Machine Learning in Physical Verification, Mask Synthesis, and Physical Design

Yibo Lin and David Z. Pan

Abstract Yield, turn-around time, and chip quality are always of significant concerns for VLSI designs. The performance and efficiency of key design steps such as physical design, mask synthesis, and physical verification are critical to guarantee fast design closure and manufacturability. Recent advances in machine learning provide various new opportunities and approaches to tackle these challenges. This chapter will discuss several applications of machine learning in the backend design flow and demonstrate its impacts to existing design automation methodology.

1 Introduction

The extreme scaling of VLSI circuits has reached to the manufacturing limitation. Various challenges have been raised from the printability issues due to lithography resolution, process variation, etc. Thus a design needs to be highly optimized and extensively verified for manufacturability. In addition, design quality and manufacturability in late stages of the design flow become increasingly sensitive to the changes in early stages, increasing turn-around time and slowing down the design closure. Thus, early stage prediction of valid designs is becoming more and more critical.

In this chapter, we focus on the backend design challenges, including physical design, mask synthesis and physical verification. Physical design implements a gate-level netlist to a layout with physical locations and geometries. It typically includes placement, clock tree synthesis, routing, etc. Mask synthesis is a follow-up step to improve the printability utilizing resolution enhancement techniques (RETs), such

Yibo Lin
University of Texas at Austin, Texas, US, e-mail: yibolin@utexas.edu

David Z. Pan
University of Texas at Austin, Texas, US, e-mail: dpan@ece.utexas.edu

as optical proximity correction (OPC) and sub-resolution assist features (SRAFs). Besides optimization stages like physical design and mask synthesis, physical verification, e.g., hotspot detection, is another crucial step to validate the manufacturability of a design.

The aforementioned stages have been developing for decades and are encountering various challenges in performance and efficiency in advanced technology nodes. The advances in machine learning stimulate new opportunities to boost the design closure of the backend flow. General machine learning can be categorized to supervised learning and unsupervised learning. Supervised learning is essentially a modeling technique in which underlying correlation between features and labels is extracted to build a model. The model is then able to predict labels when given new features. Typical algorithms for supervised learning include boosting, logistic regression, support vector machines, deep neural networks, and so on [14]. Unsupervised learning does not require labels from data samples, such as a clustering problem. It aims at learning the hidden structures in the data.

This chapter will survey recent practices and researches on the machine learning applications to the backend design automation, from late stages to early, i.e., physical verification, mask synthesis, and physical design. We will introduce the motivations of each problem and briefly explain the learning techniques to solve the problem. Section 2 focuses on the challenges in physical verification; section 3 illustrates the problems in mask synthesis; section 4 talks about the applications in physical design; section 5 concludes the chapter.

2 Machine Learning in Physical Verification

Physical verification refers to various validation steps in the manufacturing process, such as lithography hotspot detection. Machine learning mainly helps to decrease turn-around-time and manufacturing costs.

Lithography hotspots denote specific patterns that tend to fail in printing even after RETs, which are becoming more and more common due to the complexity of lithography system and process variation. The early detection of lithography hotspot remains to be a critical challenge to enhance manufacturability and reduce costs. Generally, expensive lithography simulation is required for accurate hotspot detection, often leading to long turn-around time. Therefore, developing efficient hotspot detection is desired and machine learning is a suitable technique due to its efficiency in prediction.

Hotspot detection is a classification task in which layout clips labeled as either hotspots or non-hotspots are provided. The objective is to construct a model that can accurately classify clips. Fig. 1 shows two layouts with hotspots. In practice, the performance of hotspot detection is mainly evaluated with two metrics, i.e., detection accuracy and false alarm. *Detection accuracy* is defined as the ratio between the number of correctly detected hotspots and the number of real hotspots. *False alarm* is defined as the number of non-hotspots that are recognized as hotspots. The model-

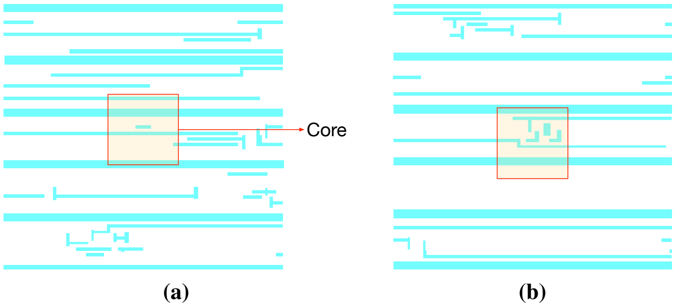


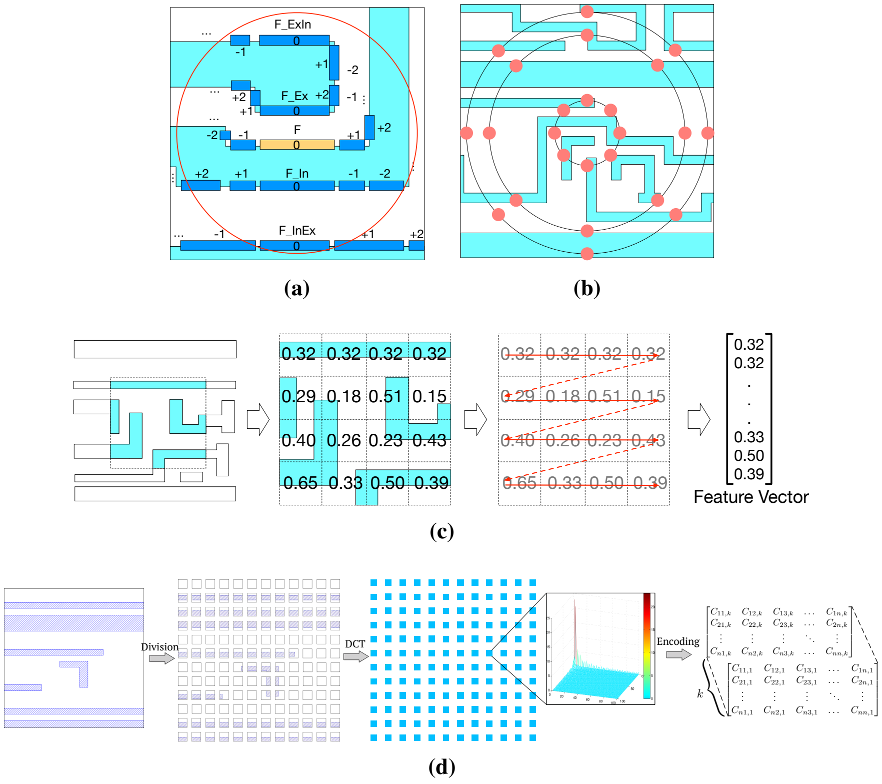
Fig. 1 Example of hotspot patterns marked in red [13].

ing objective is to first improve detection accuracy and then minimize false alarms. In other words, it is tolerable to mis-classify a few non-hotspots into hotspots, while the opposite is not encouraged.

2.1 Layout Feature Extraction and Encoding

The performance of hotspot detection is highly dependent to layout feature extraction and model selection. Layout feature represents the layout attributes to determine hotspots and non-hotspots, which is fundamental to the detection. Typical feature representations include density based feature [45, 27], fragmentation based feature, and concentric circle area sampling (CCAS) [29, 57, 58].

Fig. 2(a) illustrates the fragmentation based feature extraction. Each fragment F within a circle with an effective radius r is considered. Geometric characteristics of fragments covered by the circle are extracted as the representation of F, e.g., pattern shapes, distances between layout and corner information. Fig. 2(b) explains the concentric circles with area sampling to capture the layout information that matches the diffraction of lights. In this feature representation, a layout is sampled with different number of positions along concentric circles. Fig. 2(c) shows the density-based feature extraction. The feature is a vector of pattern densities computed from the layout density within each grid. As features are generally extracted to a feature vector, the spatial information between elements of the vector is lost [51, 24]. For example, geometrically close sampling points may not correspond to elements close to each other in indices. Hence, a feature tensor representation is proposed to keep such spatial information, as shown in Fig. 2(d). The original clip is converted to a hyper-image after feature tensor extraction. It is divided into 12×12 blocks and each block is converted to 100×100 images. The feature tensor is obtained after applying discrete cosine transformation (DCT) to each block.

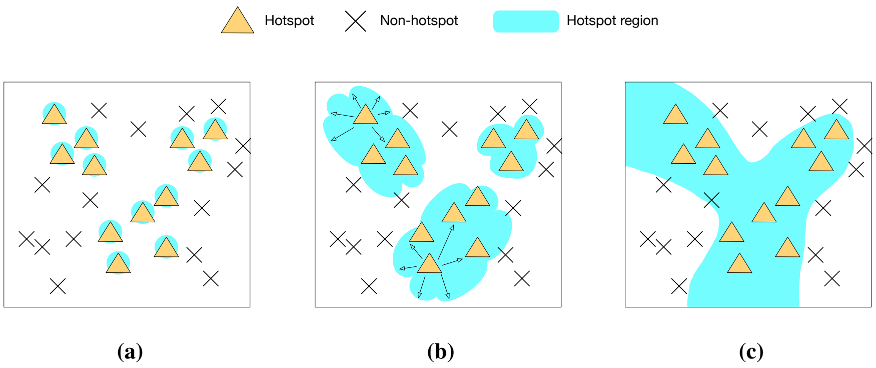


(a) **(b)**

(c)

(d)

Fig. 2 (a) Fragmentation based hotspot signature extraction[9]. (b) CCAS feature extraction [23]. (c) Density-based pattern representation[58]. (d) Feature tensor generation[51].



(a) **(b)** **(c)**

Fig. 3 A 2D-space example of hotspot region decision. (a) Pattern matching. (b) Fuzzy Pattern Matching. (c) Machine learning. [23]

2.2 Machine Learning Models for Hotspot Detection

Conventionally pattern matching is widely used in hotspot detection, as show in Fig. 3(a), while it cannot handle the situation when a pattern is not found in the pre-built library [1, 55]. Then fuzzy pattern matching is developed to dynamically tune the regions around the known hotspots, as shown in Fig. 3(b) [23]. Machine learning based approaches recently demonstrate even better generality, as shown in Fig. 3(c).

Besides feature extraction, various machine learning models have been used as hotspot detection kernels to achieve high accuracy and low false alarms, including support vector machine (SVM) [10, 56], artificial neural network (ANN) [10], and boosting methods [27, 57]. Zhang et al. [57] also propose an online learning scheme to verify newly detected hotspots and incrementally update the model.

To mitigate the impacts from feature representation to the detection accuracy, deep neural network (DNN) has been proposed for hotspot detection [39, 28]. DNN can avoid the manual efforts for the selection of feature extraction approaches, because it takes high-dimensional layout as input and perform automatic feature extraction during training. Promising empirical results have been observed with DNN in several papers [39, 28, 49, 50]. A typical configuration of DNN structure is shown in Fig. 4. It consists of four convolution layers and two fully connected layers. Each convolution layer uses a set of kernels to perform convolution on an input tensor $\mathbf{F}$ as follows,

$$\mathbf{F} \otimes \mathbf{K}(j,k) = \sum_{i=1}^{c} \sum_{m_0=1}^{m} \sum_{n_0=1}^{m} \mathbf{F}(i, j-m_0, k-n_0)\mathbf{K}(m_0, n_0), \tag{1}$$

where $\mathbf{F} \in \mathscr{R}^{c\times n\times n}$ and kernel $\mathbf{K} \in \mathscr{R}^{c\times m\times m}$. Each fully connected layer performs linear transformation to an input vector as follows,

$$x \mapsto \mathbf{W}x, \tag{2}$$

where $x \in \mathscr{R}^{m\times 1}$ is the input vector and $\mathbf{W} \in \mathscr{R}^{n\times m}$ represents the neuron weights. The activation function ReLU layer is defined as follows,

$$x \mapsto \begin{cases} x, & \text{if } x > 0, \\ 0, & \text{otherwise}, \end{cases} \tag{3}$$

where $x \in \mathscr{R}^{m\times 1}$ is the input vector.

The DNN in Fig. 4 is trained with mini-batch gradient descent (MGD) algorithm, as shown in Alg. 1 [51]. Variable $\mathbf{W}$ is the neuron weights, λ is the learning rate, $\alpha \in (0,1)$ denotes the decay factor of the learning rate, k denotes the decay step, $\mathbf{y}_h^*$ is the hotspot ground truth, and $\mathbf{y}_n^*$ is the non-hotspot ground truth. In each iteration, m training instances $\{\mathbf{F}_1, \mathbf{F}_2, \dots, \mathbf{F}_m\}$ are randomly sampled. The gradients are computed from line 6-8. The neuron weights $\mathbf{W}$ is updated with the gradients and the learning rate in line 14. In every k iterations, the learning rate λ is decayed to

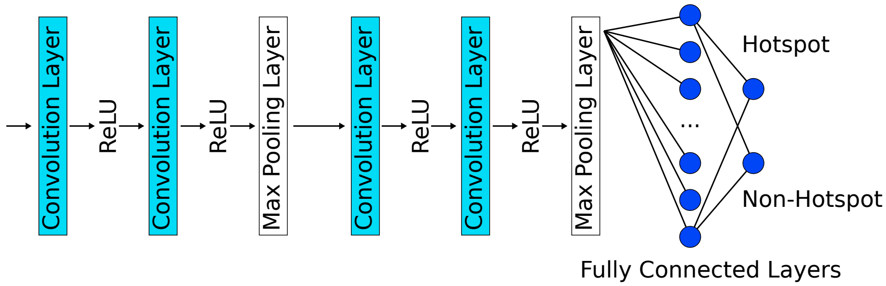


Fig. 4 An example of a neural network for hotspot detection [51].

Algorithm 1 Mini-batch Gradient Descent (MGD)

```
 1: function MGD(($\mathbf{W}, \lambda, \alpha, k, \mathbf{y}_h^*, \mathbf{y}_n^*$))
 2:     Initialize parameters $j \leftarrow 0, \mathbf{W} > 0$;
 3:     while not stop condition do
 4:         $j \leftarrow j + 1$;
 5:         Sample $m$ training instances $\{\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_m\}$;
 6:         for $i = 1, 2, \ldots, m$ do
 7:             $\mathscr{G}_i \leftarrow backprop(\mathbf{F}_i)$;
 8:         end for
 9:         Calculate gradient $\bar{\mathscr{G}} \leftarrow \frac{1}{m} \sum_{i=1}^{m} \mathscr{G}_i$;
10:         Update weight $\mathbf{W} \leftarrow \mathbf{W} - \lambda \bar{\mathscr{G}_i}$;
11:         if $j$ mod $k = 0$ then
12:             $\lambda \leftarrow \alpha\lambda, j \leftarrow 0$;
13:         end if
14:     end while
15:     return Trained model $f$;
16: end function
```

$\alpha\lambda$, as shown in line 10-11. MGD function returns the best model for the training set.

Despite the convenience in automatic feature extraction, the best configuration of DNN still requires manual trial and error process, such as searching for the number and types of layers. Later, Matsunawa et al.[28] and Yang et al.[51] further propose two different DNN structures that can improve the accuracy and reduce false alarms.

Table 1 shows the comparison between various state-of-the-art hotspot detectors on both ICCAD 2012 contest benchmarks and industrial designs [51, 49, 24]. The number of clips with hotspots is represented as “HS#” and the number of clips without hotspots is represented as “NHS#”. Column “Accu” denotes the accuracy and column “FA” denotes the false alarm. While different hotspot detectors may have different objectives in their problem formulations, the table reports the accuracy and false alarm for reference. Generally, deep learning achieves high accuracy with relatively low false alarm [51, 49]. Detectors like the online boosting algorithm [57] mainly try to reduce the overall detection and simulation time (ODST) using online learning with reasonable accuracy.

Table 1 Comparison between the state-of-the-art hotspot detectors [51, 49]

Bench	Train		Test		AdaBoost [27]		Online [57]		Deep [51]		Deep [49]	
	HS#	NHS#	HS#	NHS#	FA#	Accu (%)	FA#	Accu (%)	FA#	Accu (%)	FA#	Accu (%)
ICCAD	1204	17096	2524	13503	2919	84.2	4497	97.7	3413	98.2	1776	97.36
Industry1	34281	15635	17157	7801	557	93.2	1136	89.9	680	98.9	307	98.41
Industry2	15197	48758	7520	24457	1320	44.8	7402	88.4	2165	93.6	793	90.56
Industry3	24776	49315	12228	24817	3144	44.0	8609	82.3	4196	91.3	1723	83.63
Avg.	-	-	-	-	2397	66.6	5411	89.6	2613	95.5	1150	92.49
Ratio	-	-	-	-	0.92	0.70	2.07	0.94	1.0	1.0	0.44	0.97

3 Machine Learning in Mask Synthesis

As the technology nodes scale to the limit of light wavelength, various resolution enhancement techniques (RETs), such as optical proximity correction (OPC), source mask co-optimization, and sub-resolution assist features (SRAFs), become a necessity. Machine learning can be applied to various RETs to improve the turn-around time of mask synthesis.

3.1 Mask Synthesis Flow

Fig. 5(a) gives a standard mask synthesis flow in which target patterns (layout) are taken as input and mask patterns are generated after iterative optimization procedures including SRAF generation, OPC, mask rule check (MRC) and lithography compliance check (LCC) [24]. In SRAF generation, sub-resolution assist features are inserted to benefit the printing of target patterns. In OPC, the edge segments of target patterns are optimized for robust lithography printing. To ensure mask manufacturing friendliness, mask manufacturing rules should be checked after these optimization procedures in MRC. Then, LCC performs the lithography simulation under a set of process windows to check printability. Here process windows denote different $\{$focus, dose$\}$ conditions to generate printing contours, such as nominal, inner, and outer contour, as shown in Fig. 5(b). To quantify the process windows, two metrics are introduced: edge placement error (EPE) evaluates the distance between the target pattern contour and the nominal contour; process variation (PV) band evaluates the area between the inner and outer contour. A typical objective of RETs is to minimize EPE and PV band.

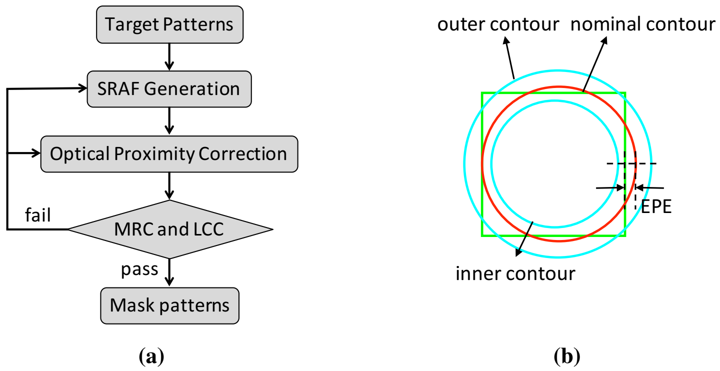


Fig. 5 Mask synthesis: (a) a standard mask synthesis flow, (b) printing contours under different {focus, dose} conditions [47].

3.2 Machine Learning for Sub-Resolution Assist Features

SRAFs are small rectangles within the sub-resolution domain to assist the printing of target patterns. In other words, they will not be actually printed, even though they are on masks. The effectiveness of SRAFs for an isolated contact is illustrated in Fig. 6. It can be seen that Fig. 6(c) (with SRAFs) achieves much smaller PV band than Fig. 6(b) (without SRAFs). This is because SRAFs deliver light to the positions of target patterns in a proper phase, improving the robustness of printing. In advanced technology nodes, developing fast yet high-quality SRAF generations is increasingly critical to the yield [48, 47].

There are two types of conventional SRAF generation approaches, model-based and rule-based. Model-based SRAF generation ensures high-quality and robustness but is computationally expensive [37, 41, 53, 38, 34, 21]. Thus, it is not scalable to large layout designs. On the other hand, rule-based SRAF generation enables super fast turn-around time by complicated look-up-tables [36, 20, 22], while its performance highly depends on the quality of look-up-tables which require adjustment with significant engineering efforts [47].

Supervised learning is promising to efficiently approximate model-based SRAF generation to improve turn-around time and meanwhile maintain high quality [47]. The training data comes from model-based SRAF generation. The model is trained to predict whether a pixel should be covered by SRAFs. The actual SRAFs are generated with the guidance of the model, subjecting to SRAF rules. Learning-based SRAF generation is formulated into a classification problem in which feature vectors are extracted with CCAS and the kernel models adopt both logistic regression and SVM.

The mathematical formulation of logistic regression is as follows [11],

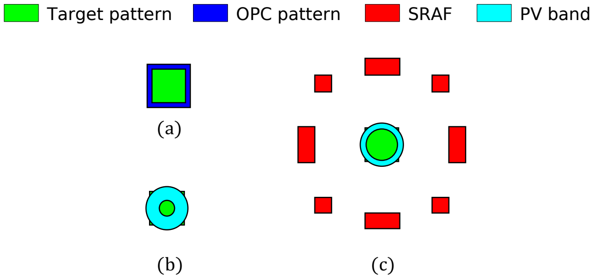


Fig. 6 (a) An isolated contact, (b) printing with OPC only, (c) printing with SRAF generation and OPC [47].

$$\min_{\boldsymbol{w}} \quad \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_i \log(1+e^{-y_i\boldsymbol{w}^T\boldsymbol{x_i}}), \tag{4}$$

where $\boldsymbol{w}$ is the weight parameters determined during training, $\boldsymbol{x_i}$ and y_i are features and label (-1 or 1 for two-class classification) for i^{th} data sample, respectively. The first term $\frac{1}{2}\boldsymbol{w}^T\boldsymbol{w}$ is the $L2$ regularization to avoid overfitting utilizing maximum likelihood method [17]. The second term is the overall error cost. Parameter C sets the importance of the regularization term. Thus the objective for training is to minimize the overall error cost with $L2$ regularization.

On the other hand, the mathematical formulation of support vector machine with linear kernel is as follows [4],

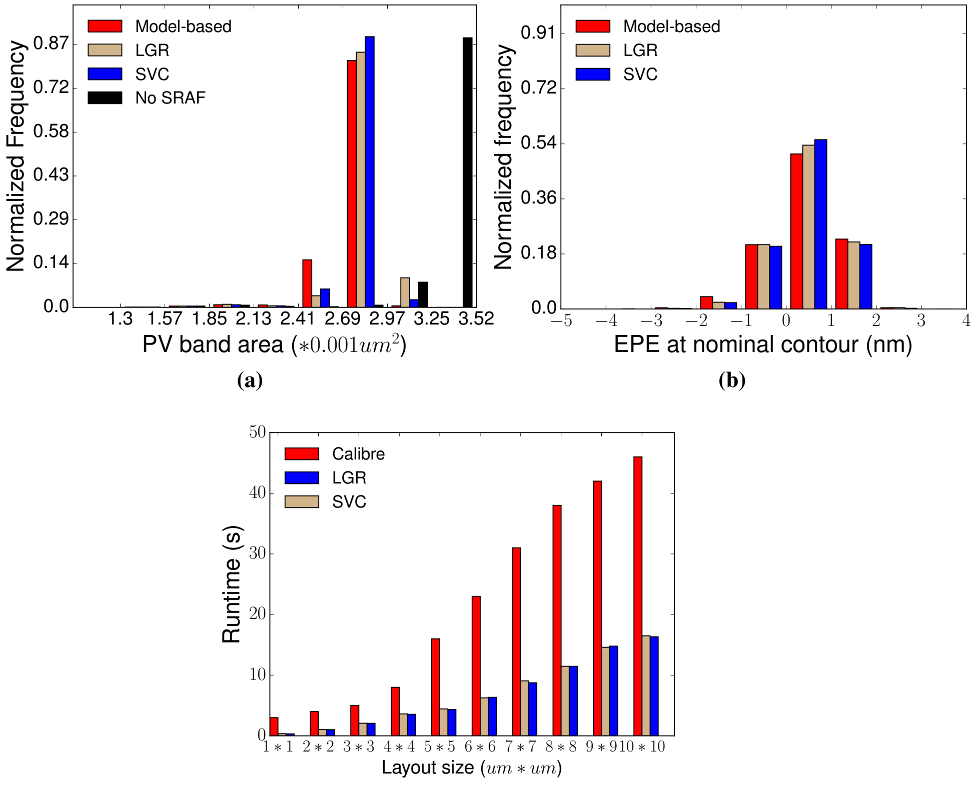
$$\min_{\boldsymbol{w},b,\xi} \quad \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_i \xi_i, \tag{5a}$$

$$\text{s.t.} \quad y_i(\boldsymbol{w}^T\boldsymbol{x_i}+b) \geq 1-\xi_i, \tag{5b}$$

$$\xi_i \geq 0, \quad \forall i, \tag{5c}$$

where $\boldsymbol{w}, b, \xi$ are variables to be determined during training. Variable b is the bias for the hyperplane and ξ denotes the error for the i^{th} data sample. The objective function contains two terms: one term for error minimization; the other for $L2$ regularization like that in logistic regression. Essentially SVM defines a hyperplane to maximize the margin between the decision boundaries.

The comparison of various SRAF generation approaches in EPE, PV band, and runtime, is shown in Fig. 7. The model-based SRAF generation uses Mentor Calibre with industrial-strength setup. As shown in Fig. 7(a), SVM based classification leads to better approximation to the model-based approach than does logistic regression in terms of PV band. The differences in EPE are marginal, as shown in Fig. 7(b). The major benefit of the learning-based approaches comes from the runtime, as shown in Fig. 7(c). Over 3X speedup for a layout clip with $10\mu m \times 10\mu m$ size can be achieved by both logistic regression and SVM due to efficient prediction.



(a)

(b)

(c)

Fig. 7 Comparison among different schemes in terms of, (a) PV band distribution, (b) EPE distribution at nominal conditions, and (c) runtime [47].

3.3 Machine Learning for Optical Proximity Correction

OPC is another important RET to improve the performance of advanced lithography. Fig. 8 demonstrates the effectiveness of OPC, where the edges of target patterns are fragmented and each segment is shifted in a way that the target patterns can be robustly printed, i.e., the EPE values are minimized. Conventional model-based OPC approaches are notorious for their runtime overhead [32, 16]. To overcome the runtime issue, regression models are proposed to enable fast full-chip OPC with an acceptable performance loss, such as linear regression [19, 16] and nonlinear regression [26, 25].

Fig. 9 shows the flow of regression-based OPC, which consists of training and testing phases. Besides standard steps for both model-based OPC and machine learning based approaches like edge fragmentation, training phase requires both model-based OPC and feature extraction for model calibration, while testing phase only needs feature extraction for model validation. Current regression-based techniques suffer from overfitting issues, degrading the accuracy of OPC results in the testing phase. In addition, increasingly complicated designs result in complex optical proximity effects toward the sub-resolution domain, causing the difficulty in achieving accurate regression models.

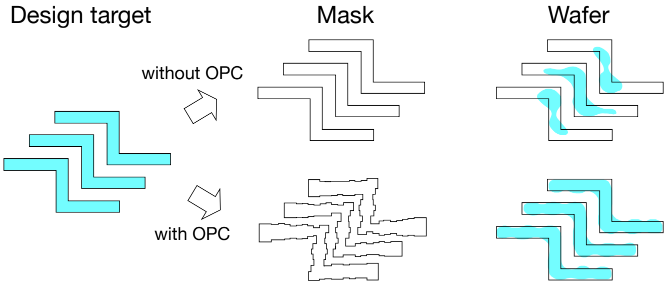


Fig. 8 Wafer patterns w./w.o. OPC [30].

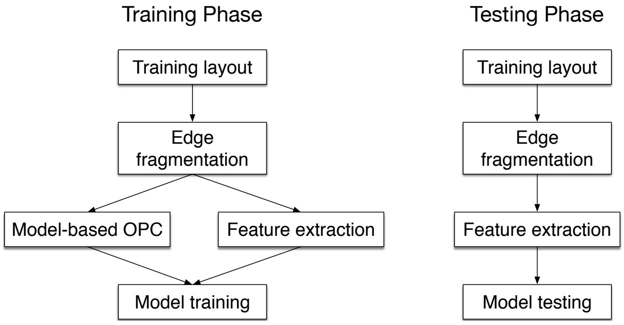


Fig. 9 Machine learning based OPC flow [30].

To overcome the aforementioned challenges, a hierarchical Bayes model (HBM) is proposed for the OPC problem with CCAS feature extraction [30]. In the HBM, a generalized linear mixed model (GLMM) is trained with explicit consideration of different edge types, including normal, convex, concave, line-end edge, etc. GLMM handles these edge types by regarding them as a random effect with a random variance. For unknown variables, the HBM assumes a non-informative prior distribution, thus avoiding the lack of prior information. Therefore, better OPC results can be generated by HBM compared with previous regression approaches.

The comparison between HBM-based approach and model-based (MB) approach is demonstrated in Fig. 10. MB_ik denotes OPC results from the k^{th} iteration of the MB-approach. The HBM-based approach can achieve EPE results comparable to

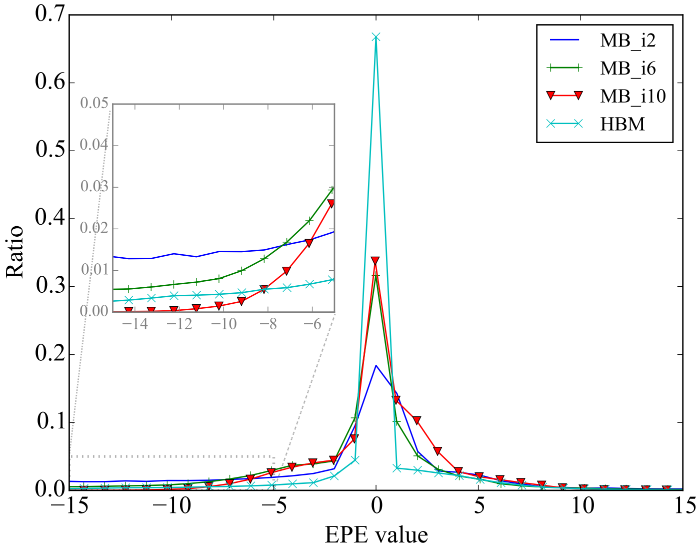


Fig. 10 Compare HBM-based and model-based OPC in terms of EPE distributions [30].

that of the MB-approach at the 10th iteration, while the former is much faster. Hence, it is suggested to initialize OPC conditions with HBM-based approach and use MB-based approach to finish the rest OPC iterations, such that the overall runtime can be reduced [30].

4 Machine Learning in Physical Design

Machine learning can not only benefit late stages in the backend design flow, but also early stages, such as physical design. Physical design contains many difficult combinatorial problems that are hard to solve optimally. These problems are getting even more complicated due to increasing constraints from design configurations and manufacturing in advanced technology nodes.

4.1 Machine Learning for Datapath Placement

Wirelength, such as steiner tree wirelength (StWL), is a widely used metric in VLSI placement for random logic designs. It is the first-order approximation to interconnect delay and capacitance. As minimizing HPWL is generally NP-hard [7], analytical placement with iterative optimizations is developed. Logic designs often contain datapaths in which cells are characterized with a high degree of bit-wise

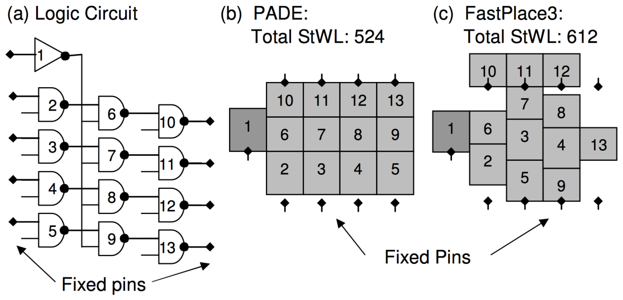


Fig. 11 An example of datapath-aware placement (PADE) achieves a 14% improvement in StWL compared to conventional placement (Fast-Place3) [42].

parallelism. Conventional analytical placement usually handles them sub-optimally [43]. For designs with many embedded datapaths, it is critical to extract and place them appropriately for high quality placement [42, 6, 46].

An example in which modern placers fail to handle datapaths effectively is shown in Fig. 11 [42, 54]. Fig. 11(a) sketches a datapath circuit in which there are three bit-stacks of cells, {2, 3, 4, 5}, {6, 7, 8, 9}, and {10, 11, 12, 13}, with fixed I/O pins. Two placement solutions, one from a datapath-aware placer, PADE [42], and the other from a conventional placer, Fast-Place3 [40], are shown in Fig. 11(b) and Fig. 11(c), respectively. PADE is able to achieve smaller StWL because it packs and aligns each bit-stack more tightly than does Fast-Place3.

The automatic extraction of datapath is critical to eventual placement quality, in which machine learning techniques can bring benefits. In the datapath extraction proposed by [42], datapaths are evaluated with SVM and ANN techniques and then extracted according to their importance. A combined SVM and ANN learning approach is developed to classify datapath and non-datapath patterns in the initial netlist for efficient modeling. SVM is able to achieve global optimal in maximizing the separation margin, while it is susceptible to data noise. ANN, on the other hand, is more robust to noise, but more difficult to achieve optimal training accuracy. A pattern is considered to be datapath if and only if both the SVM model and the ANN model output positive predictions, utilizing the advantanges of SVM and ANN.

4.2 Machine Learning for Routability-Driven Placement

Machine learning can also be applied to routability prediction in placement, which is very critical to the solution quality of modern placement. This is motivated by the gap between conventional routing congestion metrics, such as global routing congestion and the actual detailed routing violations [2, 3]. Fig. 12 shows an example in which the routing hotspots reported by global routing are quite different from those in actual detailed routing. Fig. 13 shows a similar comparison in which

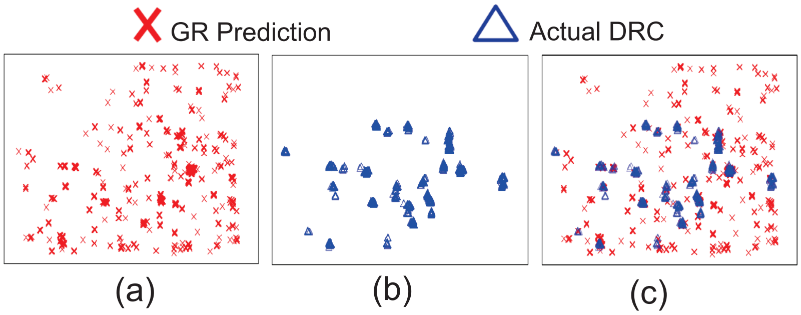


Fig. 12 (a) Routing hotspots predicted by global routing and (b) actual routing hotspots and (c) an overlay of predicted and actual routing hotspot [3].

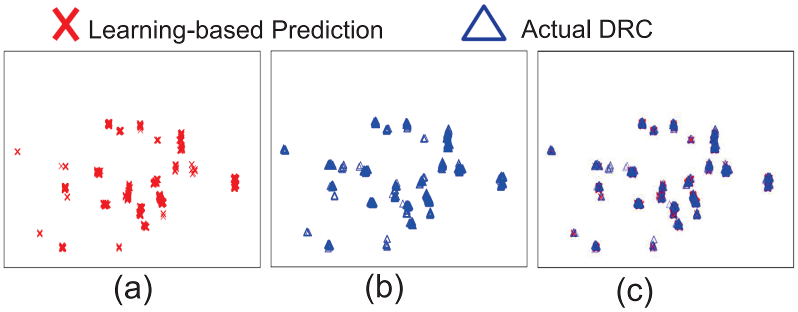


Fig. 13 (a) Routing hotspots predicted by machine learning and (b) actual routing hotspots and (c) an overlay of predicted and actual routing hotspot [3].

a SVM model with radius basis function (RBF) is adopted to build the correlation between layout features and routing hotspots. The routing hotspots reported by machine learning models are much more consistent with the actual routing hotspots. Various features in placement are extracted to train an accurate routing congestion predictor, including density parameters (e.g., local pin and cell density), global routing parameters (e.g., local overflow, demand, and capacity), pin proximity, cells tending to result in congestion (e.g., multi-height, sequential cells), connectivity parameters (e.g., number of nets in local windows), and structural parameters (e.g., number and depth of fanin and fanout logic stages). Hence, congestion prediction is very effective in placement.

Based on the routing hotspot predictor, a routability optimization algorithm is developed to redistribute white space, as shown in Fig. 14. White space around hotspots is extracted and redistributed by incrementally moving cells to improve routability. Incremental legalization is needed to remove overlaps between cells. Experimental results demonstrate an average of 20.6% and a maximum of 76.8% reduction in the number of DRC violations with negligible degradation in wirelength and timing [3].

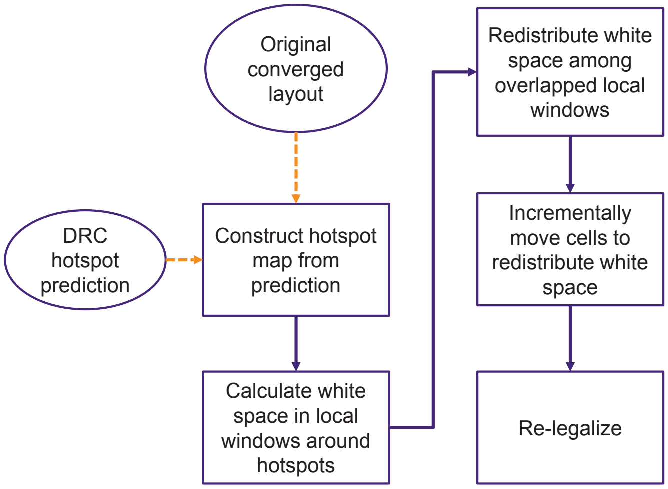

Original converged layout
DRC hotspot prediction
Construct hotspot map from prediction
Calculate white space in local windows around hotspots
Redistribute white space among overlapped local windows
Incrementally move cells to redistribute white space
Re-legalize

Fig. 14 Routability optimization guided by routing hotspot prediction [3].

4.3 Machine Learning for Clock Optimization

Besides placement, machine learning can also benefit the design of clock networks. Latch optimization including clustering and placement is of significant importance in modern VLSI designs to optimize skew and power consumption in clock networks. Latches in one cluster share a common local clock buffer (LCB) and are generally placed physically together [35, 5, 18]. Fig. 15 shows that latches are tightly clustered around LCBs, dramatically reducing the overall wirelength of local clock trees. A learning based latch optimization methodology is proposed in [44] with a genetic algorithm for initial latch placement and decision tree induction for latch template matching. It is reported that 20-30% average reduction in local clock tree capacitance is achieved in industrial designs.

Classification with decision tree predicts a target class F based on an input vector E in which $(E,F) = (e_1, e_2, e_3, ..., e_k, F)$ with $(1 \leq k \leq |E|)$. A decision tree learns by recursively partitioning the source data into subsequent subsets based on the attribute test [44]. This method has following advantages [33]:

- no prior probability distributions are required to data;
- greedy induction approaches provide good approximation to finding an optimal decision tree;
- fast prediction with worse case $\mathcal{O}(\omega)$ complexity is possible, where ω is the depth of the tree.

The decision tree induction algorithm is mainly used for structured template selection. This problem requires a quick decision on the best template given even an unknown set of input requirements. In addition, the algorithm can handle categorical

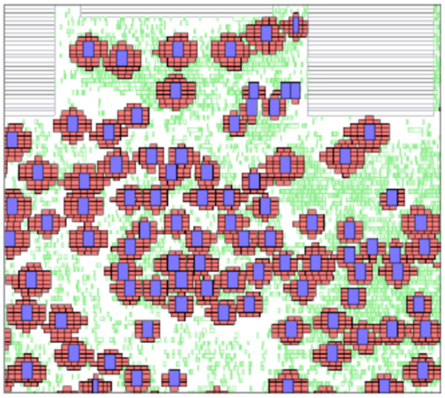


Fig. 15 Multi-GHz design showing clustered latches, where red cells are latches and purple cells are local clock buffers (LCB) [44].

Algorithm 2 Decision Tree Induction

```
 1: function TreeBuild($E, F$)
 2:     if $stopping\_condition(E,F) = TRUE$ then
 3:         Create a new node from the $leaf$;
 4:         Classify the $leaf$;
 5:         Return $leaf$;
 6:     else
 7:         Create a node $root$;
 8:         Find the best split and set it equal to the $root$ test condition;
 9:         Let $V$ = the set of possible outcome test conditions of the $root$ node;
10:         for each $v \in V$ do
11:             $E_v$ = training records given the $root$ test condition;
12:             $child$ = TreeBuild($E_v, F$);
13:             Add $child$ as descendent of $root$ and label;
14:         end for
15:     end if
16:     Return $root$;
17: end function
```

variables with multiple classes and is easy to implement and maintain. The template selection problem is a suitable application of this algorithm.

The decision tree induction algorithm is presented in Alg. 2 [44]. When the stopping condition is met, a new node is created with either a test condition or a class label, and then classify and return the final decision, in line 3-5. Let V be the set of possible class labels in node $leaf$ When the stopping condition is not yet met, a new node called *root* is created and the best split based on specific metrics is searched in line 7-8. For each possible label in *root*, line 11-13 recursively call for evaluating a new node.

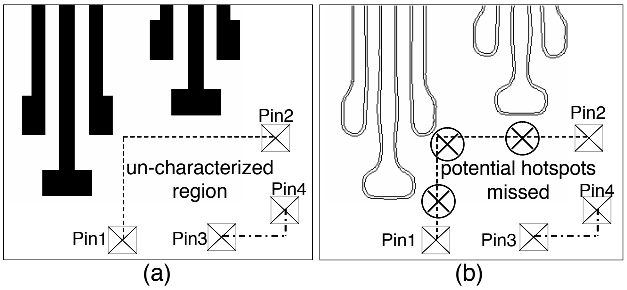


Fig. 16 The challenge of hotspot detection in detailed routing [8].

4.4 Machine Learning for Lithography Friendly Routing

Despite other applications, the last application of machine learning we introduce in this chapter is to improve lithography friendliness in early stage. Lithography hotspot mitigation in the post-routing stage lacks the flexibility and thus requires early consideration [31]. Rule-based approaches have been developed for hotspot correction [52, 12]. However, with learning-based hotspot prediction, hotspots can be identified with high accuracy to guide routing effectively.

The major challenge of hotspots detection in routing lies in the requirement of early prediction before a real routing path is obtained. A layout region with metal blockages and unrouted pins, Pin1 to Pin4, is shown in Fig. 16(a). Due to the existence of unrouted nets, general hotspot detection approaches fail to work, while potential hotspots may be caused by the routing segments from Pin1 to Pin2, as shown in Fig. 16(b). To tackle this problem, a pre-built hotspot prediction model and a routing path prediction model are developed for hotspot detection in routing [8]. The routing path prediction model explores possible routing solutions with given available routing resources and identify preferable routes according to routing congestion and hotspots predicted by the hotspot prediction model. Due to complexity of data, ANN is adopted as the classifier. The techniques report an average of 50% reduction on lithography hotspots with 18%∼30% runtime overhead compared with existing lithography friendly routing works.

5 Conclusion

Machine learning has demonstrated promising benefits to various key steps in the VLSI backend design flow. This chapter surveys several critical issues in physical verification, mask synthesis, and physical design, such as hotspot detection, SRAF generation, OPC, placement and routing optimization. There are also tremendous ongoing research in this area to develop effective and efficient techniques based on machine learning. This will enable fast closure of the backend design flow as well as increasing the quality of backend synthesis solutions such that timing, power, area, and yield can be eventually improved.

In addition, backend VLSI design is still at its early stage in applying machine learning techniques. For example, in the SRAF generation, pixel-by-pixel prediction is required and only linear models are used, limiting the application of more complicated models due to high computational expense. Similarly, OPC is only affordable to adopt linear models as well. Such kind of optimization problems essentially need to generate a new mask image with a given layout image. It is worth exploring whether generative learning techniques can be applied [15]. For placement and routing problems, manual selection of important features is still required, while it is not clear whether general representation of layout information exists and whether automatic feature selection can be developed. Furthermore, unlike fields with extensive research on machine learning like image recognition in which large amount of data is available, it is generally difficult and expensive to obtain enough data in VLSI design for training robust and accurate models. Therefore, it is critical to develop techniques to improve modeling accuracy with relaxed requirement of big data so that machine learning can be widely adopted. All these challenges remain to be explored in the future.

References

1. Andrew B. Kahng Chul-Hong Park, X.X.: Fast dual graph based hotspot detection. In: Proceedings of SPIE (2006)
2. Chan, W.T.J., Du, Y., Kahng, A.B., Nath, S., Samadi, K.: Beol stack-aware routability prediction from placement using data mining techniques. In: Computer Design (ICCD), 2016 IEEE 34th International Conference on, pp. 41–48. IEEE (2016)
3. Chan, W.T.J., Ho, P.H., Kahng, A.B., Saxena, P.: Routability optimization for industrial designs at sub-14nm process nodes using machine learning. In: ISPD, pp. 15–21 (2017)
4. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology **2**, 27:1–27:27 (2011). Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
5. Cho, M., Xiang, H., Ren, H., Ziegler, M.M., Puri, R.: LatchPlanner: latch placement algorithm for datapath-oriented high-performance VLSI designs. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 342–348 (2013)
6. Chou, S., Hsu, M.K., Chang, Y.W.: Structure-aware placement for datapath-intensive circuit designs. In: ACM/IEEE Design Automation Conference (DAC), pp. 762–767 (2012)
7. Chowdhury, S.: Analytical approaches to the combinatorial optimization in linear placement problems. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD) **8**(6), 630–639 (1989)
8. Ding, D., Gao, J.R., Yuan, K., Pan, D.Z.: AENEID: a generic lithography-friendly detailed router based on post-RET data learning and hotspot detection. In: ACM/IEEE Design Automation Conference (DAC), pp. 795–800 (2011)
9. Ding, D., Torres, J.A., Pan, D.Z.: High performance lithography hotspot detection with successively refined pattern identifications and machine learning. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD) **30**(11), 1621–1634 (2011)
10. Ding, D., Yu, B., Ghosh, J., Pan, D.Z.: Epic: Efficient predition of ic manufacturing hotspots with a unified meta-classfification formulation. In: IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC) (2012)

11. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: A library for large linear classification. Journal of machine learning research **9**(Aug), 1871–1874 (2008)
12. Gao, J.R., Jawandha, H., Atkar, P., Walimbe, A., Baidya, B., Rizzo, O., Pan, D.Z.: Self-aligned double patterning compliant routing with in-design physical verification flow. In: Proceedings of SPIE, vol. 8684, p. 868408 (2013)
13. Gao, J.R., Yu, B., Pan, D.Z.: Accurate lithography hotspot detection based on pca-svm classifier with hierarchical data clustering. In: Proc. SPIE, vol. 9053, p. 90530E (2014)
14. Goodfellow, I., Bengio, Y., Courville, A.: Deep learning. MIT press (2016)
15. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in neural information processing systems, pp. 2672–2680 (2014)
16. Gu, A., Zakhor, A.: Optical proximity correction with linear regression. IEEE Transactions on Semiconductor Manufacturing (TSM) **21**(2), 263–271 (2008)
17. Hastie, T., Tibshirani, R., Friedman, J., Franklin, J.: The elements of statistical learning: data mining, inference and prediction. The Mathematical Intelligencer **27**(2), 83–85 (2005)
18. Held, S., Schorr, U.: Post-routing latch optimization for timing closure. In: ACM/IEEE Design Automation Conference (DAC), pp. 7:1–7:6 (2014)
19. Jia, N., Lam, E.Y.: Machine learning for inverse lithography: using stochastic gradient descent for robust photomask synthesis. Journal of Optics **12**(4), 045,601:1–045,601:9 (2010)
20. Jun, J.H., Park, M., Park, C., Yang, H., Yim, D., Do, M., Lee, D., Kim, T., Choi, J., Luk-Pat, G., et al.: Layout optimization with assist features placement by model based rule tables for 2x node random contact. In: Proceedings of SPIE, pp. 94,270D–94,270D (2015)
21. Kim, B.S., Kim, Y.H., Lee, S.H., Kim, S.I., Ha, S.R., Kim, J., Tritchkov, A.: Pixel-based sraf implementation for 32nm lithography process. In: Proceedings of SPIE, pp. 71,220T–71,220T (2008)
22. Kodama, C., Kotani, T., Nojima, S., Mimotogi, S.: Sub-resolution assist feature arranging method and computer program product and manufacturing method of semiconductor device (2014). US Patent 8,809,072
23. Lin, S.Y., Chen, J.Y., Li, J.C., Wen, W.y., Chang, S.C.: A novel fuzzy matching model for lithography hotspot detection. In: ACM/IEEE Design Automation Conference (DAC) (2013)
24. Lin, Y., Xu, X., Ou, J., Pan, D.Z.: Machine learning for mask/wafer hotspot detection and mask synthesis. In: Photomask Technology, vol. 10451, p. 104510A. International Society for Optics and Photonics (2017)
25. Luo, K.S., Shi, Z., Yan, X.L., Geng, Z.: SVM based layout retargeting for fast and regularized inverse lithography. Journal of Zhejiang University SCIENCE C **15**(5), 390–400 (2014)
26. Luo, R.: Optical proximity correction using a multilayer perceptron neural network. Journal of Optics **15**(7), 075,708 (2013)
27. Matsunawa, T., Gao, J.R., Yu, B., Pan, D.Z.: A new lithography hotspot detection framework based on adaboost classifier and simplified feature extraction. In: Proceedings of SPIE, vol. 9427 (2015)
28. Matsunawa, T., Nojima, S., Kotani, T.: Automatic layout feature extraction for lithography hotspot detection based on deep neural network. In: Proceedings of SPIE (2016)
29. Matsunawa, T., Yu, B., Pan, D.Z.: Optical proximity correction with hierarchical bayes model. In: Proceedings of SPIE, vol. 9426 (2015)
30. Matsunawa, T., Yu, B., Pan, D.Z.: Optical proximity correction with hierarchical bayes model. Journal of Micro/Nanolithography, MEMS, and MOEMS **15**(2), 021,009–021,009 (2016)
31. Mitra, J., Yu, P., Pan, D.Z.: RADAR: RET-aware detailed routing using fast lithography simulations. In: ACM/IEEE Design Automation Conference (DAC), pp. 369–372 (2005)
32. Miyama, S., Yamamoto, K., Koyama, K.: Large-area optical proximity correction with a combination of rule-based and simulation-based methods. Japanese Journal of Applied Physics **35**(12S), 6370 (1996)
33. Moshkov, M.J.: Time complexity of decision trees. In: Transactions on Rough Sets III, pp. 244–459. Springer (2005)
34. Pang, L., Liu, Y., Abrams, D.: Inverse lithography technology (ilt): a natural solution for model-based sraf at 45nm and 32nm. In: Proceedings of SPIE, pp. 660,739–660,739 (2007)

35. Papa, D., Alpert, C., Sze, C., Li, Z., Viswanathan, N., Nam, G.J., Markov, I.L.: Physical synthesis with clock-network optimization for large systems on chips. IEEE Micro **31**(4), 51–62 (2011)
36. Ping, Y., McGowan, S., Gong, Y., Foong, Y.M., Liu, J., Qiu, J., Shu, V., Yan, B., Ye, J., Li, P., et al.: Process window enhancement using advanced ret techniques for 20nm contact layer. In: Proceedings of SPIE, pp. 90,521N–90,521N (2014)
37. Sakajiri, K., Tritchkov, A., Granik, Y.: Model-based sraf insertion through pixel-based mask optimization at 32nm and beyond. In: Proceedings of SPIE, pp. 702,811–702,811 (2008)
38. Shang, S.D., Swallow, L., Granik, Y.: Model-based sraf insertion (2011). US Patent 8,037,429
39. Shin, M., Lee, J.H.: Accurate lithography hotspot detection using deep convolutional neural networks. In: Journal of Micro/Nanolithography, MEMS, and MOEMS (JM3) (2016)
40. Viswanathan, N., Pan, M., Chu, C.: FastPlace 3.0: A fast multilevel quadratic placement algorithm with placement congestion control. In: IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC), pp. 135–140 (2007)
41. Viswanathan, R., Azpiroz, J.T., Selvam, P.: Process optimization through model based sraf printing prediction. In: Proceedings of SPIE, pp. 83,261A–83,261A (2012)
42. Ward, S., Ding, D., Pan, D.Z.: PADE: a high-performance placer with automatic datapath extraction and evaluation through high dimensional data learning. In: ACM/IEEE Design Automation Conference (DAC), pp. 756–761 (2012)
43. Ward, S.I., Papa, D.A., Li, Z., Sze, C.N., Alpert, C.J., Swartzlander, E.: Quantifying academic placer performance on custom designs. In: ACM International Symposium on Physical Design (ISPD), pp. 91–98 (2011)
44. Ward, S.I., Viswanathan, N., Zhou, N.Y., Sze, C.C., Li, Z., Alpert, C.J., Pan, D.Z.: Clock power minimization using structured latch templates and decision tree induction. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 599–606 (2013)
45. Wen, W.Y., Li, J.C., Lin, S.Y., Chen, J.Y., Chang, S.C.: A fuzzy-matching model with grid reduction for lithography hotspot detection. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD) **33**(11), 1671–1680 (2014)
46. Xiang, H., Cho, M., Ren, H., Ziegler, M., Puri, R.: Network flow based datapath bit slicing. In: ACM International Symposium on Physical Design (ISPD), pp. 139–146 (2013)
47. Xu, X., Lin, Y., Li, M., Matsunawa, T., Nojima, S., Kodama, C., Kotani, T., Pan, D.Z.: Sub-Resolution Assist Feature Generation with Supervised Data Learning. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD) **PP**(99) (2017)
48. Xu, X., Matsunawa, T., Nojima, S., Kodama, C., Kotani, T., Pan, D.Z.: A Machine Learning Based Framework for Sub-Resolution Assist Feature Generation. In: ACM International Symposium on Physical Design (ISPD), pp. 161–168 (2016)
49. Yang, H., Lin, Y., Yu, B., Young, F.E.: Lithography hotspot detection: From shallow to deep learning. In: IEEE International System-on-Chip Conference (SOCC) (2017)
50. Yang, H., Luo, L., Su, J., Lin, C., Yu, B.: Imbalance aware lithography hotspot detection: A deep learning approach. In: Proceedings of SPIE (2017)
51. Yang, H., Su, J., Zou, Y., Yu, B., Young, F.E.: Layout hotspot detection with feature tensor generation and deep biased learning. In: ACM/IEEE Design Automation Conference (DAC) (2017)
52. Yang, J., Rodriguez, N., Omedes, O., Gennari, F., Lai, Y.C., Mankad, V.: DRCPlus in a router: automatic elimination of lithography hotspots using 2D pattern detection and correction. In: Proceedings of SPIE, vol. 7641, p. 76410Q (2010)
53. Ye, J., Cao, Y., Feng, H.: System and method for model-based sub-resolution assist feature generation (2011). US Patent 7,882,480
54. Yu, B., Pan, D.Z., Matsunawa, T., Zeng, X.: Machine learning and pattern matching in physical design. In: Design Automation Conference (ASP-DAC), 2015 20th Asia and South Pacific, pp. 286–293. IEEE (2015)
55. Yu, Y.T., Chan, Y.C., Sinha, S., Jiang, I.H.R., Chiang, C.: Accurate process-hotspot detection using critical design rule extraction. In: ACM/IEEE Design Automation Conference (DAC) (2012)

56. Yu, Y.T., Lin, G.H., Jiang, I.H.R., Chiang, C.: Machine learning based hotspot detection using topological classification and critical feature extraction. In: ACM/IEEE Design Automation Conference (DAC) (2013)
57. Zhang, H., Yu, B., Evangeline, Y.F.: Enabling online learning in lithography hotspot detection with information-theoretic feature optimization. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD) (2016)
58. Zhang, H., Zhu, F., Li, H., Young, F.E., Yu, B.: Bilinear lithography hotspot detection. In: ACM International Symposium on Physical Design (ISPD) (2017)