Layout Hotspot Detection with Feature Tensor Generation and Deep Biased Learning

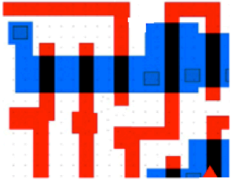
Haoyu Yang[1], Jing Su[2], Yi Zou[2], Bei Yu[1], Evangeline F. Y. Young[1]

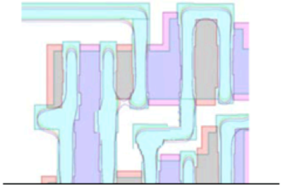
[1]The Chinese University of Hong Kong
[2]ASML Brion Inc.

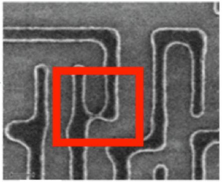
Outline

Outline

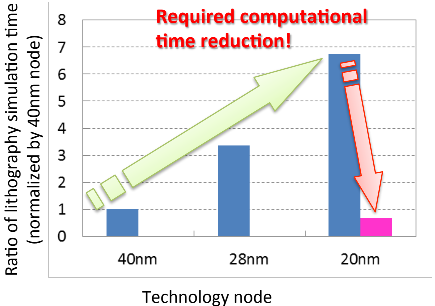
Lithography Hotspot Detection



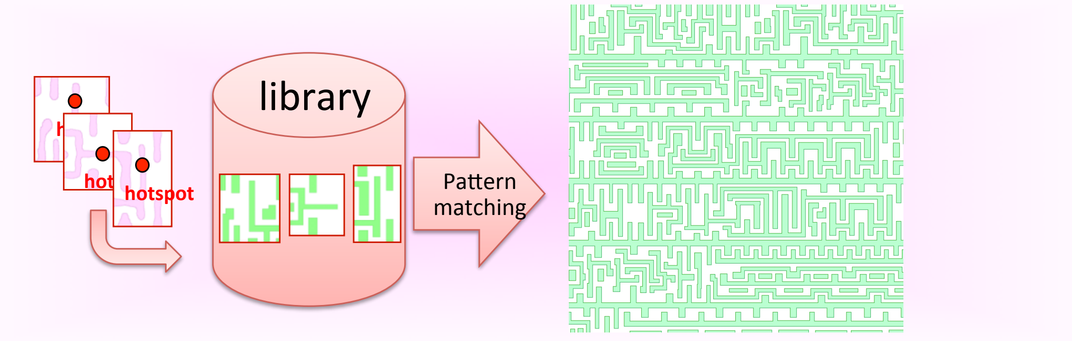




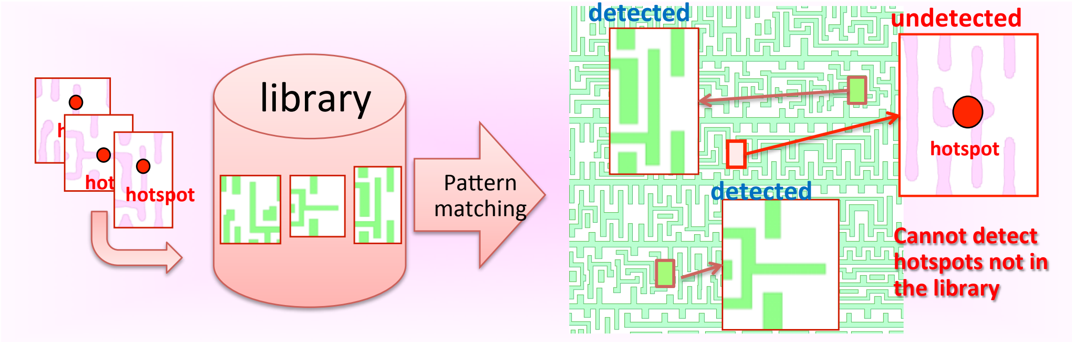
- RET: OPC, SRAF, MPL
- Still hotspot: low fidelity patterns
- Simulations: extremely CPU intensive



Pattern Matching based Hotspot Detection

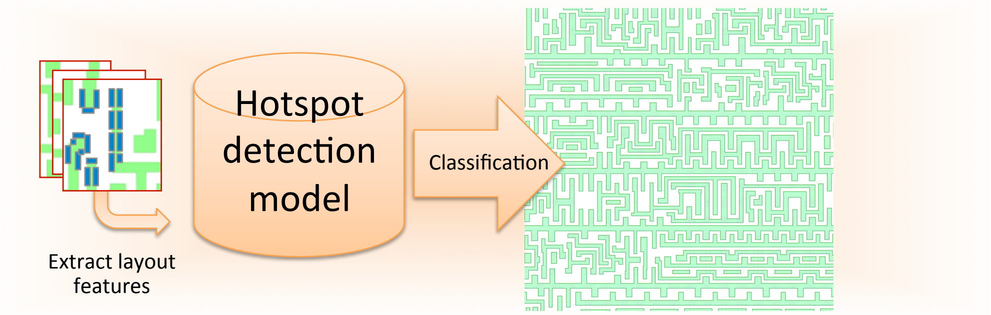


Pattern Matching based Hotspot Detection

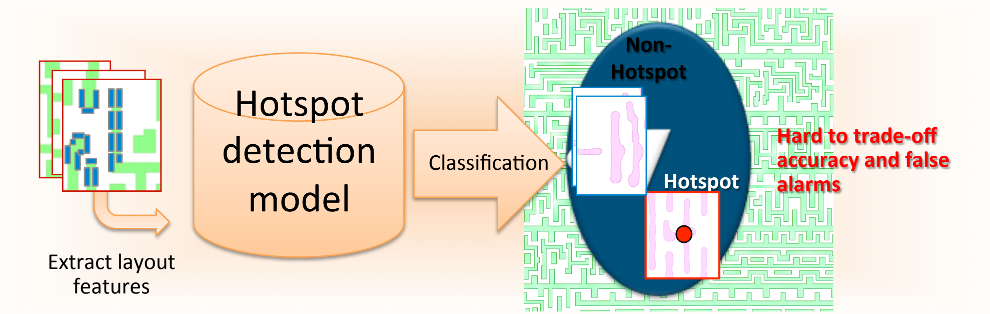


- Fast and accurate
- [Yu+,ICCAD'14] [Nosato+,JM3'14] [Su+,TCAD'15]
- Fuzzy pattern matching [Wen+,TCAD'14]
- Hard to detect non-seen pattern

Machine Learning based Hotspot Detection



Machine Learning based Hotspot Detection



- Predict new patterns
- Decision-tree, ANN, SVM, Boosting, Bayesian, ...
- [Ding+,TCAD'12][Yu+,JM3'15][Matsunawa+,SPIE'15][Yu+,TCAD'15][Zhang+,ICCAD'16][Wen+,TCAD'14]
- Feature reliability and model scalability

Why Deep Learning?

1. **Feature Crafting v.s. Feature Learning**
   - Manually designed feature–> Inevitable information loss
   - Learned feature–> Reliable
2. **Scalability**
   - More pattern types
   - More complicated patterns
   - Hard to fit millions of data with simple ML model
3. **Mature Libraries**
   - `Caffe` [Jia+,ACMMM'14]
   - `Tensorflow` [Martin+,TR'15]

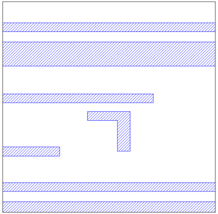
Special Issues for Layout Hotspot Detection

Layout image size is large ($\approx 1000 \times 1000$)

- Compared to ImageNet ($\approx 200 \times 200$)
- Associated CNN model is large
- Not storage and computational efficient

Hotspot detection accuracy is more important

- Hotspot –> Circuit Failure
- False Alarm –> Runtime Overhead
- Consider methods for better trade-off between accuracy and falsealarm

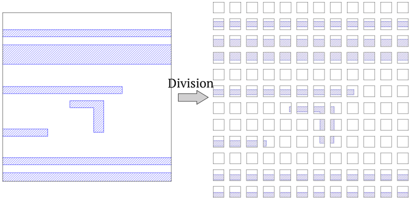


Layout clip with $1nm$ precision has resolution 1200×1200

Outline

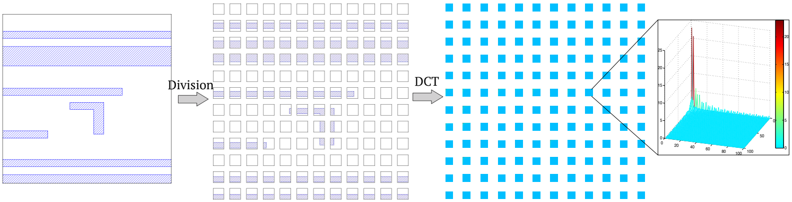
Feature Tensor Generation

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



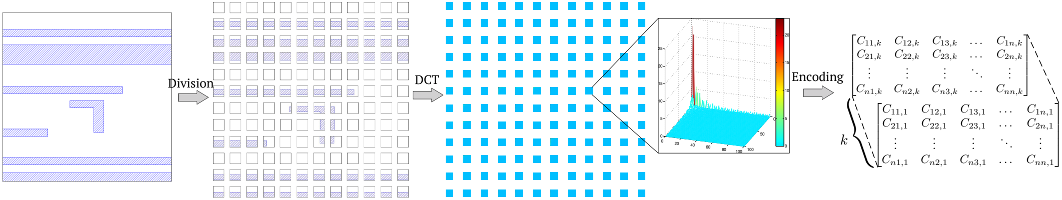
Feature Tensor Generation

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



Feature Tensor Generation

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

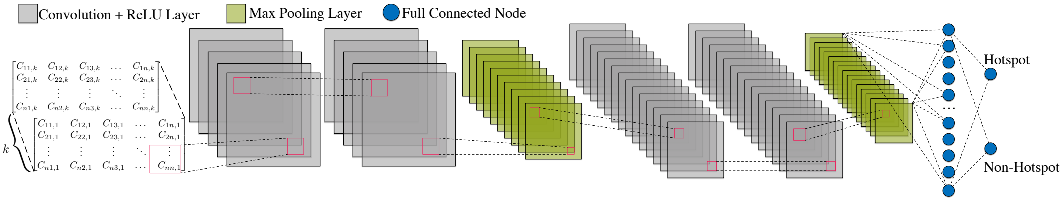


CNN Architecture

Feature Tensor

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

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



Outline

Recall The Training Procedure

- Minimize difference with ground truths

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

$$\mathbf{F} \in \begin{cases} \mathcal{N}, & \text{if } \mathbf{y}(0) > 0.5 \\ \mathcal{H}, & \text{if } \mathbf{y}(1) > 0.5 \end{cases} \tag{2}$$

Recall The Training Procedure

- Minimize difference with ground truths

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

$$\mathbf{F} \in \begin{cases} \mathcal{N}, & \text{if } \mathbf{y}(0) > 0.5 \\ \mathcal{H}, & \text{if } \mathbf{y}(1) > 0.5 \end{cases} \tag{2}$$

- Shifting decision boundary

$$\mathbf{F} \in \begin{cases} \mathcal{N}, & \text{if } \mathbf{y}(0) > 0.5 + \lambda \\ \mathcal{H}, & \text{if } \mathbf{y}(1) > 0.5 - \lambda \end{cases} \tag{3}$$

Recall The Training Procedure

- Minimize difference with ground truths

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

$$\mathbf{F} \in \begin{cases} \mathcal{N}, & \text{if } \mathbf{y}(0) > 0.5 \\ \mathcal{H}, & \text{if } \mathbf{y}(1) > 0.5 \end{cases} \tag{2}$$

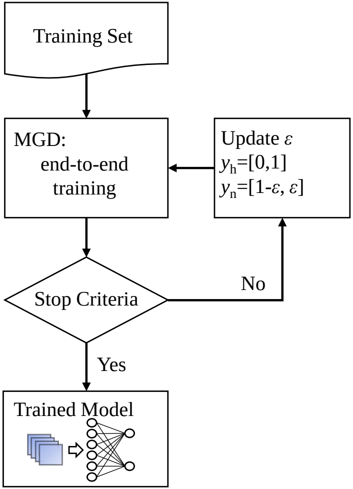
- Shifting decision boundary (✗)

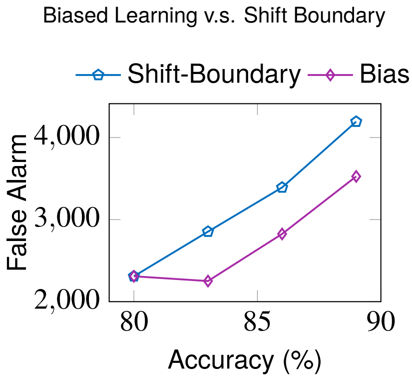
$$\mathbf{F} \in \begin{cases} \mathcal{N}, & \text{if } \mathbf{y}(0) > 0.5 + \lambda \\ \mathcal{H}, & \text{if } \mathbf{y}(1) > 0.5 - \lambda \end{cases} \tag{3}$$

- Biased ground truth

$$\mathbf{y}_n^* = [1 - \epsilon, \epsilon] \tag{4}$$

The Biased Learning Algorithm

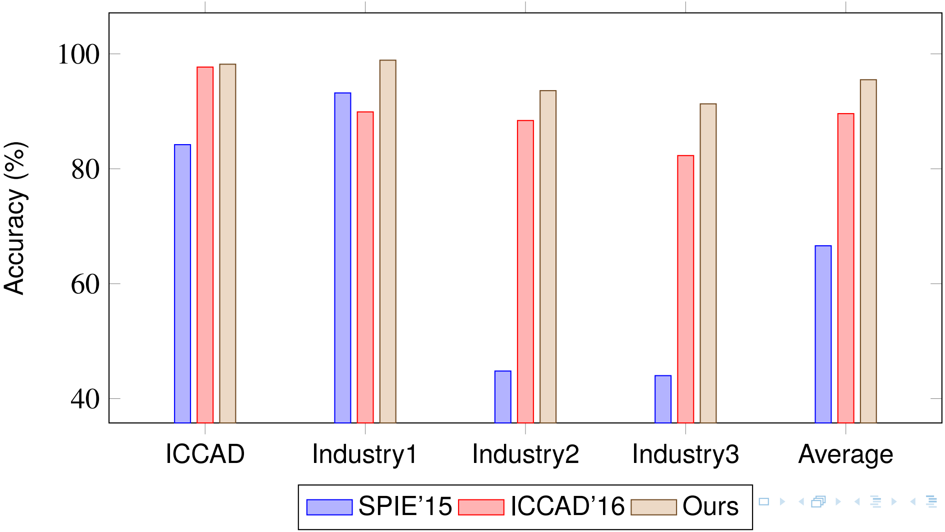




Outline

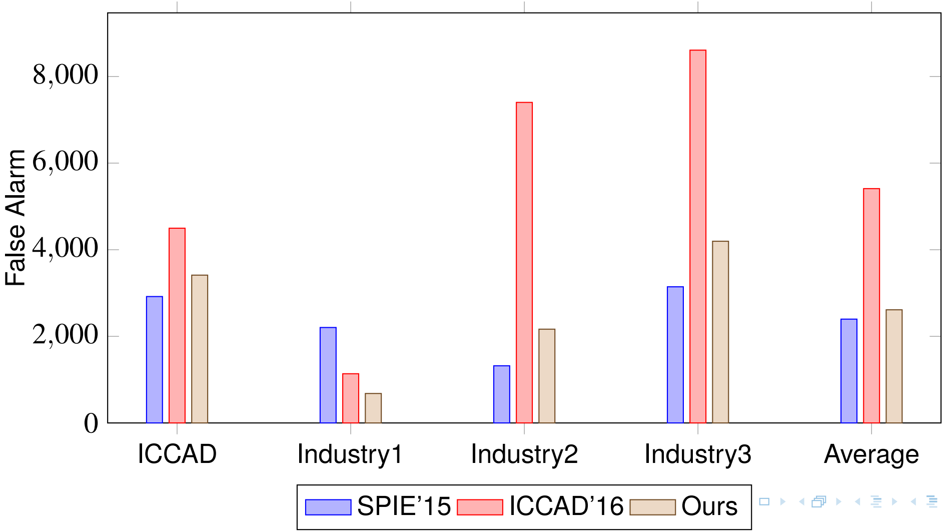
Comparison with Two Hotspot Detectors

- Detection accuracy improved from 89.6% to 95.5%



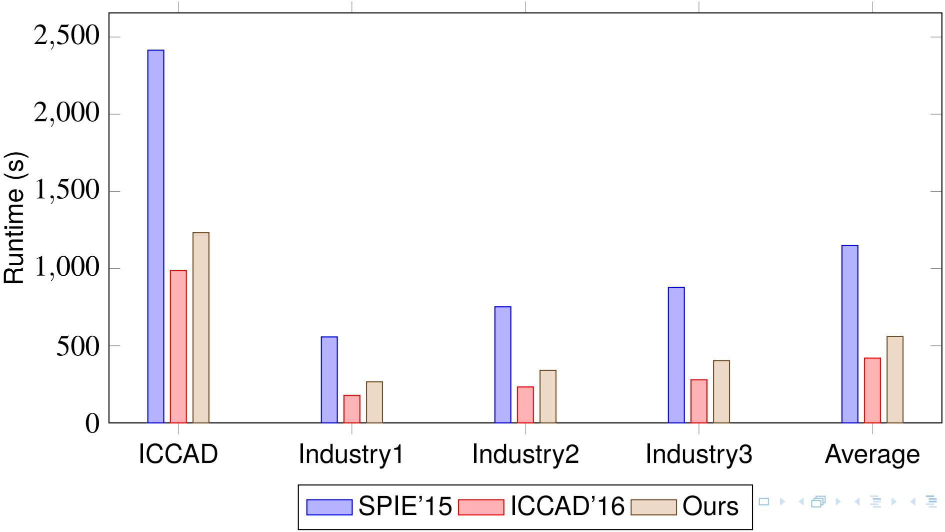
Comparison with Two Hotspot Detectors

- Comparable false alarm penalty



Comparison with Two Hotspot Detectors

- Comparable testing runtime



Thank You