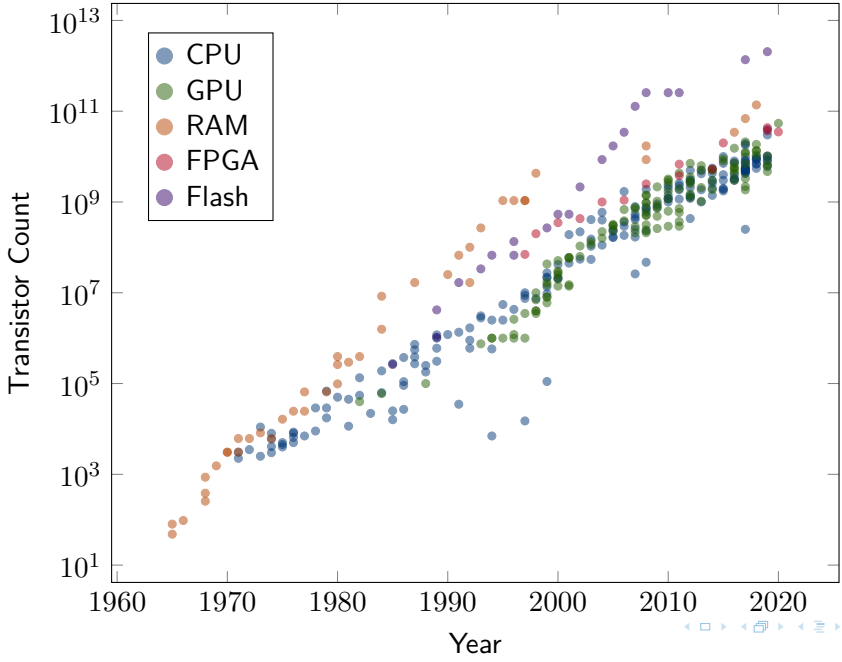
香港中文大學
The Chinese University of Hong Kong

VLSI Mask Optimization: From Shallow To Deep Learning

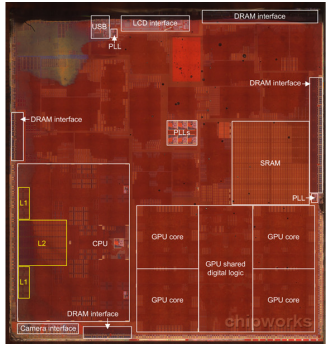
Bei Yu

Department of Computer Science and Engineering
The Chinese University of Hong Kong

Moore's Law to Extreme Scaling

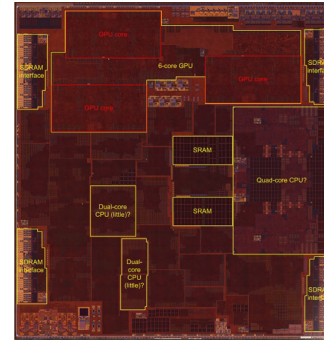


Scaling of Apple SOC



Apple A7 (2013)

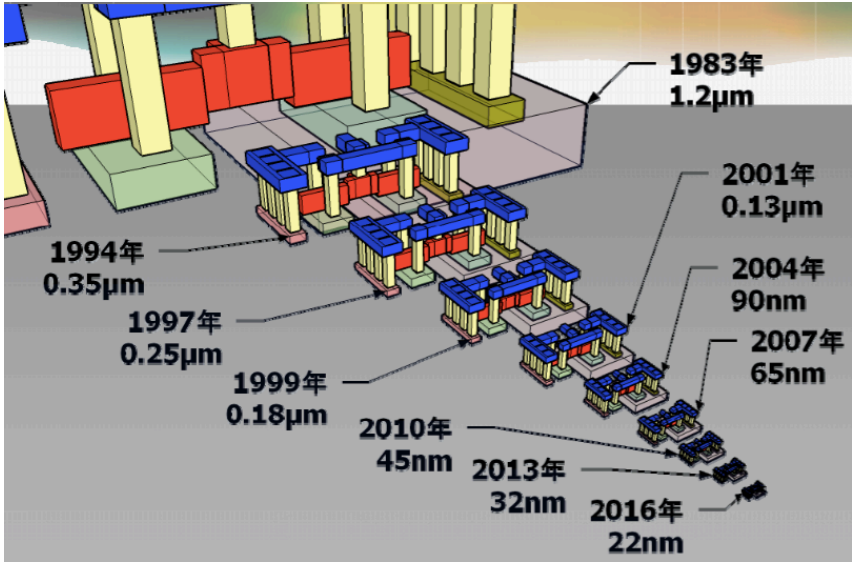
- 1,000,000,000 Transistors
- $102mm^2$ die size
- 1.3GHz



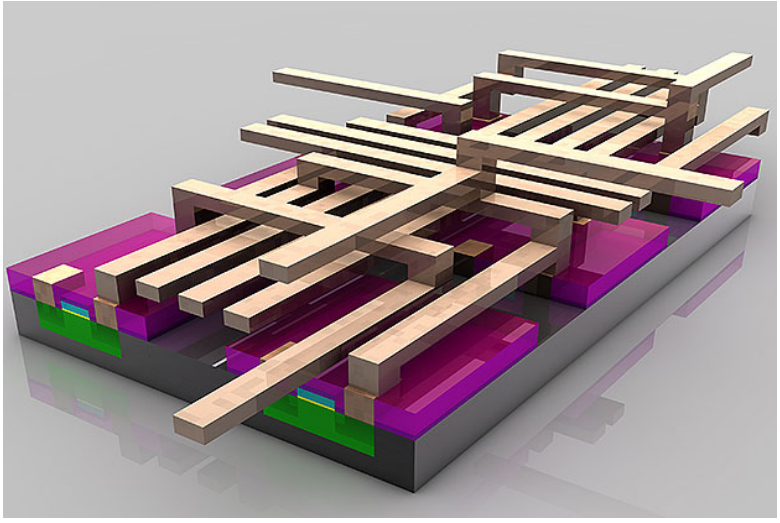
Apple A10 (2016)

- 3,300,000,000 Transistors
- $125mm^2$ die size
- 2.34GHz

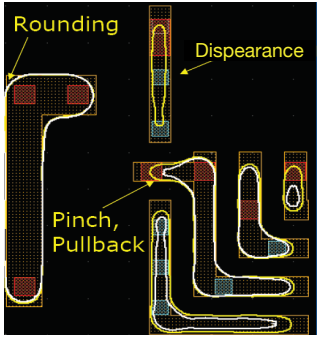
An Inverter Example

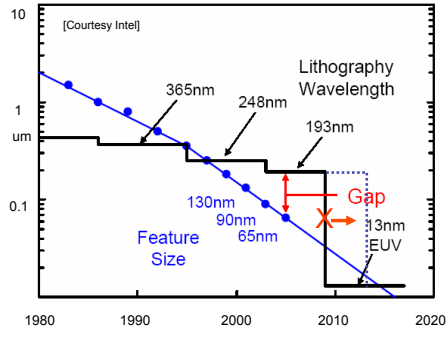


Detailed View of Layout



Manufacturing Issues of Layout





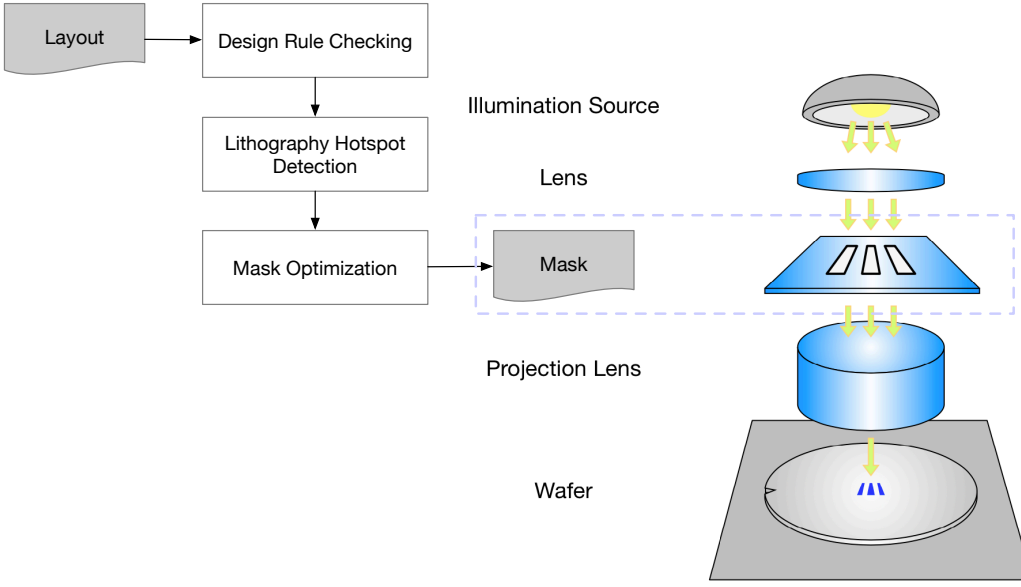
Manufacturability Status & Challenges

Mainstream Tools

Market Anually: USD 100M!

1. `Calibre` by Mentor Graphics
2. `Brion Tool` by ASML
3. `IC Validator` by Synopsys
4. `Pegasus` by Cadence

From Layout to Chip



Outline

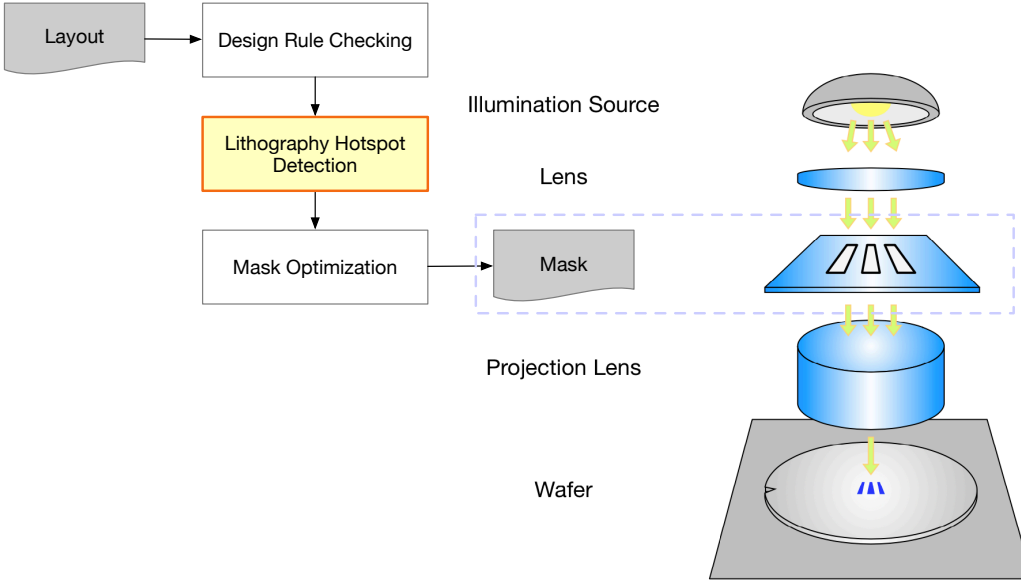
Task 1: Hotspot Detection

Task 2: Mask Optimization

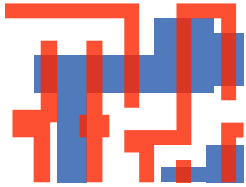
Outline

Task 1: Hotspot Detection

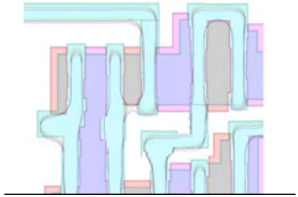
Task 2: Mask Optimization


Layout
Design Rule Checking
Lithography Hotspot Detection
Mask Optimization
Mask
Illumination Source
Lens
Projection Lens
Wafer

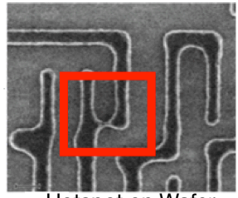
Challenge: Failure (Hotspot) Detection



Pre-OPC Layout

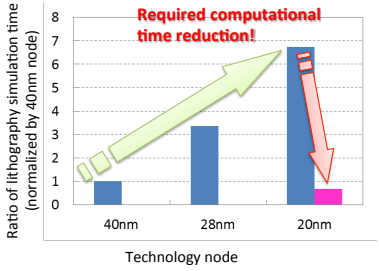


Post-OPC Mask

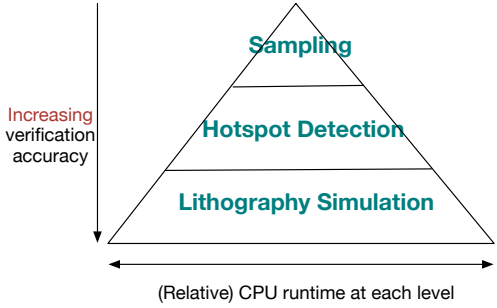


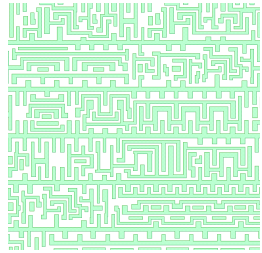
Hotspot on Wafer

- RET: OPC, SRAF, MPL
- Still hotspot: low fidelity patterns
- Simulations: extremely CPU intensive



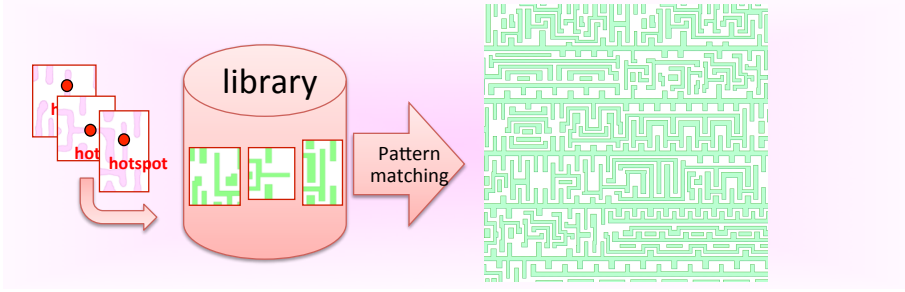
Hotspot Detection Hierarchy



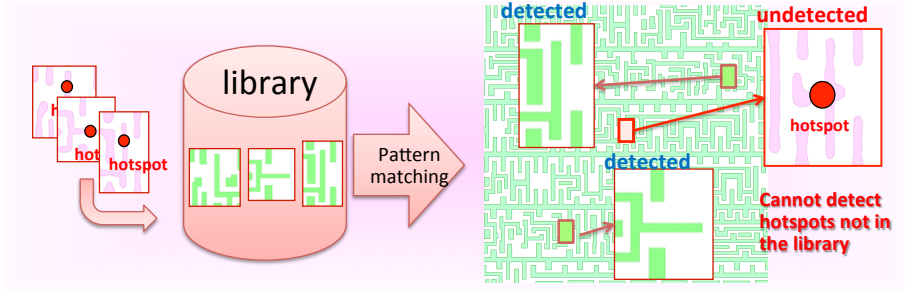


- **Sampling** (DRC Checking):
  scan and rule check each region
- **Hotspot Detection**:
  verify the sampled regions and report potential hotspots
- **Lithography Simulation**:
  final verification on the reported hotspots

Pattern Matching based Hotspot Detection

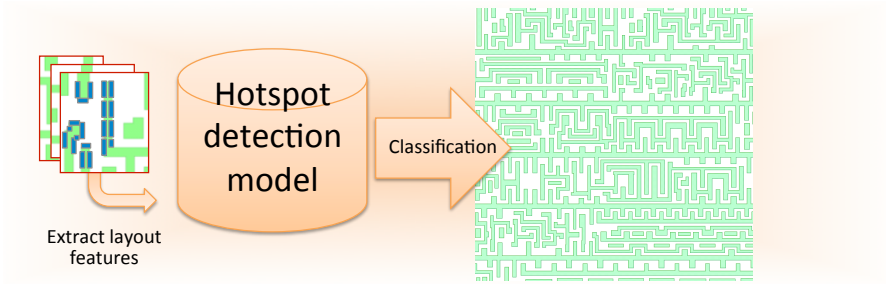


Pattern Matching based Hotspot Detection

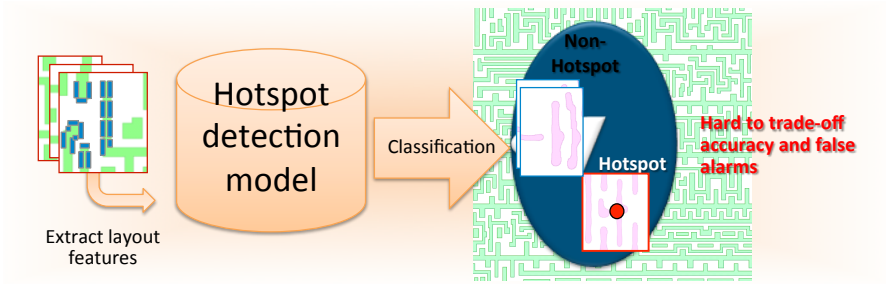


- Fast and accurate
- [Yu+,ICCAD'14] [Nosato+,JM3'14] [Su+,TCAD'15]
- Fuzzy pattern matching [Wen+,TCAD'14]
- Hard to detect non-seen pattern

Classification based Hotspot Detection



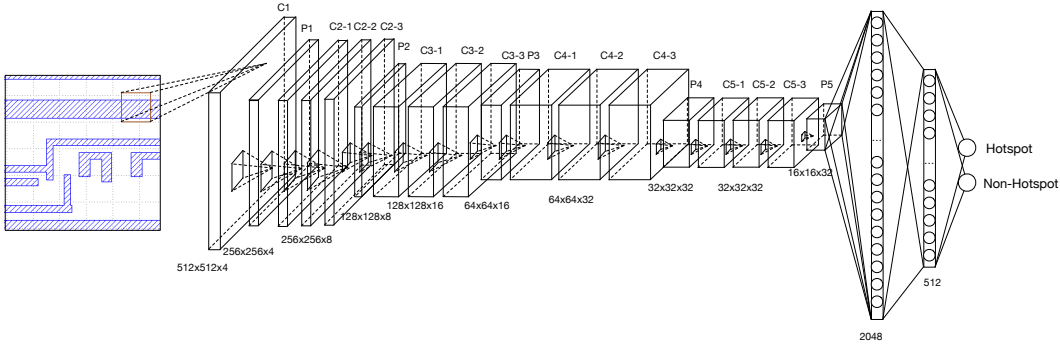
Classification based Hotspot Detection



- Predict new patterns
- Decision-tree, ANN, SVM, Boosting ...
- [Drmanac+,DAC'09] [Ding+,TCAD'12] [Yu+,JM3'15] [Matsunawa+,SPIE'15] [Yu+,TCAD'15]
- Hard to balance accuracy and false-alarm

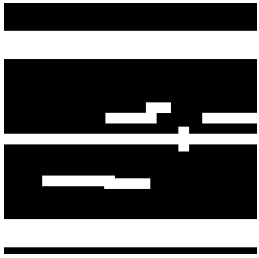
First DNN Hotspot Detection Architecture[1]

- Total 21 layers with 13 convolution layers and 5 pooling layers.
- A ReLU is applied after each convolution layer.

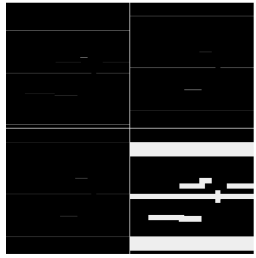


[1] Haoyu Yang, Luyang Luo, et al. (2017). "Imbalance aware lithography hotspot detection: a deep learning approach". In: *JM3* 16.3, p. 033504.

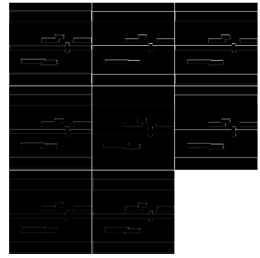
Layer Visualization



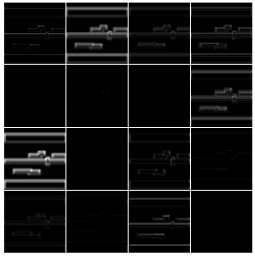
Origin



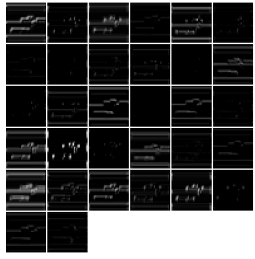
Pool1



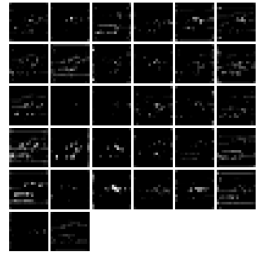
Pool2



Pool3

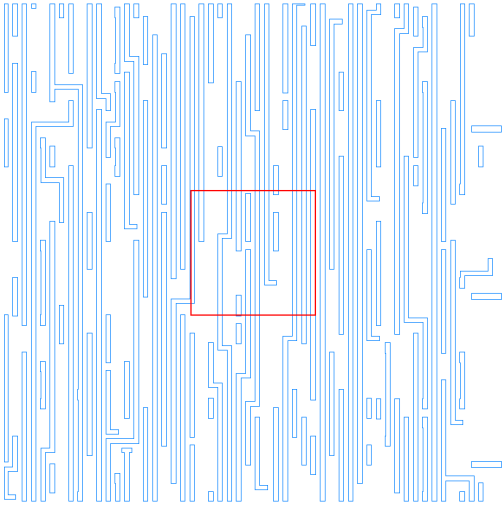


Pool4

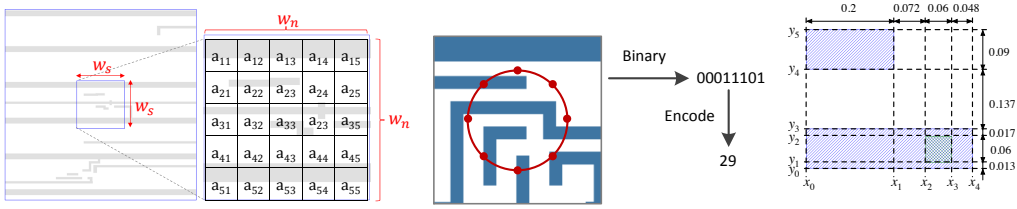


Pool5

Rethinking: “ImageHotspot” v.s. ImageNet



HSD-Research: New Representation



(a) (b) (c)

- (a) Density-based encoding [SPIE'15][2]
- (b) Concentric circle sampling [ICCAD'16][3]
- (c) Squish pattern [ASPDAC'19][4]

[2] Tetsuaki Matsunawa et al. (2015). "A new lithography hotspot detection framework based on AdaBoost classifier and simplified feature extraction". In: *Proc. SPIE*. vol. 9427.

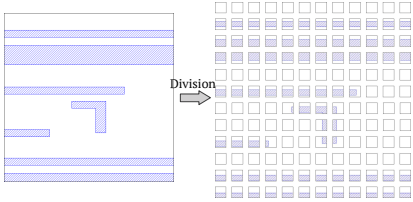
[3] Hang Zhang, Bei Yu, and Evangeline F. Y. Young (2016). "Enabling Online Learning in Lithography Hotspot Detection with Information-Theoretic Feature Optimization". In: *Proc. ICCAD*, 47:1–47:8.

[4] Haoyu Yang, Piyush Pathak, et al. (2019). "Detecting multi-layer layout hotspots with adaptive squish patterns". In: *Proc. ASPDAC*, pp. 299–304.

Simplified CNN Architecture [DAC'17][5]

Feature Tensor Generation:

- Clip Partition
- Discrete Cosine Transform
- Discarding High Frequency Components
- Feature Tensor

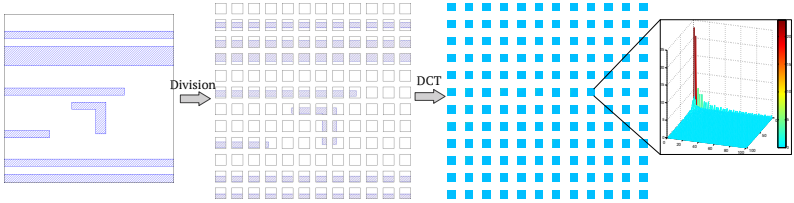


[5] Haoyu Yang, Jing Su, et al. (2017). "Layout Hotspot Detection with Feature Tensor Generation and Deep Biased Learning". In: *Proc. DAC*, 62:1–62:6.

Simplified CNN Architecture [DAC'17][5]

Feature Tensor Generation:

- Clip Partition
- Discrete Cosine Transform
- Discarding High Frequency Components
- Feature Tensor

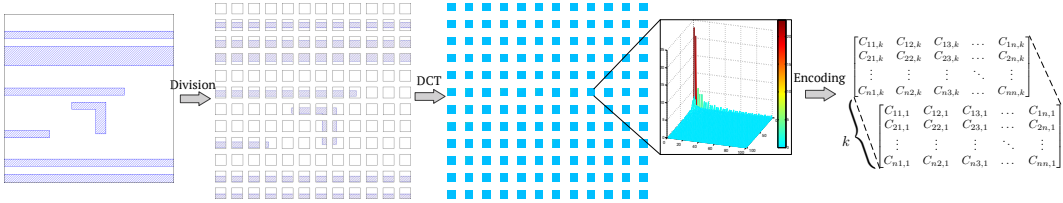


[5] Haoyu Yang, Jing Su, et al. (2017). "Layout Hotspot Detection with Feature Tensor Generation and Deep Biased Learning". In: *Proc. DAC*, 62:1–62:6.

Simplified CNN Architecture [DAC'17][5]

Feature Tensor Generation:

- Clip Partition
- Discrete Cosine Transform
- Discarding High Frequency Components
- Feature Tensor



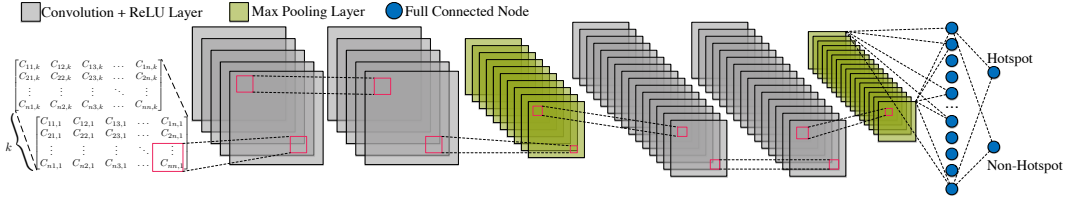
[5] Haoyu Yang, Jing Su, et al. (2017). "Layout Hotspot Detection with Feature Tensor Generation and Deep Biased Learning". In: *Proc. DAC*, 62:1–62:6.

Simplified CNN Architecture [DAC'17]

Layer	Kernel Size	Stride	Output Node #
conv1-1	3	1	$12 \times 12 \times 16$
conv1-2	3	1	$12 \times 12 \times 16$
maxpooling1	2	2	$6 \times 6 \times 16$
conv2-1	3	1	$6 \times 6 \times 32$
conv2-2	3	1	$6 \times 6 \times 32$
maxpooling2	2	2	$3 \times 3 \times 32$
fc1	N/A	N/A	250
fc2	N/A	N/A	2

Feature Tensor

- k-channel hyper-image
- Compatible with CNN
- Storage and computional efficiency



The Biased Learning Algorithm

- Minimize difference with ground truths

$$\mathrm{y}_n^* = [1, 0],\ \mathrm{y}_h^* = [0, 1].$$

$$\mathrm{F} \in \begin{cases} \mathcal{N}, & \text{if } \mathrm{y}(0) > 0.5, \\ \mathcal{H}, & \text{if } \mathrm{y}(1) > 0.5. \end{cases}$$

- Naive: Shifting decision boundary

$$\mathrm{F} \in \begin{cases} \mathcal{N}, & \text{if } \mathrm{y}(0) > 0.5 + \lambda, \\ \mathcal{H}, & \text{if } \mathrm{y}(1) > 0.5 - \lambda. \end{cases}$$

The Biased Learning Algorithm

- Minimize difference with ground truths

$$\mathrm{y}_n^* = [1, 0],\ \mathrm{y}_h^* = [0, 1].$$

$$\mathrm{F} \in \begin{cases} \mathcal{N}, & \text{if } \mathrm{y}(0) > 0.5, \\ \mathcal{H}, & \text{if } \mathrm{y}(1) > 0.5. \end{cases}$$

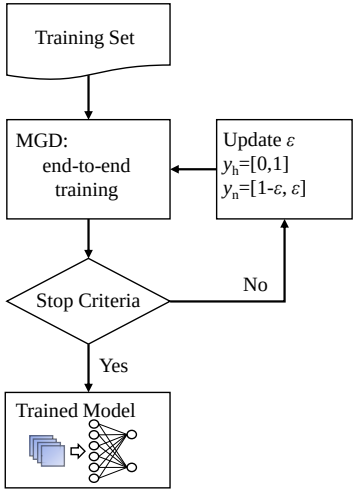
- Naive: Shifting decision boundary (✘)

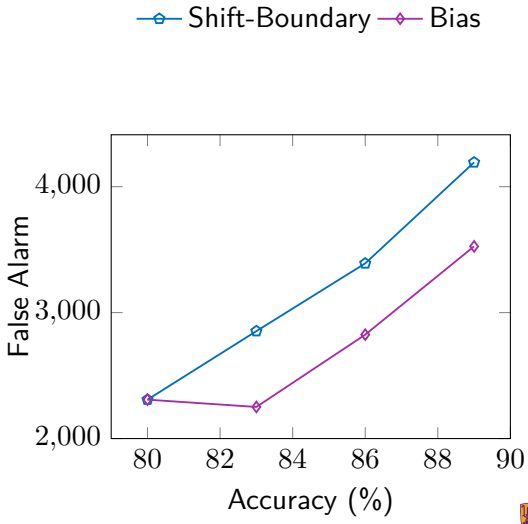
$$\mathrm{F} \in \begin{cases} \mathcal{N}, & \text{if } \mathrm{y}(0) > 0.5 + \lambda, \\ \mathcal{H}, & \text{if } \mathrm{y}(1) > 0.5 - \lambda. \end{cases}$$

- Biased ground truth:

$$\mathrm{y}_n^* = [1 - \epsilon, \epsilon].$$

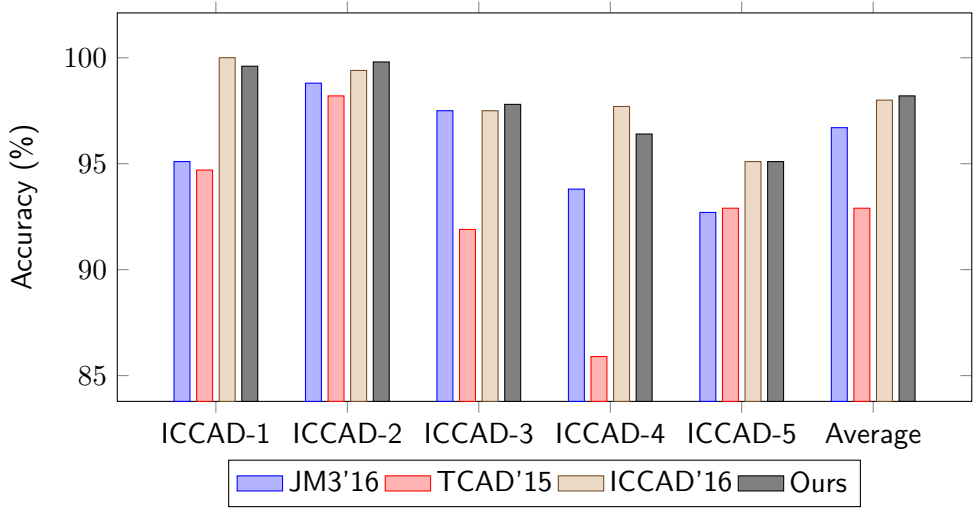
The Biased Learning Algorithm





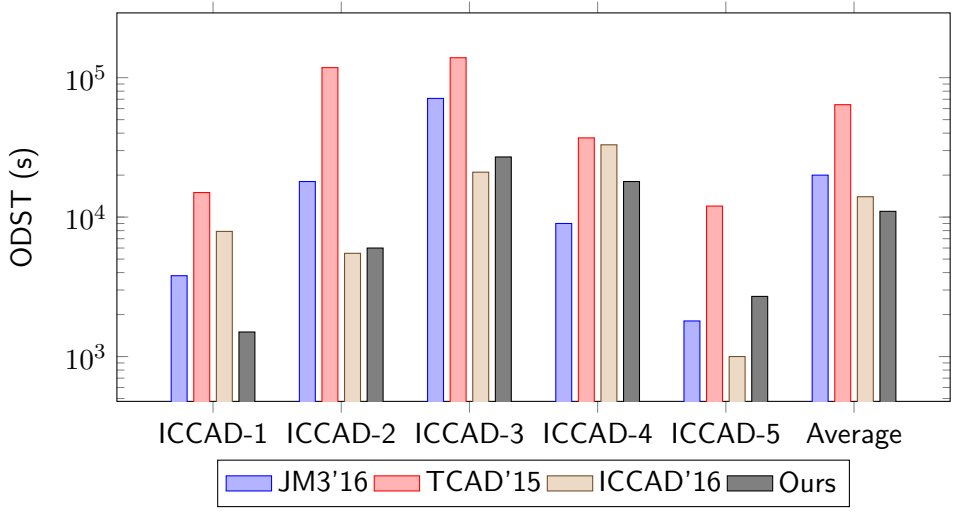
Comparison with Previous Hotspot Detectors

- Detection accuracy improved from 89.6% to 95.5%



Comparison with Previous Hotspot Detectors

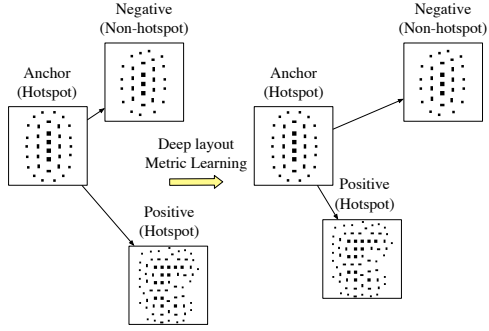
- Comparable false alarm penalty



Metric Feature Learning [ICCAD'20][6]

Motivation

- In original space, the anchor is much similar to the negative
- After deep layout metric learning, in a new manifold, the two hotspot layout clips are kept apart from the non-hotspot clip



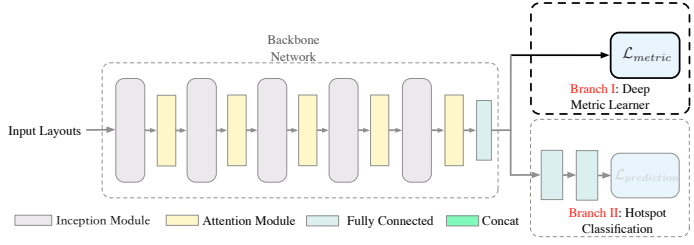
[6] Hao Geng, Haoyu Yang, et al. (2020). "Hotspot Detection via Attention-based Deep Layout Metric Learning". In: *Proc. ICCAD*.

Metric Feature Learning

- A triplet: $f_{\boldsymbol{w}}(\boldsymbol{x}_i)$, $f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^+)$, $f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^-)$
- $f_{\boldsymbol{w}}(\boldsymbol{x}_i)$: an anchor layout clip
- $f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^+)$: sharing the same label with the anchor
- $f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^-)$: having the opposite label to the anchor

$$\min_{\boldsymbol{w}} \frac{1}{n}\sum_{i=1}^{N} \max(0, M + ||f_{\boldsymbol{w}}(\boldsymbol{x}_i) - f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^+)||_2^2 - ||f_{\boldsymbol{w}}(\boldsymbol{x}_i) - f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^-)||_2^2) \quad (1)$$

$$\text{s.t. } ||f_{\boldsymbol{w}}(\boldsymbol{x}_i)||_2^2 = 1,\ \forall (f_{\boldsymbol{w}}(\boldsymbol{x}_i), f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^+), f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^-)) \in \mathcal{T}. \quad (2)$$



Metric Feature Learning

Gradients Calculation:

$$\frac{\partial \mathcal{L}_{metric}(f_{\boldsymbol{w}}(\boldsymbol{x}_i), f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^+), f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^-))}{\partial f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^+)} = \frac{2}{n}\left(f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^+) - f_{\boldsymbol{w}}(\boldsymbol{x}_i)\right) \cdot \mathbf{1}\left(\mathcal{L}_{metric}(f_{\boldsymbol{w}}(\boldsymbol{x}_i), f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^+), f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^-)) > 0\right), \tag{3a}$$

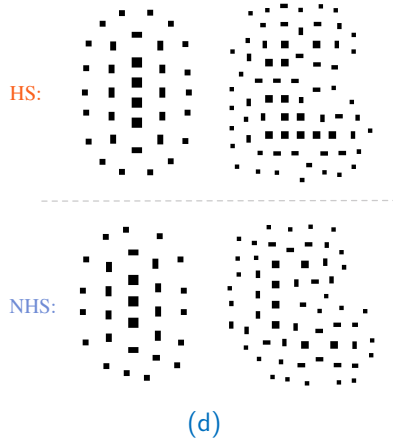
$$\frac{\partial \mathcal{L}_{metric}(f_{\boldsymbol{w}}(\boldsymbol{x}_i), f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^+), f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^-))}{\partial f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^-)} = \frac{2}{n}\left(f_{\boldsymbol{w}}(\boldsymbol{x}_i) - f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^-)\right) \cdot \mathbf{1}\left(\mathcal{L}_{metric}(f_{\boldsymbol{w}}(\boldsymbol{x}_i), f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^+), f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^-)) > 0\right), \tag{3b}$$

$$\frac{\partial \mathcal{L}_{metric}(f_{\boldsymbol{w}}(\boldsymbol{x}_i), f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^+), f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^-))}{\partial f_{\boldsymbol{w}}(\boldsymbol{x}_i)} = \frac{2}{n}\left(f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^-) - f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^+)\right) \cdot \mathbf{1}\left(\mathcal{L}_{metric}(f_{\boldsymbol{w}}(\boldsymbol{x}_i), f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^+), f_{\boldsymbol{w}}(\boldsymbol{x}_i{}^-)) > 0\right), \tag{3c}$$

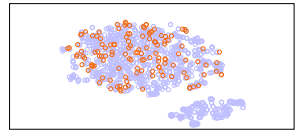
where $\mathbf{1}$ is the indicator function which is defined as:

$$\mathbf{1}(x) = \begin{cases} 1 & \text{if x is true,} \\ 0 & \text{otherwise.} \end{cases} \tag{3d}$$

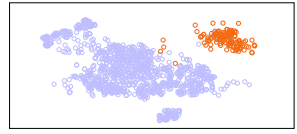
The t-SNE visualizations of feature embeddings on VIA benchmarks



(d)



(e)

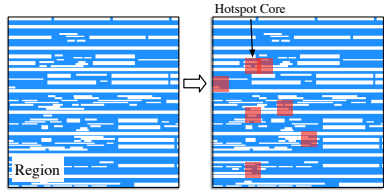


(f)

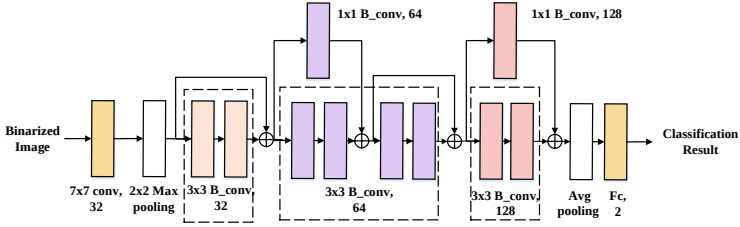
(a) The exemplars of hotspots and non-hotspots; (b) The DCT feature embeddings of TCAD'19; (c) The feature embeddings of our proposed framework.

HSD-Research: New Network Architecture

- Region-based HSD [DAC'19][7]



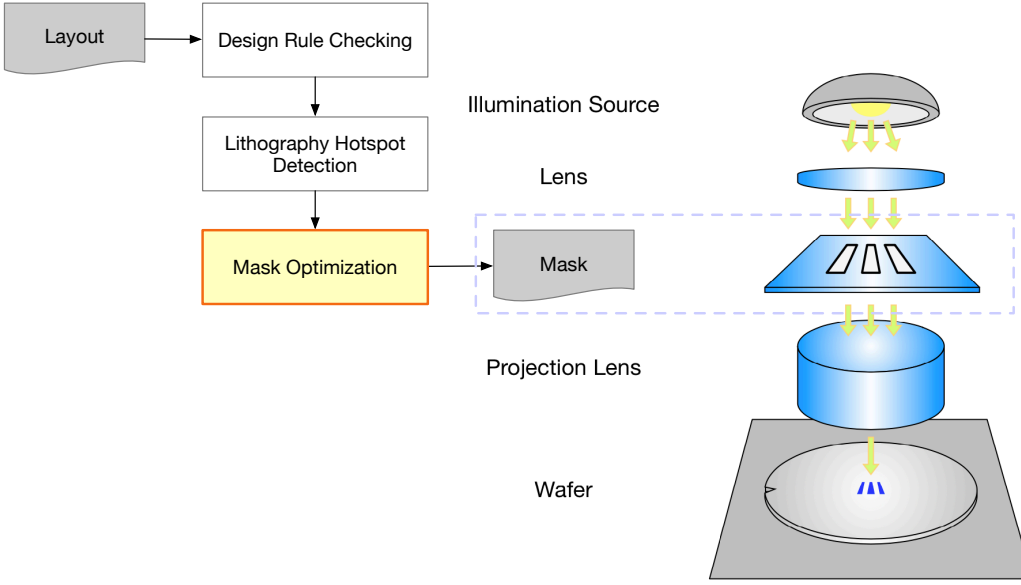
- Binarized residual neural network [DAC'19][8]



[7] Ran Chen et al. (2019). "Faster Region-based Hotspot Detection". In: *Proc. DAC*, 146:1–146:6.

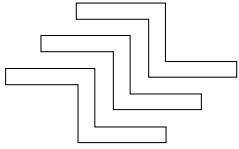
[8] Yiyang Jiang et al. (2019). "Efficient Layout Hotspot Detection via Binarized Residual Neural Network". In: *Proc. DAC*, 147:1–147:6.

Outline

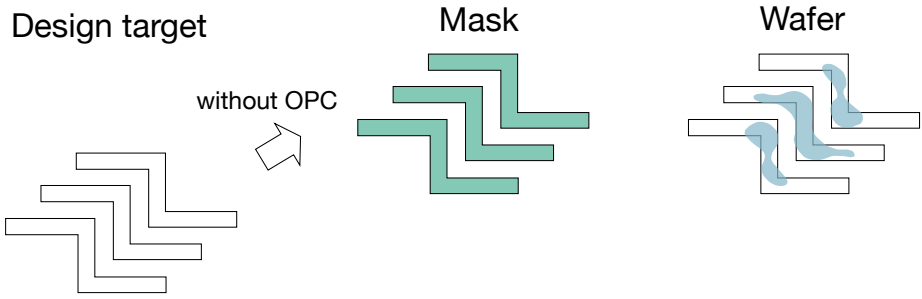

Layout
Design Rule Checking
Lithography Hotspot Detection
Mask Optimization
Mask
Illumination Source
Lens
Projection Lens
Wafer

Mask Optimization

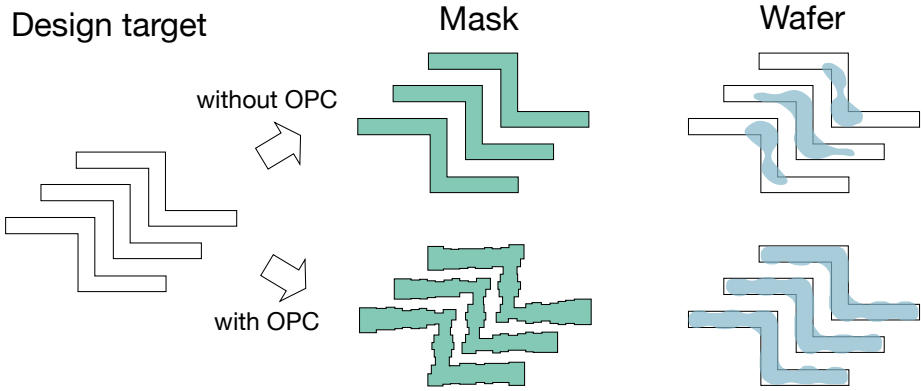
Design target



Mask Optimization



Mask Optimization



Lithography Model

- SVD Approximation of Partial Coherent System [Cobb,1998]

$$\mathrm{I} = \sum_{k=1}^{N^2} w_k |\mathrm{M} \otimes \mathrm{h}_k|^2. \tag{4}$$

- Reduced Model [Gao+,DAC'14]

$$\mathrm{I} = \sum_{k=1}^{N_h} w_k |\mathrm{M} \otimes \mathrm{h}_k|^2. \tag{5}$$

- Etch Model

$$\mathrm{Z}(x,y) = \begin{cases} 1, & \text{if } \mathrm{I}(x,y) \geq I_{th}, \\ 0, & \text{if } \mathrm{I}(x,y) < I_{th}. \end{cases} \tag{6}$$

Inverse Lithography Technique (ILT)

The main objective in ILT is minimizing the lithography error through gradient descent.

$$E = ||\mathrm{Z}_t - \mathrm{Z}||_2^2, \tag{7}$$

where Z_t is the target and Z is the wafer image of a given mask.
Apply translated sigmoid functions to make the pixel values close to either 0 or 1.

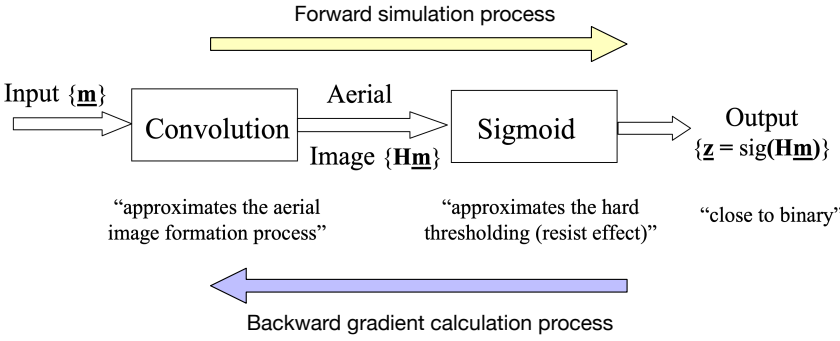
$$\mathrm{Z} = \frac{1}{1 + \exp[-\alpha \times (\mathrm{I} - \mathrm{I}_{th})]}, \tag{8}$$

$$\mathrm{M}_b = \frac{1}{1 + \exp(-\beta \times \mathrm{M})}. \tag{9}$$

Combine Equations (4)–(9) and the analysis in [Poonawala,TIP'07],

$$\begin{aligned}\frac{\partial E}{\partial \mathrm{M}} =& 2\alpha\beta \times \mathrm{M}_b \odot (1 - \mathrm{M}_b) \odot \\ & (((\mathrm{Z} - \mathrm{Z}_t) \odot \mathrm{Z} \odot (1 - \mathrm{Z}) \odot (\mathrm{M}_b \otimes \mathrm{H}^*)) \otimes \mathrm{H} + \\ & ((\mathrm{Z} - \mathrm{Z}_t) \odot \mathrm{Z} \odot (1 - \mathrm{Z}) \odot (\mathrm{M}_b \otimes \mathrm{H})) \otimes \mathrm{H}^*).\end{aligned} \tag{10}$$

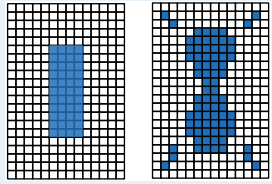
ILT flow


Forward simulation process
Input {m}
Convolution
Aerial
Image {Hm}
Sigmoid
Output
{z = sig(Hm)}
"approximates the aerial image formation process"
"approximates the hard thresholding (resist effect)"
"close to binary"
Backward gradient calculation process

ILT Methodologies [DAC'14][9] [ICCAD'17][10][11]

Typical ILT

- Mask → Image → Matrix
- Calculate gradient on each pixel.



Level-set method

- Boundary-based update
- Implicit representation; focus on boundaries

$$\begin{cases} \phi(t,\mathrm{x}) < 0 & \text{if } \mathrm{x} \in \Omega(t) \\ \phi(t,\mathrm{x}) = 0 & \text{if } \mathrm{x} \in \Gamma(t) \\ \phi(t,\mathrm{x}) > 0 & \text{if } \mathrm{x} \in \overline{\Omega(t)} \end{cases}$$

[9] Jhih-Rong Gao et al. (2014). "MOSAIC: Mask Optimizing Solution With Process Window Aware Inverse Correction". In: *Proc. DAC*. San Jose, California, 52:1–52:6.

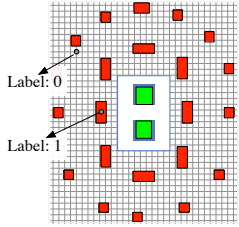
[10] Yuzhe Ma et al. (2017). "A Unified Framework for Simultaneous Layout Decomposition and Mask Optimization". In: *Proc. ICCAD*, pp. 81–88.

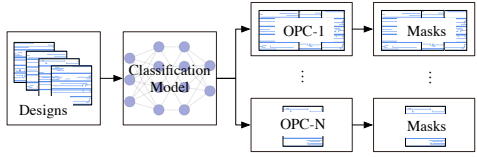
[11] Ziyang Yu et al. (2021). "A GPU-enabled Level Set Method for Mask Optimization". In: *submitted to DATE*.

Leaning-based Mask Optimization

Discriminative models [TCAD'20][12] [ASPDAC'20][13]

- Pixel-wise classification
- Printed image estimation/quality estimation





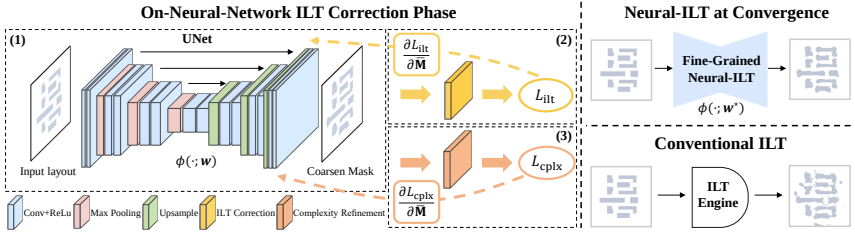
[12] Hao Geng, Wei Zhong, et al. (2020). "SRAF Insertion via Supervised Dictionary Learning". In: *IEEE TCAD*.

[13] Haoyu Yang, Wei Zhong, et al. (2020). "VLSI Mask Optimization: From Shallow To Deep Learning". In: *Proc. ASPDAC*, pp. 434–439.

Leaning-based Mask Optimization

Generative model[14][15][16]

- Image generation

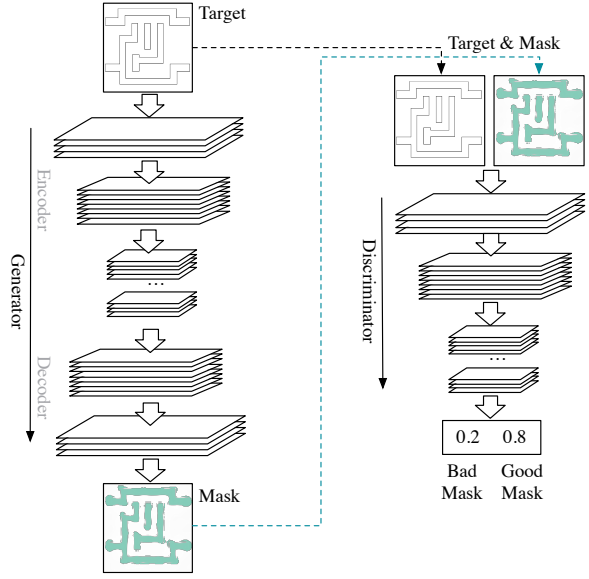


[14] Haoyu Yang, Shuhe Li, et al. (2018). "GAN-OPC: Mask Optimization with Lithography-guided Generative Adversarial Nets". In: *Proc. DAC*, 131:1–131:6.

[15] Bentian Jiang et al. (2020). "Neural-ILT: Migrating ILT to Nerual Networks for Mask Printability and Complexity Co-optimizaton" ". In: *Proc. ICCAD*.

[16] Guojin Chen et al. (2020). "DAMO: Deep Agile Mask Optimization for Full Chip Scale". In: *Proc. ICCAD*.

GAN-OPC [DAC'18][17]



[17] Haoyu Yang, Shuhe Li, et al. (2018). "GAN-OPC: Mask Optimization with Lithography-guided Generative Adversarial Nets". In: *Proc. DAC*, 131:1–131:6.

GAN Training

Based on the OPC-oriented GAN architecture in our framework, we tweak the objectives of G and D accordingly,

$$\max_{\mathrm{G}} \mathbb{E}_{\mathrm{Z}_t \sim \mathcal{Z}}[\log(\mathrm{D}(\mathrm{Z}_t, \mathrm{G}(\mathrm{Z}_t)))], \tag{11}$$

$$\max_{\mathrm{D}} \mathbb{E}_{\mathrm{Z}_t \sim \mathcal{Z}}[\log(\mathrm{D}(\mathrm{Z}_t, \mathrm{M}^*))] + \mathbb{E}_{\mathrm{Z}_t \sim \mathcal{Z}}[1 - \log(\mathrm{D}(\mathrm{Z}_t, \mathrm{G}(\mathrm{Z}_t)))]. \tag{12}$$

In addition to facilitate the training procedure, we minimize the differences between generated masks and reference masks when updating the generator as in Equation (13).

$$\min_{\mathrm{G}} \mathbb{E}_{\mathrm{Z}_t \sim \mathcal{Z}} ||\mathrm{M}^* - \mathrm{G}(\mathrm{Z}_t)||_n, \tag{13}$$

where $|| \cdot ||_n$ denotes the l_n norm.

GAN Training

Based on the OPC-oriented GAN architecture in our framework, we tweak the objectives of G and D accordingly,

$$\max_{\mathrm{G}} \mathbb{E}_{\mathrm{Z}_t \sim \mathcal{Z}}[\log(\mathrm{D}(\mathrm{Z}_t, \mathrm{G}(\mathrm{Z}_t)))], \tag{11}$$

$$\max_{\mathrm{D}} \mathbb{E}_{\mathrm{Z}_t \sim \mathcal{Z}}[\log(\mathrm{D}(\mathrm{Z}_t, \mathrm{M}^*))] + \mathbb{E}_{\mathrm{Z}_t \sim \mathcal{Z}}[1 - \log(\mathrm{D}(\mathrm{Z}_t, \mathrm{G}(\mathrm{Z}_t)))]. \tag{12}$$

In addition to facilitate the training procedure, we minimize the differences between generated masks and reference masks when updating the generator as in Equation (13).

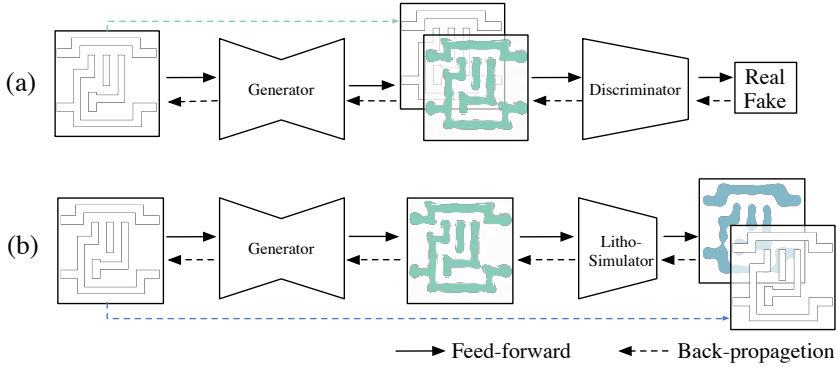
$$\min_{\mathrm{G}} \mathbb{E}_{\mathrm{Z}_t \sim \mathcal{Z}} ||\mathrm{M}^* - \mathrm{G}(\mathrm{Z}_t)||_n, \tag{13}$$

where $||\cdot||_n$ denotes the l_n norm. Combining (11), (12) and (13), the objective of our GAN model becomes

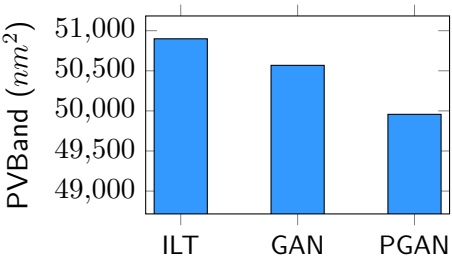
$$\min_{\mathrm{G}} \max_{\mathrm{D}} \mathbb{E}_{\mathrm{Z}_t \sim \mathcal{Z}}[1 - \log(\mathrm{D}(\mathrm{Z}_t, \mathrm{G}(\mathrm{Z}_t))) + ||\mathrm{M}^* - \mathrm{G}(\mathrm{Z}_t)||_n^n] + \mathbb{E}_{\mathrm{Z}_t \sim \mathcal{Z}}[\log(\mathrm{D}(\mathrm{Z}_t, \mathrm{M}^*))]. \tag{14}$$

ILT-guided Pre-training

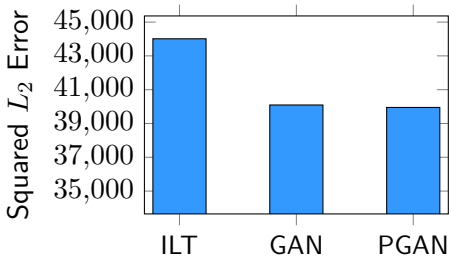
Observing that both ILT and neural network optimization share similar gradient descent procedure, we propose a jointed training algorithm that takes advantages of ILT engine, as depicted in Figure (b). We initialize the generator with lithography-guided pre-training to make it converge well in the GAN optimization flow thereafter.



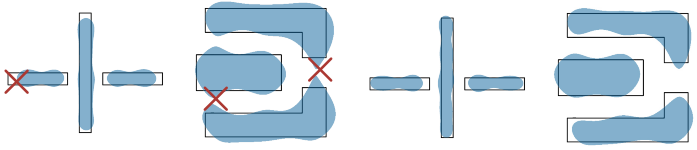
Results


PVBand (nm^2)
51,000
50,500
50,000
49,500
49,000
ILT
GAN
PGAN

(a)

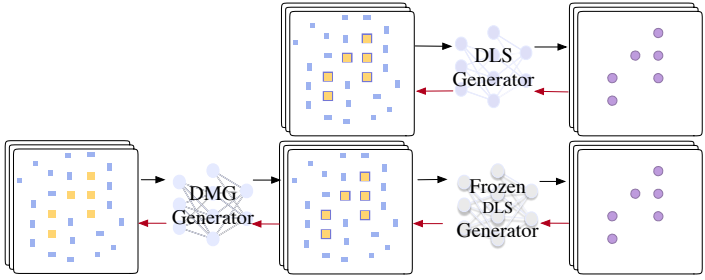

Squared L_2 Error
45,000
43,000
41,000
39,000
37,000
35,000
ILT
GAN
PGAN

(b)


ILT
PGAN

(c)

Deep Mask Optimization [ICCAD'20][18]

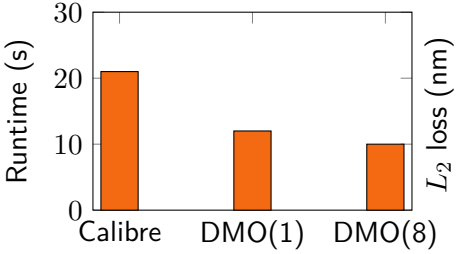


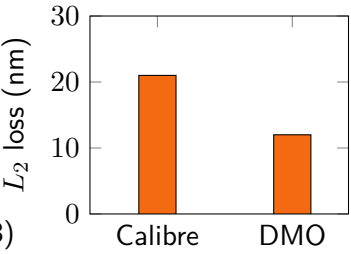
	GAN-OPC			Calibre			**DMO**		
	L_2 (nm)	PV Band (nm^2)	Runtime (s)	L_2 (nm)	PV Band (nm^2)	Runtime (s)	L_2 (nm)	PV Band (nm^2)	Runtime (s)
case 1	7456	11424	284	5159	11671	1417	**4631**	**11166**	352
case 2	7321	11215	281	4987	11463	1406	**4432**	**10955**	336
case 3	7102	11265	285	5420	11516	1435	**4802**	**11032**	367
case 4	8032	11642	322	5382	11910	1606	**4835**	**11265**	399
Average	7478	11386	293	5237	11640	1466	**4675**	**11104**	363
Ratio	1.60	1.03	0.80	1.12	1.05	4.04	**1.00**	**1.00**	1.00

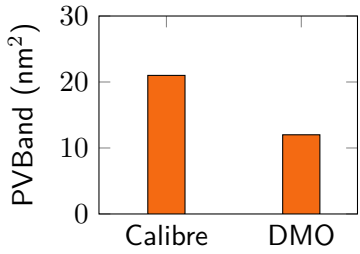
[18] Guojin Chen et al. (2020). "DAMO: Deep Agile Mask Optimization for Full Chip Scale". In: *Proc. ICCAD*.

Scalability on Full-chip Layout

- Tested on $10\mu\text{m} \times 10\mu\text{m}$ layout.

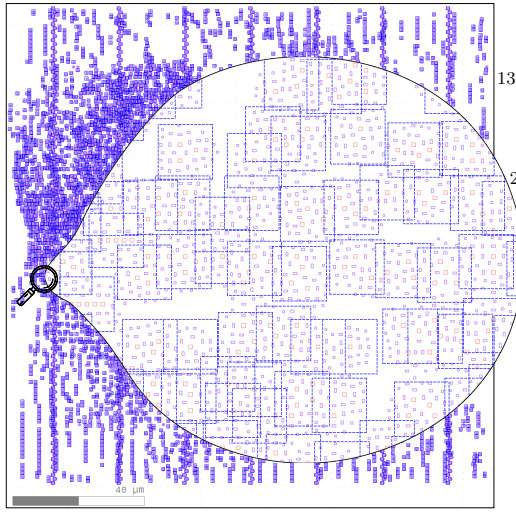


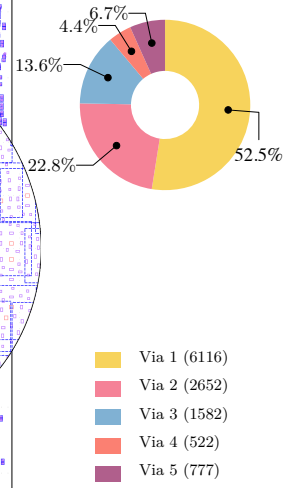




- DMO(1) – Single GPU card;
- DMO(8) – 8 GPU cards

Results on ISPD 2019 datasets





Thank You!