Understand and Detect: Lithographic Hotspot Detection by the Interpretable Graph Attention Network

Andy Liu[1*], Silin Chen[1*], Guohao Wang[3], Wenzheng Zhao[3], Yuxiang Fu[1,2], 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]ZetaTech Co., Ltd., China

***Abstract*—Lithography hotspot detection plays a crucial role in the design-for-manufacturing (DFM) process. Recent developments in machine learning have demonstrated significant advantages in improving feature extraction capabilities, computational efficiency, and reducing false alarms in hotspot detection. However, deep learning models remain black-box approaches, with the interpretability challenge yet to be addressed. The topological features of the local patterns causing hotspot classification results also remain unknown. In this paper, we propose the first interpretable GNN framework for lithography hotspot detection, which achieves both high detection accuracy and precise hotspot localization within the layout. Our framework maps the geometric structure of layouts into graph representation. Then, we introduce a novel graph attention network (GAT) framework, encoding local topological features through attention queries on neighbors. Additionally, a novel graph interpretability method is designed by leveraging latent variables in edge distributions and subgraphs optimization, enabling the extraction of local topological features and providing detailed explanations of hotspot localization. Experimental results demonstrate that our approach achieves state-of-the-art (SOTA) performance on the ICCAD-2012 and ICCAD-2019 benchmarks. Moreover, we validate the interpretability of our GNN model on the ICCAD-2016 benchmark, accurately identifying hotspot locations within the lithographic design.**

***Index Terms*—Design For Manufacturability, Lithography Hotspot Detection, Graph Neural Networks, Interpretable AI**

I. Introduction

As Very-Large-Scale Integration (VLSI) technology continues to advance, the feature sizes in chip designs are shrinking, making lithographic process errors more significant. Lithography hotspot detection becomes particularly crucial at the design stage, enabling the early identification and prevention of potential printing defects to reduce manufacturing costs. Early detection methods, such as lithography simulation [1], [2], offer high accuracy but are computationally intensive and time-consuming. Pattern-matching methods [3], [4] are highly effective at identifying patterns already present in the hotspot library, but they exhibit limited generalization ability.

With the development of AI technology, Machine Learning (ML)-based hotspot detection methods [5]–[7], especially those based on Deep Learning (DL) [8], [9], [11], [12], [16], [17], have made significant improvements in detection efficiency, accuracy, and generalization capabilities. CNN-based detection methods [5], [8], [9], [11] apply spatial feature transformations to layout clip images, which are subsequently fed into convolutional network layers for feature extraction. Sequence-based detection methods [13], utilizing Transformer architectures, capture spatial or structural patterns in layout data to effectively detect hotspots by analyzing sequential or contextual relationships. However, these methods, which use clip images as input, face the issue of data sparsity. Moreover, they lack the ability to model the topological features of lithographic layouts.

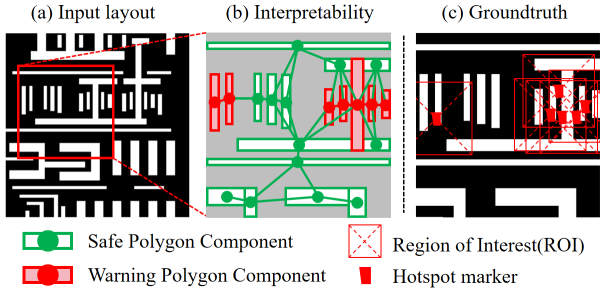

Fig. 1. Our Interpretability Results on a hotspot sample. (a) An input GDS layout. The warning polygon components in (b) indicate the layout structures that are likely to cause hotspots locationally. (c) The hotspot ground truth from ICCAD-2016 contest through EUV lithography simulation.

In contrast, Graph Neural Network [14], [15] (GNN)-based hotspot detection methods stand out due to their robust ability to extract topological features. GNN-based detection methods encode the layout into a graph representation, allowing more effective extraction of topological features through the aggregation and updating of node information. Sun et al. [16] proposed a new graph representation method for layout patterns, designed rich node and edge features, and introduced a modified GNN framework with multi-type edges. Yan et al. [17] proposed a lightweight 3-hop message-passing GNN framework based on the Modified Transitive Closure Graph (MTCG). Their method significantly reduces false alarms while achieving high accuracy [16]. Moreover, the graph-based layout data significantly lowers storage and computational

*Andy Liu and Silin Chen contributed equally to this work.
†Corresponding Author: nzou@nju.edu.cn. This work was supported in part by Postgraduate Research Practice Innovation Program of Jiangsu Province (Grant Number: KYCX25_0394) and the National Natural Science Foundation of China (62341408).

costs, demonstrating high potential for practical applications.

However, these DL-based methods lack interpretability in the detection results and fail to adequately address the features of hotspot patterns. DL-based hotspot detection methods are typically considered black-box approaches [22]. Most existing frameworks treat hotspot detection as a classification task for layout clips, neither localizing the specific areas responsible for hotspots nor explaining what kinds of local topological features contribute to them. Only Sun et al. [18] and Jiang et al. [19] have provided CNN-based interpretability results. However, CNN-based hotspot explainers suffer from lower detection accuracy, and the coverage of the heatmaps they generate often exceeds the actual hotspot positions, resulting in higher computational costs.

In this work, we pioneeringly introduce an improved Graph Attention (GAT)-based detection flow and GNN interpretability into the field of lithography hotspot detection. Our feature extraction architecture encodes layout geometric information through rich aggregation operations, with a multi-head GAT mechanism introduced to capture the interactions among local pattern polygons. Our graph interpretability module utilizes the latent variable distribution of edge features from the feature extraction framework to backpropagate through and optimize the subgraph. Finally, the subgraph is optimized to precisely locate the local patterns responsible for hotspot classification results.

The main contributions of this paper are as follow:

- We propose the first Interpretable GNN framework for lithography hotspot detection, which accurately localizes the patterns responsible for hotspot formation in layouts, providing local topological interpretability for GNN-based detection flows.
- We present a novel GDS2Graph representation method that emphasizes both the geometric properties of pattern polygons and the topological features of their neighboring structures.
- We propose a novel GAT feature extraction framework that encodes simple geometric features of input layout clips into a rich graph topological feature space, providing robust feature embeddings for subsequent hotspot detection and interpretability modules.
- In the ICCAD-2012, ICCAD-2016 and ICCAD-2019 benchmarks, our hotspot detection framework achieves state-of-the-art performance in both accuracy and false alarm rates. Furthermore, we present both visualized and quantitative interpretability results on the ICCAD-2016 dataset, demonstrating our framework's superior detection accuracy and interpretability capabilities.

II. PRELIMINARIES

To assess the hotspot detection performance of our proposed framework, we define the hotspot classification task.

Problem 1 (Hotspot Detection): This is a binary classification problem. Given a collection of clip patterns labeled as hotspots and non-hotspots, our goal is to train a classification model that maximizes accuracy (Acc) and minimizes false alarms (FA). The evaluation metrics are defined as follows:

Definition 1 (Acc): The proportion of correctly predicted hotspot samples out of all hotspot (HS) samples.

Definition 2 (FA): The proportion of non-hotspot (NHS) samples incorrectly classified as hotspots.

To evaluate the interpretability task, we define the graph-based hotspot interpretability problem.

Problem 2 (Interpretability): For each pattern, graph-level interpretability aims to identify subgraphs that are crucial for the hotspot classification results, corresponding to the local pattern polygons responsible for the hotspot. The objective of the interpretability task is to maximize the interpretability hotspot accuracy (HA) and minimize the false positive rate (FPR), which are defined as follows:

Definition 3 (HA): The proportion of rectangular components from pattern polygons that are correctly interpreted as hotspots, relative to the total number of hotspot regions.

Definition 4 (FPR): The proportion of rectangular components from pattern polygons that are incorrectly interpreted as hotspots, relative to the total number of non-hotspot regions.

The details of the hotspot detection and interpretability experiments are provided in Section V. Additionally, we provide visualized results demonstrating the interpretability of hotspot detection.

III. PROPOSED METHOD

An overview of our interpretable GAT-based lithographic hotspot detection framework is presented in Figure 3. Firstly, we introduce a novel GDS2graph lithographic layout representation method, along with a detailed explanation of the encoding approach for geometric features. Secondly, we present the proposed novel graph attention-based feature extraction framework. Lastly, we pioneeringly emphasize a GNN interpretability method for hotspot detection.

A. GDS2graph Representation

The GDS2graph representation method projects the voluminous GDS information into a compact graph structure, reducing storage requirements while preserving the layout's geometric information. Figure 2 illustrates our graph representation method using a GDS layout clip example. We denote a graph as $G = (V, E)$, where V represents the set of nodes and E represents the set of edges.

First, we perform rectangular partitioning of the polygons to construct the nodes, where each rectangle is treated as a node. Subsequently, each rectangle is assigned features based on its intrinsic properties and the geometric relationships with its local neighbors. The length l and width w serve as the simplest intrinsic properties of each rectangle. The minimum neighbor distance represents the shortest straight-line distance between a node and its neighboring rectangles. The coordinates x and y of the rectangle are used for visualization and interpretability.

Finally, we design edge connection rules among the rectangles, considering that adjacent polygons may contribute to hotspots. On the one hand, nodes belonging to different

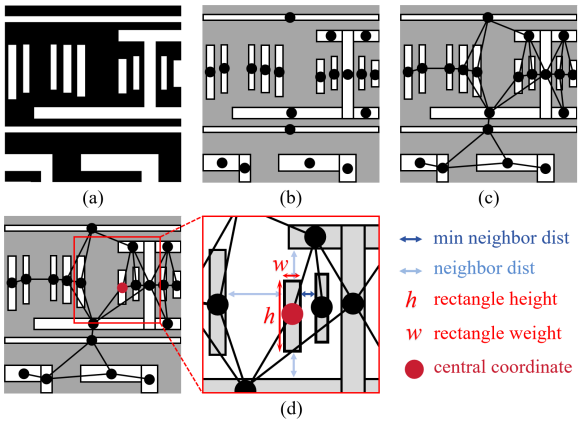

Fig. 2. Proposed graph representation method. (a) A layout sample. (b) Node representation with rectangular segmentation for polygons. (c) Edge connections. (d) Description of the specific node feature creation method.

polygons will be connected if (1) they are neighbors and (2) the neighbor distance is less than a threshold t, which is determined by the feature size of the technology. On the other hand, to ensure the graph's connectivity, nodes within the same polygon are connected by edges. Independent polygons are also connected to the main graph.

B. GNN Feature Extraction Framework

Considering the importance of local topological features in lithography hotspot detection, we propose a graph attention-based feature extraction approach to uncover high-level topological features of the layout. As illustrated in Figure 3.(b), our feature extractor is designed with a rich aggregation structure. The Graph Attention mechanism is introduced to better focus on the interactions within the neighboring nodes in the layout. The detailed steps are presented in Algorithm 1.

With the GDS graph structure embedding, we use the adjacency matrix A and the set of node features $\mathbf{h} = \{h_1, h_2, \ldots, h_N\}$, where $h_i \in \mathbb{R}^{F_1}$ as input. In our framework, the node feature h_i represents the initial feature vector for each node (corresponding to a region in the layout polygons) and contains the basic information of the node. For node i, the interactions with its neighboring nodes $j \in N_i$ are first calculated by concatenating the features, which are then passed through MLPs. Next, the neighbor edge features e_i are computed using the neighbor concatenation operation $f_{\text{edge}}(\cdot)$ with element-wise max. The aggregated feature H_i for node i is computed by combining the node's own feature and its edge features with the aggregation operation $f_{\text{aggr}}(\cdot)$:

$$f_{\text{aggr}} = \sigma(\mathcal{A}\left((\text{MLP}_2(h_i), e_{ij})\right)). \tag{1}$$

Then, node feature set is updated to $\mathbf{H} = \{H_1, H_2, \ldots, H_N\}$, $H_i \in \mathbb{R}^{F_2}$.

After that, the graph attention mechanism is introduced to aggregate local topological feature by combining the node's own feature and its neighbor polygon features. For node i and its set of neighboring nodes $\mathcal{N}_i$, the core of Graph Attention is

Algorithm 1 Graph Attention-based Feature Extraction

Require: Graph structure $G(V, E)$, adjacency matrix $A \in \mathbb{R}^{N\times N}$, initial node features $\mathbf{h} = \{h_1, h_2, \ldots, h_N\}$, where $h_i \in \mathbb{R}^{F_1}$, neighbor set of node i: N_i.
Ensure: Latent node features $\mathbf{h}' = \{h'_1, h'_2, \ldots, h'_N\}$, where $h'_i \in \mathbb{R}^{F'}$.

for each node $i \in V$ **do**
  for each node $j \in N_i$ **do**
    $e_{ij}^{(1)} = f_{\text{concat}}(h_i, h_j)$
    $e_{ij}^{(2)} = \text{MLP}_1(e_{ij}^{(1)})$
  end for
  $e_i = f_{\text{edge}}\left(\max_{j\in N_i}\{e_{ij}^{(2)} | \ \forall j \in N_i\}\right)$
  $H_i = f_{\text{aggr}}\left(\text{MLP}_2(h_i), e_i\right), \forall j \in N_i$
  for $k = 1$ to K **do**
    for each node $j \in N_i$ **do**
      $\alpha_{ij} = \text{GAT}(H_i, \text{MLP}_3(H_j))$
    end for
    ${h^{(k)}}'_i = \sigma\left(\sum_{j\in\mathcal{N}_i} \alpha_{ij} W H_j\right)$
  end for
  ${h'}_i = f_{\text{concat}}({h^{(k)}}'_i), k = 1, 2, \ldots, K$
end for
return $\mathbf{h}'$

to compute the attention score α_{ij} for each neighboring node with respect to node i:

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(\mathbf{a}^T[W H_i \parallel W H_j]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\text{LeakyReLU}\left(\mathbf{a}^T[W H_i \parallel W H_k]\right)\right)}, \tag{2}$$

where $a : \mathbb{R}^{F'} \times \mathbb{R}^{F'} \to \mathbb{R}$ is a parameterized single-layer feedforward neural network. Both $\mathbf{a} \in \mathbb{R}^{2F'}$ and $\mathbf{W} \in \mathbb{R}^{F'\times F}$ are parameter matrices for linear transformations. The symbol $\parallel$ denotes the concatenation operation. In computing the attention scores, a softmax function is applied to ensure that the attention scores are suitable for global node normalization. Additionally, the LeakyReLU activation function is employed to enhance the model's nonlinear expressive capabilities.

The attention scores, serving as importance coefficients, are applied during the aggregation step to update the feature representation $\mathbf{h}' = \{h'_1, h'_2, \ldots, h'_N\}$, where $h'_i \in \mathbb{R}^{F'}$ for each node. We introduce multi-head graph attention to aggregate neighbor feature representations across K independent channels:

$$h'_i = \Big\Vert_{k=1}^{K} \sigma\Big(\sum_{j\in\mathcal{N}_i} \alpha_{ij}^k W^k h^{(k)}{}_j\Big), \tag{3}$$

where $h_j^{(k)}, j \in \mathcal{N}_i$ stands for the k-th attention head output. Each pass through the feature extraction layer involves the aggregation and updating of neighborhood features for each node.

Stacking multiple layers enables capturing a broader range of interactions between nodes. The resulting node features are subsequently used for downstream tasks, such as classification and interpretability. In this way, we can focus attention on the

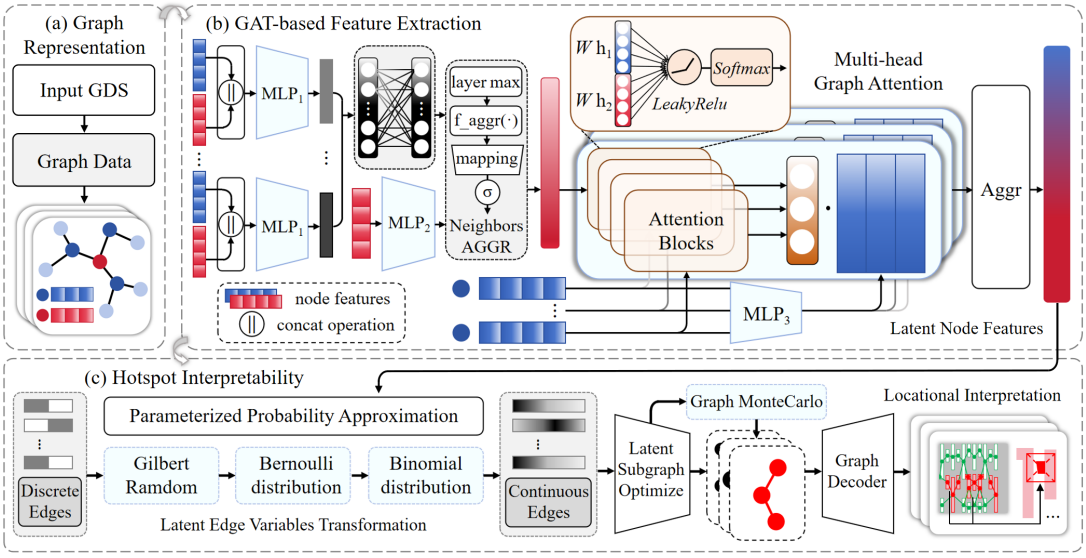

Fig. 3. Overview of the proposed Interpretable Graph Attention framework for Lithography Hotspot Detection.

neighboring polygon components that have a greater impact on causing hotspots.

C. Graph Interpretability for Hotspot Detection

The graph-based interpretability framework enables node-level analysis to identify critical subgraphs and topological features responsible for hotspot polygons formation. As illustrated in Figure 3.(c), we employ a neural-network-parameterized probabilistic model to approximate the edge feature distribution, facilitating effective learning of latent structures. Our method builds upon assumptions from previous work, emphasizing the identification of critical subgraphs based on local topology.

Assumption 1 (Gilbert Random Graph)**.** In the subgraph sampling process, we assume that the input graph G follows a Gilbert random graph model:

$$P(G) = \prod_{(i,j)\in\mathcal{E}} P(e_{ij}), \tag{4}$$

where $\mathcal{E}$ represents the set of edges in the graph G, and $P(e_{ij})$ is the probability that the edge e_{ij} exists.

Assumption 2 (Edge Bernoulli Distribution)**.** For each edge e_{ij}, we assume that $P(e_{ij})$ follows a Bernoulli distribution $e_{ij} \sim \text{Bernoulli}(\theta_{ij})$, where $\theta_{ij} \in [0,1]$ represents the probability of the existence of the edge between nodes i and j.

These assumptions allow us to model the graph with edge probabilities, which facilitates efficient subgraph sampling. Additionally, these probabilistic assumptions serve as a simulation of the inherent uncertainty in hotspot occurrence, capturing the variations and unpredictability of hotspot patterns in the layout.

Lemma 1 (Probability Approximation Property)**.** In the subgraph random sampling process, when approximating a Bernoulli distribution of edges, the latent distribution of edge weights $\hat{e}_{ij} \in (0,1)$ is differentiable.

The objective function of our interpretability model is to minimize the conditional entropy in order to obtain the latent subgraphs that preserve the invariance of the output, where the subgraphs correspond to the potential layout polygon components that may cause hotspots.

$$\min_{G_s} H(Y_0|G = G_s), \tag{5}$$

where H represents the conditional entropy, and Y_0 is the original prediction, with the complete graph G as input. G_s is a subgraph of G, and there are 2^M possible subgraphs for a graph with M edges. Enumeration is not adopted due to its inefficiency and the inability to share across all instances.

Therefore, based on Assumption 1, we assume that the input graph in the GNN interpretability model follows a Gilbert random graph. With Assumption 2, the objective function can be written as:

$$\begin{aligned}\min_{G_s} H(Y_0|G = G_s) &= \min_{G_s} \mathbb{E}_{G_s}\left[H(Y_0|G = G_s)\right] \\ &\approx \min_{\Theta} \mathbb{E}_{G_s\sim q(\Theta)}\left[H(Y_0|G = G_s)\right],\end{aligned} \tag{6}$$

where $q(\Theta)$ represents the distribution of the subgraph G_s parameterized by θ.

However, the interactions between polygons responsible for local hotspots are not balanced. Therefore, we introduce a parameterized probabilistic model to transform the binarized Bernoulli distribution into a continuous distribution within the range $(0,1)$. The subgraph sampling process $G_s \sim q(\Theta)$ is approximated by the function:

$$G_s \approx \hat{G}_s = f_\Omega(G_0, \tau, \varepsilon), \tag{7}$$

with parameters Ω, temperature τ, and an independent random variable ε.

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

Benchmark	TCAD'2020 [10]		TODAES'2022 [32]		DATE'2023 [30]		GLSVLSI'2024 [31]		TCAD'2025 [17]		Ours	
	Acc(%)	FA(%)	Acc(%)	FA(%)	Acc(%)	FA(%)	Acc(%)	FA(%)	Acc(%)	FA(%)	Acc(%)	FA(%)
ICCAD-2019-1	80.90	**2.50**	87.20	9.70	91.60	8.60	62.30	3.70	91.30	8.40	**95.20**	12.07
ICCAD-2019-2	89.80	83.90	90.30	84.10	90.50	83.90	84.30	64.20	**94.60**	81.80	88.43	**15.71**
Average	85.30	43.20	88.75	46.90	91.05	46.25	73.30	33.95	**92.95**	45.10	**91.82**	**11.58**

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

Method	Acc(%)	FA(%)	Inference Time(s)
TODAES'2019 [28]	97.60	5.60	-
DAC'2021 [29]	98.25	6.10	-
DATE'2022 [16]	98.42	12.80	3.2
DATE'2023 [30]	96.20	6.40	7.5
GLSVLSI2024 [31]	88.30	**0.20**	103.4
TODAES'2025 [19]	98.10	4.00	6.5
Ours	**98.47**	7.44	**2.9**

According to Lemma 1, we replace the conditional entropy with the cross-entropy loss $H(Y_0, \hat{Y}_s)$, where $\hat{Y}_s$ represents the predicted output given the subgraph $\hat{G}_s$ as input. Based on the aforementioned conditions, we employ Monte Carlo methods to approximate the optimization of the objective function.

$$\begin{aligned}&\min_{\Omega} \mathbb{E}_{\varepsilon\sim\text{Uniform}(0,1)}\left[H(Y_0, \hat{Y}_s)\right]\\&= \min_{\Omega} -\frac{1}{K}\sum_{k=1}^{K}\sum_{c=1}^{C} P_\Phi(Y_c|G=G_o)\log P_\Phi(Y_c|G=\hat{G}_s^{(k)}),\end{aligned} \tag{8}$$

where Y_c stands for $Y = c$, Ω denotes the parameters to be optimized, Φ represents the input parameters of the parameterized feature extraction architecture, K is the number of instances in the input graph, c is the number of predicted labels, and $\hat{G}_s^{(k)}$ refers to the k-th predicted latent subgraph.

In addition, within the optimized parameters $\Omega = g_\Psi(G_o, Z)$, this interpretability method is capable of providing latent subgraph explanations for all instances in the input data. The objective function can be expressed as:

$$\min_{\Psi} -\sum_{i\in\mathcal{I}}\sum_{k=1}^{K}\sum_{c=1}^{C} P_\Phi(Y_c|G=G_o^{(i)})\log P_\Phi(Y_c|G=\hat{G}_s^{(i,k)}), \tag{9}$$

where Z represents the input graph structure and node feature embeddings from our proposed methods in the previous sections. Ψ denotes the shared parameters within the network. $G^{(i)}$ is the input graph and $\hat{G}_s^{(i,k)}$ is the k-th sampled graph. By learning from a dataset containing a sufficient number of hotspot samples, we could achieve locational interpretability, precisely identifying the polygon components responsible for hotspots represented by subgraph output.

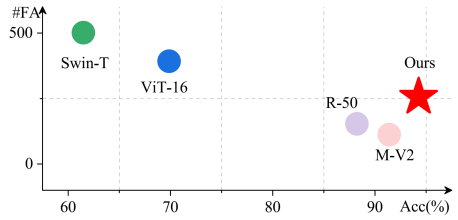

Fig. 4. Performance Comparison on ICCAD-2016 benchmark with the state-of-the-art methods.

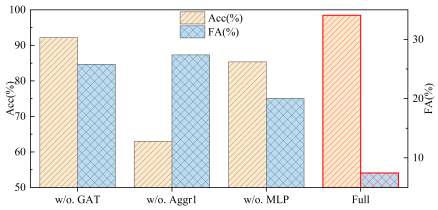

Fig. 5. Ablation study on ICCAD-2012 benchmark.

D. Benchmarks Information

For the hotspot detection classification task, we test our framework on layout clips from the 28nm process node in the ICCAD-2012 benchmark [20]. To mitigate severe class imbalance, we applied vertical and horizontal flips to hotspot samples in the training set, preserving their geometric properties. Besides, we adopt the more challenging ICCAD-2019 benchmark [21], which contains more Truly Never-Seen-Before and Hard-To-Classify patterns in the test set.

For the hotspot interpretability task, we adopt an unsupervised approach, using unlabeled GDS layout clips as input. Clip samples from the ICCAD-2016 benchmark [23], based on the 7nm process node, provide hotspot locations as locational ground truth, obtained through EUV lithography simulation. This allows for a direct validation of the interpretability performance of our framework.

IV. EXPERIMENTAL DETAILS

A. Training Strategies

We use layout clips of size $1024 \times 1024\,\text{nm}^2$ as input. Then, we convert the GDS layout to a graph using the graph representation method proposed in Section III.A, which reduces the memory consumption of the data by a factor of 5 to 12. The input graph data includes the adjacency matrix $A \in \mathbb{R}^{B\times N\times N}$ and node features $\mathbf{h} \in \mathbb{R}^{B\times N\times I_h}$, where B is the batch size, N is the maximum number of nodes in the

TABLE III
QUANTITATIVE EVALUATION OF INTERPRETABILITY PERFORMANCE.

N-hop Aggregation	HA(%)	FPR(%)
1	92.31	30.90
2	**95.06**	**29.66**
3	93.10	30.54
4	88.63	32.56
5	60.49	45.27

input graph data, and $I_h = 5$ is the feature dimension of the input nodes.

In the hotspot detection classification task, we set $K = 8$ attention heads in the proposed GAT-based feature extraction framework in Section III.B. We use the cross-entropy loss function during training and evaluate the hotspot detection performance. For the interpretability task, the objective function optimization is presented in Equation (9) in Section III.C. Additionally, we set the learning rate lr $= 0.001$ and the dropout rate Dr $= 0.01$. The Adam optimizer is used for parameter updates with a weight decay of 5×10^{-4}. Our experimental framework is implemented using PyTorch, with all models trained on 8 NVIDIA GeForce RTX A6000 GPUs (48 GB memory).

V. EXPERIMENTAL RESULTS

A. Hotspot Detection Classification

To validate the effectiveness of our proposed framework, we conducted lithography hotspot detection classification tasks on the ICCAD-2012, ICCAD-2016 and ICCAD-2019 benchmarks. Tables I, II and Figure 4 summarize the comparative experimental results with recent state-of-the-art hotspot detectors.

In the ICCAD-2012 benchmark, our framework achieved a high detection accuracy of 98.47%, surpassing most existing baseline methods. In the more challenging ICCAD-2019 benchmark, our framework achieved state-of-the-art results in both Acc and FA. In particular, on the ICCAD-2019-2 benchmark, our method significantly reduced the false alarm rate to 15.7%, far outperforming most existing hotspot detectors. This demonstrates the robust expressive capability of our framework in capturing hotspot latent features, as well as its superior detection performance.

In the ICCAD-2016 benchmark, we evaluated our method, which achieved the highest accuracy of 94.26%, outperforming several state-of-the-art backbone networks, including ResNet50 (R-50) [33], MobileNetV2 (M-V2) [34], ViT-16 [35], and Swin-Transformer Tiny (Swin-T) [36]. Meanwhile, we maintain the false alarm rate at an acceptable level.

Moreover, ablation studies on the ICCAD-2012 benchmark are conducted to demonstrate the importance of different components within our framework. In addition to the full pipeline, we provide three control groups: (1) w/o. GAT: removal of multi-head GAT blocks, (2) w/o. Aggr1: removal of the aggregation structure prior to the GAT blocks, and (3) w/o. MLP: removal of certain MLP layers. As presented in Fig 5, the 18.37% increase in false alarms (w/o. GAT) confirms its role in reducing FA. Accuracy drops of 35.50% (w/o. Aggr1) and 13.12% (w/o. MLP) highlight the importance of our feature aggregation structure in capturing latent topological patterns.

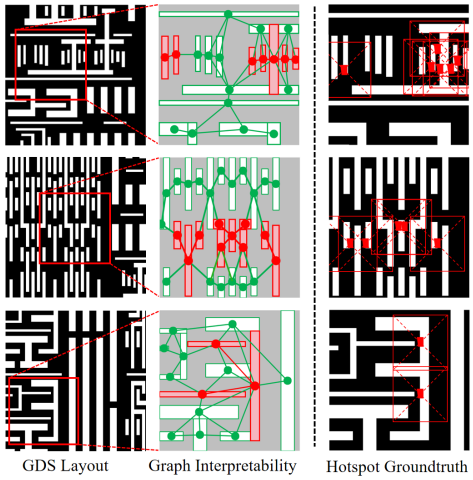

Fig. 6. Visual results of our hotspot interpretability on the ICCAD-2016 benchmark. The local components of the layout polygons responsible for the hotspots are precisely localized, with the red annotations corresponding to the explained subgraphs.

B. Hotspot Interpretability

We pioneeringly introduce graph-based interpretability to lithographic hotspot detection. We performed the interpretability task in an unsupervised manner on the ICCAD-2016 benchmark, where the layout polygon components responsible for the hotspots were accurately localized. The interpretability module generates positional indices for each graph node. Consequently, the resulting interpretability subgraphs can be mapped to the corresponding rectangular components of the layout polygons.

We provide visualized examples of hotspot interpretability, as shown in Figure 6. Each row corresponds to a layout sample. The middle column presents the interpretability results, which are the polygon component indices returned by the graph decoder. Fluorescent green nodes represent polygons that are part of a safe design, while red nodes indicate the polygon components detected as causing hotspots. Table III presents the quantitative evaluation results. The best interpretability performance is achieved at a 2-hop aggregation level, with the highest HA(95.06%) and lowest FPR(29.66%).

VI. CONCLUSION

In conclusion, our proposed Interpretable Graph Attention framework not only achieves state-of-the-art performance in hotspot detection but also provides a novel approach for the interpretability of hotspot localization at the layout level. This enables engineers to precisely identify the polygons responsible for hotspot formation, significantly reducing overall detection costs.

REFERENCES

[1] M. J. Mitra, P. Yu, and D. Z. Pan, "RADAR: RET-aware detailed routing using fast lithography simulations," Proceedings of the 42nd Annual Design Automation Conference, pp. 369-372, 2005.

[2] C. A. Mack, "Thirty years of lithography simulation," Optical Microlithography XVIII, vol. 5754, SPIE, San Jose, California, USA, pp. 1-12, 2005.

[3] W. W. Y. Wen, J. C. Li, S. Y. Lin, et al., "A fuzzy-matching model with grid reduction for lithography hotspot detection," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 33, no. 11, pp. 1671-1680, 2014.

[4] S. Y. Lin, J. Y. Chen, J. C. Li, et al., "A novel fuzzy matching model for lithography hotspot detection," Proceedings of the 50th Annual Design Automation Conference, pp. 1-6, 2013.

[5] H. Yang, L. Luo, J. Su, et al., "Imbalance aware lithography hotspot detection: a deep learning approach," Journal of Micro/Nanolithography, MEMS, and MOEMS, vol. 16, no. 3, pp. 033504-033504, 2017.

[6] 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," in Proc. 50th Annu. Design Autom. Conf., 2013, pp. 1–6.

[7] D. Ding, X. Wu, J. Ghosh, et al., "Machine learning based lithographic hotspot detection with critical-feature extraction and classification," in 2009 IEEE International Conference on IC Design and Technology, IEEE, 2009, pp. 219-222.

[8] H. Yang, J. Su, Y. Zou, et al., "Layout hotspot detection with feature tensor generation and deep biased learning," in Proceedings of the 54th Annual Design Automation Conference, 2017, pp. 1-6.

[9] H. Geng, H. Yang, L. Zhang, et al., "Hotspot detection via attention-based deep layout metric learning," in Proceedings of the 39th International Conference on Computer-Aided Design, 2020, pp. 1-8.

[10] Y. Jiang, F. Yang, B. Yu, et al., "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.

[11] D. Wu, S. Wang, M. Kamal, et al., "Enhancing layout hotspot detection efficiency with YOLOv8 and PCA-guided augmentation," arXiv preprint arXiv:2407.14498, 2024.

[12] S. Goyal and J. C. Rajapakse, "Self-supervised learning for hotspot detection and isolation from thermal images," Expert Systems with Applications, vol. 237, p. 121566, 2024.

[13] Y. Chen, Y. Wu, J. Wang, et al., "LLM-HD: Layout Language Model for Hotspot Detection with GDS Semantic Encoding," in Proc. 61st ACM/IEEE Design Automation Conference, 2024, pp. 1-6.

[14] F. Scarselli, M. Gori, A. C. Tsoi, et al., "The graph neural network model," IEEE Transactions on Neural Networks, vol. 20, no. 1, pp. 61-80, 2008.

[15] P. Veličković, G. Cucurull, A. Casanova, et al., "Graph attention networks," arXiv preprint arXiv:1710.10903, 2017.

[16] S. Sun, Y. Jiang, F. Yang, et al., "Efficient hotspot detection via graph neural network," in Proc. 2022 Design, Automation & Test in Europe Conference & Exhibition (DATE), IEEE, 2022, pp. 1233-1238.

[17] H. Yan, Y. Wang, P. Gao, et al., "A lightweight heterogeneous graph embedding framework for hotspot detection," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2025.

[18] H. Sun, C. Jiang, X. Ye, et al., "Interpretable CNN-Based Lithographic Hotspot Detection Through Error Marker Learning," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2024.

[19] C. Jiang, H. Sun, D. Feng, et al., "LithoExp: Explainable two-stage CNN-based lithographic hotspot detection with layout defect localization," ACM Transactions on Design Automation of Electronic Systems, 2025.

[20] J. A. Torres, "ICCAD-2012 CAD contest in fuzzy pattern matching for physical verification and benchmark suite," in 2012 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 349-350, 2012.

[21] 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.

[22] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you? Explaining the predictions of any classifier," in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 1135-1144.

[23] 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.

[24] Z. Ying, D. Bourgeois, J. You, et al., "GNNExplainer: Generating explanations for graph neural networks," Advances in Neural Information Processing Systems, vol. 32, 2019.

[25] D. Luo, W. Cheng, D. Xu, et al., "Parameterized explainer for graph neural network," Advances in Neural Information Processing Systems, vol. 33, pp. 19620-19631, 2020.

[26] L. Faber, K. Moghaddam, and R. Wattenhofer, "When comparing to ground truth is wrong: On evaluating GNN explanation methods," in Proc. 27th ACM SIGKDD Conf. Knowledge Discovery & Data Mining, 2021, pp. 332-341.

[27] Y. Rong, G. Wang, Q. Feng, et al., "Efficient GNN explanation via learning removal-based attribution," ACM Trans. Knowledge Discovery from Data, vol. 19, no. 2, pp. 1-23, 2025.

[28] X. He, Y. Deng, S. Zhou, et al., "Lithography hotspot detection with FFT-based feature extraction and imbalanced learning rate," ACM Trans. Design Autom. Electron. Syst. (TODAES), vol. 25, no. 2, pp. 1–21, 2019.

[29] Y. Xiao, M. Su, H. Yang, et al., "Low-cost lithography hotspot detection with active entropy sampling and model calibration," in Proc. 58th ACM/IEEE Design Autom. Conf. (DAC), IEEE, 2021, pp. 907–912.

[30] Z. Chen, F. Yang, L. Shang, et al., "Automated and agile design of layout hotspot detector via neural architecture search," in Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE), IEEE, 2023, pp. 1–6.

[31] C. Wang, Y. Fang, S. Zhang, "Feature fusion based hotspot detection with R-EfficientNet," in Proc. Great Lakes Symp. VLSI, 2024, pp. 446–451.

[32] Y. Jiang, F. Yang, B. Yu, et al., "Efficient layout hotspot detection via neural architecture search," ACM Trans. Design Autom. Electron. Syst. (TODAES), vol. 27, no. 6, pp. 1–16, 2022.

[33] K. He, X. Zhang, S. Ren, et al., "Deep residual learning for image recognition," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 770–778.

[34] M. Sandler, A. Howard, M. Zhu, et al., "Mobilenetv2: Inverted residuals and linear bottlenecks," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2018, pp. 4510–4520.

[35] A. Dosovitskiy, L. Beyer, A. Kolesnikov, et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in Int. Conf. Learn. Represent. (ICLR), 2021.

[36] Z. Liu, Y. Lin, Y. Cao, et al., "Swin transformer: Hierarchical vision transformer using shifted windows," in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), 2021, pp. 10012–10022.