SRAF Insertion via Supervised Dictionary Learning

Hao Geng
CSE Department, CUHK
hgeng@cse.cuhk.edu.hk

Haoyu Yang
CSE Department, CUHK
hyyang@cse.cuhk.edu.hk

Yuzhe Ma
CSE Department, CUHK
yzma@cse.cuhk.edu.hk

Joydeep Mitra
Cadence Design Systems Inc.
joydeepm@cadence.com

Bei Yu
CSE Department, CUHK
byu@cse.cuhk.edu.hk

ABSTRACT

In modern VLSI design flow, sub-resolution assist feature (SRAF) insertion is one of the resolution enhancement techniques (RETs) to improve chip manufacturing yield. With aggressive feature size continuously scaling down, layout feature learning becomes extremely critical. In this paper, for the first time, we enhance conventional manual feature construction, by proposing a supervised online dictionary learning algorithm for simultaneous feature extraction and dimensionality reduction. By taking advantage of label information, the proposed dictionary learning engine can discriminatively and accurately represent the input data. We further consider SRAF design rules in a global view, and design an integer linear programming model in the post-processing stage of SRAF insertion framework. Experimental results demonstrate that, compared with a state-of-the-art SRAF insertion tool, our framework not only boosts the mask optimization quality in terms of edge placement error (EPE) and process variation (PV) band area, but also achieves some speed-up.

1 INTRODUCTION

As feature size of semiconductors entering nanometer era, lithographic process variations are emerging as more severe issues in chip manufacturing process. That is, these process variations may result in manufacturing defects and a decrease of yield. Besides some design for manufacturability (DFM) approaches such as multiple patterning and litho-friendly layout design [1, 2], a de facto solution alleviating variations is mask optimization through various resolution enhancement techniques (RETs) (e.g. [3, 4]).

Sub-resolution assist feature (SRAF) [5] insertion is one representative strategy among numerous RET techniques. Without printing SRAF patterns themselves, the small SRAF patterns can transfer light to the positions of target patterns, and therefore SRAFs are able to improve the robustness of the target patterns under different lithographic variations. A lithographic simulation example demonstrating the benefit of SRAF insertion is illustrated in Figure 1. Here process variation (PV) band (i.e. yellow circuit) area is applied to measure the performance of lithographic process window. As a matter of fact, better printing performance, smaller area of the PV band. In Figure 1(a), only optical proximity correction (OPC) is conducted to improve the printability of the target pattern, while in Figure 1(b) both SRAF insertion and OPC are exploited. We can see that, through SRAF insertion, the PV band area of the printed target pattern is effectively reduced from 2688 nm^2 as in Figure 1(a) to 2318 nm^2 as in Figure 1(b).

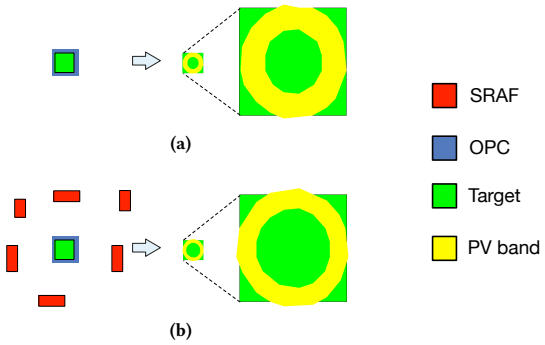

Figure 1: (a) Printing with OPC only (2688 nm^2 PV band area); (b) Printing with both OPC and SRAF (2318 nm^2 PV band area).

There is a wealth of literature on the topic of SRAF insertion for mask optimization, which can be roughly divided into three categories: rule-based approach, model-based approach, and machine learning-based approach [6, 7]. Rule-based method is able to achieve high performance on simple designs, but it cannot handle complicated target patterns. Although model-based approach has a better performance, it is unfortunately very time-consuming. Recently, Xu *et al.* investigated an SRAF insertion framework based on machine learning techniques [7]. By calibrating a mathematical model based on the training data set, the calibrated model draws inferences that can guide SRAF insertion from testing data set. However, on account of coarse feature extraction techniques and lack of global view in SRAF designs, the simulation results may not be good enough.

In a machine learning-based SRAF insertion flow, before fed into the learning engine, raw clips should be preprocessed in feature extraction stage. One of the key takeaways of previous arts is the importance of features extracted from clips that leverage prior gained knowledge to achieve expected results. Namely, with more representative, generalized and discriminative layout features, the calibrated model performs better. In this paper, we argue that the label information utilized in learning stage can be further imposed in feature extraction stage, which in turn will benefit the learning

ASPDAC '19, January 21–24, 2019, Tokyo, Japan

ACM ISBN 978-1-4503-6007-4/19/01…$15.00
https://doi.org/10.1145/3287624.3287684

counterpart. In accordance with this argument, we propose a supervised online dictionary learning algorithm, which converts features from high-dimension space into a low-dimension space, meanwhile label information is integrated into the feature representation. To the best of our knowledge, this is the first layout feature extraction work seamlessly combining with label information. There is no prior art in applying the dictionary learning techniques or further supervised dictionary approaches into SRAF insertion issue. Our main contributions are listed as follows.

- Leverage supervised online dictionary learning algorithm to handle a large amount of layout patterns.
- Our proposed feature is more discriminative and representative, and is embedded into SRAF insertion framework.
- The SRAF insertion with design rules is modeled as an integer linear programming problem.
- Experimental results show that our method not only boosts F_1 score of machine learning model, but also achieves some speed-up.

The rest of this paper is organized as following. Section 2 introduces the problem to be addressed in the paper and illustrates the whole working flow of our framework to insert SRAFs. Section 3 firstly describes the specific feature extraction method, and then proposes our supervised online dictionary learning method. Section 4 reveals the integer linear programming framework in post-processing stage. Section 5 presents the experiment results, followed by conclusion in Section 6.

2 PRELIMINARY

2.1 Problem Formulation

Given a machine learning model, F_1 score is used to measure its accuracy. Specifically, the higher, the better. Besides F_1 score, we also exploit other two metrics, PV band area and edge placement error (EPE), to quantify lithographic simulation results. We define SRAF insertion problem as follows.

Problem 1 (SRAF Insertion). Given a training set of layout clips and specific SRAF design rules, the objective of SRAF insertion is to place SRAFs in the testing set of layout clips so that the corresponding PV band and the EPE under nominal condition are minimized.

2.2 Overall Flow

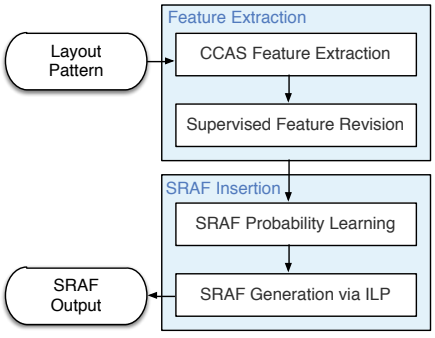

Figure 2: The proposed SRAF insertion flow.

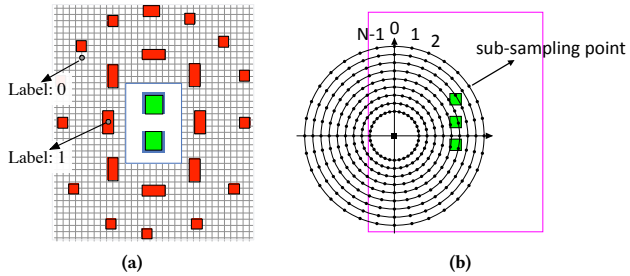

Figure 3: (a) SRAF label; (b) CCAS feature extraction method in machine learning model-based SRAF generation.

The overall flow of our proposed SRAF insertion is shown in Figure 2, which consists of two stages: feature extraction and SRAF insertion. In the feature extraction stage, after feature extraction via concentric circle area sampling (CCAS), we propose supervised feature revision, namely, mapping features into a discriminative low-dimension space. Through dictionary training, our dictionary consists of atoms which are representatives of original features. The original features are sparsely encoded over a well-trained dictionary and described as combinations of atoms. Due to space transformation, the new features (i.e. sparse codes) are more abstract and discriminative with little important information loss for classification. Therefore, proposed supervised feature revision is expected to avoid over-fitting of a machine learning model. In the second stage, based on the predictions inferred by learning model, SRAF insertion can be treated as a mathematical optimization problem accompanied by taking design rules into consideration.

3 FEATURE EXTRACTION

In this section, we firstly introduce the CCAS feature extraction method, and specify supervised feature revision. By the end, we give the details about our supervised online dictionary learning algorithm and corresponding analysis.

3.1 CCAS Feature Extraction

With considering concentric propagation of diffracted light from mask patterns, recently proposed CCAS [8] layout feature is used in SRAF generation domain.

In SRAF insertion, the raw training data set is made up of layout clips, which include a set of target patterns and model-based SRAFs. Each layout clip is put on a 2-D grid plane with a specific grid size so that real training samples can be extracted via CCAS method at each grid. For every sample, according to the model-based SRAFs, the corresponding label is either "1" or "0". As Figure 3(a) illustrates, "1" means inserting an SRAF at this grid and "0" vice versa. Figure 3(b) shows the feature extraction method in SRAF generation.

However, since adjacent circles contain similar information, the CCAS feature has much redundancy. In fact, the redundancy will hinder the fitting of a machine learning model.

3.2 Supervised Feature Revision

With CCAS feature as input, the dictionary learning model is expected to output the discriminative feature of low-dimension. In the topic of data representation [9], a self-adaptive dictionary learning

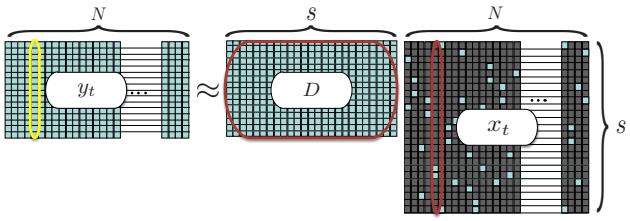

Figure 4: The overview of a dictionary learning model.

model can sparsely and accurately represent data as linear combinations of atoms (i.e., columns) from a dictionary matrix. This model reveals the intrinsic characteristics of raw data.

In recent arts, sparse decomposition and dictionary construction are coupled in a self-adaptive dictionary learning framework. As a result, the framework can be modeled as a constraint-optimization problem. The joint objective function of a self-adaptive dictionary model for feature revision problem is proposed as Equation (1):

$$\min_{\boldsymbol{x},\boldsymbol{D}} \frac{1}{N}\sum_{t=1}^{N}\{\frac{1}{2}\left\|\boldsymbol{y}_t-\boldsymbol{D}\boldsymbol{x}_t\right\|_2^2+\lambda\left\|\boldsymbol{x}_t\right\|_p\}, \tag{1}$$

where $\boldsymbol{y}_t \in \mathbb{R}^n$ is an input CCAS feature vector, and $\boldsymbol{D}=\left\{\boldsymbol{d}_j\right\}_{j=1}^{s}, \boldsymbol{d}_j \in \mathbb{R}^n$ refers to the dictionary made up of atoms to encode input features. $\boldsymbol{x}_t \in \mathbb{R}^s$ indicates sparse codes (i.e. sparse decomposition coefficients) with p the type of norm. Meanwhile, N is the total number of training data vectors in memory. The above equation, illustrated in Figure 4, consists of a series of reconstruction error, $\|\boldsymbol{y}_t-\boldsymbol{D}\boldsymbol{x}_t\|_2^2$, and the regularization term $\|\boldsymbol{x}_t\|_p$. In Figure 4, every grid represents a numerical value, and dark grid of $\boldsymbol{x}_t$ indicates zero. It can be seen that the motivation of dictionary learning is to sparsely encode input CCAS features over a well-trained dictionary.

However, from Equation (1), it is easy to discover that the main optimization goal is minimizing the reconstruction error in a mean squared sense, which may not be compatible to the goal of classification. Therefore, we try to explore the supervised information, and then propose our joint objective function as Equation (2). An assumption has been made in advance that each atom is associated with a particular label, which is true as each atom is selected to represent a subset of the training CCAS features ideally from one class (i.e. occupied with an SRAF or not).

$$\min_{\boldsymbol{x},\boldsymbol{D},\boldsymbol{A}} \frac{1}{N}\sum_{t=1}^{N}\{\frac{1}{2}\left\|\left(\boldsymbol{y}_t^\top,\sqrt{\alpha}\boldsymbol{q}_t^\top\right)^\top-\begin{pmatrix}\boldsymbol{D}\\ \sqrt{\alpha}\boldsymbol{A}\end{pmatrix}\boldsymbol{x}_t\right\|_2^2+\lambda\|\boldsymbol{x}_t\|_p\}. \tag{2}$$

In Equation (2), α is a hyper-parameter balancing the contribution of each part to reconstruction error. $\boldsymbol{q}_t \in \mathbb{R}^s$ is defined as discriminative sparse code of t-th input feature vector. Hence, $\boldsymbol{A} \in \mathbb{R}^{s\times s}$ transforms original sparse code $\boldsymbol{x}_t$ into discriminative sparse code. In $\boldsymbol{q}_t$, the non-zero elements indicate that the corresponding atoms share the same label with t-th input. Given dictionary $\boldsymbol{D}$, it is obvious that $\boldsymbol{q}_t$ has some fixed types.

For example, assume $\boldsymbol{D}=\{\boldsymbol{d}_1,\boldsymbol{d}_2,\boldsymbol{d}_3,\boldsymbol{d}_4\}$ with $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$ from class 1, $\boldsymbol{d}_3$ and $\boldsymbol{d}_4$ from class 2, then $\boldsymbol{q}_t$ for the corresponding input $\boldsymbol{y}_t$ is supposed to be either $(1,1,0,0)^\top$ or $(0,0,1,1)^\top$. For further explanation, we merge different types of $\boldsymbol{q}_t$ as a $\boldsymbol{Q}$ matrix:

$$\boldsymbol{Q}=\left(\boldsymbol{q}_1,\quad \boldsymbol{q}_2\right)=\begin{pmatrix}1&0\\1&0\\0&1\\0&1\end{pmatrix}, \tag{3}$$

where $(1,1,0,0)^\top$ means input sample shares the same label with $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$, and $(0,0,1,1)^\top$ indicates that the input, $\boldsymbol{d}_3$ and $\boldsymbol{d}_4$ are from the same class.

To illustrate physical meaning of Equation (2) clearly, we can also rewrite it via splitting the reconstruction term into two terms within l_2-norm as (4):

$$\min_{\boldsymbol{x},\boldsymbol{D},\boldsymbol{A}} \frac{1}{N}\sum_{t=1}^{N}\{\frac{1}{2}\left\|\boldsymbol{y}_t-\boldsymbol{D}\boldsymbol{x}_t\right\|_2^2+\frac{\alpha}{2}\left\|\boldsymbol{q}_t-\boldsymbol{A}\boldsymbol{x}_t\right\|_2^2+\lambda\|\boldsymbol{x}_t\|_p\}. \tag{4}$$

The first term $\|\boldsymbol{y}_t-\boldsymbol{D}\boldsymbol{x}_t\|_2^2$ is still the reconstruction error term. The second term $\|\boldsymbol{q}_t-\boldsymbol{A}\boldsymbol{x}_t\|_2^2$ represents discriminative error, which imposes a constraint on the approximation of $\boldsymbol{q}_t$. As a result, the input CCAS features from same class share quite similar representations.

Since the latent supervised information has been used, the label information can also be directly employed. After adding the prediction error term into initial objective function Equation (2), we propose our final joint objective function as Equation (5):

$$\min_{\boldsymbol{x},\boldsymbol{D},\boldsymbol{A},\boldsymbol{W}} \frac{1}{N}\sum_{t=1}^{N}\{\frac{1}{2}\left\|\left(\boldsymbol{y}_t^\top,\sqrt{\alpha}\boldsymbol{q}_t^\top,\sqrt{\beta}h_t\right)^\top-\begin{pmatrix}\boldsymbol{D}\\ \sqrt{\alpha}\boldsymbol{A}\\ \sqrt{\beta}\boldsymbol{W}\end{pmatrix}\boldsymbol{x}_t\right\|_2^2+\lambda\|\boldsymbol{x}_t\|_p\}, \tag{5}$$

where $h_t \in \mathbb{R}$ is the label with $\boldsymbol{W} \in \mathbb{R}^{1\times s}$ the related weight vector, and therefore $\|h_t-\boldsymbol{W}\boldsymbol{x}_t\|_2^2$ refers to the classification error. α and β are hyper-parameters which control the contribution of each term to reconstruction error and balance the trade-off. So this formulation can produce a good representation of original CCAS feature.

3.3 Online Algorithm

Recently, some attempts which explore label information are proposed in succession such as Discriminative K-SVD [10], kernelized supervised dictionary learning [11], label consistent K-SVD [12] (LCK-SVD) dictionary learning and supervised K-SVD with dual-graph constraints [13].

However, most of them are based on K-SVD [14] which belongs to batch-learning method. They are not suitable for dealing with large dataset since the computation overhead (e.g. computing the inverse of a very large matrix) may be high. Online learning method applied in dictionary learning model [15, 16] is a good idea, yet these algorithms are unsupervised.

Therefore, we develop an efficient online learning method, which seamlessly combines aforementioned supervised dictionary learning. Unlike the batch approaches, online approaches process training samples incrementally, one training sample (or a small batch of training samples) at a time, similarly to stochastic gradient descent.

According to our proposed formulation (i.e. Equation (5)), the joint optimization of both dictionary and sparse codes is non-convex, but sub-problem with one variable fixed is convex. Hence, Equation (5) can be divided into two convex sub-problems. Note

that, in a taste of linear algebra, our new input with label information, i.e. $\left(\boldsymbol{y}_t^\top, \sqrt{\alpha}\boldsymbol{q}_t^\top, \sqrt{\beta}h_t\right)^\top$ in Equation (5), can be still regarded as the original $\boldsymbol{y}_t$ in Equation (1). So is the new merged dictionary consisting of $\boldsymbol{D}$, $\boldsymbol{A}$ and $\boldsymbol{W}$. For simplicity of description and derivation, in following analysis, we will use $\boldsymbol{y}_t$ referring to $\left(\boldsymbol{y}_t^\top, \sqrt{\alpha}\boldsymbol{q}_t^\top, \sqrt{\beta}h_t\right)^\top$ and $\boldsymbol{D}$ standing for merged dictionary with $\boldsymbol{x}$ as the sparse codes.

Two stages, sparse coding and dictionary constructing, alternatively perform in iterations. Thus, in t-th iteration, the algorithm firstly draws the input sample $\boldsymbol{y}_t$ or a mini-batch over the current dictionary $\boldsymbol{D}_{t-1}$ and obtains the corresponding sparse codes $\boldsymbol{x}_t$. Then use two updated auxiliary matrices, $\boldsymbol{B}_t$ and $\boldsymbol{C}_t$ to help computing $\boldsymbol{D}_t$.

The objective function for sparse coding is showed in (6):

$$\boldsymbol{x}_t \triangleq \arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{y}_t - \boldsymbol{D}_{t-1}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1. \tag{6}$$

If the regularizer adopts l_0-norm, solving Equation (6) is NP-hard. Therefore, we utilize l_1-norm as a convex replacement of l_0-norm. In fact, Equation (6) is the classic Lasso problem [17], which can be solved by any Lasso solver.

Two auxiliary matrices $\boldsymbol{B}_t \in \mathbb{R}^{(n+s+1)\times s}$ and $\boldsymbol{C}_t \in \mathbb{R}^{s\times s}$ are defined respectively in (7) and (8):

$$\boldsymbol{B}_t \leftarrow \frac{t-1}{t}\boldsymbol{B}_{t-1} + \frac{1}{t}\boldsymbol{y}_t\boldsymbol{x}_t^\top, \tag{7}$$

$$\boldsymbol{C}_t \leftarrow \frac{t-1}{t}\boldsymbol{C}_{t-1} + \frac{1}{t}\boldsymbol{x}_t\boldsymbol{x}_t^\top. \tag{8}$$

The objective function for dictionary construction is:

$$\boldsymbol{D}_t \triangleq \arg\min_{\boldsymbol{D}} \frac{1}{t}\sum_{i=1}^{t}\{\frac{1}{2}\|\boldsymbol{y}_i - \boldsymbol{D}\boldsymbol{x}_i\|_2^2 + \lambda\|\boldsymbol{x}_i\|_1\}. \tag{9}$$

Algorithm 1 Supervised Online Dictionary Learning (SODL)

Input: Input merged features $Y \leftarrow \{\boldsymbol{y}_t\}_{t=1}^N, \boldsymbol{y}_t \in \mathbb{R}^{(n+s+1)}$ (including original CCAS features, discriminative sparse code $\boldsymbol{Q} \leftarrow \{\boldsymbol{q}_t\}_{t=1}^N, \boldsymbol{q}_t \in \mathbb{R}^s$ and label information $\boldsymbol{H} \leftarrow \{h_t\}_{t=1}^N, h_t \in \mathbb{R}$).

Output: New features $\boldsymbol{X} \leftarrow \{\boldsymbol{x}_t\}_{t=1}^N, \boldsymbol{x}_t \in \mathbb{R}^s$, dictionary $\boldsymbol{D} \leftarrow \{\boldsymbol{d}_j\}_{j=1}^s, \boldsymbol{d}_j \in \mathbb{R}^{(n+s+1)}$.

1: **Initialization**: Initial merged dictionary $\boldsymbol{D}_0, \boldsymbol{d}_j \in \mathbb{R}^{(n+s+1)}$ (including initial transformation matrix $\boldsymbol{A}_0 \in \mathbb{R}^{s\times s}$ and initial label weight matrix $\boldsymbol{W}_0 \in \mathbb{R}^{1\times s}$), $\boldsymbol{C}_0 \in \mathbb{R}^{s\times s} \leftarrow \boldsymbol{0}$, $\boldsymbol{B}_0 \in \mathbb{R}^{(n+s+1)\times s} \leftarrow \boldsymbol{0}$;
2: **for** $t \leftarrow 1$ to N **do**
3: Sparse coding $\boldsymbol{y}_t$ and obtaining $\boldsymbol{x}_t$; ▷ Equation (6)
4: Update auxiliary variable $\boldsymbol{B}_t$; ▷ Equation (7)
5: Update auxiliary variable $\boldsymbol{C}_t$; ▷ Equation (8)
6: Update dictionary $\boldsymbol{D}_t$; ▷ Algorithm 2;
7: **end for**

Algorithm 1 summarizes the algorithm details of the proposed supervised online dictionary learning (SODL) algorithm. We use coordinate descent algorithm as the solving scheme to Equation (6) (line 3). To accelerate the convergence speed, Equation (9) involves the computations of past signals $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t$ and the sparse codes $\boldsymbol{x}_1, ..., \boldsymbol{x}_t$. One way to efficiently update dictionary is that introduce some sufficient statistics, i.e. $\boldsymbol{B}_t \in \mathbb{R}^{(n+s+1)\times s}$ (line 4) and $\boldsymbol{C}_t \in \mathbb{R}^{s\times s}$ (line 5), into Equation (9) without directly storing the past input data sample $\boldsymbol{y}_i$ and corresponding sparse codes $\boldsymbol{x}_i$ for $i \le t$. These two auxiliary variables play important roles in updating atoms, which summarizes the past information from sparse coefficients and input data. We further exploit block coordinate method with warm start [18] to resolve Equation (9) (line 6). As a result, through some gradient calculations, we bridge the gap between Equation (9) and sequentially updating atoms based on Equations (10) and (11).

$$\boldsymbol{u}_j \leftarrow \frac{1}{\boldsymbol{C}[j,j]}\left(\boldsymbol{b}_j - \boldsymbol{D}\boldsymbol{c}_j\right) + \boldsymbol{d}_j. \tag{10}$$

$$\boldsymbol{d}_j \leftarrow \frac{1}{\max\left(\|\boldsymbol{u}_j\|_2, 1\right)}\boldsymbol{u}_j. \tag{11}$$

For each atom $\boldsymbol{d}_j$, the updating rule is illustrated in Algorithm 2. In Equation (10), $\boldsymbol{D}_{t-1}$ is selected as the warm start of $\boldsymbol{D}$. $\boldsymbol{b}_j$ indicates the j-th column of $\boldsymbol{B}_t$, while $\boldsymbol{c}_j$ is the j-th column of $\boldsymbol{C}_t$. $\boldsymbol{C}[j,j]$ denotes the j-th element on diagonal of $\boldsymbol{C}_t$. Equation (11) is an l_2-norm constraint on atoms to prevent atoms becoming arbitrarily large (which may lead to arbitrarily small sparse codes). [19] proves that in the stage of constructing dictionary, the convex optimization problem allowing separable constraints in the updated blocks (columns) will guarantee the convergence to a global optimum.

Algorithm 2 Rules for Updating Atoms

Input: $\boldsymbol{D}_{t-1} \leftarrow \{\boldsymbol{d}_j\}_{j=1}^s, \boldsymbol{d}_j \in \mathbb{R}^{(n+s+1)}$,
$\boldsymbol{B}_t \leftarrow \{\boldsymbol{b}_j\}_{j=1}^s, \boldsymbol{b}_j \in \mathbb{R}^{(n+s+1)}$,
$\boldsymbol{C}_t \leftarrow \{\boldsymbol{c}_j\}_{j=1}^s, \boldsymbol{c}_j \in \mathbb{R}^s$.

Output: dictionary $\boldsymbol{D}_t \leftarrow \{\boldsymbol{d}_j\}_{j=1}^s, \boldsymbol{d}_j \in \mathbb{R}^{(n+s+1)}$.

1: **for** $j \leftarrow 1$ to s **do**
2: Update the j-th atom $\boldsymbol{d}_j$; ▷ Equations (10) and (11)
3: **end for**

The proposed algorithm, SODL, handles the non-convex optimization problem. So finding the global optimum is not guaranteed. Although our algorithm will converge to a stationary point of the objective function, for practical applications, stationary points are empirically enough.

4 SRAF INSERTION

4.1 SRAF Probability Learning

After feature extraction via CCAS and proposed SODL framework, the new discriminative feature in low-dimension is fed into the machine learning model. For fair comparison, we exploit the same classifier, logistic regression, as used in [7]. The classifier is calibrated by the training samples and then predict the SRAF labels as probabilities for testing instances.

4.2 SRAF Insertion via ILP

Through SODL model and classifier, the probabilities of each 2-D grid can be obtained. Based on design rules for the machine

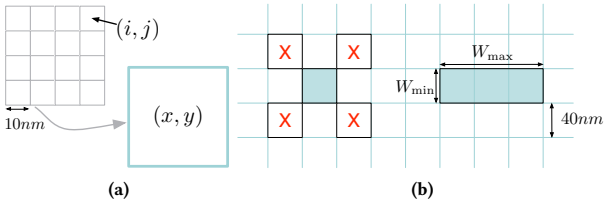

Figure 5: (a) SRAF grid model construction; (b) SRAF insertion design rule under the grid model.

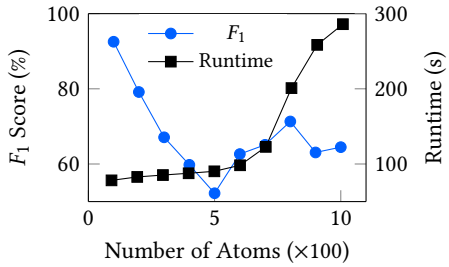

Figure 6: The trend of changing number of atoms.

learning model, the label for a grid with probability less than the threshold is "0". It means that the grid will be ignored when doing SRAF insertion. However, in [7], the scheme to insert SRAFs is a little naive and greedy. Actually, combined with some relaxed SRAF design rules such as maximum length and width, minimum spacing, the SRAF insertion can be modeled as an integer linear programming (ILP) problem. With ILP model to formulate SRAF insertion, we will obtain a global view for SRAF generation.

In the objective of the ILP approach, we only consider valid grids whose probabilities are larger than the threshold. The probability of each grid is denoted as $p(i, j)$, where i and j indicate the index of a grid. For simplicity, we merge the current small grids into new bigger grids, as shown in Figure 5(a). Then we define $c(x, y)$ as the value of each merged grid, where x, y denote the index of merged grid. The rule to compute $c(x, y)$ is as follows.

$$c(x,y) = \begin{cases} \sum_{(i,j)\in(x,y)} p(i,j), & \text{if } \exists\, p(i,j) \geq \text{threshold}, \\ -1, & \text{if all } p(i,j) < \text{threshold}. \end{cases} \tag{12}$$

The motivation behind this approach is twofold. One is to speed up the ILP. Because we can pre-determine some decision variables whose values are negative. The other is to keep the consistency of machine learning prediction.

In ILP for SRAF insertion, our real target is to maximize the total probability of valid grids with feasible SRAF insertion. Accordingly, it is manifest to put up with the objective function, which is to maximize the total value of merged grids. The ILP formulation is shown in Formula (13).

$$\max_{a(x,y)} \quad \sum_{x,y} c(x,y) \cdot a(x,y) \tag{13a}$$

$$\text{s.t.} \quad a(x,y) + a(x-1,y-1) \leq 1, \qquad \forall (x,y), \tag{13b}$$

$$a(x,y) + a(x-1,y+1) \leq 1, \qquad \forall (x,y), \tag{13c}$$

$$a(x,y) + a(x+1,y-1) \leq 1, \qquad \forall (x,y), \tag{13d}$$

$$a(x,y) + a(x+1,y+1) \leq 1, \qquad \forall (x,y), \tag{13e}$$

$$a(x,y) + a(x,y+1) + x(x,y+2) + a(x,y+3) \leq 3, \qquad \forall (x,y), \tag{13f}$$

$$a(x,y) + a(x+1,y) + x(x+2,y) + a(x+3,y) \leq 3, \qquad \forall (x,y), \tag{13g}$$

$$a(x,y) \in \{0,1\}, \qquad \forall (x,y). \tag{13h}$$

Here $a(x, y)$ refers to the insertion situation at the merged grid (x, y). According to the rectangular shape of an SRAF and the spacing rule, the situation of two adjacent SRAFs on the diagonal is forbidden by Constraints (13b) to (13e); e.g. Constraint (13b) requires the $a(x, y)$ and the left upper neighbor $a(x-1, y-1)$ cannot be 1 at the same time, otherwise which will lead to the violation against design rules. Constraints (13f) to (13g) restrict the maximum length of SRAFs. The Figure 5(b) actively illustrates these linear constraints coming from design rules.

5 EXPERIMENTAL RESULTS

We implement the framework using `python` on an 8-core 3.7GHz Intel platform. To verify the effectiveness and the efficiency of our SODL algorithm, we employ the same benchmark set as applied in [7], which consists of 8 dense layouts and 10 sparse layouts with contacts sized $70nm$. The spacing for dense and sparse layouts are set to $70nm$ and $\geq 70nm$ respectively.

Table 1 compares our results with a state-of-the-art machine learning based SRAF insertion tool [7]. Column "Benchmark" lists all the test layouts. Columns "F_1 score", "PV band", "EPE" and "CPU" are the evaluation metrics in terms of the learning model performance, the PV band area, the EPE, and the total runtime. Column "ISPD'16" denotes the experiment results by [7], while columns "SODL" and "SODL+ILP" correspond to the results of our supervised online dictionary learning framework without and with ILP model in post-processing. Note that in "SODL", a greedy SRAF generation approach as in [7] is utilized.

It can be seen from the table that the SODL algorithm outperforms [7] in terms of F_1 score by 5.5%. This indicates the predicted SRAFs by our model match the reference results better than [7]. In other words, the proposed SODL based feature revision can efficiently improve machine learning model generality.

We also feed the SRAFed layouts into Calibre [20] to go through a simulation flow that includes OPC and lithography simulation, which will generate printed contours under a given process window. The simulation results show that we get slightly better PV band and EPE results with around 20% less runtime overhead. In particular, after incorporating SRAF design rules with an ILP solution, SODL behaves even much better with an average PV band of $2.609\times10^{-3}\mu m^2$ and an average EPE of $0.774nm$ that surpass [7] with 2% less PV band and 3% less EPE.

We exemplify the trends of runtime and F_1 score with respect to the changing of number of atoms, which is depicted in Figure 6. With an increment in number of atoms, runtime ascends. Meanwhile, F_1 score goes down until number of atoms reaches a threshold. The reason is that the increase of feature dimensionality may generate over-fitting.

Table 1: Lithographic Performance Comparison with [7].

Benchmark	ISPD'16 [7]				SODL				SODL+ILP			
	F_1 score (%)	PV band ($.001\mu m^2$)	EPE (nm)	CPU (s)	F_1 score (%)	PV band ($.001\mu m^2$)	EPE (nm)	CPU (s)	F_1 score (%)	PV band ($.001\mu m^2$)	EPE (nm)	CPU (s)
Dense1	95.37	1.891	0.625	1.46	96.69	1.840	0.625	1.12	96.69	1.823	0.750	1.26
Dense2	94.77	1.960	0.313	1.35	97.00	2.033	0.500	1.06	97.00	2.003	0.438	1.20
Dense3	93.70	2.677	1.625	1.25	96.87	2.718	1.375	0.97	96.87	2.445	1.375	1.07
Dense4	93.89	2.426	1.313	1.47	96.49	2.288	1.125	1.13	96.49	2.459	1.625	1.29
Dense5	93.54	2.445	1.250	1.47	96.16	2.428	1.375	1.18	96.16	2.336	1.375	1.28
Dense6	93.02	2.933	0.750	1.17	96.86	2.871	1.000	0.91	96.86	2.886	0.250	1.00
Dense7	94.22	2.426	1.500	1.42	97.13	2.409	1.333	1.09	97.13	2.318	1.500	1.21
Dense8	93.24	2.354	1.417	1.40	96.85	2.436	1.417	1.10	96.85	2.366	1.167	1.20
Sparse1	90.51	2.937	0.438	2.61	93.62	2.866	0.563	2.04	93.62	2.803	0.375	2.18
Sparse2	87.65	2.870	0.625	7.02	94.03	2.872	0.516	5.25	94.03	2.873	0.594	5.63
Sparse3	85.75	2.882	0.556	14.07	91.68	2.911	0.535	10.74	91.68	2.829	0.528	11.50
Sparse4	85.56	2.896	0.566	23.81	93.35	2.891	0.496	18.44	93.35	2.830	0.547	19.66
Sparse5	85.69	2.889	0.565	28.96	90.48	2.931	0.571	23.18	90.48	2.850	0.580	24.80
Sparse6	84.65	2.875	0.558	41.87	91.57	2.852	0.630	32.43	91.57	2.787	0.572	34.29
Sparse7	85.00	2.881	0.540	56.95	93.01	2.921	0.611	44.87	93.01	2.841	0.575	47.58
Sparse8	84.05	2.899	0.564	74.56	92.72	2.860	0.573	59.70	92.72	2.835	0.560	63.08
Sparse9	84.71	2.885	0.586	94.93	90.21	2.940	0.549	75.34	90.21	2.833	0.568	79.58
Sparse10	84.03	2.884	0.599	106.33	92.60	2.915	0.512	82.90	92.60	2.836	0.560	88.00
Average	89.41	2.667	0.799	25.67	94.30	2.666	0.795	20.19	94.30	2.609	0.774	21.43
Ratio	1.000	1.000	1.000	1.000	**1.055**	0.999	0.994	**0.787**	**1.055**	**0.978**	**0.969**	0.835

6 CONCLUSION

In this paper, for the first time, we have introduced the concept of dictionary learning into the layout feature extraction stage and further proposed a supervised online algorithm constructing dictionary. This algorithm has been exploited into a machine learning-based SRAF insertion framework. To get a global view for SRAF generation, combined with design rules, an ILP has been built to generate SRAFs. The experimental results show that the F_1 score of machine learning model in SRAF insertion has been boosted and runtime overhead is also reduced compared with a state-of-the-art SRAF insertion tool. More importantly, the results of lithography simulations demonstrate the promising lithography performance in terms of PV band area and EPE. With the transistor size shrinking rapidly and the layouts becoming more and more complicated, we expect to apply our ideas into general VLSI layout feature learning and encoding.

ACKNOWLEDGEMENTS

This work is supported in part by The Research Grants Council of Hong Kong SAR (Project No. CUHK24209017).

REFERENCES

[1] D. Z. Pan, B. Yu, and J.-R. Gao, "Design for manufacturing with emerging nanolithography," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 32, no. 10, pp. 1453–1472, 2013.
[2] S. Shim, S. Choi, and Y. Shin, "Light interference map: A prescriptive optimization of lithography-friendly layout," *IEEE Transactions on Semiconductor Manufacturing (TSM)*, vol. 29, no. 1, pp. 44–49, 2016.
[3] S. Banerjee, Z. Li, and S. R. Nassif, "ICCAD-2013 CAD contest in mask optimization and benchmark suite," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2013, pp. 271–274.
[4] J.-R. Gao, X. Xu, B. Yu, and D. Z. Pan, "MOSAIC: Mask optimizing solution with process window aware inverse correction," in *ACM/IEEE Design Automation Conference (DAC)*, 2014, pp. 52:1–52:6.
[5] C. H. Wallace, P. A. Nyhus, and S. S. Sivakumar, "Sub-resolution assist features," Dec. 15 2009.
[6] J. Jun, M. Park, C. Park, H. Yang, D. Yim, M. Do, D. Lee, T. Kim, J. Choi, G. Luk-Pat *et al.*, "Layout optimization with assist features placement by model based rule tables for 2x node random contact," in *Proceedings of SPIE*, vol. 9427, 2015.
[7] X. Xu, T. Matsunawa, S. Nojima, C. Kodama, T. Kotani, and D. Z. Pan, "A machine learning based framework for sub-resolution assist feature generation," in *ACM International Symposium on Physical Design (ISPD)*, 2016, pp. 161–168.
[8] T. Matsunawa, B. Yu, and D. Z. Pan, "Optical proximity correction with hierarchical bayes model," in *Proceedings of SPIE*, vol. 9426, 2015.
[9] M. J. Gangeh, A. K. Farahat, A. Ghodsi, and M. S. Kamel, "Supervised dictionary learning and sparse representation-a review," *arXiv preprint arXiv:1502.05928*, 2015.
[10] Q. Zhang and B. Li, "Discriminative K-SVD for dictionary learning in face recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2010, pp. 2691–2698.
[11] M. J. Gangeh, A. Ghodsi, and M. S. Kamel, "Kernelized supervised dictionary learning," *IEEE Transactions on Signal Processing*, vol. 61, no. 19, pp. 4753–4767, 2013.
[12] Z. Jiang, Z. Lin, and L. S. Davis, "Learning a discriminative dictionary for sparse coding via label consistent K-SVD," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 1697–1704.
[13] Y. Yankelevsky and M. Elad, "Structure-aware classification using supervised dictionary learning," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 4421–4425.
[14] M. Aharon, M. Elad, and A. Bruckstein, "k-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.
[15] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *International Conference on Machine Learning (ICML)*, 2009, pp. 689–696.
[16] K. Skretting and K. Engan, "Recursive least squares dictionary learning algorithm," *IEEE Transactions on Signal Processing*, vol. 58, no. 4, pp. 2121–2130, 2010.
[17] R. Tibshirani, "Regression shrinkage and selection via the Lasso," *Journal of the Royal Statistical Society: Series B*, vol. 58, pp. 267–288, 1996.
[18] J. Friedman, T. Hastie, and R. Tibshirani, "Regularization paths for generalized linear models via coordinate descent," *Journal of Statistical Software*, vol. 33, no. 1, p. 1, 2010.
[19] D. P. Bertsekas, *Nonlinear programming*. Athena scientific Belmont, 1999.
[20] Mentor Graphics, "Calibre verification user's manual," 2008.