Novel End-to-End Production-Ready Machine Learning Flow for Nanolithography Modeling and Correction

Mohamed Habib   mohamed1611071@eng1.cu.edu.eg
Electronics and Electrical Communications Engineering Department,
Faculty of Engineering,
Cairo University,
Giza 12613, Egypt

Hossam A. H. Fahmy   hossam.fahmy@eng.cu.edu.eg
Electronics and Electrical Communications Engineering Department,
Faculty of Engineering,
Cairo University,
Giza 12613, Egypt

Mohamed F. Abu-ElYazeed   mfathyae@eng.cu.edu.eg
Electronics and Electrical Communications Engineering Department,
Faculty of Engineering,
Cairo University,
Giza 12613, Egypt

Corresponding Author: Mohamed Habib

Abstract

Mask optimization for optical lithography requires extensive processing to perform the Resolution Enhancement Techniques (RETs) required to transfer the design data to a working Integrated Circuits (ICs). The processing power and computational runtime for RETs tasks is ever increasing due to the continuous reduction of the feature size and the expansion of the chip area. State-of-the-art research sought Machine Learning (ML) technologies to reduce runtime and computational power, however ML-RETs are still not enabled for IC production flows yet. In this study, we analyze the reasons holding back ML computational lithography from being production ready. We present a novel flow that enables end-to-end mask optimization in addition to high scalability and consistency.

1. INTRODUCTION

Computational lithography aims to prepare synthesized integrated circuits (ICs) designs for accurate transfer into the semiconductor wafer. It performs elaborate correction flows that adds, subtracts

and modifies the original IC patterns with the end goal of minimizing the fabrication process errors. Such correction flows and operations are often referred to as Resolution Enhancement Techniques (RETs) and are known for being computationally expensive and time consuming.

As nanolithography technologies continue to advance towards ever decreasing minimum feature sizes, the same die area is able to host more design data. Henceforth, the data volume required to produce the same wafer area increase substantially, consequently, increasing the computational power and runtime required to perform the RET correction.

Computational lithography industry sought Machine Learning (ML) algorithms to accelerate the RET correction and reduce the computational load. However, ML-RET are not yet adopted for production to the best knowledge of the authors.

In this study, we analyze the reasons holding back ML-RET correction from utilization in RET production flows. Based on this analysis, we introduce a novel flow that mitigates those obstructions and offers an end-to-end production-ready platform for ML-RET with very high scalability. The rest of this paper is organized as follows: in section 2, we discuss the traditional way of performing RET correction. Next, we discuss and analyze the state-of-the-art of ML-RET and the roadblocks holding it from being a viable production option in section 3. We then introduce TPM-RET, a novel end-to-end production friendly ML flow to model and correct lithography in section 4, and showcase some of its results in section 5. Next, we discuss how TPM-RET flow solves the production difficulties in section 6. Finally, we present our future plans and conclusions in sections 7 and 8 respectively.

2. TRADITIONAL RET

Optical lithography machinery provided good accuracy early-on for large pattern dimensions. However, its accuracy and fidelity degraded as manufacturing requirements progressed towards smaller dimensions, even though such dimensions are still within the process theoretical limits. Literature indicates [1] that the influence of Optical, mechanical and chemical effects become more impactful as the printed features progress towards the theoretical lithography limits.

It is impractical to replace the production lines for every new lithography node, given the hefty time and monetary investments put into research and yield stabilization. Thus, it is the duty of advanced computational lithography and RET techniques to extend the lifespan of the existing infrastructure with little or no modification to the hardware.

RET techniques were invented gradually one after another as the need arose and extra correction steps were required. Such techniques are executed sequentially in standard tapeout flows that grew more complex with every technology advancement. Furthermore, each one of these techniques grew more complicated as the accuracy degradation grew more prominent. Here is a list of some famous RET techniques [2, 3]:

- Optical Proximity Correction (OPC)

- Sub-Resolution Assist Features (SRAFs)

- Off-Axis Illumination (OAI)

- Mask Process Correction (MPC)

- Phase-Shifing Masks (PSM)

In general, a RET technique operates in one of the following ways:

1. Modify the mask shape by moving its edges or introduce new features such that the results of the mask illumination is as close as possible to original target shapes.

2. Change the laser source shape or incidence angle to enhance contrast at wafer plane or give optimal illumination conditions to dominant mask feature configuration.

3. Alter photomask stack in order to introduce phase difference between bright and dark areas consequently offering better contrast for the wafer image.

The ever-growing computational, complexity and cost overhead of traditional RETs drive the industry to look for a faster and less computationally demanding approaches [4]. ML-RETs offer an attractive alternative to reduce the computational cost of the RETs, however such solutions are not yet adopted for RET production flows.

3. ML-RET STATE-OF-THE-ART ANALYSIS

In this section, we briefly discuss the current state-of-the-art ML-RET solutions and analyze the obstacles holding back this technology from reaching production-level despite the efforts expended.

3.1 State-of-the-Art Synopsis

Literature shows clear research directions and preferred methods for performing ML-RET correction. These directions are influenced by the nature of the RET correction itself as well as the latest advancements in ML industry especially in the image processing field. Based on our research, we summarize the key characteristics of the ML-RET meta as follows:

3.1.1 Image-based photomask correction

Recent state-of-the-art ML-RET methods use image-based input by converting design patterns into image slices [5, 7–9]. The ML model then translates the input images and transfers it to the optimized photomask domain to produce an image with the final photomask shapes.

3.1.2 Generative ML techniques

Generative Adversarial Networks (GANs) [10] and other advanced domain transfer ML techniques are the mainstream methods for state-of-the-art ML-RET research [6, 8, 9, 11]. This can be attributed

to the great advancements of image generation and translation techniques in recent years, henceforth, encouraging their usage to solve ML-RET as a domain transfer problem.

3.1.3 Single ML-RET applications

State-of-the-Art research focuses on ML specific treatment targeting a single RET application [5, 9, 12, 13]. OPC correction and SRAF insertion are the most researched techniques in that regards, since they are the most challenging and computationally expensive operations in the production tapeout flow.

3.1.4 Model-based reference data

Reference data required for ML model training is usually generated using state-of-the-art model-based RET solutions [6, 9, 12, 13]. Inverse Lithography Technology (ILT) [14] is often used to fulfill this role as it provides the theoretical best photomask for a given pattern.

3.2 Difficulties Facing Production-Ready ML-RET

3.2.1 Loss of mask information

The final photomask after RET correction is a binary pattern, where any pixel on it can be either bright or dark. Traditionally, the final photomask is obtained through a series of simulations and iterative correction until reaching acceptable accuracy.

For ML-RET model training, the binary photomask data is used to teach the model about the correction process and ideal corrected patterns. However, this means losing valuable information about the intermediate operations and obscures a lot of the physical and process relations. This, in turn, makes it hard for the ML-RET model to infer the correct relations governing the process behavior. Thus, the risk of over training increases and the reduces the ML models ability to handle never-seen patterns.

3.2.2 Full-chip scale issues

State-of-the-Art research focuses on the accuracy and runtime aspects of calibration and evaluation of ML-RET models. To the best knowledge of the authors, not much attention goes to enabling chip scale evaluation, which is a key aspect for enabling the technology. Since a full chip is too large to be processed as a single image, the need arise for splitting the chip into smaller slices and stitching them back after correction is done.

For a ML-enabled process, slicing the chip data is not a simple task especially for when using GANs and image translation techniques. The recurring patterns should be put consistently in the same location with respect to its window slice to make sure the correction results are consistent.

The reassembly of the chip from the corrected images slices is yet another nontrivial task that calls for complex post-processing rules. First, it requires stitching back polygons spanning across multiple windows together, and consequently, resolving any conflicts that appear at window borders. Furthermore, more post processing is required to resolve any MRC[1] violations that may appear if polygons in separate windows come too close after stitching them together.

3.2.3 Correction consistency

Consistency is an important aspect for IC manufacturing to ensure similar treatment for same patterns across a chip scale correction. This is especially desired for designs based on symmetrical device topologies in order to ensure uniform device performance, reliability and minimize electrical stress.

The topological features around the target pattern is a main factor that dictates its correction. Slight shifts in the correction window, i.e due to chip slicing in an GAN-based correction, can lead to a different ML-RET response to the same pattern resulting into inconsistent correction.

The stitching operations during full-chip assembly can be another source of correction inconsistency. This happens in the cases where the post-processing algorithm applies different treatments to the polygons near windows border while resolving stitching conflicts or MRC violations.

3.2.4 Hardware requirements

Traditional RET correction requires massive computational resources, hence FABs usually own huge processing grids with thousands of CPU cores as well as advanced monitoring and allocation management systems. ML models, however, are best trained and utilized in a GPU-based environment.

To adopt state-of-the-art ML-RET flows, FABs need to replace their CPU-based infrastructure and reinvest in a GPU-based one. This creates great deal of resistance in the adoption of ML-RET due to the accumulated experience, time and fund investments in the older infrastructure.

3.2.5 Pattern interaction distance

The physical phenomena affecting the pattern transfer from photomask to wafer depends heavily on the interactions between neighboring patterns. Computational lithography tools usually have a specified distance where such pattern interactions have an impactful contribution, usually referred to as interaction distance or radius.

Traditional RET tools calculate the interaction distance uniformly around each pattern. This is also true for ML-RET models that utilize uniformly collected features around the corrected pattern, such as models based on Support Vector Machines (SVM), as the feature collection algorithm can be programmed to restrict measurements to the correct interaction distance.

---

[1] Mask Rule Checks: Special rules defined by mask house to make sure the photomask is manufacturable.

On the other hand, GANs and similar ML-RET models correct for a complete windows as whole, thus the whole window can be considered as interacting features to any pattern inside it. This poses two issues, first, patterns at window edges will have unsymmetrical interaction distance. The second issue is that similar patterns will have variable interaction distance according to their position in the window. This can lead to variability in the correction results and further contribute to inconsistency.

3.2.6 Input resolution

Traditional RET techniques as well as feature-based ML-RET, use the IC design data directly to perform the correction or measure layout features respectively. The layout resolution in this case determines the refinement of the edge displacement which can bring forth better accuracy.

GAN-based ML-RET models, contrarily, require the conversion of the IC layout to image to be able to process the data. This adds an extra dimension to the correction process. Using too coarse image resolution sabotages the accuracy of the correction due to the information loss, making the model unable to differentiate between pattern configuration. While using too fine image resolution harms the correction runtime as well as increases the chance of model overfitting and losing the ability to generalize over the input samples.

3.2.7 Never-seen pattern integration

Traditionally, a never-seen pattern does not impact process modeling. If such pattern failed to post RET verification, the RET correction recipe is then fine tuned to account for it, or in some rare occasions, it can be entirely disallowed using design rules and hotspot detection tools.

Ml-RET models, on the other hand, are built based on the model experience with patterns that are relatively similar to real-life designs. Hence, model retraining may be necessary to account for such patterns. RET models are the most central part of the correction process, their calibration and fine tuning takes effort, experience and most importantly test photomasks to perfect. Therefore, having to retrain the models to include patterns that are outside of the seen data space is a huge overhead.

3.2.8 Partitioning of RET correction

Traditionally, RET correction is done in stages in reverse order of the process manufacturing steps. This staged approach is due to RET techniques being introduced overtime and the CAD industry formulating the correction using such techniques as separate building blocks comprising a big flow.

This staged approach is, however, heavily influencing the correction flow in ML-RET state-of-the-art research. The partition of the ML-RET flow adds extra multiple layers of complexity, requiring dataset preparation, data engineering, model training and verification for every part of the correction flow.

4. TPM-RET: NOVEL PRODUCTION-FRIENDLY ML-RET FLOW

We present True Pixel-based Machine-learning RET (TPM-RET), a novel production-friendly flow for the modeling and correction of the lithography process using convolutional neural network. TPM-RET framework addresses the concerns discussed in section 3, providing a CPU-scalable, end-to-end, consistent solution all while being full-chip ready.

In the next subsections, we discuss the design choices defining the TPM-RET flow. The main focus of such discussions is to demonstrate the benefits of this novel flow and how it addresses the aspects that makes ML-RET not appealing for mass-production.

4.1 Design Choice #1: True Pixel-Based Correction

Modern state-of-the-art methods use GANs to correct chunks of the photomask at one go using input image clips of the design data. This correction style is, in fact, pixel-based, however it generates conflicts at split boundary and consistency distrust due to correction window shifts as previously discussed in section 3.2.3. To avoid such issues, gain more control over the correction scheme and allow for flexible image resolution, we opt to apply correction for each pixel of the photomask separately, one at a time.

4.2 Design Choice #2: Minimum Model Footprint

To reduce any runtime issue from the pixel-by-pixel correction, we choose to use the minimum model that yields good accuracy. Thus, we choose Convolutional Neural Network (CNN) as the model structure of choice for TPM-RET, due to its efficiency in image classification and low computational power requirements.

4.3 Design Choice #3: Inverse Intensity Profile (IIP)

Inverse Lithography Technology [ILT] [14] solves the lithography optimization problem inversely. Assuming that we have the ideal silicon pattern $T(x, y)$ at the wafer level, we calculate the optimum photomask pattern $M^*(x, y)$ that when put at the scanner will yield such ideal silicon pattern. In other words, the ILT tries to solve the lithography problem by calculating the inverse of the manufacturing process transfer function $F()$. Hence, we can calculate the optimal photomask M^* by solving the equation (1):

$$M^*(x, y) = F^{-1}(T(x, y)) \tag{1}$$

FIGURE 1 shows an example test pattern and its corresponding ILT photomask calculated using Calibre® pxOPC® [15]. While the $M^*(x, y)$ is the clearest option for photomask pixel classification, the binary nature of the photomask conceals the process related information which makes it challenging for the model to infer it while training.
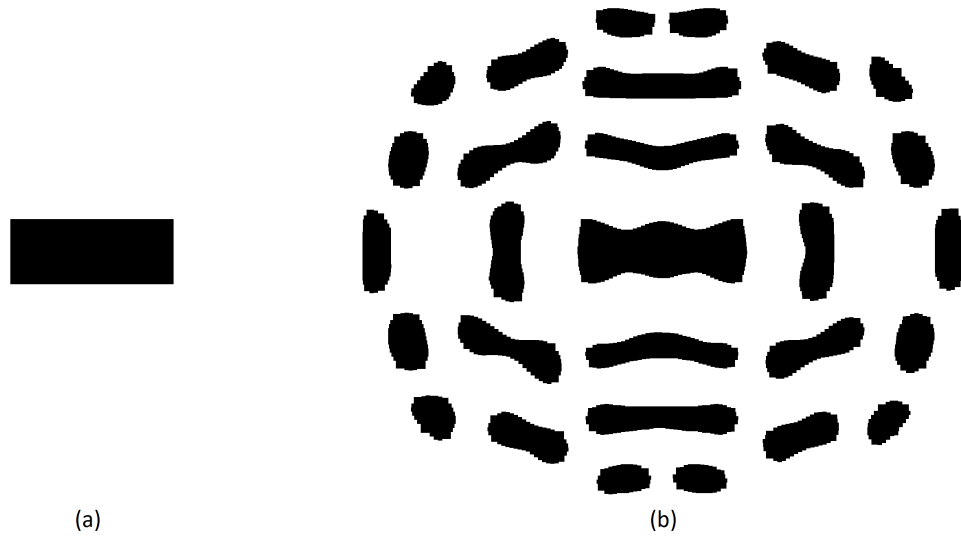
<div align="center">(a)   (b)</div>

Figure 1: Example of a simple target wafer pattern T (a), and its corresponding inverse lithography ideal photomask M^* (b)

We introduce Inverse Intensity Profile [IIP], a novel method to recover the process information hidden inside the binary photomask function. We calculate $IIP(x, y)$ as the spacial convolution product of the binary mask function $M^*(x, y)$ and a predefined Inverse Intensity Kernel (IIK) $K(x, y)$ as shown in (2):

$$IIP(x, y) = \frac{M^*(x, y) \circledast K(x, y)}{||M^*(x, y) \circledast K(x, y)||} \tag{2}$$

The function $IIP(x, y)$ is a continuous function within the $[0, 1]$ range and it provides information about the photomask transitions between "opaque" and "transparent" states. The kernel $K(x, y)$ is an optimization parameter that represents the process characteristics. FIGURE 2 shows the IIP map corresponding to the pattern in shown in FIGURE 1. Using the introduced IIP allows the model to learn the process underlying physics during the training phase and hence have more stability and coherence than when using the original binary function.

4.4 Design Choice #4: Nonuniform Image Compression

A halo region around corrected pattern, a single pixel in our case, is required to provide spacial awareness about its neighborhood during the correction process. Usually, such halo is referred to as Interaction Distance (ID) and it is the furthest distance at which another pattern can affect the correction process. ID is an optimization parameter and is selected based on the required level of accuracy in the final photomask. For a standard 32nm immersion lithography process, typical ID is around 1.2um and can be tuned according to accuracy requirements.
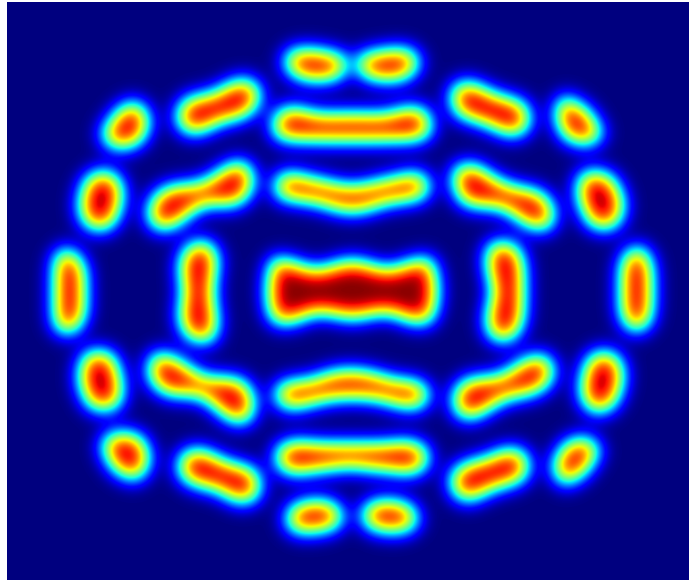
Figure 2: IIP map corresponding to target pattern in FIGURE 1.

Assuming we take a resolution of 1 pixel per $1nm^2$, an image of size 2001×2001 pixels is required to account for a ID of $1um$ for every processed pixel. Such image size is too large to use as input for a ML model as it can slow the runtime and cause overfitting. Increasing the number of pixels per nm^2 will further distend the problem.

To fix this issue, we use nonuniform image compression that uses a different compression algorithm for vertical and horizontal pixel arrangements. This is to retain any directional asymmetries in the lithography process and allow the model to distinguish different pattern orientations more easly. Using this compression technique, an image of size 2001×2001 will be reduced to 250×250 pixels.

4.5 Design Choice #5: End-to-End Correction

Partitioning the ML-RET correction flow adds unnecessary complexity while lacking the historical justifications that made such approach acceptable for traditional RET flows. Furthermore, cascading the output of ML-RET into another can magnify the error. We opt to executing the whole ML lithography correction in one shot as an end-to-end flow, especially that the required input data for all ML-RET stages already exist in the input images and the IIP map. The benefits of using an end-to-end flow also includes the simplification of dataset engineering, reduction of training and verification times, lenient hardware requirements, and appeals to the ML models power to fit complex patterns.

4.6 TPM-RET FLOW ASSEMBLY

Advancing from the previous discussion, we assemble the design pieces into the TPM-RET flow. We can split the flow into two phases:
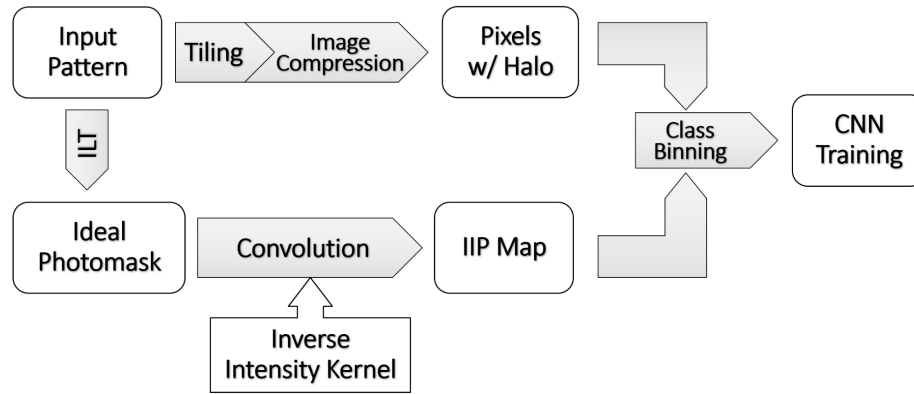

Figure 3: Flow chart for TPM-RET data preparation module.

4.6.1 Data preparation and model training

FIGURE3 shows the flow chart for the data preparation phase for the TPM-RET platform. The input pattern is used to extract the pixel images by first converting it into a high resolution image then tile and compress it in order to obtain the images corresponding to the training pixels data.

Simultaneously, the test pattern is corrected via Calibre® pxOPC® to obtain the ILT photomask. The IIP map is obtained as a function of the photomask domain, $IIP(x, y)$, by convolving the selected IIK $K(x, y)$ with the ILT photomask $M^*(x, y)$.

The images are then paired to their corresponding IIP value using their pixel coordinates, such that an image corresponding to pixel $p(x_p, y_p)$ is coupled with the value $IIP(x_p, y_p)$. The obtained data are next assigned to the classes according to their $IIP(x, y)$ value and saved for ML model training. For the sake of this study, we use 100 IIP classes. The number of classes depends on whether the intended application requires a fine IIP map or not. Increasing the number of classes does not affect the runtime for same samples count, but will also require more training samples to populate all the classes.

Once the data collection is finished, we use the Tensorflow [16] and Keras [17] libraries to construct and train the CNN model. The training data are split into three unique sets, the training and validation sets are used for the model training loop while the third set is used for testing the calibrated model and ensure its accuracy after the training is completed.

Table 1: TPM-RET Settings.

Parameter Name	Value
# Classes	100
Pixels/nm^2	4
ID	400nm
IIK	Optimized 32nm IIK

4.6.2 Model deployment and post processing

FIGURE 4 shows the flow chart for the model deployment and post processing phase. First, the target pattern is tiled into smaller images representing the corrected pixels and their respective ID distance. Such images are then compressed to match the input size of the CNN model.

The scaling manager organizes the distribution of the pixels data and the predicted IIP class to and from the CNN model instances respectively. The CNN model evaluates the pixel data $p(x_p, y_p)$ and predicts the IIP class equivalent to it $IIP(x_p, y_p)$. The calculated IIP values are then arranged in the respective order of their input pixels, which ultimately forms the complete IIP map corresponding to the target pattern.

The IIP map is then converted to a binary pattern by applying a threshold to it, hence defining the output clear-cut delineated geometries. Post-processing is then applied to convert the obtained geometries to standard layout format and apply any required clean-ups to meet restrictions on minimum pattern area or edge length.
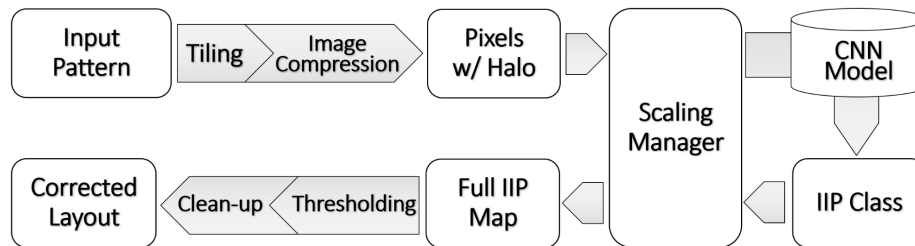

Figure 4: Flow chart for model deployment in photomask optimization application.

It is worth noting that for some applications the IIP map itself can be the desired outputs, such as electrical stress or heat map, while others require further processing to obtain binary output representation such as the corrected photomask in case of RET applications.

5. CASE STUDY

For the sake of this paper, we demonstrate the TPM-RET flow by comparing end-to-end results based on OPC and SRAF correction of a 32nm metal test-pattern. The reference tool is Calibre™ pxOPC™ which is used to generate the ideal photomasks.
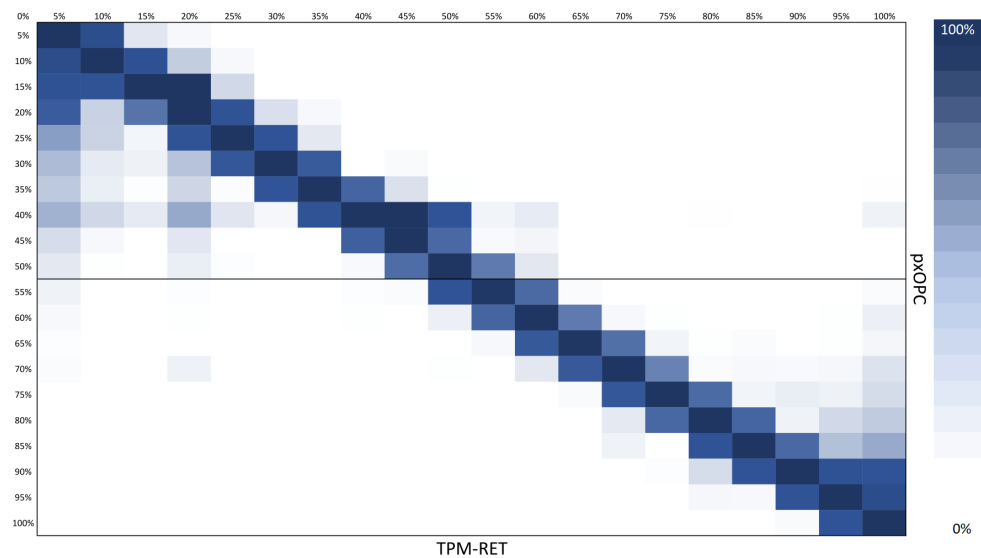
Figure 5: Confusion Matrix for TPM-RET IIP predictions versus the IIP reference generated from pxOPC photomask.

The test patterns are divided into groups according to topology, width and space. The model training flow only uses pixel data from the 40nm and 140nm isolated line and line-space patterns. The patterns from other dimensions and topologies are used only for testing and results comparison.

The TPM-RET flow prototype is implemented using Python 3.9 [18] and Calibre™ DesignRev™ internal scripting language. ML model training is done using Tensorflow and Keras, and the model of choice for this case study is mobileNetV3 [19]. The TPM-RET flow settings for the training loop are listed in TABLE 1.

Figure 6: TPM-RET flow results comparison with the reference pxOPC™ for Isolated patterns.
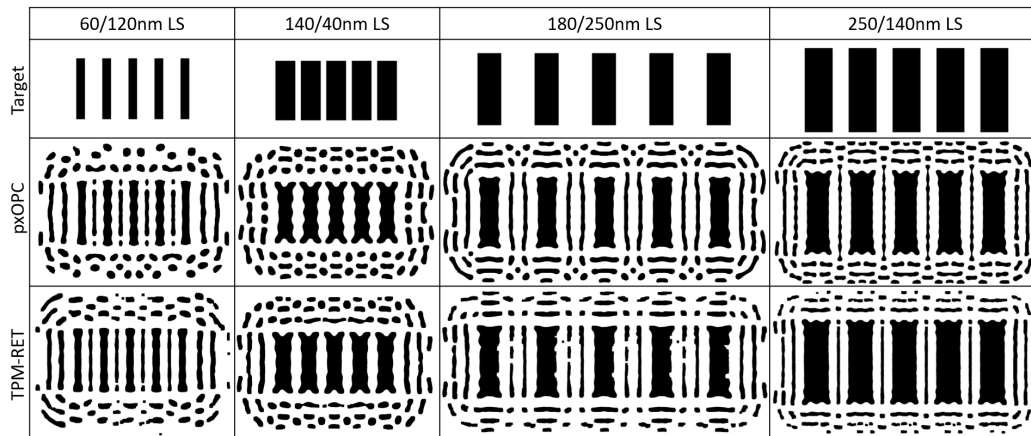
Figure 7: TPM-RET flow results comparison with pxOPC™ for Line-Space patterns.

The TPM-RET for data-preparation, IIP map prediction and cleanup was run on a 128 Intel® Xeon® CPU E7-4830 2.1GHz CPU cores and 384GB RAM machine. The ML model training was done on a separate machine with an Nvidia® Quadro® RTX 4000 GPU and 32GB RAM, which is the only step in this prototype that needs to be executed in a GPU-based environment.

The testing dataset is used to generate a confusion matrix, FIGURE 5, to compare the accuracy of TPM-RET versus the reference pxOPC™. The matrix shows well correlation with the reference results and illustrates good ML model stability.

TPM-RET flow is also used to perform end-to-end correction for the test patterns. FIGURE 6 shows the flow results for isolated lines and contacts, while FIGURE 7, shows its results for line-space patterns. The results show the correlation between the pxOPC™ and TPM-RET photomasks for various pattern dimensions although the mobileNetV3 model was only trained for $40nm$ and $140nm$ line-space and isolated line patterns. TABLE 2 shows a runtime comparison between TPM-RET and pxOPC™ for a $5 \times 5um$ test pattern.

Table 2: TPM-RET Runtime Comparison Against pxOPC.

Tool	pxOPC	TPM-RET	
	128 CPUs	32 CPUs	128 CPUs
runtime (sec)	314	1105	283

6. DISCUSSION: TPM-RET ADVANTAGES

6.1 Scalability

The true pixel based design, section 4.1, allows the model evaluation for any given pixel to be independent on neighboring pixels. We take advantage of that by distributing the execution over

multiple processing units efficiently. Furthermore, the full-chip computation process can be distributed over CPU grids, since the TPM-RET flow use computationally sparing ML models as discussed in section 4.2. This is a great advantage as silicon foundries owns huge CPU grid systems and it is their preferred execution platform.
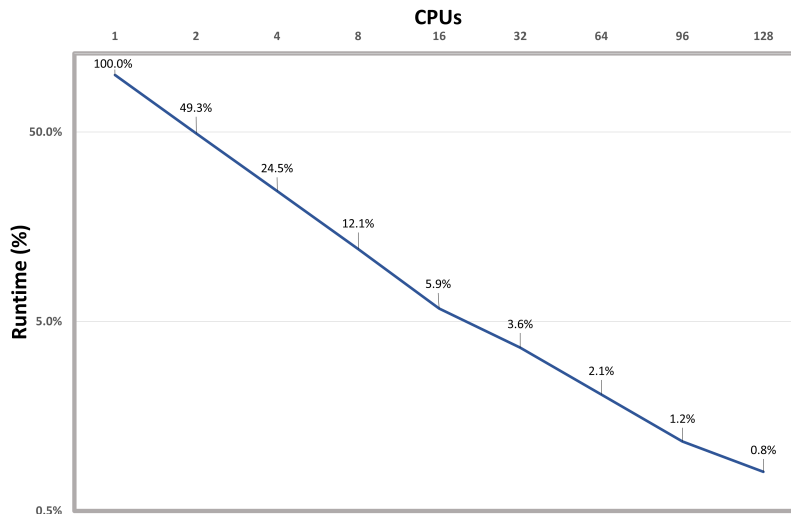

Figure 8: Logscale chart for TPM-RET runtime scaling versus number of CPUs.

TPM-RET flow scales well with great efficiency as shown in the log-scale chart in FIGURE 8, as the number of CPUs increase. The chart shows a 125x runtime speed-up running when using 128 CPUs versus a single CPU, providing over 97% scaling efficiency. We foresee the TPM-RET flow scaling at the same near 1/n rate for larger number of CPU units. The experiment was done on the same 128 CPU machine from section 5.

The distribution process it self is controlled by the scaling manager module, shown in fig 4. This module is implemented using custom python code to manage the slicing and reassembly of chip data with minimal runtime overhead.

6.2 Consistency

TMP-RET flow does not use panning and window slicing to fit the input patterns into its input size. This eliminates the boundary conflicts and the need for stitching patterns spanning across multiple correction windows, hence the flow will not generate inconsistent correction due to chip slicing.

Furthermore, the TPM-RET flow does not generate mismatching halo around identical patterns, since handling the input patterns on a single pixel basis eliminates the case of asymmetrical ID. This guarantees identical halo around identical pixels, which in turn necessitates same inference result.

6.3 Re-Correction

When a photomask error is caught in a traditional RET flow, a photomask correction process is performed by either tuning the correction recipe and rerunning the flow unto the erroneous areas or manually if the change scope is manageable.

TPM-RET allows for such re-correction process by allowing a fine-tuned ML model to run over the pixels representing the erroneous areas. Furthermore, the TPM-RET flow can handle multiple ML models performing correction at different chip areas by defining the execution ranges of operation for each of them.

6.4 Full-Chip Ready

TPM-RET flow addresses concerns regarding ML-RET full-chip processing. First, it does not require stitching or window splitting, which eliminates most of the post processing optimizations and also clears the inconsistency associated with them as discussed in 6.2.

The flow is also able to scale over large number of CPUs due to its true-pixel-based nature. This appeals to the CPU infrastructure preferred at silicon foundries, hence eliminating the friction of adopting GPU based infrastructure as a necessity to use ML-based solutions.

6.5 Recapturing Obfuscated Process Information

The data available in a reference photomask is binary where as any pixel on such mask can be opaque or bright. Even if the reference is an ideal ILT photomask, the information of the process and the intermediate steps leading to this mask are concealed.

The IIP algorithm allows the TPM-RET flow to uncover the information masked underneath the binary photomask and provides a gradual profile describing the transitions between the bright and opaque areas as discussed in section 4.3. This gives the model more information to work with and makes it less prone to overfitting.

6.6 End-to-End Solution

As discussed in 3.2.8, traditional RET flows splits the mask correction into consequent steps due to historical and flow design reasons. ML-RET applications do not have to follow such footsteps, especially with the complications and potential accuracy setbacks that come with it.

With that in mind, TPM-RET flow is designed as an end-to-end framework for any lithography applications as long as it initiates from an input layout pattern and ends with a similar sized output.

6.7 Flexibility

TPM-RET platform can handle different accuracy and resolution settings with minor or no changes. Prediction accuracy can be controlled by tuning the number of classes, the number of pixels per nm^2, the ID and the IIK. Such changes can be accounted for without any major modifications to the flow.

TPM-RET flow can as well as fit plethora of full chip applications. TPM-RET flow can account for different correction and modeling tasks by replacing the "ideal photomask" block in FIGURE 3, with the desired reference and selecting a proper IIK. For a layout simulator, the reference should be an accurate simulation of the target pattern and a suitable IIK to generate the corresponding IIP map. Similarly, an electrical stress simulator takes a full-chip stress map as its ideal reference which can also be used as the IIP map, and the flow output will not require thresholding or clean-up in that case.

7. FUTURE WORK

Our future work plans for TPM-RET include the following items:

1. Convert the TPM-RET Python code to C/C++ [20] to improve execution runtime and optimize the model evaluations for CPU platforms.

2. Develop an algorithm to filter and select the pixels data generated from the TPM-RET data-preparation module.

3. Implement re-correction and multi-model execution features.

8. CONCLUSIONS

In this study, we reviewed the traditional RET techniques as well as ML-RET state-of-the-art. We highlighted the main obstacles preventing production-ready ML-RET technologies. We then introduced TPM-RET flow, a novel ML platform capable of performing end-to-end lithography mask optimization in a consistent and production friendly manner while achieving a near 1/n CPU scalability. We demonstrated the flow based on a 32nm immersion lithography technology test patterns by performing an end-to-end correction targeting both OPC and SRAF. Finally, we shared our future work and improvement plans for the TPM-RET flow.

References

[1] Mack C. Fundamental Principles of Optical Lithography: The Science of Microfabrication. John Wiley & Sons; 2007 Dec 17.

[2] Wong AK. Resolution Enhancement Techniques in Optical Lithography. SPIE press. 2001.

[3] Lin T, Donnelly T, Schulze S. Model Based Mask Process Correction and Verification for Advanced Process Nodes. In Optical Microlithography XXII. SPIE. 2009;7274:761-770.

[4] Fritze M, Tyrrell B. Prospects and Challenges of Optical RET.(Lithography Part I of a Series). Solid State Technology. 2003;46:61-65.

[5] Yang H, Li S, Ma Y, Yu B, Young Y, et al GAN-OPC: Mask Optimization With Lithography-Guided Generative Adversarial Nets. In: 55th ACM/ESDA/IEEE Design Automation Conference (DAC). San Francisco: IEEE Publications. 2018:1-6.

[6] Chen G, Yu Z, Liu H, Ma Y, Yu B, et al. Develset: Deep Neural Level Set for Instant Mask Optimization. In: IEEE/ACM International Conference On Computer Aided Design (ICCAD). Munich, Germany: IEEE Publications. 2021:1-9.

[7] Ciou W, Hu T, Tsai YY, Hsuan CT, Yang E, et al. Machine Learning OPC With Generative Adversarial Networks. In: Lafferty NV, Kim R-H, Editors. DTCO and computational patterning. San Jose: SPIE. 2022;12052:216-235.

[8] Chen G, Chen W, Sun Q, Ma Y, Yang H, et al. DAMO: Deep Agile Mask Optimization for Full-Chip Scale. IEEE Trans Comput Aid Des Integr Circuits Syst. 2022;41:3118-3131.

[9] Alawieh MB, Lin Y, Zhang Z, Li M, Huang Q, et al. GAN-SRAF: Gan-Sraf: Subresolution Assist Feature Generation Using Generative Adversarial Networks. IEEE Trans Comput Aid Des Integr Circuits Syst. 2021;40:373-385.

[10] Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, et al. Generative Adversarial Networks. 2014. Arxiv Preprint: https://arxiv.org/pdf/1406.2661.pdf

[11] Ciou W, Hu T, Tsai Y-Y, Hsuan T, Yang E, et al. SRAF Placement With Generative Adversarial Network. In: Owa S, Phillips MC, editors. Optical Microlithography. Vol. XXXIV. SPIE [online only]. 2021;11613:1161305.

[12] Shim S, Shin Y. Etch Proximity Correction Through Machine-Learning-Driven Etch Bias Model. SPIE Proceedings. 2016;9782:58-67.

[13] Sharma M, Durvasula BS, Rao N, Bork I, Sharma R, et al. Machine Learning Guided Curvilinear Mpc. In: Rankin Jh, Preil Me, Editors. Photomask Technology. Monterey: SPIE. 2019;11148:115-121.

[14] Pang LL. Inverse Lithography Technology: 30 Years From Concept to Practical, Full-Chip Reality. J Microgr/Nanopatterning. 2021;20:030901.

[15] https://eda.sw.siemens.com/en-US/ic/calibre-manufacturing/computational-lithography/

[16] Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, et al. Tensorflow: Large-Scale Machine Learning on Heterogeneous Systems. 2015. Software available: https://www.tensorflow.org/

[17] https://keras.io

[18] Van Rossum G, Drake Jr FL. Python Reference Manual. Centrum voor wiskunde en Informatica Amsterdam. 1995.

[19] Howard A, Sandler M, Chu G, Chen L-C, Tan M et al. asudevan, V. Le, Q.V., Adam, H:
      Searching for MobileNetV3. 2019. ArXiv preprint: https://arxiv.org/pdf/1905.02244.pdf

[20] {https://www.lirmm.fr/ ducour/Doc-objets/ISO+IEC+14882-1998.pdf