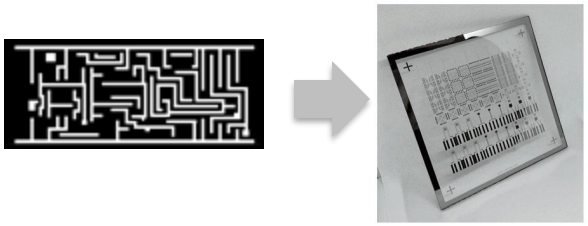
ML for Computational Lithography: Practical Recipes

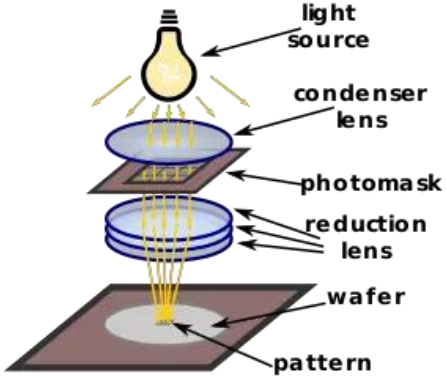
Youngsoo Shin
School of EE, KAIST

Chip Manufacturing

- **Mask synthesis**: layout to masks



- **Lithography**: pattern transfer from mask to wafer (through exposure to light)



- Wafer processing: etch, ion implantation, etc

- Packaging

Computational lithography:
mathematical and algorithmic approaches to improve the resolution attainable with lithography

ML for Lithography

- Has been popular since ~2010
  - Why: (1) ML provides "higher" modeling capability, (2) many applications are "image recognition" or "image conversion"
  - Some ML solutions are already being provided through vendor products (e.g. Synopsys, Mentor, Brion)

Motivations

- ML (for chip design and manufacturing) has its own limitations
  - Lack of benchmark and common data set
  - Difficult to analyze and debug
  - Data belongs to users; Model provided by vendors

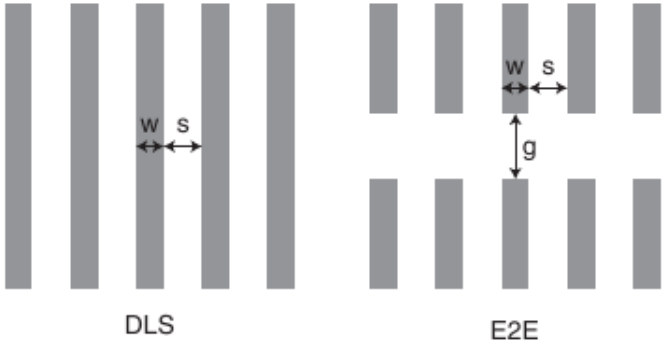
- This talk
  - Which lithography applications are more promising with ML?
  - When training samples are sparse, do we still use ML?

ML for Lithography

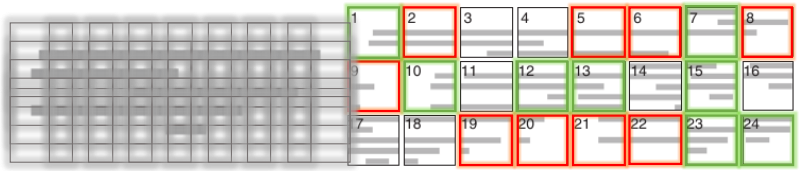
- Promising applications
  - Test pattern classification
  - Etch bias model
  - OPC and ILT
- ML may not be an ideal solution in
  - Optical model
  - Hotspot
  - Assist features

Lithography Test Patterns

- Parametric patterns
  - Represented by a few geometrical parameters (e.g. width, space)
  - Easy to build and analyze
  - Cannot cover complex patterns

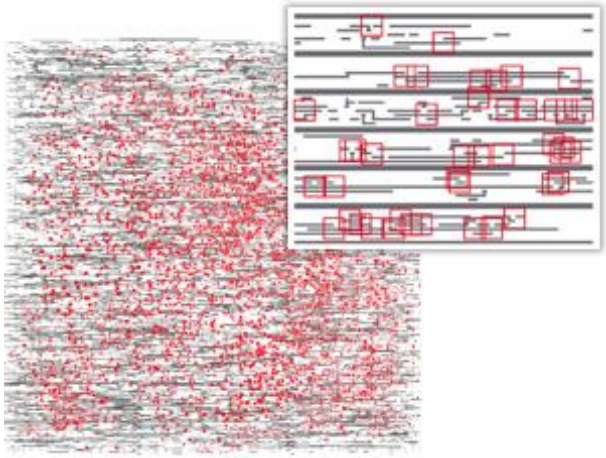


- Actual patterns
  - Extracted from sample layouts; can cover more complex shapes
  - Should be well classified



Insignificant or covered by the parametric test patterns

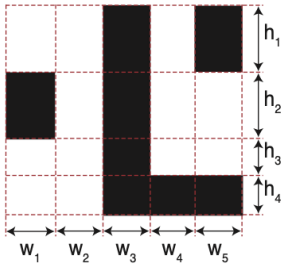
Redundant due to multiple capturing: 1-10, 7-12, 13-15, 23-24



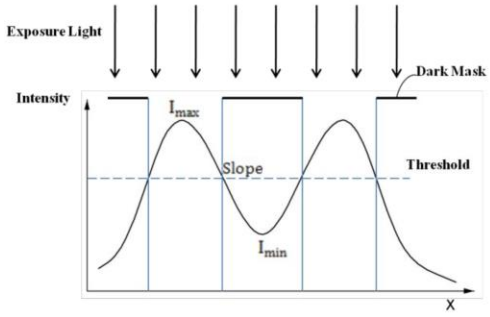
Test Pattern Classification

- Representation of sample patterns in parameter space → Clustering → Selection of representative patterns

- Representation parameters are important
  - Hanan grid (or Squish pattern)
  - Image parameter space (IPS): I_{max}, I_{min}, I_{slp}
  - ML features

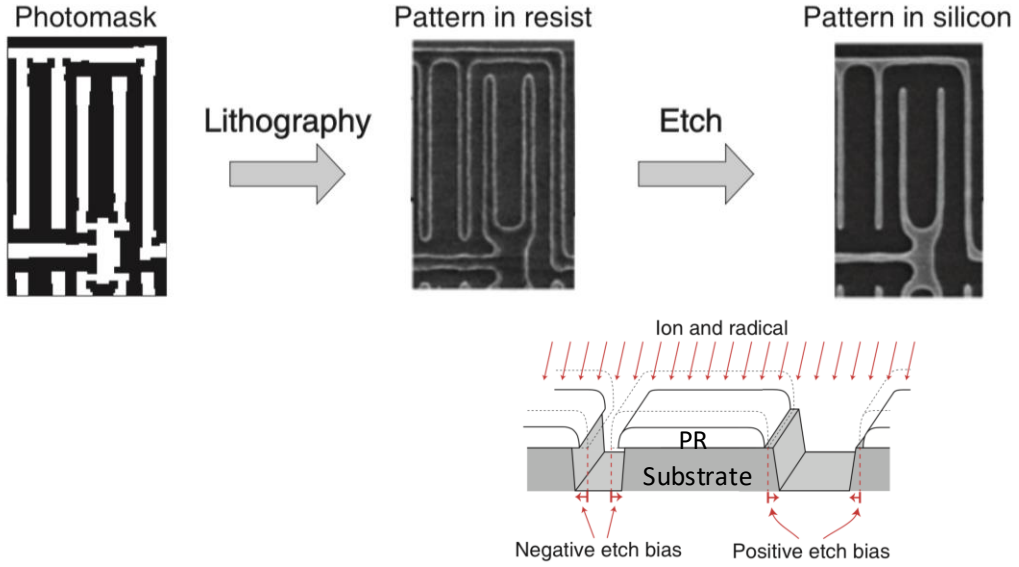


$$T = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$



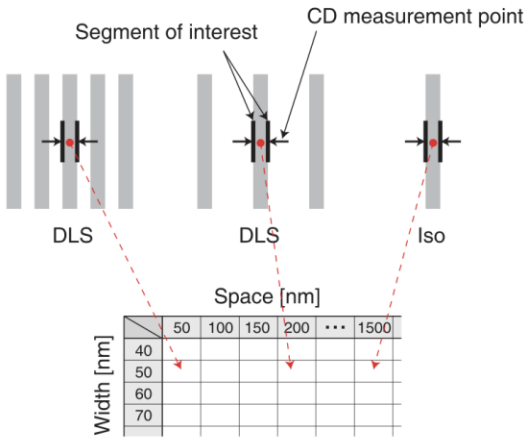
Etch Bias Model

- Etch bias
  - Amount of under-etch (positive bias) or over-etch (negative bias)
  - Etch bias model is a key for EPC (etch proximity correction) or Retargeting

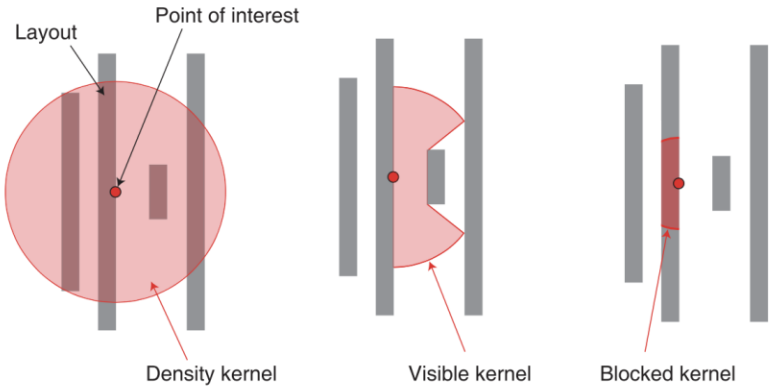


Etch Bias Model

- Current RB- or MB-models are crude and inaccurate
  - Etch process is difficult and complex to model, accurately



RB model

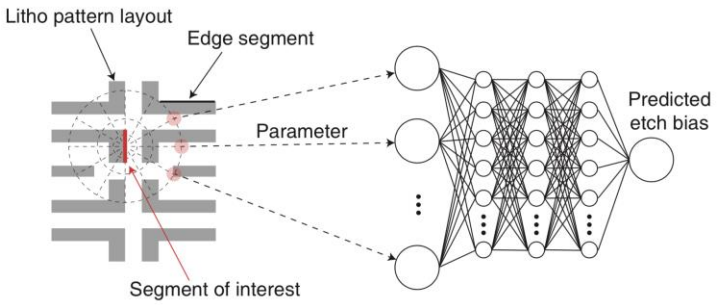


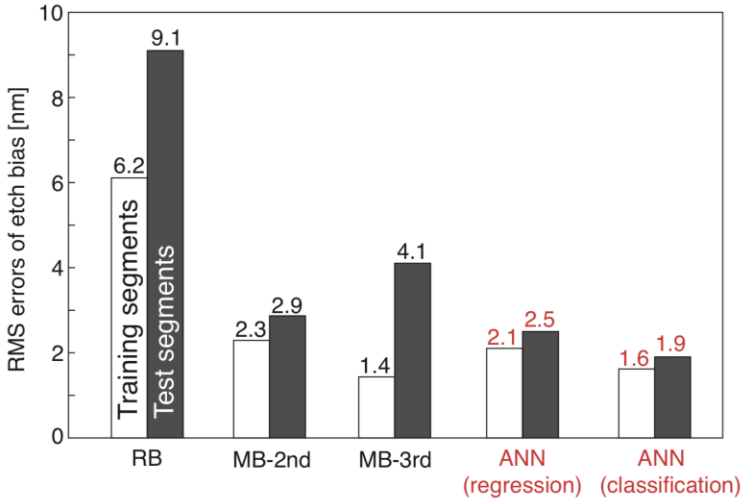
MB model

(e.g.) Etch bias = $C_0 + C_1Den + C_2Vis + C_3Blo + C_4Den^2 + \ldots$

ML Etch Bias Model

- Even a simple MLP provides pretty good accuracy
  - Choice of MLP input parameters (e.g. local pattern densities, optical kernel signals) are important
  - Choice of regression network (for weak etch process) or classification network (for strong etch process)



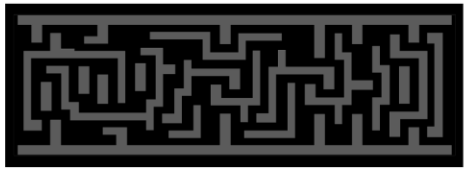


S. Shim and Y. Shin, "Machine learning-guided etch proximity correction," *IEEE Tr. on Semiconductor Manufacturing*, Feb. 2017.

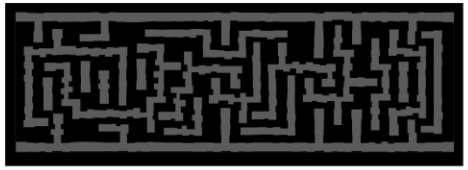
Standard (Model-Based) OPC

- Runtime increases year after year
  - More masks to process
  - More polygons to correct
  - Higher resolution requested

- Given: layout patterns



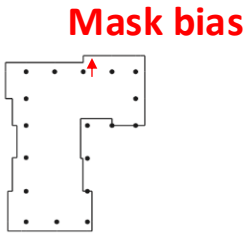
- Derive: mask patterns



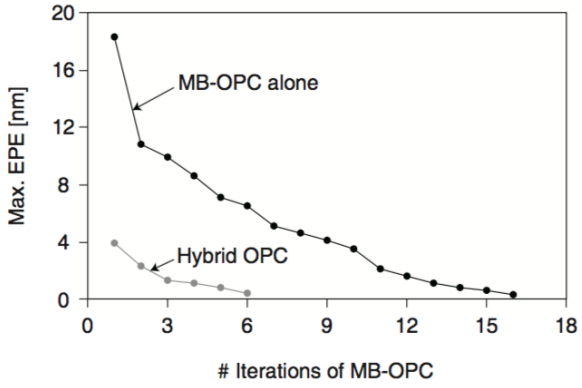
Technology Node	28nm	22nm	14nm	10nm	7nm	5nm
Production start	2011	2013	2014	2016	2017	2019
Average transistor density (billion/cm^2)	1.17	1.63	2.34	3.75	6.25	10.71
Number of critical layer masks	18	24	33	37	47	66
Normalized OPC runtime per layer per unit area	1	1.4	2	2.7	4	5.6

Y. Cao, "Application of machine learning in computational lithography," *SPIE eBeam lunch*, 2019.

OPC with ML



- Trained ML model for "Quick guess of mask bias"
  - No iterations, no simulations: extremely fast
  - But quality is not high enough
    - Practical application: ML output as initial OPC solution, provided to standard OPC: still quite fast (3 ~ 6 times)
  - Challenges
    - Choice of ML model (MLP, BRNN, GCN, CNN)
    - Feature engineering (pattern densities, optical kernel signals)
    - ML optimization & training (as usual)



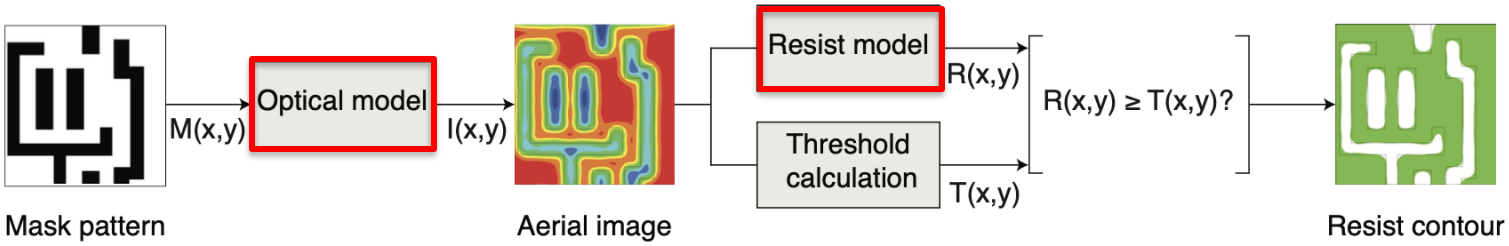
ML for Lithography

- Promising applications
  - Test pattern classification
  - Etch bias model
  - OPC and ILT

- **ML may not be an ideal solution**
  - **Optical model**
  - **Hotspot**
  - **Assist features**

Lithography Simulation

- Uses a compact model to simulate the response of PR to exposure and development



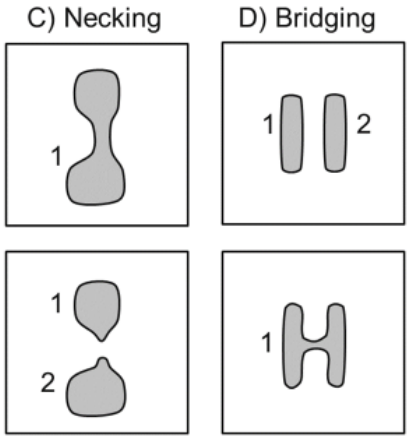
- **Optical model**
  - Captures exposure
  - Often uses SOCS (sum of coherent systems) approximation: already efficient (accuracy and computation time)

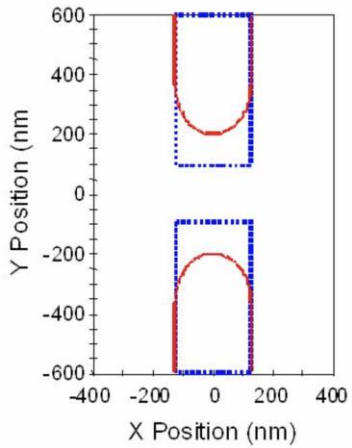
$$I(x,y) = \sum \lambda_i \left| \phi_i \otimes M(x,y) \right|^2$$

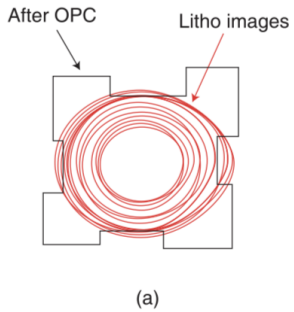
- **Resist model**
  - Captures PEB (post-exposure bake) & development

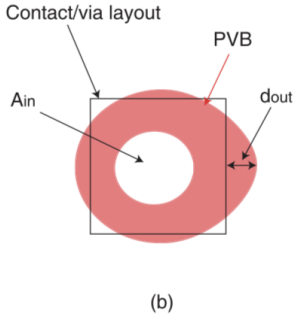
Hotspot

- Patterns that may cause pinching (critical width), bridging (critical spacing), line end shortening, etc
  1. Pattern matching (using hotspot library) to choose candidate regions
  2. "Lithography simulations" to confirm hotspot patterns through PVB (process variation band)



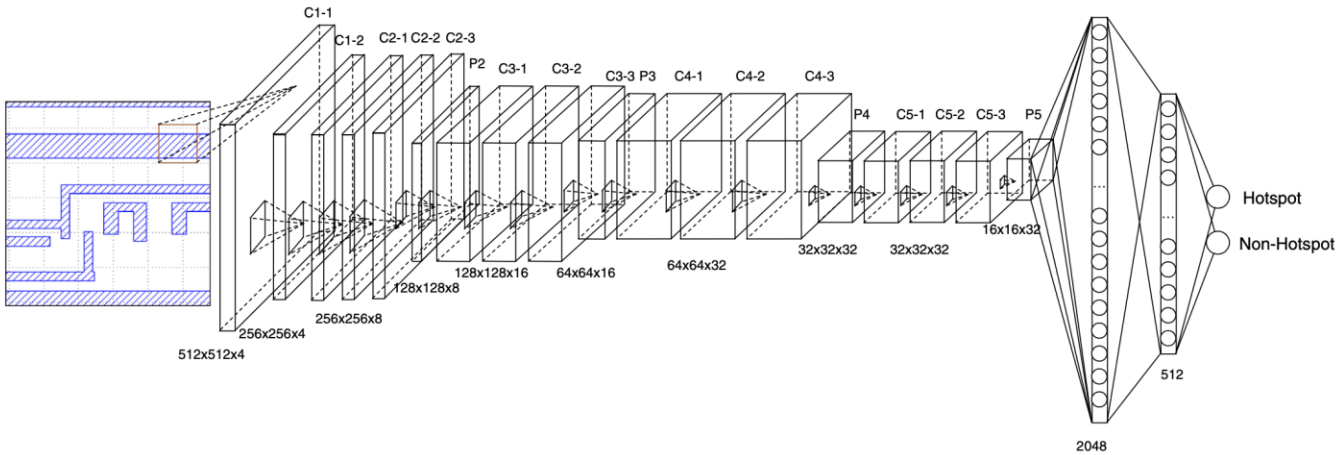






Hotspot Detection

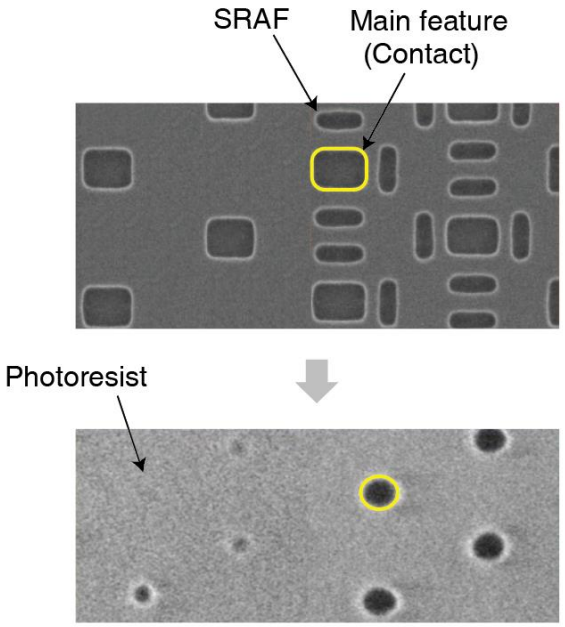
- Hotspot detection using CNN
  - Training is difficult: sample hotspots are sparse, augmentation of samples (mirroring, flipping, etc) might help
  - Detection should be perfect; miss prediction is not allowed

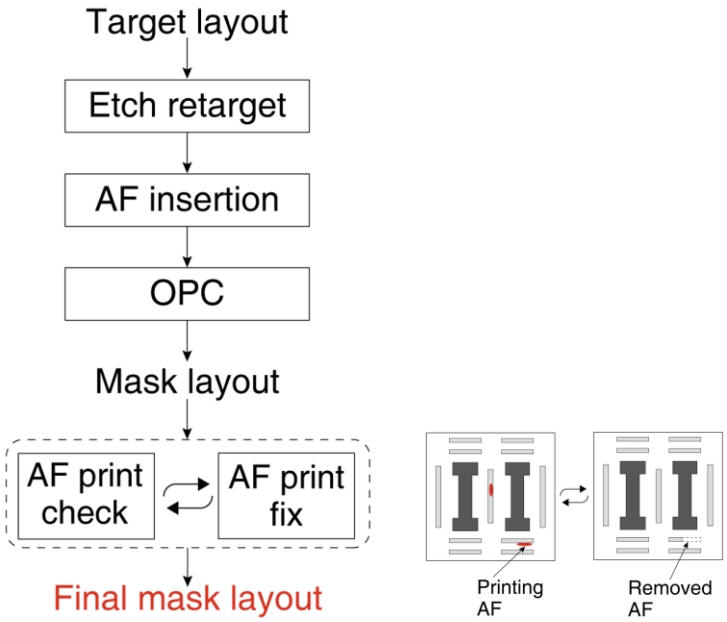


H. Yang et al, "Imbalance aware lithography hotspot detection: a deep learning approach," *Proc. SPIE Advanced Lithography*, 2017.

Assist Features (AF)

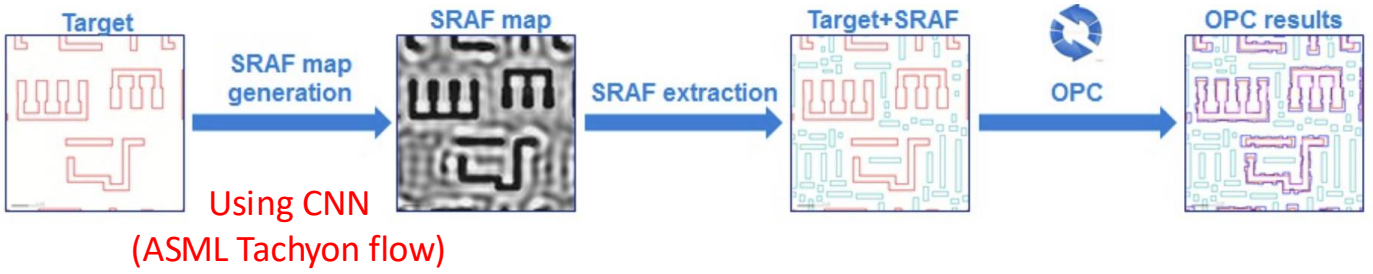
- Extra patterns added to the mask, not intended to be printed, which help nearby main patterns for better printing through constructive light interference





AF Applications with ML

- CNN to predict AF map for AF insertion
  - AF insertion runtime is reduced to 1/7 (in standard OPC flow); ILT runtime is reduced by 34%
- MLP for AF printability check
- AFs are "small" features; high accuracy of AF applications with ML is not easy
- AFs are becoming integral part of ILT



S. Wang et al, "Machine learning assisted SRAF placement for full chip," *Proc. SPIE*, 2017.

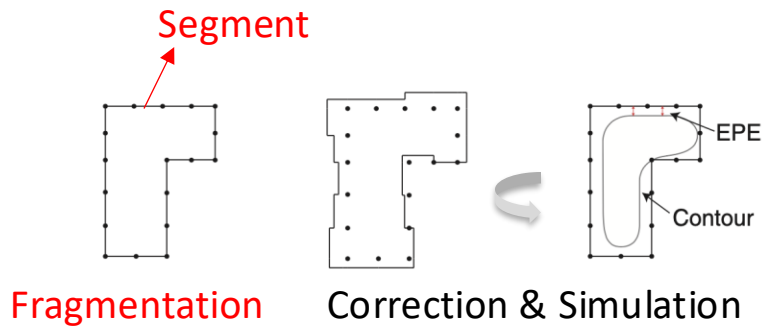
Motivations

- ML (for chip design and manufacturing) has its own limitations
  - Lack of benchmark and common data set
  - Data belongs to users; Model provided by vendors
  - Difficult to analyze and debug

- This talk
  - Which lithography applications are more promising with ML?
  - When training samples are sparse, do we still use ML?

Example: Re-Fragmentation

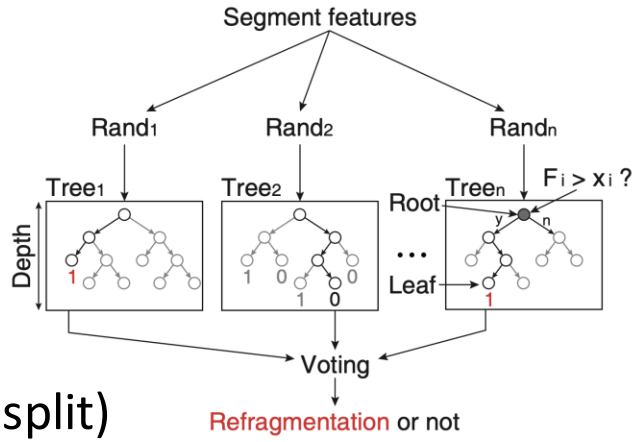
- OPC relies on fragmentation
  - Simple rules (e.g. nominal segment length) are usually applied



- Motivation
  - Only a few (critical) segments will cause many OPC iterations
  - Discover critical segments (with ML model) → they are further divided

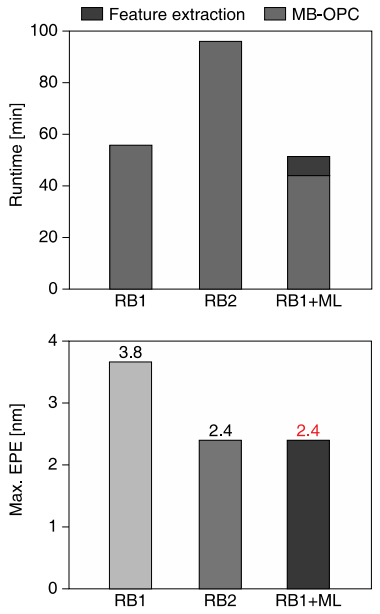
Re-Fragmentation with ML

- ML model: random forest classifier (RFC)



- Actual re-fragmentation
  - Each decision tree predicts 0 (non-split) or 1 (split)
  - Voting: split segment in half if sum of 1s > threshold

- Assessment
  - RB1: nominal length = 30nm → 7k segments
  - RB2: nominal length = 15nm → 13.8k segments
  - Proposed: RB1 + re-fragmentation → 7k + 93 segments

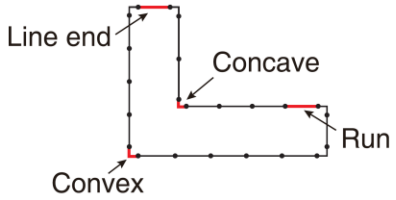


G. Cho, Y. Kwon, T. Kim, and Y. Shin, "Refragmentation through ML classifier for fast and accurate OPC," *Proc. SPIE Advanced Lithography*, 2022.

Rule-Based Re-Fragmentation

- Same amount of data (28k segments) is used to set up a few rules
  - σ for length and 2σ for |initial EPE|

Segment type	Length	|Initial EPE|
Line-end	>25nm	>31.9nm
Convex	>42nm	>7.4nm
Concave	>40nm	>11.8nm
Run (adjacent to corner)	>38nm	>8.8nm
Run (not adjacent to corner)	>44nm	>5.6nm



- Rule-based is worse (in max EPE) than RFC when data volume is enough

Refragmentation	Max. EPE [nm]	#Segments
No	3.83	7,000
RFC (big data)	2.42	7,096
Rules (big data)	3.07	7,163

RFC vs Rule-Based (in Small Data Volume)

- Sample segments are reduced: 28k → 1.4k
- RFC model is re-trained; rules are set up again

- Rule-based is better than RFC, this time
  - **RFC is over fitted**
  - **Rules are less sensitive to the amount of data**
  - **Carefully crafted "complex rules" (with hints from RFC) can be very good**

Segment type	Length	|Initial EPE| if $\phi_1 \leq 0.73$	|Initial EPE| if $\phi_1 > 0.73$
Line-end	>25nm	>29.4nm	>26.5nm
Convex	>42nm	>7.8nm	>7.0nm
Concave	>40nm	>10.2nm	>9.2nm
Run (adjacent to corner)	>37nm	>9.7nm	>8.7nm
Run (not adjacent to corner)	>44nm	>5.9nm	>5.3nm

Refragmentation	Max. EPE [nm]	#Segments
No	3.83	7,000
RFC (big data)	2.42	7,096
Rules (big data)	3.07	7,163
RFC (small data)	3.41	7,165
Rules (small data)	3.13	7,177
Revised rules (small data)	2.59	7,168

Summary

- ML is not always an ideal solution
- Rule-based (= heuristic) may be better than ML in small data volume