mathcal{I}(\mathbf{X}^I))$;
$f^S_{hs}, f^S_{nhs} = \mathrm{Proj}(\mathcal{S}(\{\mathbf{X^S_{hs}}, \mathbf{X^S_{nhs}}\}))$;
$score_{hs}, score_{nhs} = \mathrm{Softmax}(\mathrm{sim}(f^I, (f^S_{hs}, f^S_{nhs})))$;
$Y \leftarrow score_{hs}, score_{nhs}$;
return Y.

D. Absolute position embedding

The polygon-based topology is commonly associated with the definition of hotspots in design rules. We embedded the positions of the polygons into the learning process of the sequence model by the proposed APE. For a given polygon P^k, the centroid coordinates (x_k, y_k) are computed by central moments as follows:

$$x_k = \frac{\mathbf{C}^k_{10}}{\mathbf{C}^k_{00}}, y_k = \frac{\mathbf{C}^k_{01}}{\mathbf{C}^k_{00}} \tag{10}$$

As shown in Fig. 4, we exclude the position encoder typically employed in Transformer architectures, as the lack of continuous contextual relationships between polygons. Instead, we developed a straightforward multi-layer perceptron (MLP) to encode the coordinates for integration into the sequence model, yielding absolute position embedding $\mathbf{X}^{pos}_i$ as follows:

$$\mathbf{X}^{pos}_i = \{\sigma(\mathrm{FC}([\hat{x}_k, \hat{y}_k]))\}^l_{k=1} \tag{11}$$

Here, the coordinates need to be normalized as $\hat{x}_k = \frac{x_k}{W}$ and $\hat{y}_k = \frac{y_k}{H}$, W and H denotes the width and height of the input layout clip image, l denotes the number of polygon. Consistent with other approaches utilizing position embedding, we adopt summation to fuse position information into the sequence representation. Therefore, for the sequence model $\mathcal{S}(\cdot)$, the inputs are $\mathbf{X}^S_{i,input} = \mathbf{X}^{pos}_i + \mathbf{X}^S_i$.

The entire training and evaluation process for CLI-HD, which combines Layout2Seq and APE, is detailed in Alogorithm 2.

TABLE I
STATISTICS OF THE DATASET

Benchmark	Training Set		Testing Set	
	#HS	#NHS	#HS	#NHS
ICCAD2012	1204	17096	2524	13503
ICCAD2016	1300	9935	1301	2484
ICCAD2019-1	467	17758	1001	14621
ICCAD2019-2	467	17758	64310	65523

TABLE II
COMPARISON ON ICCAD-2012 BENCHMARK WITH THE STATE-OF-THE-ART METHODS.

Method	Accuracy(%)	FA(%)	Time(s)
DAC'2021 [6]	98.25	6.10	-
TODAES'2022 [14]	98.90	9.90	30.0
TCAD'2022 [10]	98.77	18.50	**3.0**
DATE'2022 [8]	98.42	12.80	3.2
DATE'2023 [5]	96.20	6.40	7.5
Ours	**99.72**	**1.20**	10

IV. EXPERIMENTAL RESULTS

Our experimental framework is implemented using PyTorch, with all models trained on 8 NVIDIA GeForce RTX A6000 GPUs (48 GB memory) for accelerated computation. We evaluate our approach on the ICCAD2012 [15], ICCAD2016 [16], and ICCAD2019 benchmarks [17]. Notably, our experiments on the ICCAD2016 dataset incorporate random cropping of image sizes, a novel augmentation technique that allows for the presence of multiple hotspots within a single image. This increases both the complexity and challenge of hotspot detection, distinguishing our methodology from prior works. Comprehensive dataset details are provided in TABLE I.

A. Comparison with State-of-the-Art Hotspot Detection Works on ICCAD2012

To demonstrate the efficacy of our proposed approach, we present the performance results on the ICCAD2012 dataset. As shown in TABLE II, a comparison is made between our method and the currently state-of-the-art techniques for hotspot detection. Our proposed method demonstrates superior accuracy compared to other state-of-the-art algorithms (98.25% of DAC'2021 [6], 98.90% of TODAES'2022 [14], 98.77% of TCAD'2022 [10], 98.42% of DATE'2022 [8] and 96.20% of DATE'2023 [5] v.s. 99.72% of ours). Moreover, the proposed method effectively addresses the problem of excessive false positives encountered by existing approaches. On the ICCAD2012 benchmark, our method reduces the false alarm rate to 1.2%. The computational efficiency of the proposed algorithm is a key consideration. Our method achieves competitive performance while maintaining an acceptable trade-off between the accuracy and speed.

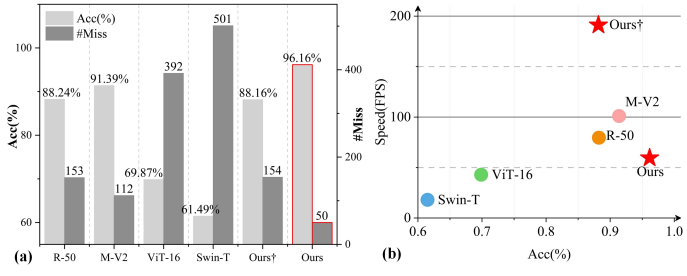

Fig. 5. Comparison with the state-of-the-art deep learning-based methods on ICCAD2016. a) Bars with solid red borders indicate optimal results. † denotes our results of supervised fine-tuning. #Miss denotes the number of false negatives. b) Comparison of model detection speed and accuracy on ICCAD2016.

TABLE III
RESULTS ON ICCAD2019-1 FOR OUR PROPOSED

Benchmark	Accuracy(%)	FA(%)
Benchmark1	76.60	29.66
Benchmark2	86.76	22.53
Benchmark3	88.63	19.45
Benchmark4	83.58	22.65
Benchmark5	93.00	20.43
Benchmark6	88.67	18.52
Benchmark7	85.75	21.36
Benchmark8	86.85	20.37
Benchmark9	93.40	9.71

B. Comparison with State-of-the-Art Deep Learning-based Classification Model on ICCAD2016

We evaluated four state-of-the-art backbone networks (ResNet50 (R-50) [18], MobileNetV2 (M-V2) [19], ViT-16 [12], and Swin-Transformer Tiny (Swin-T) [20]) using supervised learning on the ICCAD2016 datasets, and compared their performance against our proposed method. As shown in Fig. 5(a), our proposed method achieves an optimal accuracy of 96.16% with reducing the number of missed detections to only 50. Specifically, we evaluate the generalization of the proposed method by fine-tuning the learned representations under a supervised learning framework, denoted as Ours†. We attained comparable accuracy to a 50-layer ResNet (88.24% vs 88.16%) by fine-tuning contrastive learning representations with a single-layer neural network. Furthermore, as illustrated in Fig. 5(b), our fine-tuned model demonstrates significantly improved inference speed, achieving nearly 200 FPS, outperforming all comparative methods.

C. Comparison with State-of-the-Art on ICCAD2019

We perform comparative experiments using the ICCAD2019 benchmark, which consists of two subsets, ICCAD2019-1 and ICCAD2019-2, both sharing a common training set. It is considered a more accurate evaluation of the detector's capacity to detect never-before-seen hotspots. TABLE III provides detailed results for the ICCAD2019-1 dataset. TABLE IV reports the performance of our proposed method and four state-of-the-art hotspot detection algorithms, as presented in TCAD'2020 [21], TODAES'2022 [14], DATE'2023

TABLE IV
COMPARISON ON ICCAD-2019 BENCHMARK WITH THE STATE-OF-THE-ART METHODS.

Benchmark	TCAD'2020 [21]		TODAES'2022 [14]		DATE'2023 [5]		GLSVLSI'2024 [22]		Ours	
	ACC(%)	FA(%)	ACC(%)	FA(%)	ACC(%)	FA(%)	ACC(%)	FA(%)	ACC(%)	FA(%)
ICCAD2019-1	80.90	2.50	87.20	9.70	91.60	8.60	62.30	3.70	**96.10**	**1.00**
ICCAD2019-2	89.80	83.90	90.30	84.10	**90.50**	83.90	84.30	64.20	87.02	**20.53**
Average	85.30	43.20	88.75	46.90	91.05	46.25	73.30	33.95	**91.56**	**10.77**

[5], and GLSVLSI'2024 [22]. Firt, our proposed framework achieves an average accuracy of 91.56%, outperforming TCAD'2020 [21], TODAES'2022 [14], DATE'2023 [5], and GLSVLSI'2024 [22], which report accuracies of 85.30%, 88.75%, 91.05% and 73.30%, respectively. Furthermore, the results indicate that the existing methods exhibit a significant false positive rate on the ICCAD2019-2 dataset. Our framework demonstrates a substantial advantage in effectively suppressing false alarms, achieving a rate of 20.53% on ICCAD2019-2 and an average of 10.77%.

V. CONCLUSION

In this work, we propose CLI-HD, a novel hotspot detection method that leverages the topological representation of layout patterns and a contrastive learning framework. We develop a contrastive learning framework that aligns feature representations across binary images and sequence representations. The Layout2Seq is proposed to integrates geometric information of polygons in the process of pattern representation. We introduce absolute position information to learn the topological structure. Our results show that CLI-HD surpasses previous methods, achieving a 0.82%-4.77% gain in accuracy while reducing false alarm rates by 4.9%-23.18%. Future work will explore the application of topological and contrastive learning techniques across diverse downstream tasks beyond hotspot detection.

REFERENCES

[1] W.-C. Chang and I. H.-R. Jiang, "iclaire: A fast and general layout pattern classification algorithm with clip shifting and centroid recreation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 8, pp. 1662–1673, 2019.

[2] S. S.-E. Tseng, W.-C. Chang, I. H.-R. Jiang, J. Zhu, and J. P. Shiely, "Efficient search of layout hotspot patterns for matching sem images using multilevel pixelation," in *Optical Microlithography XXXII*, vol. 10961. SPIE, 2019, pp. 51–58.

[3] Z. Wang, Y. Shen, W. Zhao, Y. Bai, G. Chen, F. Farnia, and B. Yu, "Diffpattern: Layout pattern generation via discrete diffusion," in *2023 60th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2023, pp. 1–6.

[4] G. Zhou, B. Korrapati, G. R. Reddy, J. Hu, Y. Chen, and D. G. Thakurta, "Patternpaint: Generating layout patterns using generative ai and inpainting techniques," *arXiv preprint arXiv:2409.01348*, 2024.

[5] Z. Chen, F. Yang, L. Shang, and X. Zeng, "Automated and agile design of layout hotspot detector via neural architecture search," in *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2023, pp. 1–6.

[6] Y. Xiao, M. Su, H. Yang, J. Chen, J. Yu, and B. Yu, "Low-cost lithography hotspot detection with active entropy sampling and model calibration," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2021, pp. 907–912.

[7] A. B. Kahng, C.-H. Park, and X. Xu, "Fast dual-graph-based hotspot filtering," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 9, pp. 1635–1642, 2008.

[8] S. Sun, Y. Jiang, F. Yang, B. Yu, and X. Zeng, "Efficient hotspot detection via graph neural network," in *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2022, pp. 1233–1238.

[9] L. Wen, Y. Zhu, L. Ye, G. Chen, B. Yu, J. Liu, and C. Xu, "Layoutransformer: Generating layout patterns with transformer via sequential pattern modeling," in *Proceedings of the 41st IEEE/ACM international conference on computer-aided design*, 2022, pp. 1–9.

[10] H. Geng, H. Yang, L. Zhang, F. Yang, X. Zeng, and B. Yu, "Hotspot detection via attention-based deep layout metric learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 8, pp. 2685–2698, 2022.

[11] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[12] A. Dosovitskiy, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[13] M.-K. Hu, "Visual pattern recognition by moment invariants," *IRE transactions on information theory*, vol. 8, no. 2, pp. 179–187, 1962.

[14] Y. Jiang, F. Yang, B. Yu, D. Zhou, and X. Zeng, "Efficient layout hotspot detection via neural architecture search," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 27, no. 6, pp. 1–16, 2022.

[15] J. A. Torres, "Iccad-2012 cad contest in fuzzy pattern matching for physical verification and benchmark suite," in *Proceedings of the International Conference on Computer-Aided Design*, 2012, pp. 349–350.

[16] R. O. Topaloglu, "Iccad-2016 cad contest in pattern classification for integrated circuit design space analysis and benchmark suite," in *2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2016, pp. 1–4.

[17] G. R. Reddy, K. Madkour, and Y. Makris, "Machine learning-based hotspot detection: Fallacies, pitfalls and marching orders," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2019, pp. 1–8.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[19] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[20] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[21] Y. Jiang, F. Yang, B. Yu, D. Zhou, and X. Zeng, "Efficient layout hotspot detection via binarized residual neural network ensemble," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 7, pp. 1476–1488, 2020.

[22] C. Wang, Y. Fang, and S. Zhang, "Feature fusion based hotspot detection with r-efficientnet," in *Proceedings of the Great Lakes Symposium on VLSI 2024*, 2024, pp. 446–451.