pNeurFill: Enhanced Neural Network Model-Based Dummy Filling Synthesis With Perimeter Adjustment

Zhaoting Chen, Junzhe Cai, Changhao Yan, *Member, IEEE*, Zhaori Bi, *Member, IEEE*, Yuzhe Ma, *Member, IEEE*, Bei Yu, *Senior Member, IEEE*, Wenchuang Hu, *Senior Member, IEEE*, Dian Zhou, *Member, IEEE*, and Xuan Zeng, *Senior Member, IEEE*

***Abstract*—Dummy filling is widely applied to significantly improve the planarity of topographic patterns for the chemical mechanical polishing (CMP) process in VLSI manufacturing. In the dummy filling flow, dummy synthesis works as the key step to adjust the post-CMP profile height. However, existing dummy synthesis optimization approaches usually fail to balance the filling quality and efficiency. This article proposes a novel model-based dummy filling synthesis framework NeurFill, integrated with multiple starting points-sequential quadratic programming (MSP-SQP) optimization solver. Inside this framework, a full-chip CMP simulator is first migrated to the neural network, achieving 8134× speedup on gradient calculation by backward propagation. Entrenched in the CMP neural network models, we further implement an improved version of NeurFill (pNeurFill) to alleviate the post-CMP height variation caused by dummy perimeter. After each iteration of dummy density optimization, an additional perimeter adjustment based on a given candidate dummy pattern set is applied to search for the optimal perimeter fill amount. The experimental results show that the proposed NeurFill outperforms existing rule- and model-based methods. The extra perimeter adjustment strategy in pNeurFill can achieve an average 66.97 Å decreasing in height variation and 8.92% quality improvement compared to NeurFill. This will provide guidance for DFM so as to increase IC chip yield.**

***Index Terms*—Chemical mechanical polishing (CMP), design for manufacturability, metal fill synthesis, neural network, sequential quadratic programming (SQP).**

Manuscript received 14 December 2022; revised 7 May 2023 and 5 September 2023; accepted 24 September 2023. Date of publication 28 September 2023; date of current version 22 January 2024. This work was supported in part by the National Key R&D Program of China 2020YFA0711900 and Grant 2020YFA0711904, and in part by the National Natural Science Foundation of China (NSFC) Research Projects 61974032, 62141407, 62304052, 61929102, 61674042. The preliminary version has been presented at the 58th ACM/IEEE Design Automation Conference (DAC) in 2021 [DOI: 10.1109/DAC18074.2021.9586325]. This article was recommended by Associate Editor I. H.-R. Jiang. *(Corresponding authors: Changhao Yan; Zhaori Bi; Xuan Zeng.)*

Zhaoting Chen, Changhao Yan, Zhaori Bi, and Xuan Zeng are with the School of Microelectronics, State Key Laboratory of Integrated Chips and System, Fudan University, Shanghai 201203, China (e-mail: yanch@fudan.edu.cn; zhaori_bi@fudan.edu.cn; xzeng@fudan.edu.cn).

Junzhe Cai is with the Morefun Studios, Tencent Inc., Shanghai 201103, China.

Yuzhe Ma is with the Microelectronics Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511458, China.

Bei Yu is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong.

Wenchuang Hu is with the Precision Medicine Center, West China Hospital, Sichuan University, Chengdu 610017, China.

Dian Zhou is with the Department of Electrical Engineering, The University of Texas at Dallas, Richardson, TX 75080 USA.

Digital Object Identifier 10.1109/TCAD.2023.3320629

I. Introduction

CHEMICAL mechanical polishing (CMP) has been an indispensable operation in the modern semiconductor industry for oxide dielectric and metal layer planarization. Typically, the post-CMP profile heights depend on the distribution of metal patterns in polishing windows and vary largely within one layer of the polished layout. This varying effect can accumulate through layers and cause a yield decrease in IC manufacturing, especially in the very deep-submicrometer and very large-scale multilayer interconnection process. To mitigate the problem, dummies are inserted into the layout to control post-CMP topography. These dummies do not directly influence the functionality of the circuit [1], but will inevitably induce extra parasitic capacitance resulting in undesirable circuit performance degradation, such as timing degradation [2], [3]. Thus, dummy filling algorithms are supposed to increase chip yield while trying to maintain acceptable circuit performance.

Dummy filling flow classically includes two stages, namely, dummy synthesis and dummy insertion [4]. Dummy synthesis optimizes the fill amount in each filling window to improve metrics of layout planarity and circuit performance degradation. Dummy insertion determines the dummy location in each filling window to satisfy design rules and minimize induced parasitic capacitance. Estimating parasitic capacitance of critical nets to promote post-CMP circuit performance during dummy insertion has been widely investigated [5], [6], [7], [8], [9]. In this article, we focus on dummy synthesis optimization, which dominates the planarity of the polished layout.

The methods of dummy synthesis are categorized as rule-based and model-based. Rule-based methods adopt empirical knowledge of the CMP process to guide optimization, which is normally based on rules of density variance, density gradient, etc. Kahng et al. [10] presented a practical rule-based formulation with which linear programming (LP) was manipulated to solve the filling problem of a fixed r-dissection layout. Then, an improved variant with an increased number of adjacent density calculation windows was proposed by Chen et al. [11].

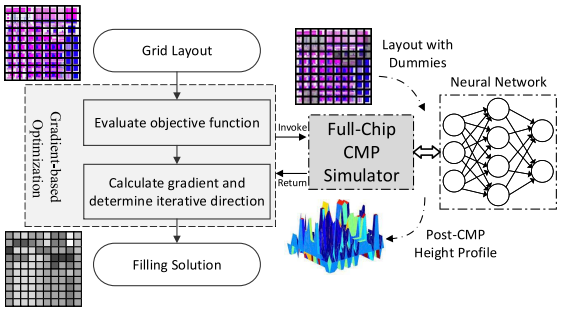

Fig. 1. Flow of model-based dummy filling synthesis and the motivation of leveraging neural network-based CMP simulators.

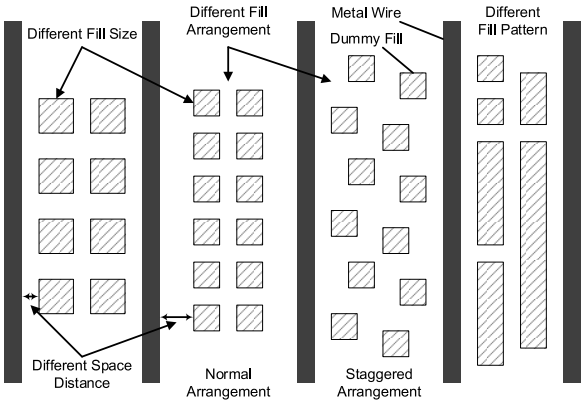

Fig. 2. Different fill modes, i.e., fill size, buffer space distance, fill arrangement, and fill pattern, can change surface topography and coupling capacitance after the CMP process [22].

However, these two methods only considered the relationship between interlayer-dielectric (ILD) thickness and local metal density. To achieve better quality, multiple metrics, such as overlay, line deviation, and outliers, were introduced to comprehensively evaluate the filling result [12], [13], [14], [15]. With the advancement of technology nodes, the intrinsic incompleteness of empirical rules limits the rule-based filling quality [16].

Model-based methods leverage analytical expressions to describe the phenomena like physical abrasion and chemical erosion in the CMP process. Tian et al. [17] proposed a two-step model-based filling algorithm, in which an LP problem was constructed for global density assignment that is followed by a local dummy feature adjustment. The approach generated more uniform post-CMP topographies than the results of rule-based methods. A novel electro-chemical plating (ECP) model-based dummy filling algorithm proposed by Sinha et al. [18] significantly reduced the post-CMP height variation and the amount of inserted dummies. A dummy synthesis model proposed by [19] enables implementation of complex design guidelines, and is also suitable for standard fill patterns in the insertion stage. However, model-based methods usually simplify those complicated models for efficiency. As a result, the filling quality is influenced by the accuracy of adopted models.

Recently, Cai et al. [16] found that it was unnecessary to be limited to semi-empirical or analytical ECP/CMP models. Instead, they implemented a dummy filling framework integrating a calibrated full-chip CMP simulator, and a sequential quadratic programming (SQP) algorithm was engaged to optimize the quality score computed via the simulation result. Full-chip CMP simulators have become mature sign-off tools in manufacture under 45 nm so that accuracy is guaranteed. Therefore, this simulator model-based approach gained better post-CMP layout planarity than other model-based methods.

Fig. 1 illustrates the basic flow of full-chip CMP model-based dummy filling synthesis. The target layout is divided into thousands of uniform windows, so the optimization problem is extremely high dimensional. Inside the optimizer, the post-CMP height profile is generated by a full-chip CMP simulator and evaluated by the metrics. Numerical gradients of the model-based metrics are used to guide the optimization, which is time-consuming. Huge invocations of CMP simulators on numerical gradient calculation have become the bottleneck of model-based filling synthesis. Besides, CMP is necessary for sub-65-nm process, and calibration of model parameters can be laborious.

In fact, massive simulator invocations can be easily observed in other model-based optimization problems, e.g., the optical proximity correction (OPC) problem. Recently, Jiang et al. [20] proposed Neural-ILT for model-based OPC problem, leveraging neural network-based simulation for acceleration. Motivated by their work, we develop NeurFill to solve the efficiency bottleneck of model-based dummy filling synthesis. Because of the similarity to the image segmentation problem and the local effect of the CMP process, the full-chip CMP simulator is migrated to a pretrained neural network as Fig. 1, where gradient calculation can be efficiently performed by backward propagation. A preliminary version of NeurFill was published in [21].

However, only considering dummy density is not enough for the post-CMP chip planarity. As illustrated in Fig. 2, different fill modes, i.e., fill size, buffer space distance, fill arrangement, and fill pattern, have been verified to change surface topography and coupling capacitance after CMP process [22], [23], [24], [25]. For example, the dummies in the first and second columns have different fill size and buffer space distance. The last column has different patterns. The second and third columns have the same size but with different fill arrangement. Among these fill modes, fill size and fill pattern fundamentally reflect the ratio of area and perimeter of dummies. Thus, dummy perimeter should be seriously taken into consideration in dummy synthesis. Other factors, buffer space and fill arrangement not included in this article, should be considered in the phase of dummy insertion.

In this article, we further propose pNeurFill, an enhanced version of NeurFill, to take dummy perimeter along with different fill patterns into consideration. To obtain high-prediction accuracy in the new scenario, the neural network is trained by a dataset that concerning extra perimeter variation. We extend the original optimization problem formulation with expanded perimeter space and additional density and perimeter bounded constraints. To solve this extended optimization problem with both density and perimeter, we insert a perimeter adjustment step after the density optimization step in the original optimization loop. Based on the optimization result in each

filling window, we propose a heuristic dummy pattern allocation algorithm to determine the amount of each fill pattern from the given candidate pattern set (CPS).

Our main contributions are summarized as follows.

1) We propose NeurFill, a model-based dummy filling synthesis framework based on a neural network. After migrating the conventional full-chip CMP simulator into a GPU-based neural network, we can achieve 188× speedup on objective evaluation and 8134× speedup on gradient calculation.
2) Prior knowledge-based starting point is leveraged for fast dummy filling synthesis. SQP optimizer is applied to obtain better-filling quality due to the speedup in gradient calculation.
3) To further improve the filling quality, the original optimization problem is extended with extra optimization dimension of the perimeter, and pNeurFill framework is proposed to add a perimeter adjustment step in the NeurFill flow.
4) Experimental results show that the proposed NeurFill can achieve significant improvement in filling quality and overall score over traditional rule-based and model-based methods. Compared to the state-of-the-art simulator model-based method [16], NeurFill obtains average 137× speedup with comparable quality score. Besides, the refined pNeurFill achieves remarkable advancements in the post-CMP planarization of chips, i.e., decreasing average 66.97 Å in height variation from 109.0 Å and increasing average 8.92 % in quality score versus NeurFill.

The remainder of this article is organized as follows. Section II introduces the background of CMP dummy filling. Section III elaborates our motivation. Sections IV and V presents our neural network model-based dummy synthesis frameworks and the perimeter adjustment strategy. In Section VI, we provide experimental results and analytically compare the proposed frameworks with the state-of-the-art algorithms. Finally, Section VII draws the conclusion.

II. Preliminaries

In this section, we introduce some background knowledge of the ECP and CMP process in Section II-A, which explains the reason why dummy filling is indispensable in IC manufacture. Section II-B briefly interprets the mechanism of full-chip CMP simulators. In Section II-C, we enumerate several metrics adopted to measure the filling results.

A. ECP and CMP Process

From the back end of sub-130-nm technology node, the ECP process is applied to deposit metal onto the wafer filling up etched trenches and forming interconnect wires and vias. The metal and oxide thickness after ECP is not uniform across the whole chip. Normally, systematic thickness variations are found to be layout dependent [26].

Then, the nonuniformity should be reduced by the CMP process. The wafer is held on a carrier with the surface to be polished facing down toward a polishing pad on a platen.

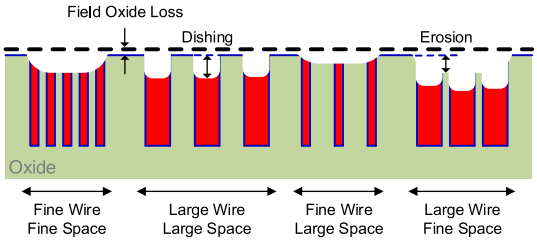

Fig. 3. Post-CMP topography of different layout patterns. The field oxide is polished, which causes thickness loss within the whole surface. Erosion and dishing describe excessive thickness loss of metal wire and of local oxide region.

Typically, both the wafer carrier and the platen are rotated in the same direction, as the carrier also has a lapping motion across the pad, and slurry composed of suspended particles in a chemical solution is delivered to the polishing platform. The whole procedure starts with homogeneous polishing, where the metal removal rate on different parts of a die is influenced by the as-plated initial topography created by layout patterns and plating characteristics. Later, the barrier layer is removed, and overpolish occurs. Ideally, the wafer can be perfectly flat after polishing. But an important degradation is that metal lines suffer from dishing and erosion problems, as shown in Fig. 3, which results in post-CMP thickness variations.

B. Full-Chip CMP Simulator

After 45-nm technology node, full-chip CMP simulators have become sign-off tools to help designers and manufacturers to overcome DFM problems from reference flow in foundries. Generally, the simulation procedure consists of four steps, as illustrated in Fig. 4.

Step 1: Grid the whole chip into multiple equally dissected windows, build equivalent pitch array structures based on metal density and perimeter in each window, and compute envelope heights of these windows [23].

Step 2: Solve contact mechanics and fluid mechanics equations to evaluate the pressure in each window [27].

Step 3: Compute the removal rates with the envelop pressure in local up and down areas using the density-step-height (DSH) model, where the critical step height H_{ex}, calibrated by experimental data, includes the information of perimeter input [28].

Step 4: Obtain the removed amount within a unit polishing time of δt by the Preston equation [29].

The last three steps will repeat until the preset total polishing time is reached. More details about full-chip CMP simulators can be found in [23] and [27].

C. Evaluation Metrics

In ICCAD 2014 dummy filling contest [12], many model-based metrics were introduced to evaluate layout planarity and circuit performance degradation after dummy filling optimization. Given an L-layer layout, each layer is divided into $N \times M$ windows according to the window size in a CMP simulator. Three objectives related to the layout planarity are height variation σ, line deviation σ^*, and outliers ol. Height

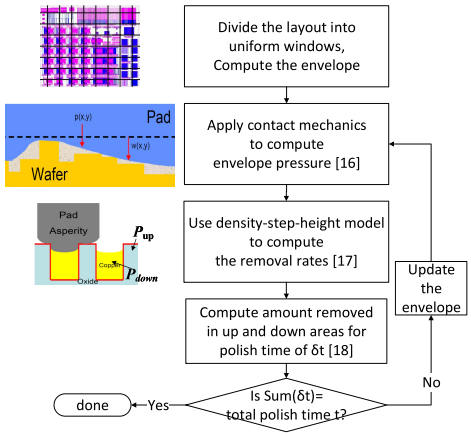

Fig. 4. Simulation flow of a full-chip CMP simulator. The layout is dissected into uniform windows and the surface height in each window is iteratively calculated within a preset simulation time.

variation is the standard deviation of profile heights in all windows of the layout, which aims at presenting uniformity at the layout level. Line deviation is the summation of column-based profile height variation. Outlier describes those profile heights outside the 3σ range. The three metrics are defined as

$$\sigma = \sum_{l=1}^{L} \sigma_l = \sum_{l=1}^{L} \sqrt{\frac{1}{N \times M} \sum_{i=1}^{N} \sum_{j=1}^{M} \left(H_{l,i,j} - \bar{H}_l\right)^2} \tag{1}$$

$$\sigma^* = \sum_{l=1}^{L} \sum_{i=1}^{N} \sum_{j=1}^{M} \left|H_{l,i,j} - \bar{H}_{l,j}\right| \tag{2}$$

$$ol = \sum_{l=1}^{L} \sum_{i=1}^{N} \sum_{j=1}^{M} \max\left(0, \left|H_{l,i,j} - \bar{H}_l\right| - 3\sigma_l\right) \tag{3}$$

where σ_l is the profile height variation in layer l. $H_{l,i,j}$ is the profile height of window $W_{l,i,j}$. $\bar{H}_l = (1/N \times M) \sum_{i=1}^{N} \sum_{j=1}^{M} H_{l,i,j}$ and $\bar{H}_{l,j} = (1/N) \sum_{i=1}^{N} H_{l,i,j}$ are the average profile height of layer l and of column j in layer l, respectively. Besides, total fill amount fa and overlay area ov are objectives concerning circuit performance degradation. The total fill amount is defined as

$$fa = \sum_{l=1}^{L} \sum_{i=1}^{N} \sum_{j=1}^{M} x_{l,i,j} \tag{4}$$

where $x_{l,i,j}$ is the fill amount in window $W_{l,i,j}$. Overlay area is calculated by four overlay types about inserted slack regions during dummy synthesis.

III. Motivation

In the CMP process, both mechanical and chemical methods are applied. Due to its complexity, the CMP simulators are regarded as nonlinear black boxes in existing model-based algorithms. Therefore, analytical gradient of the CMP model cannot be easily derived. As alternatives, numerical gradients are leveraged to drive the SQP optimization, which takes the form as

$$\nabla f_{\text{eval}} = \frac{f_{\text{eval}}(\mathbf{x} + \Delta\mathbf{x}) - f_{\text{eval}}(\mathbf{x})}{\Delta\mathbf{x}} \tag{5}$$

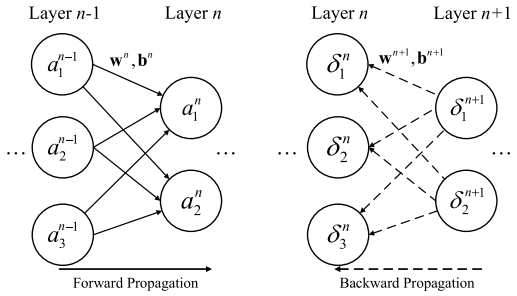

Fig. 5. Forward and backward propagation on neural networks.

where function $f_{\text{eval}}(\cdot)$ refers to an evaluation of quality score by CMP simulation. It is shown that both the case of current window fill amount $f_{\text{eval}}(\mathbf{x})$ and the case when a small dummy amount changed in each window $f_{\text{eval}}(\mathbf{x}+\Delta\mathbf{x})$ need to be simulated. Furthermore, each simulation will go through plenty of iteration steps. Huge invocations of CMP simulators are unacceptable because of the rigorous time-to-market requirement in chip design.

A. Acceleration With Neural Network

Fig. 5 illustrates the basic idea of forward and backward propagation on neural networks. In forward propagation, the activations of neurons in the nth layer $\mathbf{a}^n$ are linked with the activations of neurons in the $(n-1)$th layer $\mathbf{a}^{n-1}$ by equation

$$\mathbf{a}^n = f_{\text{act}}\left(\mathbf{w}^n \mathbf{a}^{n-1} + \mathbf{b}^n\right) \tag{6}$$

where $\mathbf{w}^n$ and $\mathbf{b}^n$ are the weight matrix and bias vector in the nth layer. f_{act} is the activation function. In backward propagation, the gradients of the nth layer δ^n are related to the gradients of the $(n+1)$th layer δ^{n+1} by equation

$$\delta^n = \left(\mathbf{w}^{n+1}\right)^T \delta^{n+1} \odot f'_{\text{act}}\left(\mathbf{z}^n\right) \tag{7}$$

where $\odot$ is the Hadamard product. f'_{act} is the gradient of activation function to $\mathbf{z}^n$ and $\mathbf{z}^n = \mathbf{w}^n \mathbf{a}^{n-1} + \mathbf{b}^n$. One significant benefit neural networks bring is that the backward propagation algorithm works far faster versus finite differential gradient approaches, which can accelerate the optimization.

B. Similarity Between CMP Model and Image Segmentation

Once the behavior of CMP simulators is learned by neural networks, the black box is opened to some extent and information required by optimization is also available.

Fortunately, we observe the underlying similarity between the CMP model and the image segmentation problem. In the image segmentation problem, the input is an image with pixels, which contains the information of RGB colors, and the output is a grayscale image indicating the areas of each segment. On the other hand, the input of the CMP model is an extracted grid layout, where each window contains parameters, such as pressure, trench height and density, perimeter of coppers, etc. The output of the CMP model is a post-CMP height profile, providing a positive height of each window. Besides, due to the contact mechanics of rough polishing pads in the CMP process, the character length in the CMP model is ranged

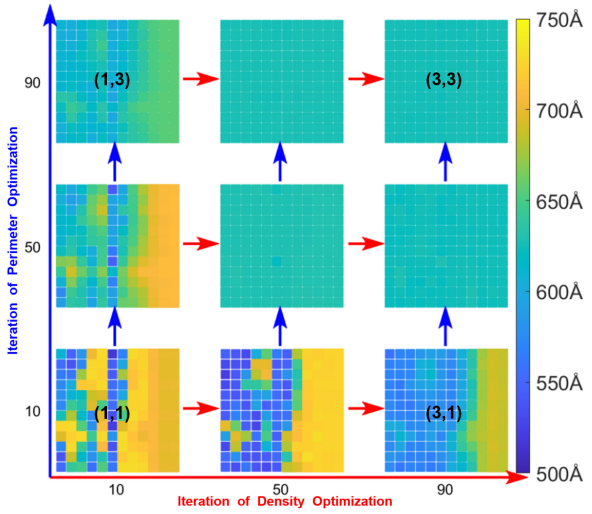

Fig. 6. Post-CMP topographies of a test layout during a iterative density and perimeter optimization.

from 20 to 100 μm [27], which limits the number of correlation windows. The local effect in the CMP model is similar to the convolutional kernels of neural networks. The intrinsic similarity between two problems, the local effect in the CMP model, and the potential acceleration motivate us to leverage pretrained neural networks replacing the conventional CMP simulator.

C. Improvement From Perimeter Adjustment

Full-chip CMP simulators use an accurate process model to predict the roughness of the chip surface. Besides the input layout topography, simulators also take inputs of process-related parameters, including the applied nominal pressure, the simulation time, and the material-removal rate, as well as the polishing pad parameters consisting of the equivalent Young's modulus, the height distribution function, auto-correlation function [27], etc. Actually, the process-related parameters and the polishing pad parameters are usually fixed by the incorporated processes and foundries. Designers can only ease the roughness degradation of post-CMP chip topographies by adjusting the dummy distribution in the layout.

Although a large number of research concentrates on the impact of dummy density, dummy perimeter influences the line width of the equivalent pitch array structures in the CMP simulation, which can sway the surface planarity of the polished wafers. To verify that the variation of metal perimeter has a significant effect on post-CMP planarity, we successively optimize dummy density and perimeter, respectively, in a test layout and observe improvements of the planarity in post-CMP topographies. As shown in Fig. 6, horizontal and vertical axes are iterations of density and perimeter optimization, respectively. For example, if only optimizing density, the planarity can be improved from (1, 1) to (3, 1). If only optimizing perimeter, it is from (1, 1) to (1, 3). If optimizing both density and perimeter, it is from (1, 1) to (3, 3). The experimental results show that optimizing density and perimeter simultaneously can obtain better-post-CMP planarity.

IV. NeurFill Framework

In this section, we first give the formulation of dummy synthesis problem in Section IV-A. The detailed architecture of the CMP neural network is demonstrated in Section IV-B. The numerical analyses of planarity score and performance degradation score are delivered in Section IV-C. As SQP algorithm is the essence of our iterative optimization, the way to find the starting point is interpreted in Section IV-D. Section IV-E shows the whole framework of NeurFill. Section IV-F explains the strategy of training the CMP neural network model.

A. Problem Formulation

The objective of dummy filling synthesis is to maximize the filling quality score S_{qual}, which can be formulated as

$$\max_{\mathbf{x}} S_{\text{qual}} = S_{\text{plan}} + S_{\text{deg}} \tag{8a}$$

$$S_{\text{plan}} = \alpha_\sigma f_\sigma(\sigma(\mathbf{H})) + \alpha_{\sigma^*} f_{\sigma^*}(\sigma^*(\mathbf{H})) + \alpha_{ol} f_{ol}(ol(\mathbf{H})) \tag{8b}$$

$$S_{\text{deg}} = \alpha_{fa} f_{fa}(fa(\mathbf{x})) + \alpha_{ov} f_{ov}(ov(\mathbf{x})) \tag{8c}$$

$$\text{s.t.} \quad x_{l,i,j} \in [0, s_{l,i,j}] \tag{8d}$$

where S_{plan} and S_{deg} indicate layout planarity score and circuit performance degradation score. $\mathbf{x}$ and $\mathbf{H}$ are vectors, including $x_{l,i,j}$ and $H_{l,i,j}$ in all windows, respectively. $s_{l,i,j}$ is the slack area in window $W_{l,i,j}$. α_σ, α_{σ^*}, α_{ol}, α_{fa}, and α_{ov} are weight factors provided by the CAD contest benchmark. As it is expected to decrease all aforementioned metrics, f_σ, f_{σ^*}, f_{ol}, f_{fa}, and f_{ov} are monotonic nonincreasing score functions about height variation, line deviation, outliers, fill amount and overlay area, which belong to the function set as

$$\left\{ f(t) = \max\left(0, 1 - \frac{t}{\beta}\right) \middle| \beta \in \mathbb{R}^+ \right\} \tag{9}$$

where β of f_σ, f_{σ^*}, f_{ol}, f_{fa}, and f_{ov} are benchmark-related parameters.

B. CMP Neural Network

As shown in Fig. 7, we choose UNet [30], a powerful convolutional neural network, to replace the full-chip CMP simulator. The input layout size is fixed to be 100×100 windows, and smaller layouts will be duplicated several times to fit it. The rest of our CMP neural network includes an extraction layer, in which pattern-related and process-related parameters are extracted into a layout parameter tensor $\mathbf{L}$, and an objective layer, in which planarity score S_{plan} is calculated according to the predicted result of profile height $\mathbf{H}_n$ from UNet. Since process-related parameters can be learned by network hyperparameters in the training stage, $\mathbf{L}$ is a $2 \times 100 \times 100$ tensor, where $L_{i,j} = (D, P)_{i,j}$ is metal density D and perimeter P of the window $W_{i,j}$ after filling $x_{i,j}$ dummy amount, where i and j are row and column indexes, respectively. The output $\mathbf{H}_n$ is a 100×100 matrix estimating the average height of each window after polishing. Robust computation functions are called to calculate height variation σ, line deviation σ^*, and outliers

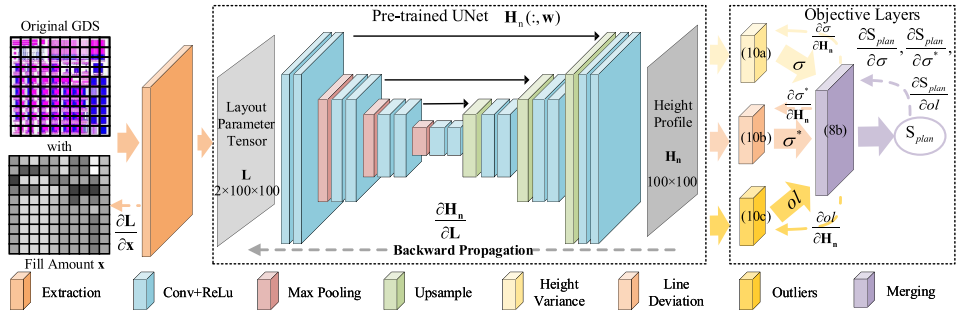

Fig. 7. Overview of CMP neural network.

ol in (1)–(3), which can be expressed as

$$\sigma = \mathrm{SQRT}(\mathrm{VAR}(\mathbf{H}_n)) \tag{10a}$$

$$\sigma^* = \mathrm{SUM}(\mathrm{ABS}(\mathbf{H}_n - \mathrm{MEAN}(\mathbf{H}_n, 1))) \tag{10b}$$

$$ol = \mathrm{SUM}(\mathrm{SIGMOID}(\eta(\mathrm{ABS}(\mathbf{H}_n - \mathrm{MEAN}(\mathbf{H}_n)) - 3 \cdot \mathrm{SQRT}(\mathrm{VAR}(\mathbf{H}_n))))) \tag{10c}$$

where SQRT, VAR, SUM, ABS, MEAN, and SIGMOID are *PyTorch* functions, and η is a hyperparameter. The differentiable sigmoid function is adopted to smooth the piecewise formulation in (3).

C. Quality Estimation

The quality score S_{qual} in (8a) is the combination of planarity score S_{plan} and circuit performance degradation score S_{deg}. According to (8b), S_{plan} consists of three terms that are separately calculated in the objective layer of the CMP neural network. Its gradient can be obtained by backward propagation through the chain rule as

$$\nabla S_{\mathrm{plan}} = \frac{\partial S_{\mathrm{plan}}}{\partial \sigma}\frac{\partial \sigma}{\partial \mathbf{H}_n}\frac{\partial \mathbf{H}_n}{\partial \mathbf{L}}\frac{\partial \mathbf{L}}{\partial \mathbf{x}} + \frac{\partial S_{\mathrm{plan}}}{\partial \sigma^*}\frac{\partial \sigma^*}{\partial \mathbf{H}_n}\frac{\partial \mathbf{H}_n}{\partial \mathbf{L}}\frac{\partial \mathbf{L}}{\partial \mathbf{x}} + \frac{\partial S_{\mathrm{plan}}}{\partial ol}\frac{\partial ol}{\partial \mathbf{H}_n}\frac{\partial \mathbf{H}_n}{\partial \mathbf{L}}\frac{\partial \mathbf{L}}{\partial \mathbf{x}}. \tag{11}$$

Gradient calculation is performed for $(\partial S_{\mathrm{plan}}/\partial \sigma)$, $(\partial S_{\mathrm{plan}}/\partial \sigma^*)$, and $(\partial S_{\mathrm{plan}}/\partial ol)$ according to (8b), and for $(\partial \sigma/\partial \mathbf{H}_n)$, $(\partial \sigma^*/\partial \mathbf{H}_n)$, and $(\partial ol/\partial \mathbf{H}_n)$ according to (10a)–(10c) in the objective layer. The backward propagation in UNet will compute $(\partial \mathbf{H}_n/\partial \mathbf{L})$, and $(\partial \mathbf{L}/\partial \mathbf{x})$ is resolved in the extraction layer.

In dummy synthesis, there are two methods to intuitively decrease induced parasitic capacitance. One is simply minimizing the fill amount, and the other is trying to plan the places of dummies into better-slack regions. The two considerations form the performance degradation score as demonstrated in (8c). Because the estimation of fill amount and overlay is not associated with CMP process, analytical gradients are able to be derived. The analytical gradient of total fill amount is calculated as

$$\nabla fa = J_{L,N,M} \tag{12}$$

where $J_{L,N,M}$ is the all-ones vector.

The overlay area remains unknown until the locations of dummies are determined. According to the wire locations in up and down metal layers, the slack regions of layer l can be divided into four types, as illustrated in Fig. 8. Type1 refers to no metal wire in vertical regions of layer $l+1$ and layer $l-1$. Type2 and Type3 refer to metal wire appearing only in layer $l+1$ and only in layer $l-1$, respectively. Type4 refer to regions of both up and down layers are paved with metal wire. Apparently, filling in Type1 region may induce the least parasitic capacitance, and in Type4 region is more likely to cause large parasitic capacitance. Within window $W_{l,i,j}$, fill amount in the four slack regions are written as $x_{l,i,j}^{T1-T4}$.

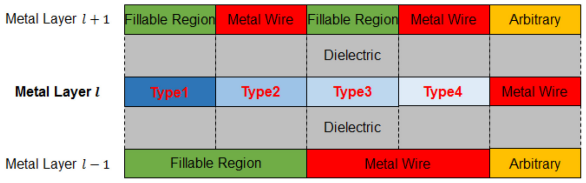

Fig. 8. Four types of fillable slack regions.

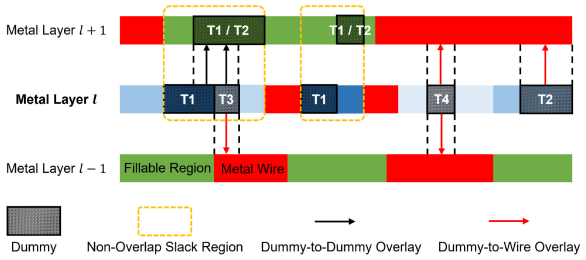

Fig. 9. Dummy-to-dummy overlay (black arrow) is the interacted area of dummies in neighboring layers. Dummy-to-wire overlay (red arrow) is the overlapped area between dummies and metal wires.

The overlay estimation analyzes dummy filled in the vertical direction of two neighboring metal layers. Fig. 9 displays all possible overlay types, which can be categorized as dummy-to-dummy overlay ov^{d-d} and dummy-to-wire overlay ov^{d-w}. The overlay area in layer l can be calculated as

$$ov_{l,i,j}^{d-d} = \max\left(0, x_{l,i,j}^{T1,T3} + x_{l+1,i,j}^{T1,T2} - s_{l,i,j}^{*}\right) \tag{13}$$

$$ov_{l,i,j}^{d-w} = x_{l,i,j}^{T2,T3} + 2x_{l,i,j}^{T4} \tag{14}$$

where $s_{l,i,j}^{*}$ indicates the area of nonoverlap slacks between $W_{l,i,j}$ and $W_{l+1,i,j}$. The total overlay in all windows can be calculated as

$$ov = \sum_{l=1}^{L-1}\sum_{i=1}^{N}\sum_{j=1}^{M} ov_{l,i,j}^{d-d} + \sum_{l=1}^{L}\sum_{i=1}^{N}\sum_{j=1}^{M} ov_{l,i,j}^{d-w}. \tag{15}$$

The analytical gradient of overlay can be derived as

$$\nabla ov_{l,i,j} = \begin{cases} 0, & x^{T1,T3}_{l,i,j} + x^{T1,T2}_{l+1,i,j} < s^{*}_{l,i,j} \\ 2, & x^{T4}_{l,i,j} > 0 \\ 1, & \text{otherwise.} \end{cases} \tag{16}$$

Therefore, analytical gradient of performance degradation score is

$$\nabla S_{\text{deg}} = -\frac{\alpha_{fa}}{\beta_{fa}} \nabla fa - \frac{\alpha_{ov}}{\beta_{ov}} \nabla ov \tag{17}$$

where α_{fa} and β_{fa} are benchmark-related constants about total fill amount, and α_{ov} and β_{ov} are constants about overlay.

D. Prior Knowledge-Based Starting Point Generation

A prior knowledge-based starting point generation strategy is used here. Intuitively, the post-CMP planarity is better when metal densities of all windows in a layer are consistent. Therefore, we assign a target density t_l for each layer, which represents the expected density in windows of layer l after dummy filling. Next, we set the initial fill amount of $W_{l,i,j}$ to make the metal density as close to t_l as possible, which can be expressed as

$$x_{l,i,j} = \begin{cases} 0, & t_l < m_{l,i,j} \\ s_{l,i,j}, & t_l > m_{l,i,j} + s_{l,i,j} \\ t_l - m_{l,i,j}, & \text{otherwise} \end{cases} \tag{18}$$

where $m_{l,i,j}$ and $s_{l,i,j}$ are the original metal density and the slack area density of window $W_{l,i,j}$. The optimal target density is determined by an exhausted linear search method that selects the value with the best-quality score, and the corresponding initial fill amount is the starting point.

E. Framework of NeurFill

The gray shadowed parts in Fig. 10 present the framework of our proposed NeurFill. In the beginning, the input layout is uniformly dissected with $100\,\mu\text{m} \times 100\,\mu\text{m}$ window size, and the starting point is generated by the prior knowledge-based method. Then, SQP optimization is applied to the updated layout in every iteration. If the step size in SQP exceeds a threshold, the dummy amounts in all windows are adjusted, and the next iteration starts. Otherwise, the flow will terminate and output the optimized layout. As hyperparameters in the CMP neural network have been fixed after training, planarity score can be calculated in forward propagation and the gradient of planarity score to the input dummy fill amount can be obtained in the input layer through backward propagation. Performance degradation score and its gradient can be directly computed by performance analysis. The two kinds of scores and gradients are merged to form the quality score and gradient in SQP optimization.

F. Pretraining of UNet Model

As depicted in Fig. 7, the UNet architecture consists of a down-sampling path to capture features and an up-sampling path to generate the post-CMP height profile. Given a set of extracted layout parameter matrix $\mathcal{L} = \{\mathbf{L}_1, \mathbf{L}_2, \ldots, \mathbf{L}_n\}$ and the corresponding post-CMP height profile set $\mathcal{H}_s = \{\mathbf{H}_{s1}, \mathbf{H}_{s2}, \ldots, \mathbf{H}_{sn}\}$ generated by the full-chip CMP simulator, the training procedure of UNet is to minimize the following objective:

$$\widetilde{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} \lambda \|\mathbf{H}_n(\mathcal{L}, \boldsymbol{w}) - \mathcal{H}_s\|_2^2 \tag{19}$$

where $\mathbf{H}_n(\cdot, \boldsymbol{w})$ is the network output with respect to weights $\boldsymbol{w}$ and λ is a configurable hyperparameter.

The input and output dimensions are fixed for neural networks. Since it is hard to find enough actual layouts for UNet training, a two-step random procedure is applied in this article to generate training data, as depicted in Fig. 11. First, layouts are divided into uniform windows. These windows are randomly assembled to generate 200 layouts of 100×100 windows. Second, random amounts of fixed $1\,\mu\text{m} \times 1\,\mu\text{m}$ square dummies are inserted into each window of the assembled layouts with no design rule violation, and then metal density and perimeter are calculated in each window for the CMP simulation. Finally, 20 000 layouts are generated by the two-step random procedure and simulated by the full-chip CMP simulator as the training set. The two-step random procedure aims to produce training instances that are close to the layouts neural networks may process in the filling optimization. Besides, the extension ability of the pretrained UNet model is verified by a testing set, which is generated by another layout not participating in the generation of the training set.

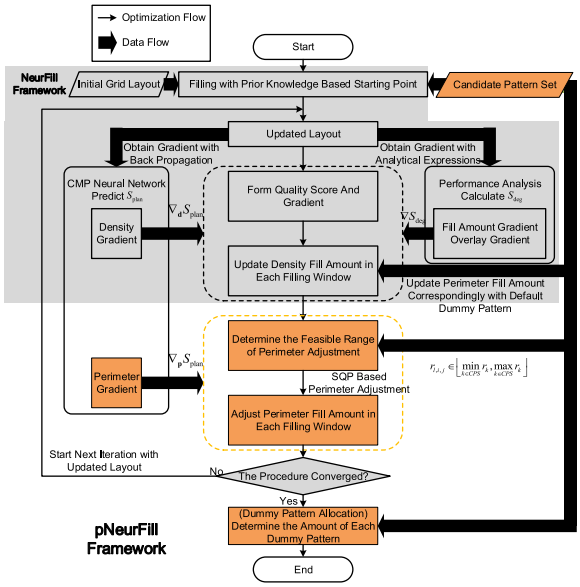

Fig. 10. NeurFill framework (gray shadowed) leverages the gradient of quality score to optimize the density fill amount of single dummy pattern in filling windows by SQP algorithm. pNeurFill framework (orange color) separates the optimization about dummy density and perimeter apart to realize perimeter adjustment and multipattern dummy filling synthesis.

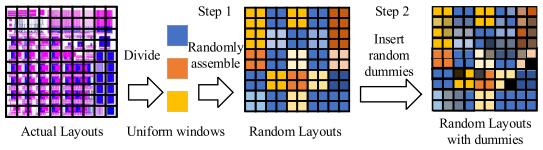

Fig. 11. Two-step random procedure of training data generation.

V. pNeurFill Framework With Perimeter Adjustment

In this section, we introduce our refined pNeurFill framework and the perimeter adjustment strategy. To conduct multipattern dummy synthesis and perimeter adjustment, we modify the original problem formulation in Section V-A. The calculation of the gradient of planarity score in the new scenario is detailed in Section V-B. Section V-C illustrates the architecture of adopted CMP neural network and affiliated training flow. In Section V-D, we propose a heuristic dummy pattern allocation algorithm to handle the multipattern dummy synthesis problem. Finally, the framework of pNeurFill and its differences from NeurFill are explained in Section V-E.

A. Perimeter Related Problem Formulation

Dummy perimeter is able to affect post-CMP topography and is meanwhile an important characterization extracted by CMP simulators. Therefore, dummy perimeter and different dummy patterns are natural to be considered in dummy filling synthesis.

1) Dummy Patterns: The choice of dummy patterns can influence the CMP uniformity and induced parasitic capacitance. For example, irregularly shaped fill patterns were adopted in [31] to reduce intralayer capacitive impact. But without loss of generality and for clarity, rectangular dummies are sufficient to implement density- and perimeter-based optimization in our dummy synthesis paradigm. Table I lists four types of rectangular dummies used in this article, where l_k and w_k are the length and width of type-k rectangular pattern. The key factor r_k of a dummy pattern is defined as the ratio of its perimeter p_k to its area a_k. All available patterns forming a CPS are fed into the optimization framework. The patterns in CPS can be replaced according to realistic process requirements.

2) Filling Constraints: The constraint in (8d) can be more accurate. Dummies are impossible to cover the whole slack area in a window as design rules require enough spacing between any two separate patterns. Moreover, the layout density satisfies allowed lower-bound and upper-bound constraints [32] as

$$d_{l,i,j} \in \left[D_{\min} - m_{l,i,j}, D_{\max} - m_{l,i,j}\right] \tag{20}$$

where $d_{l,i,j}$ is the density fill amount in window $W_{l,i,j}$. $D_{\min}$ and $D_{\max}$ stand for minimum and maximum metal density allowed in each window, and $m_{l,i,j}$ is the original metal density in window $W_{l,i,j}$.

To avoid impractical solutions based on the dummy patterns in CPS, filling perimeter and density should satisfy

$$r_{l,i,j} = \frac{p_{l,i,j}}{z d_{l,i,j}} \in \left[\min_{k\in CPS} r_k, \max_{k\in CPS} r_k\right] \tag{21}$$

where $p_{l,i,j}$ and $r_{l,i,j}$ are the perimeter fill amount and the filling ratio in window $W_{l,i,j}$, respectively. z is the area of a window.

3) Filling Objectives: Now that the filling perimeter is separated from filling density, $\mathbf{H}$ in (8b) can be specifically denoted as $\mathbf{H}(\mathbf{d}, \mathbf{p})$. In this way, the formulation of the dummy synthesis problem is described as

TABLE I
Dummy Patterns Used in Our Experiment

type-k	type-1	type-2	type-3	type-4
l_k	1 μm	2 μm	4 μm	8 μm
w_k	1 μm	1 μm	1 μm	1 μm
p_k	4 μm	6 μm	10 μm	18 μm
a_k	1 μm^2	2 μm^2	4 μm^2	8 μm^2
r_k	4 μm^{-1}	3 μm^{-1}	2.5 μm^{-1}	2.25 μm^{-1}

$$\max_{\mathbf{d},\mathbf{p}} S_{\text{qual}}(\mathbf{d},\mathbf{p}) = S_{\text{plan}}(\mathbf{d},\mathbf{p}) + S_{\text{deg}}(\mathbf{d}) \tag{22a}$$

$$\begin{aligned} S_{\text{plan}}(\mathbf{d},\mathbf{p}) = {} & \alpha_\sigma f_\sigma(\sigma(\mathbf{H}(\mathbf{d},\mathbf{p}))) \\ & + \alpha_{\sigma^*} f_{\sigma^*}\left(\sigma^*(\mathbf{H}(\mathbf{d},\mathbf{p}))\right) \\ & + \alpha_{ol} f_{ol}(ol(\mathbf{H}(\mathbf{d},\mathbf{p}))) \end{aligned} \tag{22b}$$

$$S_{\text{deg}}(\mathbf{d}) = \alpha_{fa} f_{fa}(fa(\mathbf{d})) + \alpha_{ov} f_{ov}(ov(\mathbf{d})) \tag{22c}$$

$$\text{s.t.} \quad d_{l,i,j} \in \left[D_{\min} - m_{l,i,j}, D_{\max} - m_{l,i,j}\right] \tag{22d}$$

$$r_{l,i,j} \in \left[\min_{k\in CPS} r_k, \max_{k\in CPS} r_k\right] \tag{22e}$$

where $\mathbf{H}(\mathbf{d},\mathbf{p})$, $\mathbf{d}$ and $\mathbf{p}$ are vectors, including $H_{l,i,j}$, $d_{l,i,j}$, and $p_{l,i,j}$ in all windows, respectively.

The constrained problem can be converted into a nonconstrained form by applying the Lagrangian approach, which is formulated as

$$\begin{aligned} \min_{\mathbf{d},\mathbf{p},\mathbf{V}} \mathcal{L}(\mathbf{d},\mathbf{p},\mathbf{V}) = {} & -S_{\text{qual}}(\mathbf{d},\mathbf{p}) \\ & + \mathbf{v}_1^T \cdot (\mathbf{D}_{\min} - \mathbf{m} - \mathbf{d}) \\ & + \mathbf{v}_2^T \cdot (\mathbf{d} - \mathbf{D}_{\max} + \mathbf{m}) \\ & + \mathbf{v}_3^T \cdot (\mathbf{r}_{\min} - \mathbf{r}) \\ & + \mathbf{v}_4^T \cdot (\mathbf{r} - \mathbf{r}_{\max}) \end{aligned} \tag{23}$$

where $\mathbf{V} = (\mathbf{v}_1^T, \mathbf{v}_2^T, \mathbf{v}_3^T, \mathbf{v}_4^T)^T$ are Lagrangian multipliers for constraint merging. If the values of $\mathbf{d}$, $\mathbf{p}$, and their next steps $\delta_\mathbf{d}$, $\delta_\mathbf{p}$ are determined after the tth iteration of SQP, the Lagrangian multipliers in the $(t+1)$th iteration $\mathbf{V}^{(t+1)}$ can be updated by solving

$$(\mathbf{I}, -\mathbf{I}, J_\mathbf{r}, -J_\mathbf{r}) \cdot \mathbf{V}^{(t+1)} = \mathscr{H} \cdot \left(\delta_\mathbf{d}^T, \delta_\mathbf{p}^T\right)^T + \nabla S_{\text{qual}} \tag{24}$$

where $\mathbf{I}$ is the unit matrix, $J_\mathbf{r}$ is the Jacobian matrix of $\mathbf{r}(\mathbf{d}^{(t)}, \mathbf{p}^{(t)})$, and $\mathscr{H}$ is the Hessian matrix of $\mathcal{L}(\mathbf{d}^{(t)}, \mathbf{p}^{(t)}, \mathbf{V}^{(t)})$. More details about SQP can be found in [33].

B. Planarity Estimation

The planarity score in (8b) is related to the perimeter fill amount when taking account of the dummy perimeter. In this case, we break the gradient into two parts that are separately about density and perimeter as

$$\nabla S_{\text{plan}} = \nabla_\mathbf{d} S_{\text{plan}} \cdot \hat{\mathbf{d}} + \nabla_\mathbf{p} S_{\text{plan}} \cdot \hat{\mathbf{p}} \tag{25a}$$

$$\nabla_\mathbf{d} = \left(\frac{\partial}{\partial\sigma}\frac{\partial\sigma}{\partial\mathbf{H}_n} + \frac{\partial}{\partial\sigma^*}\frac{\partial\sigma^*}{\partial\mathbf{H}_n} + \frac{\partial}{\partial ol}\frac{\partial ol}{\partial\mathbf{H}_n}\right)\frac{\partial\mathbf{H}_n}{\partial\mathbf{L}}\frac{\partial\mathbf{L}}{\partial\mathbf{d}} \tag{25b}$$

$$\nabla_\mathbf{p} = \left(\frac{\partial}{\partial\sigma}\frac{\partial\sigma}{\partial\mathbf{H}_n} + \frac{\partial}{\partial\sigma^*}\frac{\partial\sigma^*}{\partial\mathbf{H}_n} + \frac{\partial}{\partial ol}\frac{\partial ol}{\partial\mathbf{H}_n}\right)\frac{\partial\mathbf{H}_n}{\partial\mathbf{L}}\frac{\partial\mathbf{L}}{\partial\mathbf{p}} \tag{25c}$$

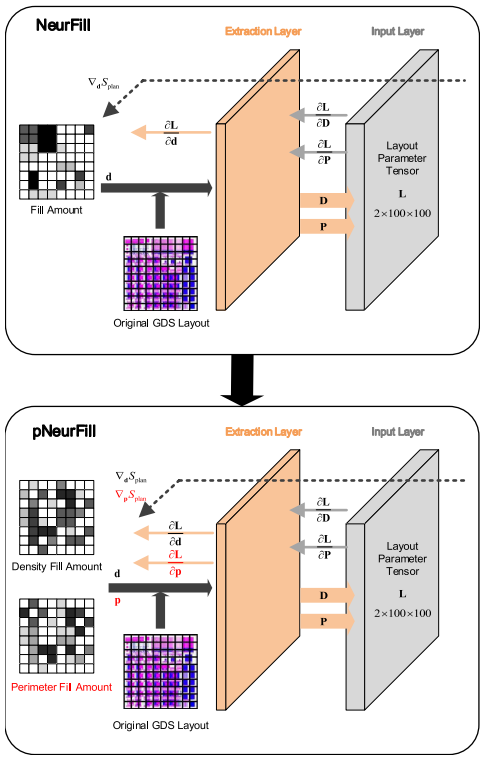

Fig. 12. Differences between NeurFill and pNeurFill in the extraction layer.

where $\nabla_{\mathbf{d}}$ and $\nabla_{\mathbf{p}}$ are gradients about density and perimeter, respectively. $\hat{\mathbf{d}}$ and $\hat{\mathbf{p}}$ are unit vectors in the parameter space.

C. Training of the Neural Network With Density and Perimeter

The structure of UNet is suitable to be employed for perimeter adjustment. Fig. 12 highlights the differences between NeurFill and pNeurFill in the extraction layer of the CMP neural network. Compared to NeurFill, pNeurFill inputs the additional perimeter fill amount $\mathbf{p}$ to the neural network, and its gradient $\nabla_{\mathbf{p}} S_{\text{plan}}$ can also be calculated via backward propagation. Therefore, gradients about density and perimeter can be separately used to optimize dummy density and to adjust dummy perimeter in each filling window.

Data augmentation is also applied to produce abundant pseudo data for neural network training. To catch the feature of perimeter variation, we create another dataset differing from the one in NeurFill as shown in Fig. 13. First, we choose certain real circuit layouts as seeds. In each window of a seed layout, random density fill amount d_{rand}, sampled from a uniform distribution in the range of $[D_{\min} - m_{l,i,j}, D_{\max} - m_{l,i,j}]$, is filled in. Then all filled windows are randomly permuted to generate a sample layout. Second, in each sample layout, a random metal perimeter amount, sampled from a uniform distribution in $[p_{\min}, p_{\max}]$, is filled into each window to generate a pseudo layout, where $p_{\min} = z \cdot d_{\text{rand}} \cdot \min_{k \in CPS} r_k$ and $p_{\max} = z \cdot d_{\text{rand}} \cdot \max_{k \in CPS} r_k$. This perimeter range guarantees that the layout can be generated using the dummy patterns in the CPS. In this way, pseudo layouts from the same sample layout have an identical density distribution but

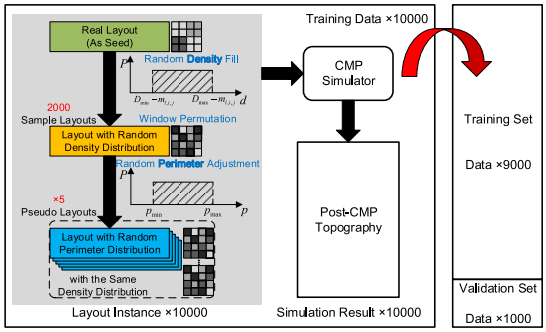

Fig. 13. Generation of training data. The data set contains 2000 batches of pseudo layouts and their simulation results. The layouts in a batch share the same metal density distribution but have different metal perimeter distributions. All data are split into a training set and a validation set.

different perimeter distributions. We repeat the first step of density randomness to create 2000 different sample layouts and repeat the second step of perimeter randomness in each of these sample layouts to generate five pseudo layouts. Finally, the 10000 (2000×5) pseudo layouts are fed into the full-chip CMP simulator to get corresponding post-CMP topographies. All training data is divided into a training set containing 9000 data and a validation set containing 1000 data. Besides, another layout is dedicated to generate a testing set containing 1000 data for ensuring the extension ability of the neural network.

D. Dummy Pattern Allocation

After dummy filling synthesis optimization in pNeurFill, the optimal density and perimeter fill amount is resolved in each window. However, inserting dummy patterns in the CPS one by one into each window will lead to a discrete variation of fill amount. Therefore, we propose a heuristic dummy pattern allocation algorithm, which determines the number of dummy patterns to fit the optimization result.

Details of the allocation are explained in Algorithm 1, where $|CPS|$ refers to the cardinal number of the CPS, and $n_{l,i,j}^{(k)}$ refers to the amount of type-k pattern in window $W_{l,i,j}$. The patterns in the CPS are sorted in descending order based on their key factors r_k. The objective of dummy allocation is to find a feasible solution that coincides with the optimized density and perimeter from dummy synthesis. The algorithm will iteratively allocate different dummy patterns in each window $W_{l,i,j}$ (line 3) when the remaining density fill amount $d > 0$ (line 5).

In each iteration, a *random* and *feasible* fill amount x is given to the type-k dummy pattern. The *random* means uniformly sampling from $(0, d]$ (line 7), which is to remove the implied priority among patterns given by the rotational order of patterns in each iteration. The *feasible* means satisfying the constraint $dz \geq xa_k$ to guarantee that there is no overfilling, and satisfying $(p - xp_k/dz - xa_k) \in [r_{\min}, r_{\max}]$ to guarantee that the optimal density and perimeter fill amounts found by the dummy synthesis are still achievable by combined patterns in the CPS in the following allocation iterations (line 8).

Algorithm 1 Dummy Pattern Allocation

Input: *CPS*; optimal density fill amount $\widetilde{d}_{l,i,j}$; optimal perimeter fill amount $\widetilde{p}_{l,i,j}$; window size z

Output: amount of patterns filled in each window $n^{(k)}_{l,i,j}$

```
sort patterns in CPS depends on r_k
r_min := min_{k∈CPS} r_k, r_max := max_{k∈CPS} r_k
for each W_{l,i,j} do
    n^(k)_{l,i,j} := 0, d := d̃_{l,i,j}, p := p̃_{l,i,j}
    while d > 0 do
        for k := 1 to |CPS| do
            sample d_0 ∈ (0, d] randomly, x := ⌊d_0 z / a_k⌋
            while dz < x a_k or (p − x p_k)/(dz − x a_k) ∉ [r_min, r_max] do
                x ← ⌊x/2⌋
            end while
            n^(k)_{l,i,j} ← n^(k)_{l,i,j} + x
            d ← d − x a_k / z
            p ← p − x p_k
        end for
    end while
end for
return n^(k)_{l,i,j}
```

E. Framework of pNeurFill With Perimeter Adjustment

The orange color in Fig. 10 demonstrates the additional steps of pNeurFill framework with perimeter adjustment. The improvements based on NeurFill mainly rest in the following aspects.

1) Besides the initial dissected layout, a CPS is also provided to constrain the range of optimized perimeter fill amount in perimeter adjustment.
2) A perimeter adjustment step is added after the density optimization to decouple the influence of density and perimeter fill amount. Thereby, the density and perimeter gradients from the CMP neural network are individually applied to different steps.
3) Finally, when iterations of SQP optimization complete, a dummy pattern allocation algorithm will convert density and perimeter fill amount into the quantities of patterns in CPS in all windows.

VI. Experimental Results

The NeurFill and pNeurFill frameworks are implemented using Python language with *PyTorch* toolkit. All experiments are performed on a Linux server with 48 CPU cores working at 2.6 GHz and an Nvidia Tesla K80 GPU (4992 Nvidia CUDA cores, 8.74 TFLOPS). The full-chip CMP simulator modeled by our neural network is calibrated under a 45-nm process of a foundry, and the accuracy is matched with the CMP Predictor [34], a commercial full-chip CMP simulator by Cadence. The window size of the full-chip CMP simulator and the dummy filling synthesis are both $100\,\mu\text{m} \times 100\,\mu\text{m}$. Dummy patterns used in pNeurFill are listed in Table I. Therefore, we have $r_{l,i,j} \in [2.25\,\mu\text{m}^{-1}, 4\,\mu\text{m}^{-1}]$ in our experiment.

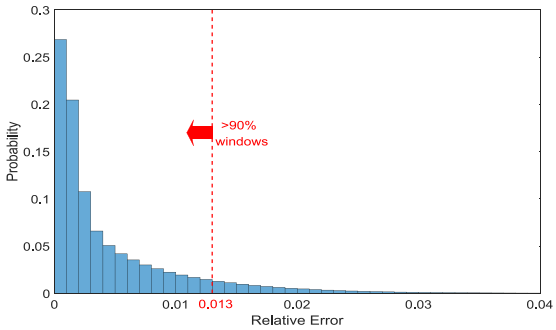

Fig. 14. Relative error distribution of the CMP neural network prediction.

TABLE II
Runtime Comparisons for Objective Evaluation and Gradient Calculation

Operation	Full-chip CMP Simulator (1c)	Full-chip CMP Simulator (64c)	CMP Neural Network (GPU)	Speedup (GPU v.s. 64c)
Objective Evaluation	4.7s	4.7s	0.025s	188×
Gradient Calculation	34100s	545s	0.067s	8134×

A. Accuracy of Pretrained Model

For NeurFill, 20 000 layout instances in the training set are generated by the two-step random procedure based on three available layouts. UNet is pretrained for 20 epochs on the GPU for 32 h. The average relative error is 0.28 % in the training set and 0.49 % in the validation set. Besides, the accuracy of the pretrained UNet model is evaluated by a testing set of 1000 layout instances. For windows in all testing set layouts, the relative prediction error of envelope height is measured. The mean and the standard deviation of relative errors are 0.60 % and 0.45 %, respectively. The maximum relative error of height is 1.77 %.

For pNeurFill, the training set are generated by the improved data augmentation method concerning perimeter variation. The average relative error is 0.43 % in the training set and 0.59 % in the validation set. In the testing set, the mean and the standard deviation of relative errors are 0.49 % and 0.70 %, respectively. The maximum relative error of height is 3.92 %. As shown in Fig. 14, the relative error of height is below 1.30 % in more than 90 % of the windows.

Since the dimensions of training spaces of NeurFill and pNeurFill are 10k and 20k, respectively, the worse-maximum relative error in pNeurFill is somewhat reasonable. Considering that the average relative error in pNeurFill is similar to NeurFill, which verifies that pNeurFill is also well-trained, therefore we did not increase the amount of training set in pNeurFill.

B. Acceleration of CMP Neural Network

Table II shows the runtime comparisons for objective evaluation and gradient calculation between the full-chip CMP simulator and the proposed CMP neural network. For single-precision performance, GPU's computation capability is up to 8.74 TFLOPS, while 64-core CPU is up to 8.12 TFLOPS.

TABLE III
OPTIMIZATION RESULTS ON THREE LAYOUT DESIGNS

Design	Method	ΔH ($\times 10$ Å)	Variation ($\times 10$ Å)	Line Deviation ($\times 10^4$ Å)	Outliers ($\times 10^2$ Å)	Runtime	Performance Degradation	Quality	File Size	Memory	Overall
A	Lin [14]	17.45	18.65	43.46	4.232	1 s	0.0000	0.3656	0.9380	0.7832	0.5103
	Tao [15]	17.47	18.02	45.03	4.185	43 s	0.8703	0.5427	0.9747	0.7832	0.6395
	Cai [16]	6.220	7.643	19.22	3.997	87 min	0.8752	0.7848	0.9898	0.7832	0.6773
	NeurFill	6.137	7.115	18.31	4.049	617 s	**0.8766**	0.7956	0.9916	0.8134	**0.7598**
	pNeurFill	5.807	6.936	18.25	4.350	1064 s	0.8726	**0.7967**	0.9874	0.7820	0.7030
B	Lin [14]	28.14	9.249	10.67	6.675	2 s	0.0000	0.4604	0.5590	0.7150	0.5590
	Tao [15]	27.83	8.659	9.814	6.141	95 s	0.4938	0.5831	0.6833	0.7150	0.6454
	Cai [16]	12.01	7.164	9.250	3.852	196 min	0.5272	0.6314	0.6665	0.7150	0.5426
	NeurFill	11.78	7.128	9.172	3.417	394 s	0.5935	0.6469	0.6379	0.8198	**0.6588**
	pNeurFill	8.532	5.055	7.557	4.933	871 s	**0.5961**	**0.7112**	0.6415	0.7447	0.6438
C	Lin [14]	23.72	6.751	7.457	6.445	12 s	0.0000	0.5301	0.6998	0.7562	0.6189
	Tao [15]	25.96	7.663	7.739	6.112	139 s	0.2548	0.6810	0.7493	0.7562	0.7187
	Cai [16]	7.845	14.90	8.892	3.589	935 min	0.2416	0.7459	0.7296	0.7562	0.6337
	NeurFill	7.611	13.56	9.130	3.348	220 s	0.2559	0.7706	0.7628	0.8323	0.7802
	pNeurFill	2.415	3.913	3.525	5.964	557 s	**0.2603**	**0.8674**	0.6761	0.7108	**0.8003**
C2	Lin [14]	8.320	15.31	21.44	4.975	9 s	0.1466	0.3613	0.9844	0.7855	0.5083
	Tao [15]	8.374	15.46	15.91	4.808	134 s	0.2619	0.5054	0.9117	0.7855	0.5972
	Cai [16]	1.283	14.22	6.590	3.907	1228 min	0.2584	0.8018	0.9816	0.7855	0.6897
	NeurFill	1.270	13.51	6.051	3.915	406 s	0.2566	0.8050	0.9634	0.8025	0.7913
	pNeurFill	1.239	2.579	2.310	5.701	850 s	**0.2637**	**0.8947**	0.9787	0.7132	**0.7994**
C3	Lin [14]	8.223	14.23	21.49	3.582	17 s	0.0000	0.3568	0.8149	0.7413	0.4933
	Tao [15]	7.938	14.07	21.15	3.593	115 s	0.2640	0.6461	0.7695	0.7413	0.6957
	Cai [16]	2.102	13.45	6.283	3.515	1076 min	0.2577	0.7998	0.7804	0.7413	0.6759
	NeurFill	1.963	13.18	6.157	3.654	309 s	0.2498	0.8082	0.7869	0.7796	0.7959
	pNeurFill	1.345	2.526	2.267	5.673	715 s	**0.2642**	**0.8958**	0.8130	0.6847	**0.8074**

The computation capabilities of two platforms are considered equivalent.

On a layout design of 100×100 windows, CMP neural network performs objective evaluation in 0.025 s by forward propagation, which achieves $188\times$ speedup. For gradient calculation, the backward propagation of CMP neural network is $8134\times$ faster than the numerical gradient calculation of the 64-core full-chip CMP simulator. The CMP neural network transform the black-box model, i.e., the full-chip CMP simulator, into a gray-box model, and combines the forward and backward propagation algorithms for acceleration. The efficiency of CMP neural network enables SQP framework for optimization.

C. Filling Quality Comparison

We compare our proposed NeurFill and pNeurFill with rule-based algorithms, Lin et al. [14], Tao et al. [15], and the best-existing model-based method, Cai et al. [16], as described in Table III. The NeurFill and pNeurFill are both neural network model-based dummy synthesis frameworks, and the pNeurFill integrates perimeter adjustment. The five test layouts all have 3 layers. Design A is a CMP test design with chip size 5 cm × 5 cm. Design B is a field programmable gate array (FPGA) design with chip size 6.7 cm × 6.3 cm. Design C, C2 and C3 are Metal 3–5 layers (M3–5), M1–3, and M2–4 of a RISC-V CPU design with chip size 10 cm × 10 cm.

Considering the filling quality, neural network model-based methods get better-quality scores than the other algorithms. The *quality* score of NeurFill is 117.6 % better than Lin et al. [14] on Design A, 10.9 % better than Tao et al. [15] on Design B, and 3.3 % better than Cai et al. [16] on Design C. Although NeurFill and simulator model-based method, Cai et al. [16] have comparable filling quality, NeurFill is $137\times$ faster than Cai et al. [16], which manifests the backward propagation of the CMP neural network is extremely efficient in contrast to the numerical gradient calculation with full-chip CMP simulation results.

Besides, pNeurFill reaches the highest-overall score and the best-post-CMP layout planarity with the least absolute height difference ΔH among these methods. In particular, pNeurFill decreases average 66.97 Å in height variation from 109.0 Å and increases average 8.92 % in quality score compared to NeurFill, which supports the validity of dummy perimeter adjustment. The experimental results in Table III show that for realistic circuit designs like FPGA and RISC-V CPU (B-C3), the quality improvements are remarkable (9.94 %–12.56 %). pNeurFill has little quality improvement ($0.7967/0.7956 = 0.14\%$) over NeurFill on Design A, perhaps because it is pitch array structures with different line widths and line spaces rather than a realistic circuit design.

As density and perimeter fill amounts are optimized in different steps in pNeurFill, the dimension of parameter space is grown compared to the only density optimization in NeurFill. Thus, the runtime in pNeurFill is approximately twice of NeurFill. To get more uniform layout planarity, which means a better-chip yield, this overhead is acceptable.

Comparison of the updates of quality score, height variation, line deviation, and outliers with iteration epoch on Design A is depicted in Fig. 15. The iterations in NeurFill and pNeurFill converge very fast and main performance improvements occur in the first 30 epochs, which shows the effectiveness of the SQP algorithm with a knowledge-based starting point.

Fig. 16 displays a detailed post-CMP height distribution in a range, including 50×50 windows on Design C. Fig. 16(a) is the height distribution of the original layout, Fig. 16(b) is that of NeurFill optimized layout, and Fig. 16(c) is that of pNeurFill optimized layout. Obviously, the whole layout surface generated by pNeurFill is more uniform than NeurFill.

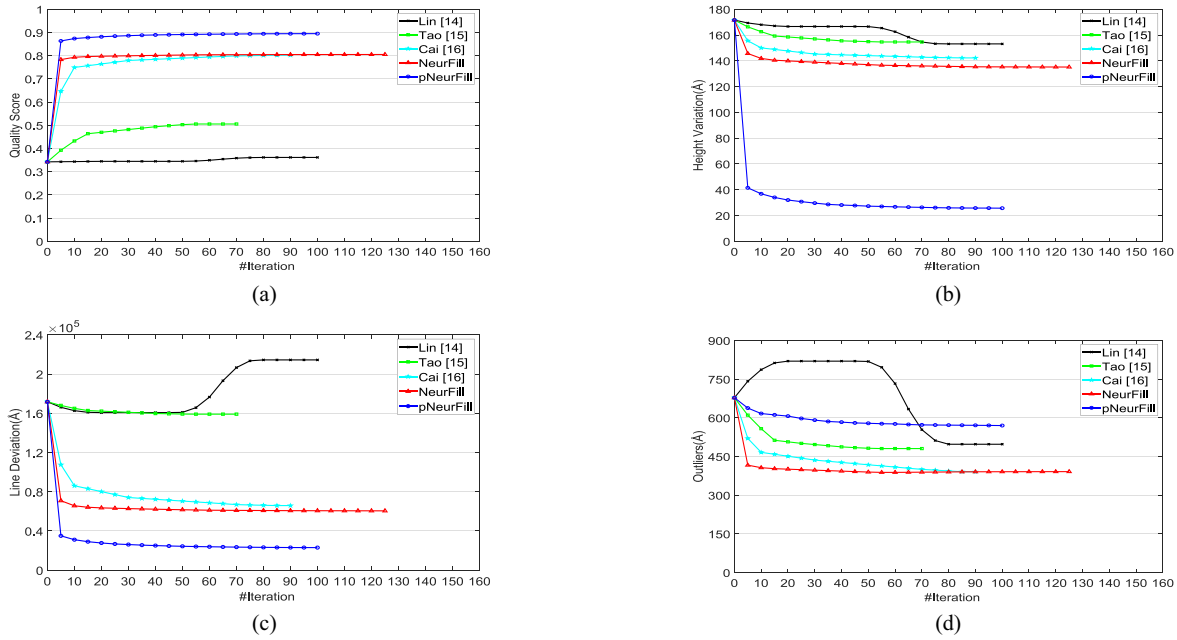

Fig. 15. Comparison about the updates of (a) quality score, (b) height variation, (c) line deviation, and (d) outliers with iteration epoch on Design C2.

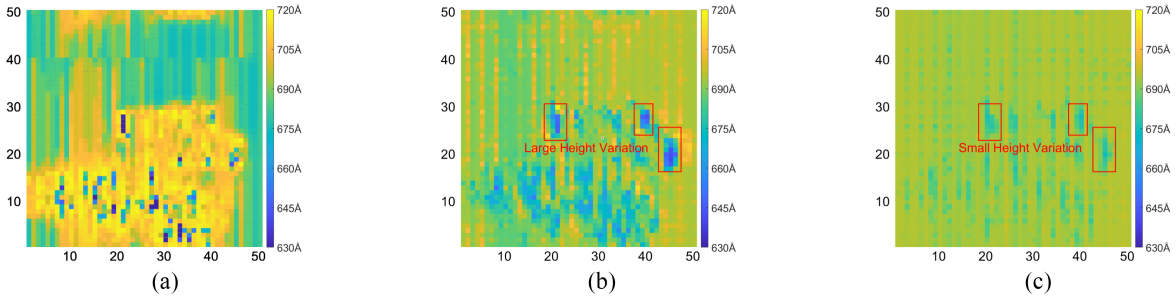

Fig. 16. Post-CMP height distributions of (a) original layout, (b) NeurFill optimized layout (w.o. perimeter adjustment), and (c) pNeurFill optimized layout (w. perimeter adjustment) on Design C.

D. Effectiveness of Dummy Allocation

Table IV shows the simulation results of two stages in pNeurFill, where "Opt" indicates the layout after dummy synthesis and "Alloc" indicates that after dummy allocation. The proposed dummy pattern allocation algorithm successfully fits the optimization results in all test layouts with merely 0.028 % quality loss on average. The loss is due to the conversion from continuous space to discrete space, where the SQP algorithm treats density and perimeter fill amount as continuous variables but the amount of dummy patterns in the allocation step is actually discrete.

Fig. 17 shows the number of different patterns placed in three test layouts. The Type-1 pattern is the square dummy used in NeurFill but is not dominant in Design A and C after optimization considering the perimeter fill amount. The other three dummy patterns also account for some proportion.

TABLE IV
RESULTS OF PNEURFILL AFTER SQP OPTIMIZATION AND AFTER DUMMY ALLOCATION ON THREE LAYOUT DESIGNS

Design	Stage	Variation ($\times 10$ Å)	Line Deviation ($\times 10^4$ Å)	Outliers ($\times 10^2$ Å)	Quality
A	Opt	6.935	18.25	4.351	0.7968
	Alloc	6.936	18.25	4.350	0.7967
B	Opt	5.047	7.537	4.931	0.7117
	Alloc	5.055	7.557	4.933	0.7112
C	Opt	3.892	3.506	5.966	0.8677
	Alloc	3.913	3.525	5.964	0.8674
C2	Opt	2.568	2.305	5.703	0.8948
	Alloc	2.579	2.310	5.701	0.8947
C3	Opt	2.514	2.260	5.676	0.8959
	Alloc	2.526	2.267	5.673	0.8958

E. Rule-Based Dummy Insertion

After determining the number of different dummy patterns in each window, dummy insertion only needs to optimize the positions of dummies to decrease induced parasitic capacitance. In Fig. 18, we put dummies into one layer of Design A based on the dummy synthesis optimization result and DRC rules to demonstrate a possible post-insertion view. It also shows that different dummy patterns are inserted into one local region of the layout.

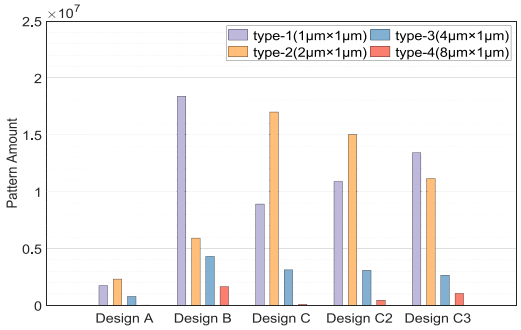

Fig. 17. Dummy amount of different patterns on test layouts.

Fig. 18. Possible Locations of inserted dummies on Design C2. Filling windows are marked with black bold lines. Metal wires are colored green and gray rectangles are filled dummies.

VII. Conclusion

In this article, we propose NeurFill to migrate full-chip CMP simulators to neural networks for dummy synthesis. Considering the height variation caused by the dummy perimeter, we further refine the framework with perimeter adjustment and propose pNeurFill to solve the related multipattern dummy synthesis problem. With empirical starting point-based SQP optimization, our methods reach better-layout planarity than rule-based methods and are much faster than the state-of-the-art model-based method. This will provide guidance for DFM in IC manufacturing. Furthermore, we believe the methodology of neural network modeling can be applied to deal with high-dimensional optimization problems in other application scenarios.

References

[1] A. B. Kahng and K. Samadi, "CMP fill synthesis: A survey of recent studies," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 1, pp. 3–19, Jan. 2008.

[2] B. E. Stine et al., "The physical and electrical effects of metal-fill patterning practices for oxide chemical-mechanical polishing processes," *IEEE Trans. Electron Devices*, vol. 45, no. 3, pp. 665–679, Mar. 1998.

[3] W.-S. Lee, K.-H. Lee, J.-K. Park, T.-K. Kim, Y.-K. Park, and J.-T. Kong, "Investigation of the capacitance deviation due to metal-fills and the effective interconnect geometry modeling," in *Proc. 4th Int. Symp. Qual. Electron. Design*, 2003, pp. 373–376.

[4] C. Feng, H. Zhou, C. Yan, J. Tao, and X. Zeng, "Efficient approximation algorithms for chemical mechanical polishing dummy fill," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 3, pp. 402–415, Mar. 2011.

[5] H. Xiang, L. Deng, R. Puri, K.-Y. Chao, and M. D. Wong, "Dummy fill density analysis with coupling constraints," in *Proc. Int. Symp. Phys. Design*, 2007, pp. 3–10.

[6] A. B. Kahng and R. O. Topaloglu, "DOE-based extraction of CMP, active and via fill impact on capacitances," *IEEE Trans. Semicond. Manuf.*, vol. 21, no. 1, pp. 22–32, Feb. 2008.

[7] T. Hirano, K. Okada, J. Hirokawa, and M. Ando, "Evaluation of effective permittivity and permeability for dummy metal fills in a CMOS chip using capacitor and inductor model," in *Proc. Int. Symp. Electromagn. Theory*, 2013, pp. 695–697.

[8] B. Jiang et al., "Fit: Fill insertion considering timing," in *Proc. 56th Annu. Design Autom. Conf.*, 2019, pp. 1–6.

[9] S.-J. Yu, C.-C. Kao, C.-H. Huang, and I. H.-R. Jiang, "Equivalent capacitance guided dummy fill insertion for timing and manufacturability," in *Proc. 25th Asia–South Pacific Design Autom. Conf. (ASP-DAC)*, 2020, pp. 133–138.

[10] A. B. Kahng, G. Robins, A. Singh, and A. Zelikovsky, "Filling algorithms and analyses for layout density control," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 18, no. 4, pp. 445–462, Apr. 1999.

[11] X. Chen, L. Xin, J. Zhang, and S. Li, "An improved rule-based dummy metal fill method FOR 65 nm ASIC design," *J. Circuits, Syst. Comput.*, vol. 22, no. 4, 2013, Art. no. 1350021.

[12] R. O. Topaloglu, "ICCAD-2014 CAD contest in design for manufacturability flow for advanced semiconductor nodes and benchmark suite," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, 2014, pp. 367–368.

[13] C. Liu et al., "An effective chemical mechanical polishing filling approach," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI*, 2015, pp. 44–49.

[14] Y. Lin, B. Yu, and D. Z. Pan, "High performance dummy fill insertion with coupling and uniformity constraints," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 36, no. 9, pp. 1532–1544, Sep. 2017.

[15] Y. Tao, C. Yan, Y. Lin, S.-G. Wang, D. Z. Pan, and X. Zeng, "A novel unified dummy fill insertion framework with SQP-based optimization method," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, 2016, pp. 1–8.

[16] J. Cai et al., "A novel and unified full-chip CMP model aware dummy fill insertion framework with SQP-based optimization method," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 40, no. 3, pp. 603–607, Mar. 2021.

[17] R. Tian, D. F. Wong, and R. Boone, "Model-based dummy feature placement for oxide chemical-mechanical polishing manufacturability," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 20, no. 7, pp. 902–910, Jul. 2001.

[18] S. Sinha, J. Luo, and C. Chiang, "Model based layout pattern dependent metal filling algorithm for improved chip surface uniformity in the copper process," in *Proc. Asia–South Pacific Design Autom. Conf.*, 2007, pp. 1–6.

[19] R. O. Topaloglu, "Energy-minimization model for fill synthesis," in *Proc. 8th Int. Symp. Qual. Electron. Design (ISQED)*, 2007, pp. 444–451.

[20] B. Jiang, L. Liu, Y. Ma, H. Zhang, B. Yu, and E. F. Y. Young, "Neural-ILT: Migrating ILT to neural networks for mask printability and complexity co-optimization," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD)*, San Diego, CA, USA, 2020, pp. 1–9.

[21] J. Cai, C. Yan, Y. Ma, B. Yu, D. Zhou, and X. Zeng, "NeurFill: Migrating full-chip CMP simulators to neural networks for model-based dummy filling synthesis," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, 2021, pp. 187–192.

[22] T. Ma, L. Chen, and J. Fang, "Study of optimal dummy fill modes in chemical–mechanical polishing process," *IEEE Trans. Compon., Packag. Manuf. Technol.*, vol. 2, no. 6, pp. 1043–1047, Jun. 2012.

[23] T. E. Gbondo-Tugbawa, "Chip-scale modeling of pattern dependencies in copper chemical mechanical polishing processes," Ph.D. dissertation, Dept. Electr. Eng. Comput. Sci., Massachusetts Inst. Technol., Cambridge, MA, USA, 2002.

[24] J. Luo, Q. Su, C. Chiang, and J. Kawa, "A layout dependent full-chip copper electroplating topography model," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, 2005, pp. 133–140.

[25] D. Fukuda, T. Shibuya, N. Idani, and T. Karasawa, "Full-chip CMP simulation system," in *Proc. Int. Conf. Planarization/CMP Technol.*, 2007, pp. 1–8.

[26] T. H. Park, "Characterization and modeling of pattern dependencies in copper interconnects for integrated circuits," Ph.D. dissertation, Dept. Electr. Eng. Comput. Sci., Massachusetts Inst. Technol., Cambridge, MA, USA, 2002.

[27] C. Feng, C. Yan, J. Tao, X. Zeng, and W. Caia, "A contact-mechanics-based model for general rough pads in chemical mechanical polishing processes," *J. Electrochem. Soc.*, vol. 156, no. 7, pp. H601–H611, 2009.

[28] T. Tugbawa, T. Park, B. Lee, and D. Boning, "Modeling of pattern dependencies for multi-level copper chemical-mechanical polishing processes," *MRS Online Proc. Library*, vol. 671, p. 43, Mar. 2011.

[29] L. M. Cook, "Chemical processes in glass polishing," *J. Non-Crystalline Solids*, vol. 120, nos. 1–3, pp. 152–171, 1990.

[30] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.*, 2015, pp. 234–241.

[31] M. Nelson, "Optimized pattern fill process for improved CMP uniformity and interconnect capacitance," in *Proc. 15th Biennial Univ./Government/Ind. Microelectron. Symp.*, 2003, pp. 374–375.
[32] X. Bai et al., "Timing-aware fill insertions with design-rule and density constraints," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 10, pp. 3529–3542, Oct. 2022.
[33] P. T. Boggs and J. W. Tolle, "Sequential quadratic programming," *Acta Numerica*, vol. 4, pp. 1–51, Jan. 1995.
[34] "CMP predictor." Accessed: Sep. 2023. [Online]. Available: https://www.cadence.com/en_US/home/tools/digital-design-and-signoff/silicon-signoff/cmp-predictor.html

Zhaoting Chen received the B.E. degree from the Department of Electronic Science and Technology, University of Science and Technology of China, Langfang, China, in 2020. He is currently pursuing the Ph.D. degree with the State Key Laboratory of Integrated Chips and Systems, Microelectronics Department, Fudan University, Shanghai, China.

His current research interests include design for manufacturability, analog circuit design automation, and optimization.

Junzhe Cai received the B.S. and M.S. degrees in microelectronics from Fudan University, Shanghai, China, in 2019 and 2022, respectively.

He is currently working with Tencent Inc., Shanghai, China.

Changhao Yan (Member, IEEE) received the B.E. and M.E. degrees from the Huazhong University of Science and Technology, Wuhan, China, in 1996 and 2002, respectively, and the Ph.D. degree in computer science from Tsinghua University, Beijing, China, in 2006.

He is currently a Full Professor with the Microelectronics Department, Fudan University, Shanghai, China. His current research interests include the parasitic parameter extraction of interconnects, parallel algorithms for large scale computation, design for manufacturability, the robust analysis of circuits, and AI and machine learning in medicine.

Zhaori Bi (Member, IEEE) received the B.Eng. degree in electronic information engineering from the Wuhan University of Technology, Wuhan, China, in 2011, and the M.S. and Ph.D. degrees in electrical engineering and computer engineering from the University of Texas at Dallas, Richardson, TX, USA, in 2013 and 2017, respectively.

He is currently an Assistant Professor with the Fudan University, Shanghai, China. His current research interests include mixed-signal system-on-a-chip design, circuit performance optimization, and applications in medical AI.

Yuzhe Ma (Member, IEEE) received the B.E. degree from the Department of Microelectronics, Sun Yat-sen University, Guangzhou, China, in 2016, and the Ph.D. degree from the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, in 2020.

He is currently an Assistant Professor with Microelectronics Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou. His research interests include agile VLSI design methodologies, machine learning-aided VLSI design, and hardware-friendly machine learning.

Dr. Ma received the Best Paper Awards from ICCAD 2021, ASPDAC 2021, and ICTAI 2019, and the Best Paper Award Nomination from ASPDAC 2019.

Bei Yu (Senior Member, IEEE) received the Ph.D. degree from The University of Texas at Austin, Austin, TX, USA, in 2014.

He is currently an Associate Professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong.

Dr. Yu received nine Best Paper Awards from DATE 2022, ICCAD 2021 and 2013, ASPDAC 2021 and 2012, ICTAI 2019, Integration, the *VLSI Journal* in 2018, the ISPD 2017, the SPIE Advanced Lithography Conference 2016, and six ICCAD/ISPD Contest Awards. He has served as the TPC Chair for ACM/IEEE Workshop on Machine Learning for CAD and in many journal editorial boards and conference committees. He is an Editor of IEEE TCCPS Newsletter.

Wenchuang Hu (Senior Member, IEEE) received the B.S. degree from Peking University, Beijing, China, in 1999, and the Ph.D. degree from the University of Notre Dame, Notre Dame, IN, USA, in 2004.

He spent a year as a Postdoctoral Research Fellow with the Department of Electrical Engineering, University of Michigan at Ann Arbor, Ann Arbor, MI, USA. In 2005, he joined The University of Texas at Dallas, Richardson, TX, USA, as a Faculty Member and became a Full Professor with the Department of Electrical Engineering in 2017. In 2021, he joined the West China Hospital, Sichuan University, Sichuan, China, as a Distinguished Professor and an Associate Director of the Center for Precision Medicine. His research has been focused on biosensors and bio-chips, molecular diagnostics, nanofabrication, and nanomaterials.

Prof. Hu is a member of Sigma Xi, AVS, MRS, ACS, and SPIE.

Dian Zhou (Member, IEEE) received the B.S. degree in physics and the M.S. degree in electrical engineering from Fudan University, Shanghai, China, in 1982 and 1985, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana–Champaign, Champaign, IL, USA, in 1990.

He joined the University of North Carolina at Charlotte, Charlotte, NC, USA, as an Assistant Professor in 1990, where he became an Associate Professor in 1995. He joined The University of Texas at Dallas, Richardson, TX, USA, as a Full Professor in 1999. His research interests include high-speed VLSI systems, CAD tools, mixed-signal ICs, and algorithms.

Xuan Zeng (Senior Member, IEEE) received the B.S. and Ph.D. degrees in electrical engineering from Fudan University, Shanghai, China, in 1991 and 1997, respectively.

She is currently a Full Professor with the Microelectronics Department, Fudan University, where she served as the Director of the State Key Laboratory of Application Specific Integrated Circuits and Systems from 2008 to 2012. She was a Visiting Professor with the Department of Electrical Engineering, Texas A&M University, College Station, TX, USA, and the Microelectronics Department, Technische Universiteit Delft, Delft, The Netherlands, in 2002 and 2003, respectively. Her current research interests include analog circuit modeling and synthesis, design for manufacturability, high-speed interconnect analysis and optimization, and circuit simulation.

Prof. Zeng received the Changjiang Distinguished Professor with the Ministry of Education Department of China in 2014, the Chinese National Science Funds for Distinguished Young Scientists in 2011, the First-Class of Natural Science Prize of Shanghai in 2012, the 10th For Women in Science Award in China in 2013, and the Shanghai Municipal Natural Science Peony Award in 2014. She received the Best Paper Award from the 8th IEEE Annual Ubiquitous Computing, Electronics and Mobile Communication Conference 2017. She is an Associate Editor of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART II: EXPRESS BRIEFS, IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, and *ACM Transactions on Design Automation of Electronic Systems*.