Node2Node: Node Adaptation with Transformer for Cross-Node Hotspot Detection

Wenbo Xu[1]*, Silin Chen[1]*, Yibo Huang[1], Xinyun Zhang[2], Zixiao Wang[2], Bei Yu[2], Ningmu Zou[1,3]†
[1]Nanjing University
[2]The Chinese University of Hong Kong
[3]Interdisciplinary Research Center for Future Intelligent Chips (Chip-X)

***Abstract*—As semiconductor manufacturing advances to smaller process nodes, hotspot detection has become critical for ensuring the manufacturability and reliability of integrated circuit (IC) layouts. However, existing detection methods rely heavily on labeled data tailored to specific nodes, resulting in poor generalizability across nodes due to variations in layout geometries and fabrication processes. Labeling new data at advanced nodes is also costly and time-consuming. To overcome these challenges, we propose Node2Node, the first adaptation framework explicitly designed for cross-node hotspot detection. Node2Node integrates a novel node-invariant encoder with a node-specific encoder to jointly capture transferable and node-dependent features. To further improve robustness, we introduce a bidirectional center alignment strategy, which refines pseudo-labels by leveraging a small amount of labeled data from the target node. Additionally, a cross-node distribution loss is introduced to explicitly align feature distributions between nodes. Extensive experiments demonstrate that Node2Node substantially improves cross-node generalization and achieves state-of-the-art hotspot detection performance.**

I. Introduction

As semiconductor manufacturing advances, chip design is transitioning to smaller process nodes. In the chip manufacturing process, patterns must be transferred onto the silicon wafer through steps like photolithography. However, physical variations during manufacturing can cause deviations in patterns, leading to defects that affect the chip's functionality. These areas, prone to failure due to process variations, are known as hotspots. The detection of such hotspots has become crucial for ensuring the manufacturability and reliability of integrated circuit (IC) layouts. As nodes shrink and complexity increases, accurate hotspot detection is essential for improving yield and manufacturability in advanced semiconductor fabrication.

Existing hotspot detection methods rely on either lithography simulation, pattern matching, or machine learning. While traditional lithography simulation provides high accuracy, it is computationally expensive and impractical for large-scale applications. On the other hand, pattern matching [1]–[3] is fast but lacks generalization capability, failing to detect unseen hotspot patterns. Machine learning-based detector [4] provide a promising alternative but face a trade-off between accuracy and false alarm rate. Recent convolutional neural networks (CNNs) [5]–[10] have demonstrated strong performance. For example, Yang [11] develop a cnn-based model capable of detecting hotspots on a clip-by-clip basis while effectively addressing challenges caused by imbalanced data distribution. To address the inability of CNNs to capture long-range dependencies in large layouts, transformer-based models [12]–[15] have been introduced. Chen [13] first introduced large language models to better capture dependencies within large layouts.

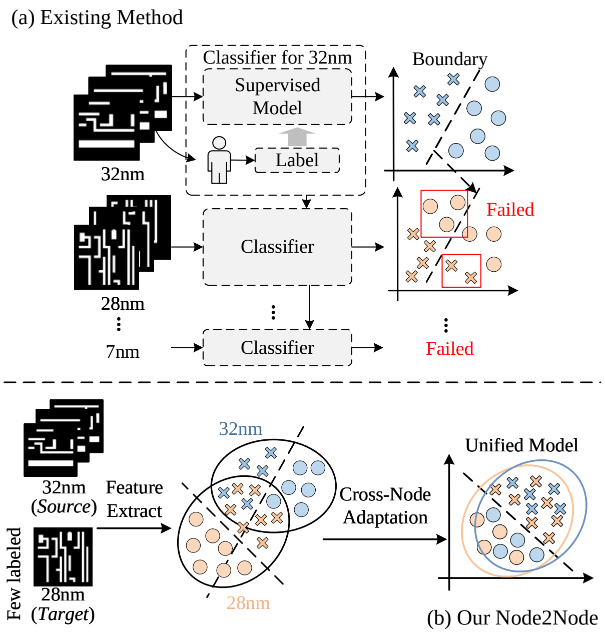

Fig. 1. The illustration of the hotspot detection flow. (a) The pipeline of existing supervised methods. (b) The pipeline of our proposed Node2Node, which is the first node adaptation framework designed explicitly for cross-node hotspot detection.

However, as shown in Fig. 1(a), existing deep learning-based approaches require supervised training with large amounts of labeled hotspot data. These models are typically trained on a single process node and suffer from a severe performance drop when deployed on different process nodes due to differences in layout geometry, manufacturing processes, and optical effects, to the point where they may fail to effectively identify hotspots. This cross-node generalization problem remains a major challenge in hotspot detection.

Although deep learning-based hotspot detection has achieved remarkable progress, its application across different process nodes remains constrained due to the following chal-

*These authors contributed equally to this work.
†Corresponding Author: nzou@nju.edu.cn.

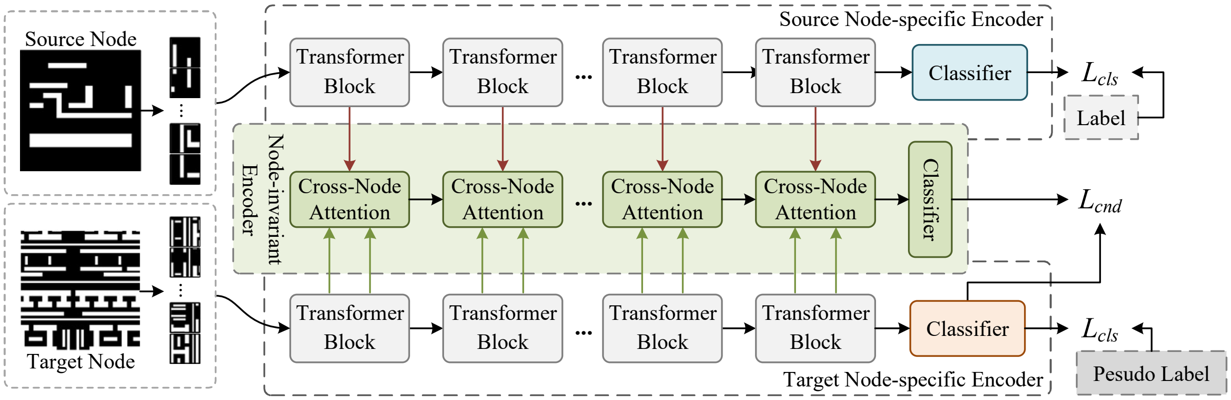

Fig. 2. The framework of our proposed Node2Node. The framework consists of three branches: the source Node-Invariant Encoder branch, the Node-Specific Encoder branch and the target Node-Invariant Encoder branch.

lenges. (1) expensive labeling cost: each new process node requires the collection and annotation of a large amount of training data, with high-quality hotspot annotations typically relying on simulation tools or expert manual analysis, which incurs significant costs. (2) cross-node distribution shift: due to considerable differences in layout features across different process nodes, such as metal spacing, line width, and manufacturing tolerances, directly testing existing models on a new process node often leads to significant performance degradation. (3) feature misalignment: convolutional neural networks mainly rely on local spatial information. However, hotspot features across different process nodes may vary in scale and shape, making it difficult for the feature space of a pre-trained model to align effectively with the target node data. Although traditional Transformers have global modeling capabilities, they struggle with cross-domain adaptation under unsupervised conditions, resulting in insufficient generalization ability.

To overcome these challenges, we propose Node2Node as shown in Fig. 1(b), a novel node adaptation framework for cross-node hotspot detection that effectively learns node-invariant representations while preserving node-specific characteristics. The main contributions of this paper are summarized as follows:

- We first propose a node adaptation framework designed explicitly for cross-node hotspot detection, which can provide significant assistance for the manufacturing of the advanced process nodes in the future.
- We design a novel tri-stream transformer-based architecture consisting of two node-invariant branches and a node-specific branch to effectively capture both node-invariant and node-specific features, enhancing model adaptability for cross-node hotspot detection.
- We propose a bidirectional center alignment strategy to improve pseudo-label quality by dynamically aligning the feature distributions of source and target nodes.
- We design a cross-node distribution loss to explicitly minimize the distribution gap between different process nodes, ensuring better feature alignment and improved hotspot detection accuracy.

II. Preliminary

In the chip manufacturing process, all patterns must be transferred onto the silicon wafer through steps such as photolithography. However, due to various physical variations during the manufacturing process, some patterns may experience deviations, resulting in defects. These defects can cause failures in the circuitry, thereby affecting the chip's functionality. The design areas that are prone to failure due to process variations during photolithography are referred to as hotspots.

In semiconductor manufacturing, different process nodes introduce unique variations in layout patterns due to differences in lithographic process conditions. These variations pose a significant challenge for hotspot detection models, as a model trained on one process node may not generalize well to another. Traditional supervised hotspot detectors [5]–[10], [12], [13] relies on labeled data for a specific node, but obtaining labeled data for every new node is expensive and time-consuming.

To address this, we focus on cross-node hotspot detection, where a model trained on a source node must adapt to a target node without direct supervision. The goal is to correctly identify hotspots in the target node while minimizing false alarms. To evaluate the performance, we define the following metrics.

Definition 1 (Accuracy). The ratio between the number of correctly detected hotspots in the target node and the number of ground truth hotspots in the target node.

Definition 2 (False Alarm). The proportion of non-hotspot samples in the target node that are incorrectly predicted as hotspots by the classifier.

With these metrics, we formulate the cross-node hotspot detection problem as follows:

Problem 1 (Cross-Node Hotspot Detection). Given a labeled source node dataset and a few labeled target node dataset, the objective is to train a detector that maximizes the Accuracy in the target node while minimizing the False Alarm, despite the distribution shift between process nodes.

III. Methodology

A. Overview

As shown in Fig. 2, the cores of our proposed Node2Node are the three branches: the source Node-Invariant Encoder branch, the Node-Specific Encoder branch and the target Node-Invariant Encoder branch. The architecture follows a teacher-student framework, where the Node-Specific branch and the target Node-Invariant branch serve as the teacher and the target Node-Invariant branch acts as the student. The cross-node attention mechanism in the Node-Specific branch helps the target branch learn knowledge from the source node, enhancing the model's cross-node learning capability.

The model input consists of data pairs processed by bidirectional center alignment strategy. The classifier is trained using both classification loss and cross-node distribution loss, ensuring effective hotspot classification and improving the model's generalization ability.

B. Bidirectional Center Alignment

This section details our proposed bidirectional center alignment method, which includes the bidirectional feature matching strategy and the center alignment and pseudo-label refinement mechanism

Bidirectional Feature Matching. In the task of cross-node hotspot detection, significant discrepancies exist in data distribution across different process nodes. A unidirectional matching approach may lead to distribution shift, thereby impairing the quality of pseudo-label. To mitigate this issue, we introduce bidirectional feature matching, which establishes bidirectional correspondences between the most similar sample pairs in the source node and the target node. This strategy ensures that the training samples comprehensively cover the data distribution of the target.

Specifically, given a source node dataset $\mathcal{S}$ and a target node dataset $\mathcal{T}$, we first find the most similar samples t^* in the target to those in the source to get the source-to-target data pair $\mathcal{P}_{\mathcal{S}\to\mathcal{T}}$:

$$\mathcal{P}_{\mathcal{S}\to\mathcal{T}} = \{(s,t^*) \mid t^* = \arg\min_{t\in\mathcal{T}} D\left(m_s, m_t\right), \forall s \in \mathcal{S}\}, \quad (1)$$

where s and t are the source node and target node data, m_i represents the feature of the image i, $D\left(m_i, m_j\right)$ represents the feature distance between image i and j.

This strategy makes full use of the source node data, but its disadvantage is that it may lead to an imbalanced matching of samples in the target node. To further reduce this bias, we perform reverse matching, where we find the most similar source sample s^* for each target sample in the target node to get the target-to-source data pair $\mathcal{P}_{\mathcal{T}\to\mathcal{S}}$:

$$\mathcal{P}_{\mathcal{T}\to\mathcal{S}} = \{(s^*,t) \mid s^* = \arg\min_{s\in\mathcal{S}} D\left(m_s, m_t\right), \forall t \in \mathcal{T}\}. \quad (2)$$

Finally, our data sample pairs are jointly composed of the following:

$$\mathcal{P} = \mathcal{P}_{\mathcal{S}\to\mathcal{T}} \cup \mathcal{P}_{\mathcal{T}\to\mathcal{S}}. \quad (3)$$

In this way, we can maximize the utilization of both source node and target node data, enhancing the performance of cross-node learning.

Center Alignment and Pseudo-label Refinement. Due to the reliance of bidirectional feature matching on feature similarity, and the potential distribution shift in feature space during cross-node tasks, directly using the bidirectional feature matching may introduce significant pseudo-label errors. To address this, we further introduce the center alignment method to optimize pseudo-labels and improve the reliability of matching.

Inspired by [16], we find that the pre-trained model of the source data is also useful to further improve the accuracy.

First, we predict the class distribution of target node samples using the pre-trained model from the source node:

$$p_k^t = P(y_t = k \mid m^t), \quad (4)$$

where p_k^t represents the probability that the target sample t belongs to class k, m^t denotes the feature representation of target sample t, extracted by the network.

Based on this, we compute the initial class center c_k^0 of class k in the target node using a weighted mean:

$$c_k^0 = \frac{\sum_{t\in\mathcal{T}} p_k^t m^t}{\sum_{t\in\mathcal{T}} p_k^t}. \quad (5)$$

To improve the reliability of pseudo-labels, we incorporate 10% of the labeled target node data, denoted as $\mathcal{T}_L$. The labeled target samples are used to refine the class centers by combining them with the pseudo-labeled samples. The refined class center c_k^0 is computed as:

$$c_k^0 = \frac{\sum_{t\in\mathcal{T}} p_k^t m^t + \sum_{t\in\mathcal{T}_L} 1(y_t = k) m^t}{\sum_{t\in\mathcal{T}} p_k^t + \sum_{t\in\mathcal{T}_L} 1(y_t = k)}, \quad (6)$$

where $1(y_t = k)$ is an indicator function that denotes whether the labeled target sample t belongs to class k.

After obtaining the class centers, we assign pseudo-labels based on the nearest center:

$$\hat{y}_t = \arg\min_k D(c_k, m^t), \quad (7)$$

where $D(c_k, m^t)$ measures the cosine distance between the sample m^t and the class center c_k.

To further reduce pseudo-label errors, we iteratively update the class centers by incorporating both pseudo-labeled and labeled target samples:

$$c_k^1 = \frac{\sum_{t\in\mathcal{T}} 1(\hat{y}_t = k) m^t + \sum_{t\in\mathcal{T}_\mathcal{L}} 1(y_t = k) m^t}{\sum_{t\in\mathcal{T}} 1(\hat{y}_t = k) + \sum_{t\in\mathcal{T}_\mathcal{L}} 1(y_t = k)}. \quad (8)$$

Finally, we introduce a consistency filtering mechanism: a target sample is retained only if its pseudo-label matches the ground-truth label of its most similar source sample; otherwise, it is discarded:

$$\mathcal{T}^* = \{t \in \mathcal{T} \mid \hat{y}_t = y_p, (s,t) \in \mathcal{P}\}. \quad (9)$$

where y_p denotes the label of the source-target pair $(s,t) \in \mathcal{P}$.

This ensures that the model utilizes only high-confidence target node samples for training, while also leveraging the labeled target data to improve robustness and accuracy.

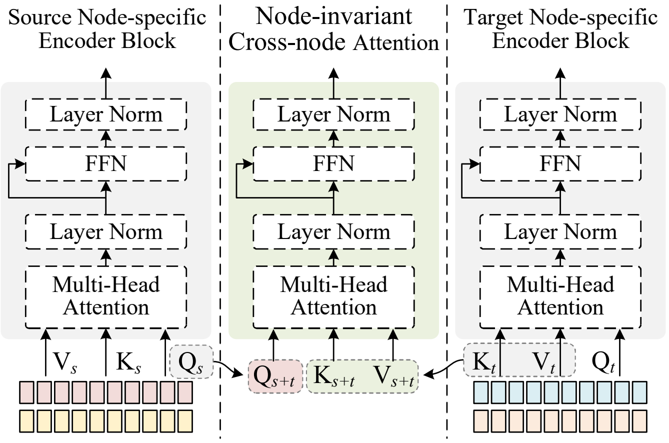
Source Node-specific Encoder Block
Node-invariant Cross-node Attention
Target Node-specific Encoder Block
Layer Norm
FFN
Layer Norm
Multi-Head Attention
V_s
K_s
Q_s
Q_{s+t}
K_{s+t}
V_{s+t}
K_t
V_t
Q_t

Fig. 3. Illustration of the Node-Specific Encoder and Node-Invariant Encoder.

C. Model Structure in Node2Node

Backbone. Different from previous works like [6], [8], we don't adapt the cnn-based architecture which rely on local receptive fields like ResNet [17] as the backbone of our model Node2Node. In contrast, we employ DeiT [18] (Transformer-based) as the backbone feature extractor, making it well-suited for understanding spatial correlations in layouts. Most importantly, DeiT integrates distillation tokens to improve data efficiency, making it more effective in the situation of cross-node hotspot detection where labelled data is seriously insufficient while maintaining the powerful representation learning capability.

The Node-Specific Encoder. In our model, there are two parallel node-specific encoder branches for the source and target nodes, respectively. Both are implemented as similar Transformer-based encoders and play a crucial role in capturing global dependencies and spatial relationships.

As shown in Fig. 3, the node-specific encoder block consists of a multi-head self-attention module and a feed-forward network (FFN). The multi-head self-attention mechanism allows the model to capture both local and global context by attending to various parts of the layout simultaneously. This is important for identifying hotspots that may not be immediately adjacent but are contextually significant across the entire layout. To incorporate spatial information, we add fixed positional encodings to the input of each attention layer. These positional encodings help the self-attention mechanism account for the layout's structure, ensuring the model understands the spatial relationships relevant to hotspot classification.

The node-specific encoder processes the feature sequence in parallel, enabling efficient computation while maintaining the ability to detect complex hotspot patterns at different scales. The output from the encoder, enriched with both local features and global context, is directly used for the final classification of hotspots.

The Node-Invariant Encoder. It is designed to model the interactions between the source and target branches by aligning their features across nodes. Like the node-specific encoder, the node-invariant encoder leverages the transformer-based architecture, but its key role lies in enabling information flow between different branches of the model (source and target), allowing the network to learn cross-node dependencies.

TABLE I
Benchmark Information

Benchmark	Data Source	of Hotspots	of Clips #HS	 #NHS
32nm	ICCAD2012-1	325	1950	3954
28nm	ICCAD2012-5	67	67	4827
7nm	ICCAD2016-2	79	50	1000
	ICCAD2016-3	2821	2807	1000
	ICCAD2016-4	162	140	2000

As shown in Fig. 3, the node-invariant encoder incorporates a multi-head cross-attention mechanism and a feed-forward network. The cross-attention mechanism works by attending to both the source and target features simultaneously. It generates contextually aligned representations by computing attention scores between the source and target branches, allowing the model to learn which parts of the source node are most relevant for the target node.

In the node-invariant encoder, the source branch features are used as queries, and the target branch features serve as keys and values. By attending to these features, the cross-attention mechanism generates a unified representation that captures relevant cross-node information.

Classifier. The classifier is implemented as a standard fully connected network layer without bias. It's worth noting that all three branches (source branch, target branch, and source-target branch) share the same classifier due to the same label of two images.

D. Loss Function

Different from previous hotspot detectors, our proposed model Node2Node includes not only the standard classification loss but also the cross-node distribution loss, ensuring that the target branch can benefit from knowledge transfer from the source branch.

The cross-node distribution loss is introduced to facilitate knowledge transfer between the source and target branches. This loss function allows the target branch as the student to learn from the source-target branch as the teacher. We consider the probability distribution of the classifier in the source-target branch as a soft label that can be used to further supervise the target branch. The Cross-Node Distribution Loss is computed as:

$$\mathcal{L}_{cnd} = \sum_{k} q_k \log p_k, \tag{10}$$

where q_k and p_k are the probabilities of category k from the source-target branch as the teacher and the target branch as the student.

Therefore, the total loss is as follows:

$$\mathcal{L} = \mathcal{L}_{cnd} + \mathcal{L}_{cls}, \tag{11}$$

where $\mathcal{L}_{cls}$ is the cross-entropy loss.

TABLE II
Comparison with Generic Vision Methods.

Benchmark	ResNet [17]		MobileNetV2 [19]		HRNet [20]		EfficientNet [21]		Ours	
	Acc(%)	FA(%)	Acc(%)	FA(%)	Acc(%)	FA(%)	Acc(%)	FA(%)	Acc(%)	FA(%)
32nm→28nm	**97.01**	62.21	49.25	48.81	97.01	62.50	38.81	48.46	95.52	**8.14**
32nm→7nm	82.80	55.13	47.36	39.38	**96.69**	95.83	46.83	43.33	93.12	**24.32**
28nm→32nm	4.26	1.26	0	**0**	10.62	0.38	0	**0**	**90.00**	13.32
28nm→7nm	9.05	**0**	0.73	5.80	17.10	30.28	6.21	1.70	**93.69**	24.97
7nm→32nm	48.97	35.66	36.56	33.59	95.18	99.29	2.31	**12.75**	**99.74**	23.97
7nm→28nm	2.99	41.35	94.03	97.95	16.42	61.82	0	31.26	**95.52**	**21.95**
Avg	40.85	32.60	37.99	37.59	55.50	58.35	15.69	22.92	**94.60**	**19.45**

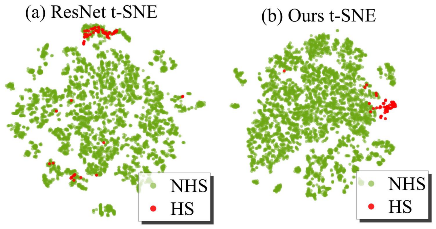

Fig. 4. Qualitative comparison of ResNet and Node2Node on 32nm→28nm. It is evident that our Node2Node can better distinguish the feature embeddings of hotspots and non-hotspots.

TABLE III
Comparison on 32nm with state-of-the-art supervised methods.

Method	Method Type	Acc(%)	FA#
JM3'2016 [24]	Fully Labeled Target	95.10	386
ICCAD'2016 [25]	Fully Labeled Target	**100**	788
JM3'2017 [11]	Fully Labeled Target	**100**	1037
TODAES'2019 [26]	Fully Labeled Target	**100**	1398
TCAD'2021 [9]	Fully Labeled Target	99.56	401
IWAPS'2022 [27]	Fully Labeled Target	99.60	1761
DAC'2024 [13]	Fully Labeled Target	99.74	197
Ours	Few Labeled Target	99.74	**161**

Note: The theoretical upper bound of our node adaptation method cannot surpass the upper bound of fully supervised methods that utilize target node labels.

IV. Experimental Results

Our experiments are all implemented using PyTorch, with all models trained on AMD EPYC 9554 64-core CPUs and 8 NVIDIA GeForce RTX A6000 GPUs (48 GB memory) for accelerated computation.

The proposed framework was evaluated on standard benchmarks across three process nodes (7 nm, 28 nm, and 32 nm).

The 32 nm and 28 nm datasets were derived from the case-1 and case-5 of the ICCAD-2012 benchmark [22]. We applied data augmentation using geometric transformations. Since hotspot identification remains unaffected by rotations and mirror flips, we performed 90°, 180°, and 270° rotations along with vertical and horizontal mirroring to balance the sample distribution across different nodes.

The 7nm dataset originates from the ICCAD-2016 benchmark [23]. Notably, our experiments on the 7nm dataset incorporate random cropping, 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. More comprehensive details about the dataset are shown in TABLE I.

A. Comparisons with Existing General Methods

To evaluate the cross-node generalization capability of different models, we conduct a series of node adaptation experiments across multiple process nodes. Each experiment follows a node adaptation setting, where one process node is treated as the source node (with labeled data) and another as the target node (with few labeled data). Therefore, the setting indicates that we use source node data and target node data during training. The model is trained under this source-target pairing, and the final performance is evaluated exclusively on the test set of target node. This setting mimics a practical scenario in semiconductor manufacturing where obtaining a large amount of data from advanced nodes is challenging.

As shown in Table II, we repeat this setup across six various node pairs, including 32nm→28nm, 7nm→28nm, and so on, to comprehensively assess model robustness under different node gaps. Table II presents a comparison between our proposed method and several generic vision baselines, including ResNet [17], MobileNetV2 [19], HRNet [20] and EfficientNet [21], evaluated on six cross-node hotspot detection benchmarks. Overall, ResNet, MobileNetV2 and HRNet exhibit a certain level of capability in cross-node hotspot detection, while EfficientNet demonstrate almost no ability to generalize across nodes. In contrast, our method achieves the best performance across all benchmarks, with an average accuracy of 94.60% and an average false alarm rate of only 19.45%, demonstrating strong cross-node generalization ability. Furthermore, we present a qualitative comparison between a representative generic vision model (ResNet) and our proposed Node2Node framework, as illustrated in Fig. 4. The results demonstrate that the Node2Node more effectively differentiates between hotspot and non-hotspot feature embeddings, outperforming existing general vision baselines in distinguishing hotspots from non-hotspots.

B. Comparisons with Fully Supervised Methods

We compare our method with state-of-the-art learning-based supervised approaches on 32nm, 28nm and 7nm technology

TABLE IV
COMPARISON ON 28NM WITH STATE-OF-THE-ART SUPERVISED METHODS.

Method	Method Type	Acc(%)	FA#
JM3'2016 [24]	Fully Labeled Target	92.70	172
ICCAD'2016 [25]	Fully Labeled Target	95.10	94
JM3'2017 [11]	Fully Labeled Target	95.10	394
DAC'2017 [5]	Fully Labeled Target	95.10	95
JM3'2019 [28]	Fully Labeled Target	97.30	80
TODAES'2019 [26]	Fully Labeled Target	95.12	43
TCAD'2019 [5]	Fully Labeled Target	95.12	95
TCAD'2021 [9]	Fully Labeled Target	95.24	**24**
DAC'2021 [29]	Fully Labeled Target	97.60	34
DAC'2024 [13]	Fully Labeled Target	97.56	395
Ours	Few Labeled Target	**98.51**	393

Note: The theoretical upper bound of our node adaptation method cannot surpass the upper bound of fully supervised methods that utilize target node labels.

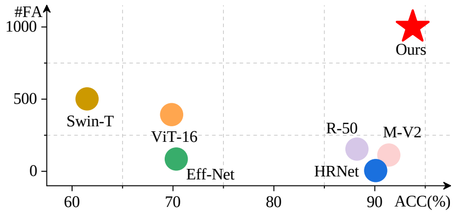

Fig. 5. Comparison on 7nm with state-of-the-art supervised methods.

nodes, in order to evaluate the effectiveness of our model against fully supervised counterparts under different process settings. The supervised methods are trained and tested using labeled data, while our method is evaluated in a setting with access to only few labeled data during training. Meanwhile, to better demonstrate the performance of our model, the final experimental results are reported based on the best performance achieved across multiple settings with 10%-30% of labeled target data.

Table III presents a comparison between our method and the state-of-the-art supervised methods on 32nm node. Our method achieves a competitive accuracy of 99.74%, which is slightly lower than ICCAD'2016 [25], JM3'2017 [11] and TODAES'2019 [26]. It's worth noting that our method achieves the lowest FA among all methods, including the current state-of-the-art model DAC'2024 [13].

Table IV compares our method with state-of-the-art supervised methods on 28nm node. Our method achieves an optimal accuracy of 98.51%. Although our method has a gap in FA compared to the lowest FA method TCAD'2021 [9], we stand out among all models when considering both accuracy and FA.

Fig. 5 further illustrates the comparison between our method and representative supervised generic vision models on the challenging 7nm node. Our Node2Node achieves 93.75% accuracy, outperforming all supervised methods including ResNet [17], Mobilenet-V2 [19], ViT [30], Swin-Transformer [31], EfficientNet [21] and HRNet [20]. Although our FA on 28nm and 32nm is higher than most supervised models, it remains within a reasonable range given that our method does require few labeled data from the target node.

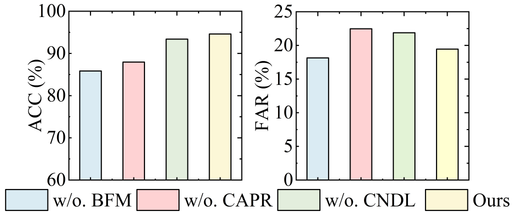

Fig. 6. Comparison among different configurations on average accuracy and average false alarm.

Although the theoretical upper bound of our node adaptation framework cannot surpass that of supervised methods, our results still demonstrate that our approach achieves competitive performance with supervised methods across all comparisons on the 32nm, 28nm, and 7nm nodes.

C. Ablation Study

To evaluate the contribution of each component in our Node2Node framework, we conduct ablation studies by incrementally removing key modules: Bidirectional Feature Matching, Center Alignment and Pseudo-label Refinement, and Cross-Node Distribution Loss. The results are presented in Fig. 6. In the figure, we use BFM, CAPR, and CNDL to represent the three proposed key modules, respectively. When removing bidirectional feature matching, the average accuracy drops by 8.76%, indicating that bidirectional matching plays a crucial role in establishing reliable source-target correspondences. Removing center alignment and pseudo-label refinement leads to a 6.65% decrease in accuracy and a 3.02% increase in false alarm rate, demonstrating the importance of dynamic pseudo-label refinement in improving both precision and robustness. Excluding cross-node distribution loss results in a 1.21% drop in accuracy and a 2.43% increase in false alarm rate, confirming that explicitly regularizing cross-node feature alignment helps the model better generalize under domain shift. In summary, all three components make complementary contributions. The complete model achieves the best overall performance, improving accuracy by up to 8.76% compared to the weakest configuration, while maintaining a low false alarm rate. This validates the effectiveness of our design in tackling cross-node hotspot detection.

V. CONCLUSION

We proposed Node2Node, the first node adaptation framework that explicitly addresses the problem of cross-node hotspot detection and significantly reduces the reliance on labeled data from the target node. By introducing a tri-stream Transformer-based architecture with Node-invariant and Node-specific encoders, along with a bidirectional center alignment strategy and cross-node distribution loss, Node2Node effectively bridges the domain gap between process nodes. Extensive experiments demonstrate its superior generalization and detection performance, highlighting its potential for practical deployment in advanced semiconductor manufacturing.

References

[1] S.-Y. Lin, J.-Y. Chen, J.-C. Li, W.-Y. Wen, and S.-C. Chang, "A novel fuzzy matching model for lithography hotspot detection," in *Proceedings of the 50th Annual Design Automation Conference*, 2013, pp. 1–6.

[2] 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 Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 11, pp. 1671–1680, 2014.

[3] 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.

[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 *Design-process-technology Co-optimization for Manufacturability IX*, vol. 9427. SPIE, 2015, pp. 201–211.

[5] H. Yang, J. Su, Y. Zou, B. Yu, and E. F. Young, "Layout hotspot detection with feature tensor generation and deep biased learning," in *Proceedings of the 54th Annual Design Automation Conference 2017*, 2017, pp. 1–6.

[6] R. Chen, W. Zhong, H. Yang, H. Geng, X. Zeng, and B. Yu, "Faster region-based hotspot detection," in *Proceedings of the 56th Annual Design Automation Conference 2019*, 2019, pp. 1–6.

[7] H. Yang, P. Pathak, F. Gennari, Y.-C. Lai, and B. Yu, "Detecting multi-layer layout hotspots with adaptive squish patterns," in *Proceedings of the 24th Asia and South Pacific Design Automation Conference*, 2019, pp. 299–304.

[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 *Proceedings of the 39th International Conference on Computer-Aided Design*, 2020, pp. 1–8.

[9] T. Gai, T. Qu, S. Wang, X. Su, R. Xu, Y. Wang, J. Xue, Y. Su, Y. Wei, and T. Ye, "Flexible hotspot detection based on fully convolutional network with transfer learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 11, pp. 4626–4638, 2021.

[10] H.-C. Shao, G.-Y. Chen, Y.-H. Lin, C.-W. Lin, S.-Y. Fang, P.-Y. Tsai, and Y.-H. Liu, "Lithohod: A litho simulator-powered framework for ic layout hotspot detection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2024.

[11] H. Yang, L. Luo, J. Su, C. Lin, and B. Yu, "Imbalance aware lithography hotspot detection: a deep learning approach," *Journal of Micro/Nanolithography, MEMS, and MOEMS*, vol. 16, no. 3, pp. 033 504–033 504, 2017.

[12] B. Zhu, R. Chen, X. Zhang, F. Yang, X. Zeng, B. Yu, and M. D. Wong, "Hotspot detection via multi-task learning and transformer encoder," in *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. IEEE, 2021, pp. 1–8.

[13] Y. Chen, Y. Wu, J. Wang, T. Wu, X. He, J. Yu, and H. Geng, "Llm-hd: Layout language model for hotspot detection with gds semantic encoding," in *Proceedings of the 61st ACM/IEEE Design Automation Conference*, 2024, pp. 1–6.

[14] W. Xu, S. Chen, J. Li, K. Di, Y. Fu, and N. Zou, "When transformer meets layout hotspot: An end-to-end transformer-based detector with prior lithography," in *Proceedings of the Great Lakes Symposium on VLSI 2025*, 2025, pp. 612–618.

[15] Y. Chen, Q. Sun, S. Zheng, X. Zhang, B. Yu, and H. Geng, "Hydas: Hybrid domain deformed attention for selective hotspot detection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2025.

[16] J. Liang, D. Hu, and J. Feng, "Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation," in *International conference on machine learning*. PMLR, 2020, pp. 6028–6039.

[17] 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.

[18] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *International conference on machine learning*. PMLR, 2021, pp. 10 347–10 357.

[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] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 5693–5703.

[21] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114.

[22] 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.

[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] M. Shin and J.-H. Lee, "Accurate lithography hotspot detection using deep convolutional neural networks," *Journal of Micro/Nanolithography, MEMS, and MOEMS*, vol. 15, no. 4, pp. 043 507–043 507, 2016.

[25] H. Zhang, B. Yu, and E. F. Young, "Enabling online learning in lithography hotspot detection with information-theoretic feature optimization," in *2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2016, pp. 1–8.

[26] X. He, Y. Deng, S. Zhou, R. Li, Y. Wang, and Y. Guo, "Lithography hotspot detection with fft-based feature extraction and imbalanced learning rate," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 25, no. 2, pp. 1–21, 2019.

[27] M. Lin, F. Zeng, and Y. Shen, "Lithography hotspot detection with resnet network," in *2022 International Workshop on Advanced Patterning Solutions (IWAPS)*. IEEE, 2022, pp. 1–4.

[28] J. Chen, Y. Lin, Y. Guo, M. Zhang, M. B. Alawieh, and D. Z. Pan, "Lithography hotspot detection using a double inception module architecture," *Journal of Micro/Nanolithography, MEMS, and MOEMS*, vol. 18, no. 1, pp. 013 507–013 507, 2019.

[29] B. Wang, L. Jiang, W. Zhu, L. Guo, J. Chen, and Y.-W. Chang, "Two-stage neural network classifier for the data imbalance problem with application to hotspot detection," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2021, pp. 175–180.

[30] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[31] 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.