Bridging the Gap Between Layout Pattern Sampling and Hotspot Detection via Batch Active Learning

Haoyu Yang, Shuhe Li, Cyrus Tabery, Bingqing Lin, and Bei Yu, *Member, IEEE*

***Abstract*—Layout hotpot detection is one of the main steps in modern very-large-scale-integration (VLSI) chip design. A typical hotspot detection flow is extremely time consuming due to the computationally expensive mask optimization and lithographic simulation. Recent researches try to facilitate the procedure with a reduced flow, including feature extraction, training set generation, and hotspot detection, where feature extraction methods and hotspot detection engines are deeply studied. However, the performance of hotspot detectors relies highly on the quality of reference layout libraries which are costly to obtain and usually predetermined or randomly sampled in previous works. In this article, we propose an active learning-based layout pattern sampling and hotspot detection flow, which simultaneously optimizes the machine-learning model and the training set that aims to achieve similar or better hotspot detection performance with much smaller number of training instances. Experimental results show that our proposed method can significantly reduce lithography simulation overhead while attaining satisfactory detection accuracy on designs under both DUV and EUV lithography technologies.**

***Index Terms*—Active learning, hotspot detection, lithography, sampling.**

I. Introduction

ALONG with aggressive feature size scaling, even equipped with various resolution enhancement techniques and hierarchical design strategy, modern chip designs are more complicated and greatly challenged by manufacturability issues. In chip design, one of the most critical steps is to detect layout hotspots, which are potentially problematic patterns after chip manufacturing. A hotspot is costly to estimate because of the complicated mask optimization and lithography simulation [1]. Many researches have been conducted to facilitate the procedure which usually share a flow as shown in Fig. 1, including *feature extraction*, *training set generation*, and *hotspot detection*.

Feature extraction aims to convert layout geometry information (e.g., density [2], [3], frequency [4], [5], design rule [6], and geometry [7], [8]) into reduced mathematical representations, which are expected to improve hotspot detection accuracy. Recently, deep neural networks also exhibit powerful feature learning ability that can obtain layout representations without prior knowledge [5], [9]–[13]. In the hotspot detection stage, all selected samples and labels are fed into hotspot detection engines based on, mostly, pattern matching and machine learning. In a pattern matching flow, similar patterns within a specific radius are clustered together based on the constraints of translation, area, and/or edge displacements [3], [14], [15]. Lithography simulation will be performed on the representative clips results from which will be then labeled to the whole cluster. Above process indicates that fuzzy matching results are drastically affected by in-cluster variance. We will show the breakdown of the cluster distributions in advanced technology process that fuzzy matching possibly fails with an 95% area constraint. Although aggressive constraints can be introduced to ensure a low in-cluster variance, additional cluster count will significantly increase lithography simulation overhead. On the other hand, machine learning technologies tackle the problem through fitting layout representations into efficient machine learning models. Yu *et al.* [6] and Ding *et al.* [16] employed support vector machine for efficient hotspot detection. Matsunawa *et al.* [2] and Zhang *et al.* [17] enhanced hotspot detectors with boosting algorithms and additional learning strategies. References [5], [9]–[11] adopt emerging deep neural networks that automatically learn layout representations and perform classification. Still, overfitting is inevitable due to weakly distributed training data.

Previous works show that although pattern matching-based methods and machine learning-based methods exhibit different functionalities, they all rely highly on the quality of reference layout libraries. For example, in-cluster pattern variance directly affects pattern matching results and pattern diversity contributes to the generality of trained machine learning models. Layout pattern sampling problems are addressed by several works that are, to some extent, related to clustering approaches. Representative methods include clustering on frequency domain [4], [18], Bayesian clustering [19], and clustering based on layout topology [14], [15], [18], [20]. However, sampling and hotspot detection are mostly conducted exclusively which ignores the beneath integrity between them.

In this article, we will propose an active learning-based framework that can bridge the gap between the layout pattern sampling procedure and the hotspot detection problem. Active learning targets at machine learning problems with massive

Manuscript received February 18, 2019; revised June 28, 2019, October 29, 2019, February 19, 2020, and May 24, 2020; accepted July 16, 2020. Date of publication August 11, 2020; date of current version June 18, 2021. This work was supported in part by the Research Grants Council of Hong Kong SAR under Project CUHK24209017. This article was recommended by Associate Editor S. Held. *(Corresponding author: Bei Yu.)*

Haoyu Yang, Shuhe Li, and Bei Yu are with the Department of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong (e-mail: byu@cse.cuhk.edu.hk).

Cyrus Tabery is with ASML Brion Inc., Santa Clara, CA 95131 USA.

Bingqing Lin is with the College of Mathematics and Statistics, Shenzhen University, Shenzhen 518060, China.

Digital Object Identifier 10.1109/TCAD.2020.3015903

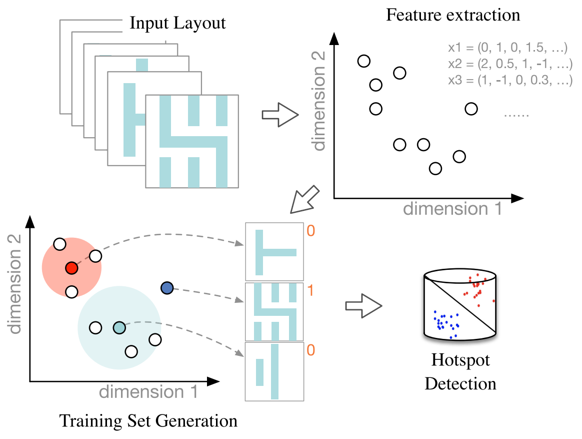


Fig. 1. Conventional process of layout hotspot training set and detection model generation.

data that is costly and time consuming to label. A major step of active learning is querying instances to determine whether the instance should be labeled and added into the training set from a perspective of machine learning model generality [21]. Representative querying strategies include uncertainty sampling (US [22]), query by committee (QBC [23]), and expected model change (EMC [24]). US aims to find the instances which the prediction model is most uncertain about and have the posterior probability around 0.5; QBC selects instances based on the disagreement among multiple classifiers; and EMC labels the most influential data in terms of the existing model. A common idea behind these strategies is labeling instances that are hardly distinguished by the classifier. However, there are several drawbacks of existing active learning strategies: 1) only one sample is selected in each iteration in most active learning flows which is lacking in efficiency; 2) machine learning models have to be retrained from raw state once the training set is updated; and 3) training set diversity is not considered in sampling flow which might cause serious overfitting problem [21], [25], [26]. Although Kullback–Leibler (KL) divergence [27] on posterior probabilities of unlabeled samples can be applied for diversity analysis [28], the effectiveness is limited on binary classification problems.

To address these concerns, we propose a batch mode active learning method that considers both model uncertainty and training set diversity for a better hotspot detection results. We embed the active learning engine into deep neural networks thus data sampling and incremental model training can be conducted alternatively. Guaranteed by the online property of the stochastic gradient descent, we only need to finetune the neuron weights according to new labeled instances instead of training model from scratch in each iteration. The rapid development of deep neural networks also makes it possible to learn representative features from raw image, which becomes another reason that we pick CNN as our learning engine. We take advantage of this characteristic and construct a diversity matrix of automatically learned features which will contribute as a partial criterion for data sampling in each iteration.

The main contributions of this article are listed as follows.

1) A novel layout pattern sampling and hotspot detection (PSHD) flow is proposed to simultaneously optimize training set and machine learning model.
2) We develop a batch mode active learning engine that samples multiple instances in each iteration according to the training set diversity and data uncertainty, where a specific distance metric is designed to guarantee a convex objective that makes the sampling procedure more efficient.
3) We conduct experiments on metal layers under 7 nm and 28 nm technology nodes which demonstrate the generality of the proposed flow that significantly increases hotspot detection accuracy while minimizing lithography simulation overhead.

The reminder of this article is organized as follows. Section II introduces the basic terminologies and definition. Section III lists theoretical and algorithmic details. Section IV presents experiment settings and results, followed by conclusion in Section V.

II. Preliminaries

A. Some Terminologies and Problem Formulation

This section introduces some terminologies and related problem formulation. Throughout this article, scalers are written as lowercase letters (e.g., x), vectors are bold lowercase letters (e.g., $\boldsymbol{x}$), and matrices are represented as bold uppercase letters (e.g., $\boldsymbol{X}$). Particularly, we use $J_n(\cdot)$ to represent the Bessel function of the first kind of order n. All layout images are with a resolution of 1 nm/pixel. The framework evaluation metrics are defined as follows.

Definition 1 (Hit): A hit is defined as when the detector reports hotspot on a clip of which at least one defect occurs at the core region. We also denote the ratio between number of hits and total hotpsot clips as detection accuracy.

Here the *detector* in this article refers to a hotspot detector with its input being a layout clip and its output being a label indicating whether the clip contains manufacturing issues.

Definition 2 (Extra): An extra is defined as when the detector reports hotspot on a clip of which no defect occurs at the core region.

Definition 3 (Litho-Clip): A litho-clip is a pattern in the training set or a pattern that is labeled hotspot by the machine learning model. The count of litho-clips reflects the lithography simulation overhead.

According to the evaluation metrics above, we define the problem of layout PSHD as follows.

Problem 1 (PSHD): Given a layout design, the objective of PSHD is sampling representative clips that will generalize the hotspot pattern space and maximize the machine learning model generality, i.e., maximizing the detection accuracy while minimizing the number of litho-clips.

From the definition, we can observe that our problem formulation differs from traditional hotspot detection in previous works. Here, we are dealing with a practical application, where no training set and testing set are given in advance. In short, we seek to conduct full-chip hotspot detection with smallest lithography simulation overhead. Accordingly, the input of

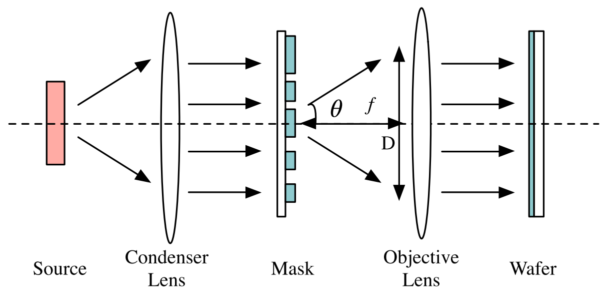


Fig. 2. Simple lithography imaging system.

our framework will be *full-chip layout* design and the outputs include a *training set* (labeled dataset), a *trained model*, and an *unlabeled dataset*. The evaluation metrics of such framework include overall hotspot detection accuracy

$$\mathrm{Acc} = \frac{\mathrm{HS}_{\mathrm{Train}} + \mathrm{Hits}}{\mathrm{HS}_{\mathrm{Total}}} \tag{1}$$

where $\mathrm{HS}_{\mathrm{Train}}$, Hits, and $\mathrm{HS}_{\mathrm{Total}}$ denote hotspot number in the training set, correctly predicted hotspot number in the unlabeled dataset, and total hotspot number, and the lithography overhead as in the following equation:

$$\mathrm{Litho} = Tr + \mathrm{FA} \tag{2}$$

where Tr represents the total number of clips in the training set and FA is the detection false alarm in the unlabeled dataset.

B. Lithography Proximity Effect

Most challenges in optical lithography come from the proximity effects caused by diffraction as light goes through the mask stage, as shown in Fig. 2. In a lithography imaging system, the electric field of the diffracted pattern is given by the Fourier transform of the original mask pattern. Afterward, diffracted patterns will be collected by the objective lens to project images on the wafer. Because of the limited size of objective lens, higher order diffraction patterns will be discarded when forming the image on the wafer that results in a lower pattern fidelity [29]. Typically, to ensure the mask image can be transferred onto the wafer as accurate as possible, at least the zero and $\pm$1st diffraction order should be captured by the objective lens. Accordingly, the smallest design pitch can be defined as in the following equation:

$$\frac{1}{p} = \frac{\mathrm{NA}}{\lambda} \tag{3}$$

where p denotes the design pitch, λ is the wavelength of the light source, and NA is the numerical aperture of the objective lens which determines how much information can be collected by the objective lens and is given by

$$\mathrm{NA} = n \sin\theta_{\max} = \frac{D}{2f} \tag{4}$$

where n is the index of refraction of the medium, $\theta_{\max}$ is the largest half-angle of the diffraction light that can be collected by the objective lens, D denotes the diameter of physical aperture seen in front of the objective lens, and f represents the focal length [30].

The existence of diffraction makes it also interesting to analyze the minimum distance when two shapes stop affecting the aerial images of each other. Fraunhofer diffraction occurs in classic lithography system, where the diffracted patten is determined by the Fraunhofer diffraction integral [30]. For simplicity, we consider the contact hole as an example whose diffraction pattern resembles the Airy disk. The light intensity in terms of observation angle θ at the entrance of the objective lens is shown in the following equation:

$$I(\theta) = \left(\frac{2J_1(kr\sin\theta)}{kr\sin\theta}\right)^2 \cdot I_0 \tag{5}$$

where I_0 denotes the center intensity of airy disk, $r = (D/2)$ is the radius of the entrance pupil, and $k = (2\pi/\lambda)$ is the wavenumber. According to the properties of the Bessel function, dark regions of Airy disk that correspond to zeros of $I(\theta)$ appear periodically with a degradation of total energy. The total energy within an observation angle can be derived by integrating (5) over θ

$$P(\theta) = \left[1 - J_0^2(kr\sin\theta) - J_1^2(kr\sin\theta)\right] \tag{6}$$

which reflects by how much the diffraction information can be collected. If we pick the 6th zero point of $I(\theta)$ at $kr\sin\theta \approx 19$, we can derive $P(\theta) \approx 96.73\%$, which is the fraction of diffraction collected with in a given window size. Besides, we assume $n = 1$ in the air

$$\sin\theta = \frac{19}{kr} = \mathrm{NA}. \tag{7}$$

Combining (7) and (4)

$$D = 6.05 \cdot \frac{\lambda}{\mathrm{NA}}. \tag{8}$$

Here, D determines the minimum distance when two shapes can be regarded as isolated patterns, which can be derived to be 230 nm using $\mathrm{NA} = 0.35$ and $\lambda = 13.5$ nm under extreme ultraviolet (EUV) lithography technologies.

III. Algorithm

In this section, we will discuss the details of our PSHD flow, including initial training set selection, batch active sampling algorithm, and some mathematical analysis.

Because it is extremely costly to label layout clips, our flow aims to sample as small number of clips as possible while ensuring good machine learning model generality. Conventional layout pattern sampling methods conduct clustering on layout clips and obtain representative patterns based on the results of pattern matching or clustering. Although the clustering can effectively reduce the sample number, it does not consider the behavior or requirement of, especially, machine learning-based hotspot detectors. As shown in Fig. 3(a), pattern matching collects a lot of less critical patterns that lie far from the decision boundary while ignoring important patterns. In conventional active learning-based sampling [see Fig. 3(b)], prediction uncertainty of each clip is included in the selection criteria. That is, patterns with posterior probability around 0.5 will be sampled with higher priority. However, in a layout PSHD task, we care more about

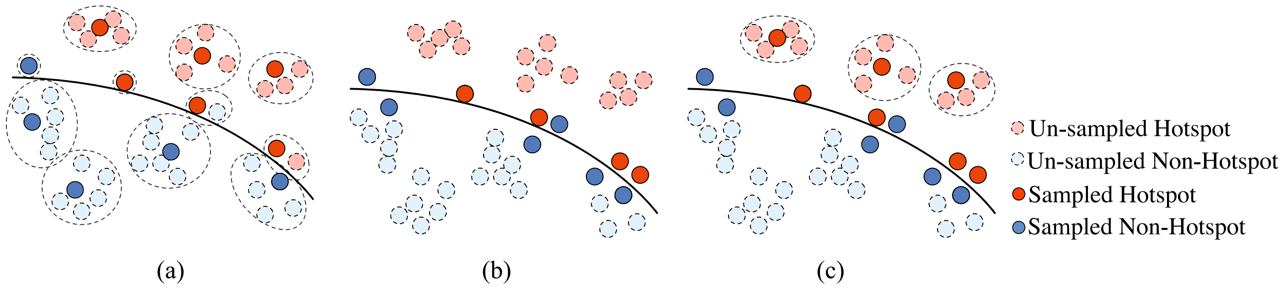


Fig. 3. Visualization of different layout pattern sampling methods. (a) Pattern matching. (b) Conventional active learning. (c) Proposed layout pattern sampling.

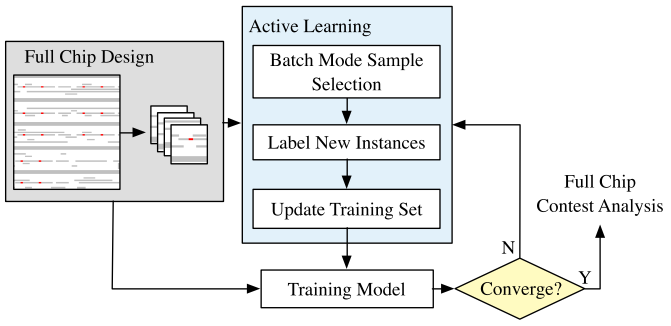


Fig. 4. PSHD flow.

hotspot regions. Therefore, apart from considering diversity of training instances, we tend to select clips with higher probability being hotspot in our sampling approach, as illustrated in Fig. 3(c).

A. Flow Summary

The proposed layout PSHD flow is illustrated in Fig. 4. To analyze the printability of a full-chip design, we dispatch the layout into clips based on the lithography proximity effect analysis, such that the whole chip is covered by the core region of each clip that contains enough information to conduct printability estimation. And then, the training set and the machine learning model will be updated until convergence, when all the clips will be either labeled or dropped. Finally, the full-chip hotspot detection will be conducted on the dropped clips with the final learning model. We go through the framework details in the rest of this section.

B. Diversity-Aware Batch Sampling

Discriminative machine learning models are usually designed to find the optimal hyperplane that separates the whole data space. The quality of a model is measured by its generative loss which is associated with the prediction error on the future instances. In this section, we will discuss an instance selection policy considering both *model uncertainty* and *data diversity*, thus the selected instances are expected to contribute most on the trained model generality. The uncertainty describes how confident of the classifier when recognizing new instances. A model is uncertain on a given instance if the prediction probability draws around 0.5 according to the posterior distribution or the instance is too close to the hypothesis plane in the feature space. Data diversity corresponds to the instance distribution in the dataset. The underneath idea is to label instances into the training set such that the training set entropy is maximized. Most active learning algorithms, such as US [22], QBC [23], and EMC [24], are designed to pick one instance in each iteration, which is not efficient as problem sizes grow. Even these methods are applied for batch selection, samples touch the selection criteria are labeled into the training set, when redundant instance samples are more likely to be chosen. Here, we consider a *batch selection* mechanism that takes both model uncertainty and data diversity into account.

Suppose we have a training set $\mathcal{L}_t$ and an unlabeled set $\mathcal{U}_t$ at time t. Let $\boldsymbol{w}_t$ be the classifier parameters trained on $\mathcal{L}_t$. The objective is to select a batch $\mathcal{B}$ with k points from $\mathcal{U}_t$ so that the future learner $\boldsymbol{w}_{k+1}$, trained on $\mathcal{L}_t \cup \mathcal{B}$, has maximum generalization capability. Let $\mathcal{Y} = \{0, 1\}$ be the set of possible classes in the problem. For a given unlabeled layout clip $\boldsymbol{x}_i$, we denote the related posterior probability as $p(y|\boldsymbol{x}_i; \boldsymbol{w}_t)$. Usually, the uncertainty of the unlabeled instance $\boldsymbol{x}_i$ is defined as the entropy of the predicted probabilities, as shown in the following equation:

$$c(i) = -\sum_{j \in \mathcal{Y}} p(y = j|\boldsymbol{x}_i; \boldsymbol{w}_t) \log p(y = j|\boldsymbol{x}_i; \boldsymbol{w}_t). \tag{9}$$

However, in the layout pattern sampling problems, problematic instances are of more interests. We therefore pick a simple but more practical representation of $c(i)$

$$c(i) = p(y = 1|\boldsymbol{x}_i; \boldsymbol{w}_t) \tag{10}$$

which corresponds to the probability of a given instance being hotspot. Usually, the redundancy between unlabeled points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ can be calculated through the KL divergence, which measures how two training instances differ from each other in a statistic point of view. In the domain of layout hotspot detection, however, we are dealing with yes or no problem, which is less informative for diversity analysis. To benefit the layout analysis problem, we use inner-product of two instances in the normalized feature space as shown in the following equation:

$$E(i, j) = \boldsymbol{x}_i^\top \boldsymbol{x}_j. \tag{11}$$

We can further formulate the diversity matrix $\boldsymbol{D} \in \mathbb{R}^{n\times n}$, whose entries are defined by (11).

Given the matrix $\boldsymbol{D}$, the batch mode active learning problem is shown in mathematical formulation (12), where the objective is to select a batch of points with high aggregate uncertainty scores and high divergences among the samples

$$\min_{\boldsymbol{m}} \ \boldsymbol{m}^{\top}\boldsymbol{D}\boldsymbol{m} \tag{12a}$$
$$\text{s.t.} \ m_i \in \{0, 1\} \quad \forall i \tag{12b}$$
$$\sum_i m_i = k \quad \forall i. \tag{12c}$$

Here, k is the number of patterns that will be selected into the training set, m_i is a binary variable, and $m_i = 1$ if pattern $\boldsymbol{x}_i$ is selected in the batch $\mathcal{B}$. It should be noted that (12) is binary quadratic programming, which is NP-hard. We relax the integer constrains and derive the following problem:

$$\min_{\boldsymbol{m}} \ \boldsymbol{m}^{\top}\boldsymbol{D}\boldsymbol{m} \tag{13a}$$
$$\text{s.t.} \ m_i \in [0, 1] \quad \forall i \tag{13b}$$
$$\sum_i m_i = k \quad \forall i \tag{13c}$$

which is a standard quadratic programming problem and can be solved efficiently. It can be seen here one advantage of the proposed distance metric over the KL divergence and the Euclidean distance is that (11) ensures the objectives of (12) and (13) to be convex by $\boldsymbol{D} \succeq 0$. Finally, the integer solution can be recovered by picking k largest entries in $\boldsymbol{m}$.

C. Layout Pattern Sampling and Hotspot Detection

The rapid development of deep neural networks makes it possible to learn representative features from raw image and complete effective classification jobs. Therefore, in this project, we pick up a well-designed shallow convolutional neural networks from [5] as the preferred machine learning model which will be embedded into the active learning flow. In particular, features obtained from the fully connected layers are fed into (11) to calculate the divergence matrix. Most neural networks are trained with mini-batch gradient descent (MGD), where a random small batch of training samples are fed into the neural networks to update neuron weights. The online property of MGD makes it easier to update the model on new instances without retraining the model from scratch compared to the traditional support vector machine or logistic regression. It should be noted that the proposed classification-driven active learning flow is very general that it can be plugged into any incremental hotspot detectors.

In most cases $\boldsymbol{D}$ will be extremely large, especially for EUV specific layers, which makes (13) hard to solve. We therefore stochastically sample a subset $\hat{\mathcal{U}}_t \subseteq \mathcal{U}_t$ before entering the quadratic programming phase to further reduce the computational cost. Finally, the neural network can be accordingly updated as $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \alpha[\partial l/(\partial \boldsymbol{w}_t)]$. Here, α denotes the updating rate and l is the average cross-entropy loss of sampled instances, defined as follows:

$$l = \frac{1}{k}\sum_{i=1}^{k} \log p(y_i = 1|\boldsymbol{x}_i; \boldsymbol{w}_t). \tag{14}$$

It should be noted that although the neural networks may need multiple iterations to finish training, the computational cost is much less than training from a raw model.

Algorithm 1 Batch Active Sampling

```
Require: $\mathcal{L}_0, \mathcal{U}_0, n, \sigma$.
Ensure: $\boldsymbol{w}, \mathcal{D}$.
1: Initialize $\boldsymbol{w} \sim \mathcal{N}(0, \sigma)$;
2: $\mathcal{L} \leftarrow \mathcal{L}_0, \mathcal{U} \leftarrow \mathcal{U}_0, \mathcal{D} \leftarrow \emptyset$;
3: $\boldsymbol{w} \leftarrow$Train the machine learning model based on $\mathcal{L}$.
4: while $\mathcal{U} \neq \emptyset$ do
5:     $\hat{\mathcal{U}} \leftarrow$ Sample $n$ instances with highest probability (predicted with current $\boldsymbol{w}$) being hotspot from $\mathcal{U}$;
6:     $\mathcal{U} \leftarrow \mathcal{U} \backslash \hat{\mathcal{U}}$;
7:     $\mathcal{B} \leftarrow$Select $k$ instances by solving Problem (13);
8:     $\mathcal{L} \leftarrow \mathcal{L} \cup \mathcal{B}$;
9:     $\mathcal{D} \leftarrow \hat{\mathcal{U}} \backslash \mathcal{B} \cup \mathcal{D}$;
10:    $\boldsymbol{w} \leftarrow$Update machine learning model based on $\mathcal{L}$;
11: end while
12: return $\boldsymbol{w}, \mathcal{D}$.
```

Reference [5] has shown that biased label is able to provide better tradeoffs on hotspot detection problem during the fine-tune procedure. However, by our observation, stepped bias significantly disturbs the pretrained model. We therefore improves this technique by letting the bias change linearly along with the training step.

Algorithm 1 presents the details of the layout pattern sampling flow. The algorithm requires an initial training set $\mathcal{L} = \mathcal{L}_0$ with labeled patterns, an unlabeled pattern pool $\mathcal{U} = \mathcal{U}_0$, number of patterns to be queried n, and a standard deviation σ used to initialize the machine learning models (lines 1 and 2); we first train an initial machine learning model based on $\mathcal{L}_0$ (line 3). In each sampling iteration, we fetch n instances from $\mathcal{U}$ without replacement and form a query set $\hat{\mathcal{U}}$ (lines 5 and 6); k instances are sampled into a set $\mathcal{B}$ by solving Problem (13) (line 7); then k instances will be added up to the training set and rest $n - k$ instances will be dropped and accordingly, new training set $\mathcal{L}$, discarded set $\mathcal{D}$, and the machine learning model are updated (lines 7–9). The algorithm ends when the unlabeled instance pool is empty and returns the trained model and remaining unlabeled patterns to be verified by the machine learning model.

D. Initial Training Set Generation

Note that Algorithm 1 requires an initial labeled dataset $\mathcal{L}_0$ to obtain a pretrained model that will be used to extract features for future layout patterns. Thus, $\mathcal{L}_0$ is critical on the performance of the whole flow. Because it is almost impossible to know which pattern is more likely to have defects at beginning and hotspots are fetal but rare in layout spaces, we select the initial training set through analyzing the distribution of the unlabeled dataset, assuming that hotspots occur with lowest posterior probabilities.

Details of initial training set generation can be found in Algorithm 2 that takes the unlabeled dataset $\mathcal{U}$, the number of principle components n_c, and the number of initial sampled instances n_i as inputs. We first convert layout patterns into frequency domain via feature tensor extraction [5] (line 1)

Algorithm 2 Initial Sampling

```
Require: $\mathcal{U}, n_c, n, n_i$.
Ensure: $\mathcal{L}_0, \mathcal{U}_0$.
1: $\mathcal{D} \leftarrow$FeatureTensorExtraction($\mathcal{U}$);
2: $\mathcal{F} \leftarrow$PCA($\mathcal{D}, n_c$);
3: $\boldsymbol{\mu} \leftarrow \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{f_i}, \Sigma \leftarrow$ cov($\mathcal{F}$);
4: for $f_i \in \mathcal{F}, i = 1, 2, ..., n$ do
5:     $p_i = P(f_i; \boldsymbol{\mu}, \Sigma)$;
6: end for
7: $\mathcal{I}_0 \leftarrow$Get the indices of $n_i$ patterns with smallest posterior probabilities;
8: $\mathcal{L}_0 \leftarrow \mathcal{U}_{\mathcal{I}_0}, \mathcal{U}_0 \leftarrow \mathcal{U}\backslash\mathcal{L}_0$;
9: return $\mathcal{L}_0, \mathcal{U}_0$.
```

followed by one step principle component analysis for further dimensionality reduction (line 2). A general Gaussian model is established by calculating the mean and the covariance (line 3). We then calculate the posterior probabilities of the unlabeled dataset based on the estimated mean and covariance (lines 4–6). Finally, we sample n_i instances with smallest posterior probabilities to form the initial training set (lines 7–9).

E. Algorithm Analysis

In this section, we will discuss and analyze some technique details of our proposed framework. As described in the previous section, we relax the integer constraints when solving the sampling problem (12). Because each queried instance will be sampled or dropped by solving the problem in (13), the entries of the optimal solution will be rounded into binary values. Here, we will analyze the loss of optimality of problem (13) when reconstructing an integer solution as the sampling choice, as claimed in Theorem 1.

Theorem 1: Let $\boldsymbol{m}$ be the optimal solution of problem (13) that is binarized into $\boldsymbol{m}_b$ by setting k largest entries to 1 and rest $n-k$ entries to 0, then

$$f(\boldsymbol{m}) \leq f(\boldsymbol{m}_b) \leq 2f(\boldsymbol{m}) + 2\lambda_n\left(k - \frac{k^2}{n}\right) \tag{15}$$

where $f(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{D}\boldsymbol{x}$, n is total number of instances in each query iteration, k is the number of instances that will be sampled into training set, and $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ are the eigenvalues of $\boldsymbol{D}$.

Proof: $f(\boldsymbol{m}) \leq f(\boldsymbol{m}_b)$ is trivial, and we will show that $f(\boldsymbol{m}_b) \leq 2f(\boldsymbol{m}) + 2\lambda_n(k - [k^2/n])$. According to (11), the distance matrix $\boldsymbol{D}$ can be written as $\boldsymbol{D} = \boldsymbol{F}^\top\boldsymbol{F}$, where each column of $\boldsymbol{F}$ is the feature vector of each queried instance, thus $\boldsymbol{D} \succeq \boldsymbol{0}$ and f is convex. By definition

$$\frac{1}{2}f(\boldsymbol{m}) + \frac{1}{2}f(\boldsymbol{m}_b - \boldsymbol{m}) \geq f\left(\frac{1}{2}\boldsymbol{m}_b\right) \tag{16}$$

i.e.,

$$\frac{1}{2}\boldsymbol{m}_b^\top\boldsymbol{D}\boldsymbol{m}_b - \boldsymbol{m}^\top\boldsymbol{D}\boldsymbol{m} \leq (\boldsymbol{m}_b - \boldsymbol{m})^\top\boldsymbol{D}(\boldsymbol{m}_b - \boldsymbol{m}). \tag{17}$$

By the Rayleigh–Ritz theorem [31]

$$(\boldsymbol{m}_b - \boldsymbol{m})^\top\boldsymbol{D}(\boldsymbol{m}_b - \boldsymbol{m}) \leq \lambda_n\|\boldsymbol{m}_b - \boldsymbol{m}\|_2^2. \tag{18}$$

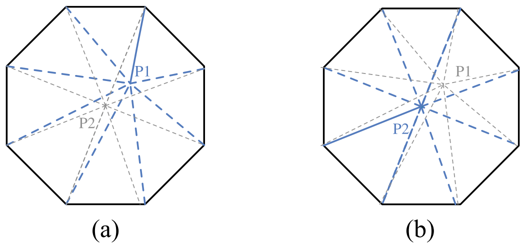


Fig. 5. Geometric view of $\boldsymbol{m}_b - \boldsymbol{m}$. P_1 and P_2 denote the end points of $\boldsymbol{m}$, and in particular, P_2 lies in the center of the base polygon. The solid segments in each figure represents $\|\boldsymbol{m}_b - \boldsymbol{m}\|_2^2$ for a given $\boldsymbol{m}$.

Claim that

$$\begin{aligned}\|\boldsymbol{m}_b - \boldsymbol{m}\|_2^2 &\leq \max_{\boldsymbol{y}\in\mathcal{T}}\|\boldsymbol{m}_b - \boldsymbol{y}\|_2^2 \\ &= \max_{\boldsymbol{y}\in\mathcal{T}} \min_{\boldsymbol{x}\in\mathcal{T}_b, \boldsymbol{y}\in\mathcal{T}} \|\boldsymbol{x} - \boldsymbol{y}\|_2^2\end{aligned} \tag{19}$$

where $\mathcal{T} = \{\boldsymbol{x} \in \mathbb{R}^n | \sum_1^n x_i = k, x_i \in [0, 1] \quad \forall i\}$ and $\mathcal{T}_b = \{\boldsymbol{x} \in \mathbb{R}^n | \sum_1^n x_i = k, x_i \in \{0, 1\} \quad \forall i\}$. Without loss of generality, we assume all the entries of a given $\boldsymbol{y}$ are placed in an order

$$1 \geq y_{\delta_1} \geq y_{\delta_2} \geq \cdots \geq y_{\delta_n} \geq 0 \tag{20}$$

thus according to the rounding strategy, $\boldsymbol{m}_b$ is defined as follows:

$$m_{b,i} = \begin{cases} 1 & \forall i \in \{\delta_1, \delta_2, \ldots, \delta_k\} \\ 0 & \text{otherwise.} \end{cases} \tag{21}$$

Then as claimed in (19):

$$\begin{aligned}\|\boldsymbol{m}_b - \boldsymbol{y}\|_2^2 &= \sum_{i=1}^{k}\left(1 - y_{\delta_i}\right)^2 + \sum_{i=k+1}^{n} y_{\delta_i}^2 \\ &= k + \sum_{i=1}^{n} y_{\delta_i}^2 - 2\sum_{i=1}^{k} y_{\delta_i} \\ &\leq k + \sum_{i=1}^{n} y_{\delta_i}^2 - 2\sum_{i=1}^{k} y_{\eta_i} \\ &= \|\boldsymbol{x} - \boldsymbol{y}\|_2^2 \quad \forall \boldsymbol{x} \in \mathcal{T}_b, \boldsymbol{y} \in \mathcal{T}.\end{aligned} \tag{22}$$

Consider a right pyramids with a regular base, which has its apex at the origin, C_n^k edges defined by the vectors defined in $\mathcal{T}_b$ and a regular polygon base A lies in the hyperplane $\sum_{i=1}^n x_i = k$. As shown in Fig. 5, $\|\boldsymbol{m}_b - \boldsymbol{m}\|_2^2$ reaches its maximum value when $\boldsymbol{m}$ lies in the center of the pyramid base.

That is, $\boldsymbol{m} = [(k/n),)(k/n), \ldots, (k/n)]^\top$, and

$$\begin{aligned}\max_{\boldsymbol{m}_b, \boldsymbol{m}} \|\boldsymbol{m}_b - \boldsymbol{m}\|_2^2 &= k\left(1 - \frac{k}{n}\right)^2 + (n-k)\left(\frac{k}{n}\right)^2 \\ &= k - \frac{k^2}{n}\end{aligned} \tag{23}$$

which, combined with (18), justifies the theorem. ■

Theorem 1 provides a theoretical guidance on choosing proper n and k in the batch sampling procedure, which can also be intuitively explained by the fact that if we sample all or one instances in each querying iteration, we have no risk on the integer relaxation error, however, at the cost of diversity loss.

TABLE I
BENCHMARK DETAILS

Benchmarks	CD (nm)	HS #	NHS #	Tech (nm)
ICCAD12-1	45	325	5019	32
ICCAD12-2	45	672	46583	28
ICCAD12-3	45	2717	50976	28
ICCAD12-4	45	272	36342	28
ICCAD12-5	45	67	22043	28
ICCAD12-28	45	3728	155944	28
ICCAD19	-	65778	163680	28/32
ICCAD16-1	16	0	63	7
ICCAD16-2	16	56	967	7
ICCAD16-3	16	1100	3916	7
ICCAD16-4	16	157	1678	7

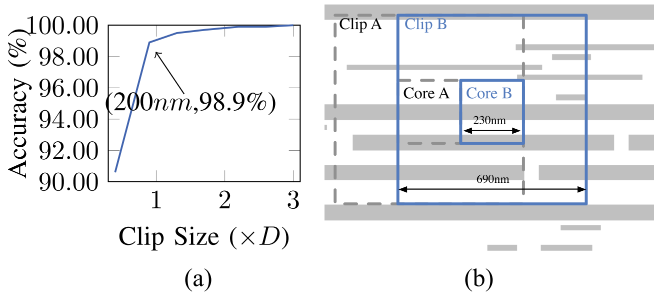


Fig. 6. Dispatching layouts based on estimated D. Because there is no spacing and overlapping between adjacent core regions of adjacent clips, each layout is fully scanned in the sampling and detection flow. Particularly, exact matching has a detection rate of 98.9% with the clip size in the original contest setting. (a) Influence of clip size. (b) Clip-based scan.

IV. EXPERIMENTAL RESULTS

A. Dataset and Configurations

Our layout PSHD flow is tested on two industrial benchmark sets: ICCAD12 [32] and ICCAD16 [33].

Table I lists the benchmark details. To verify the efficiency of our proposed method on EUV-oriented designs, we shrink ICCAD16 layouts to reach a CD under 7-nm technology node as indicated in the column "CD (nm)". Columns "HS #" and "NHS #" are numbers of hotspot and nonhotspot clips in each benchmark and "Tech (nm)" is the technology nodes of each design. ICCAD12 contains five benchmark sets from [32] with labels which can be directly input to our flow. To verify the scalability of our algorithm, we also merge all 28-nm designs from ICCAD12 into a much larger dataset ICCAD12-28 with 3728 hotspot patterns and 0.15M nonhotspot patterns. Very recently, Reddy *et al.* [34] found that the designs from ICCAD12 may lack truly never seen before and hard to classify patterns, and hence the dataset is less effective as an evaluation of the machine learning models. Derived from designs in ICCAD12, they synthesized a new challenging benchmark suit that contains both 28 nm and 32 nm designs with ~40% hotspot patterns. Statistics of the new benchmark suit are shown as ICCAD19 in Table I. ICCAD16 contains four layouts that are original designed for fuzzy matching tasks. To locate defects in those layouts, we apply ASML Tachyon optical proximity correction (OPC) and layout manufacturability checker (LMC) tools on scaled layouts using EUV lithography models for 7-nm metal layer. In the LMC stage, we only consider three types of defects that are edge placement error, bridge, and neck which contribute most to circuit failures. Then all the locations where edge placement error, bridging, and necking occur are marked as defects. To perform efficient and parallel testing, clip-based scan is usually applied in classic hotspot detection flow, where the clip size and scanning stride are empirically determined according to the optical diameter ($D = 230$ nm) under given lithography specifications. Fig. 6(a) shows that fail detected hotspot count of exact pattern matching reduces to zero as clip size increases to around $3 \times D = 690$ nm that will be chosen as the clip size in our experiment. It should also be noted that original ICCAD16 contains predefined marker layer covering a fraction of the layouts. To conduct full-chip detection, we manually replaced the original marker layer with a new layer that contains uniformly distributed markers that cover whole layouts.

According to the estimated D of 7-nm EUV lithography system, we adopt an overlapped dispatching method that covers the whole layout with reasonably small clip size that contains enough information to determine whether the center core region is hotspot or not. Fig. 6(b) illustrates the details of the dispatching procedure. We use a 690 nm $\times$ 690 nm sliding window to scan the whole layout with scanning stride being $(1/3)$ of the clip size, which ensures that center 230×230 core regions of each clips are exactly covering the whole chip. Note that to ensure that clips contain more than 96% information to estimate the printability of their core region, the smallest distance from the core boundary to the clip boundary is intentionally selected as 230 nm. Furthermore, each clip will be marked as hotspot clip if defects occur at its core region as shown in Fig. 6(b). Statistics of ICCAD16 benchmarks are also listed in columns "HS #" and "NHS #." We can notice that the smallest layout ICCAD16-1 is defect-free, therefore the case ICCAD16-1 is ignored in the following experiments. For the rest of ICCAD16 cases, ICCAD16-2 has 56 hotspots out of 1023 clips, ICCAD16-3 has 1100 hotspots out of 5016 clips, and ICCAD16-4 has 157 hotspots out of 1835 clips. It should be noted that although layout ICCAD16-4 is much larger than other cases. However, it is much more regular and a large fraction of the patterns violate EUV direct print metal layer design. We therefore only extract clips from DRC-clean regions and that is why the total clip count is less than ICCAD16-3. We can also see the out of expected behaviors on ICCAD16-4 in the experiments in the following sections.

To accommodate the shallow neural networks and the computational requirements, we conduct feature tensor extraction [5] on each clips in all benchmark cases that convert layout images into reduced frequency domain.

B. Effectiveness of Batch Active Sampling

In the first experiment, we will compare the batch active sampling method with fuzzy matching under different area constraints. The procedures of Algorithm 1 on four benchmark sets are depicted in Fig. 7, where the x-axis represents the total number of patterns sampled into training set and the

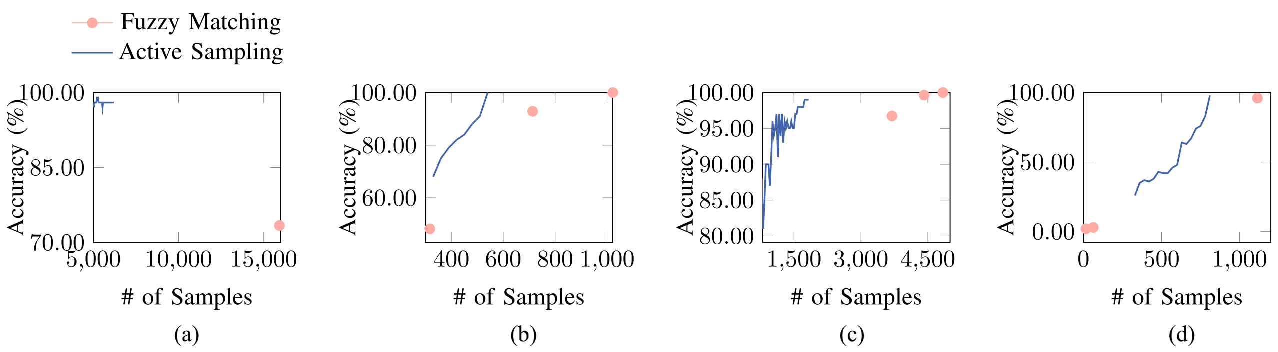


Fig. 7. Learning model performance versus sampling count. The blue curve is the reference performance obtained from fuzzy matching with different area constrains reflected as different sampling count. The red curve shows the sampling results based on Algorithm 1. (a) `ICCAD12-28`. (b) `ICCAD16-2`. (c) `ICCAD16-3`. (d) `ICCAD16-4`.

TABLE II
FULL-CHIP PSHD ON ICCAD12 BENCHMARKS

Benchmarks	PM-exact [15]			PM-a95^{12} [15]			PM-a90^{12} [15]			PM-e2^{2} [15]			FT [35]			Greedy [17]			Ours		
	Acc	Litho		Acc	Litho		Acc	Litho		Acc	Litho		Acc	Litho		Acc	Litho		Acc	Litho	
	(%)	#	(%)	(%)	#	(%)	(%)	#	(%)	(%)	#	(%)	(%)	#	(%)	(%)	#	(%)	(%)	#	(%)
`ICCAD12-1`	100.0	3839	71.84	93.17	732	13.70	81.66	398	7.45	100.0	3828	71.63	68.31	1226	22.94	99.08	1140	21.33	99.70	1436	26.87
`ICCAD12-2`	100.0	31812	67.32	99.54	13091	27.70	93.84	5720	12.10	100.0	32601	68.99	5.95	3024	6.40	2.36	1036	2.19	97.42	2648	5.60
`ICCAD12-3`	100.0	31513	58.69	98.54	20904	38.93	91.79	9335	17.39	100.0	30769	57.31	93.95	9244	15.50	96.22	10579	19.70	96.55	4484	8.35
`ICCAD12-4`	100.0	29438	80.40	90.46	4527	12.36	62.00	1411	3.85	100.0	28067	76.66	8.82	5511	15.05	82.35	2723	7.44	95.64	6842	18.69
`ICCAD12-5`	100.0	14988	67.79	91.20	3045	13.77	84.91	1204	5.45	100.0	14849	67.16	8.15	1651	7.47	91.30	1486	6.72	94.12	1227	5.55
`ICCAD12-28`	100.0	127746	80.01	96.83	38879	24.35	73.38	15923	9.97	100.0	124320	77.86	32.14	20000	12.53	24.57	26945	16.88	96.99	8210	5.14
`ICCAD19`	99.93	157238	68.53	-	-		-	-		-	-		63.66	65168	28.40	82.01	86917	37.88	96.30	69743	30.39
Average	99.99	56653	70.65	-	-		-	-		-	-		29.38	14986	15.47	68.27	18689	16.02	96.67	13512	14.37
Ratio	1.03	4.19	4.92	-	-		-	-		-	-		0.30	1.11	1.08	0.77	1.02	1.12	1.00	1.00	1.00

1 Experiments on all `ICCAD12` cases are conducted on the center 600 × 600 region of each clip because the area constrained fuzzy matching cannot be finished within one week using original clip size of 1200 × 1200.
2 Experiments on `ICCAD19` case cannot finish with given time limit even on the center 600 × 600 region of each clip.

TABLE III
FULL-CHIP PSHD ON ICCAD16 BENCHMARKS

Benchmarks	PM-exact [15]			PM-a95 [15]			PM-a90 [15]			PM-e2 [15]			FT [35]			Greedy [17]			Ours		
	Acc	Litho		Acc	Litho		Acc	Litho		Acc	Litho		Acc	Litho		Acc	Litho		Acc	Litho	
	(%)	#	(%)	(%)	#	(%)	(%)	#	(%)	(%)	#	(%)	(%)	#	(%)	(%)	#	(%)	(%)	#	(%)
`ICCAD16-2`	100.0	1022	99.90	92.86	717	69.70	48.21	328	30.99	100.0	1022	99.90	91.07	782	76.25	51.79	475	46.43	100.0	881	86.12
`ICCAD16-3`	100.0	4838	96.45	99.64	4420	88.04	96.73	3717	73.70	99.91	4777	95.22	86.18	1854	32.70	73.82	2496	49.76	99.74	3589	71.55
`ICCAD16-4`	95.54	1134	60.71	2.55	65	3.43	1.91	20	0.82	78.34	842	43.54	50.32	573	28.88	50.32	668	36.40	98.09	1608	87.63
Average	98.51	2325	85.69	65.01	1730	53.72	48.95	1343	35.17	92.75	2199	79.55	77.13	983	45.94	58.64	1213	44.20	99.27	2026	81.77
Ratio	0.992	1.15	1.05	0.655	0.85	0.66	0.493	0.66	0.43	0.934	1.09	0.97	0.78	0.49	0.56	0.59	0.60	0.54	1.00	1.00	1.00

y-axis denotes the detection accuracy. According to the analysis, we avoid choosing the (k/n) that results in big rounding error (i.e., 0.5). Considering that sample count also affects the training performance and the lithography simulation overhead, we pick $k = 30$, $n = 90$ in all benchmarks. On the other hand, because `ICCAD12/19` are much larger benchmark sets that contain more than 150 000 clips, early stopping is applied in the batch active sampling procedure, where we pick the maximum sampling number to be 5% of total instance number in each dataset.

The discrete dots in Fig. 7 correspond to fuzzy matching results with area constraints 90%, 95%, and 100%, respectively. It can be seen that our batch sampling converges at a reasonably high detection accuracy on both DUV and EUV specific layers while requiring much less training instances than exact pattern matching. In other words, our proposed method can significantly reduce lithography simulation overhead. Particularly, for the case `ICCAD12`, exact matching samples more than 10^5 clips among the whole dataset, while our method achieves similar results with only 6200 clips. Because our framework adopts CNN as the learning engine, uncertainty behavior will be introduced by weight initialization and batching sampling. However, with the help of good weight initialization, reasonable sampling ratio in diversity-aware sampling (see Section III-E), and dynamic learning rate, we are able to attain a ±5 hits variation cross multiple runs.

TABLE IV
RESULT COMPARISON WITH CONVENTIONAL ACTIVE LEARNING SOLUTIONS ON ICCAD12 BENCHMARKS

Benchmarks	US [22]			EMC [24]			Ours		
	Acc	Litho		Acc	Litho		Acc	Litho	
	(%)	#	(%)	(%)	#	(%)	(%)	#	(%)
ICCAD12-1	98.46	788	20.90	98.15	1542	35.80	99.70	1436	26.87
ICCAD12-2	2.340	1601	3.39	2.330	2812	5.95	97.42	2648	5.60
ICCAD12-3	85.50	2938	5.47	75.12	3229	6.01	96.55	4484	8.35
ICCAD12-4	73.72	2913	7.96	90.28	4798	13.10	95.64	6842	18.69
ICCAD12-5	76.47	908	4.11	81.71	2325	10.52	94.12	1227	5.55
ICCAD12-28	95.92	8642	5.41	96.33	11010	6.90	96.99	8210	5.14
ICCAD19	74.24	32958	14.36	69.14	30299	13.20	96.30	69743	30.39
Average	71.54	7296	8.80	72.93	8055	13.07	96.67	13512	14.37
Ratio	0.74	0.54	0.61	0.75	0.60	0.91	1.00	1.00	1.00
ICCAD19-C	93.74	75068	32.72	93.34	76109	33.17	96.30	69743	30.39

TABLE V
RESULT COMPARISON WITH CONVENTIONAL ACTIVE LEARNING SOLUTIONS ON ICCAD16 BENCHMARKS

Benchmarks	US [22]			EMC [24]			Ours		
	Acc	Litho		Acc	Litho		Acc	Litho	
	(%)	#	(%)	(%)	#	(%)	(%)	#	(%)
ICCAD16-2	71.43	543	53.08	73.21	546	53.37	100.0	881	86.12
ICCAD16-3	84.60	1939	38.66	87.30	1958	39.04	99.74	3589	71.55
ICCAD16-4	66.03	870	47.41	57.69	875	47.68	98.08	1608	87.63
Average	74.02	1117	46.38	72.73	1126	46.70	99.27	2026	81.77
Ratio	0.75	0.55	0.57	0.733	0.56	0.57	1.00	1.00	1.00

We compare the sampling results on ICCAD12/16/19 with exact/fuzzy matching methods and two recent sampling methods, as listed in Tables II and III. Columns "PM-exact," "PM-a95," "PM-a90," and "PM-e2" correspond to the results derived from pattern matching using a state-of-the-art pattern analysis tool [15], where PM-exact denotes only exactly same patterns can be clustered together, PM-a95 and PM-a90 refer to any clips that satisfy 95% and 90% area constraints are clustered together and PM-e2 groups clips with less than 2-nm edge displacements. Here the area and edge constraints are defined following [33]. Column "FT" lists the result of clustering on frequency domain of layout patterns that is similar to the flow proposed in [35]. Columns "Litho" cover the number (#) and the ratio (%) of clips being labeled out of total pattern numbers, including the clips sampled into training sets and all the detection extras.

PM-exact, as the reference method, shows 100% accuracy on all test cases except for ICCAD16-4. According to the lithography simulation results of the layout in ICCAD16-4, we notice all defects appear at the patterns belong to a different design space, which possibly makes the lithography model and optical proximity analysis inaccurate. Therefore, we observe minor prediction error and extra on ICCAD16-4. The result also shows exact pattern matching can achieve extremely high verification accuracy, however, at the cost of simulating and labeling a large fraction clip patterns in the whole dataset. On the contrary, the proposed batch sampling method achieves similar detection accuracy querying only 7% of total layout clips for ICCAD-12 and 300 less clips on average for ICCAD-16 compared to the best baseline PM-exact. In particular, our method shows even higher detection accuracy on ICCAD16-4 than exact pattern matching because of the effective Gaussian initial sampling, which also demonstrates our assumption holds in Algorithm 2. It should be noted that it is normal that the instances in a training set is not completely separable, which explains why our method behaves even better than exact pattern matching. We can also observe that our algorithm behaves much better on ICCAD12 than ICCAD16, which can be explained by the fact that ICCAD12 contains more than $10\times$ sample candidates that fits our algorithm better.

For three fuzzy matching options, various area or edge constraints offer different level of tradeoffs between verification performance and lithography overhead. PM-a95 and PM-e2 can still maintain good prediction accuracy on ICCAD16-2 and ICCAD16-3 with slightly less litho count, but the total number of labeled instances is still much larger than our method. Moreover, fuzzy matching fails to extract problematic instances on a more difficult testcase ICCAD16-4 with looser constraints that they all reach less than 50% prediction accuracy. Experiments on ICCAD19 also manifest the scalability issue of fuzzy matching solutions. As can be seen, area or edge displacement constrained fuzzy matching cannot be done within days when the design pool is extremely large and with complicated patterns. It should be noted that we did not further narrow down the matching window because matching on regions much smaller than the estimated optical diameter does not make sense for hotspot detection purpose.

Shim *et al.* [35] proposed to use frequency-domain representation to sample layout patterns with similar property and detect hotspots. Here, we conduct additional experiments by clustering layout clips based on their Fourier transform results. Clips closest to a cluster center will be selected as the representative clip that indicates the property of the whole cluster. By the results in the column "FT," we can observe that with similar sampling number, batch active sampling exhibits much better than frequency-domain clustering. We also conduct an experiment using the greedy sampling method [17] where each instance being predicted as hotspot will be incrementally added into the training set. Although the greedy sampling method can successfully select partial hotspot clips in some test cases, the performance is highly affected by the initial learning model, where prediction error will be gradually amplified in the greedy sampling procedure. As listed in "Greedy," the greedy method [17] only achieves $\sim$60% average detection accuracy on ICCAD16, $\sim$74% average detection accuracy on ICCAD12, and $\sim$80% on ICCAD19.

To show that the proposed sampling solution is efficiently and carefully designed for layout printabilility estimation purpose, we also compare the PSHD results with two popular active learning solutions US [22] and EMC [24]. Both algorithms are implemented and integrated into our deep learning framework. As shown in Tables IV and V, our method exhibits obvious advantage on detection accuracy on ICCAD12, ICCAD16, and ICCAD19 cases over US and

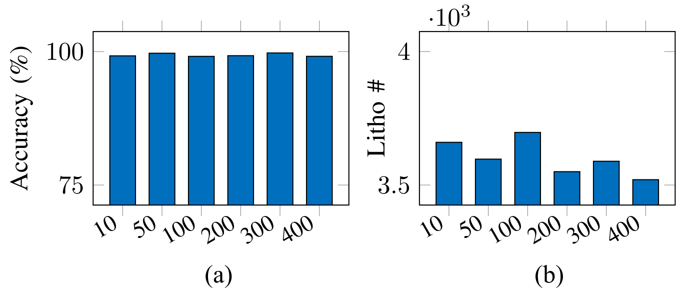


Fig. 8. Influence of initial sampling size. (a) Accuracy(%). (b) Litho #.

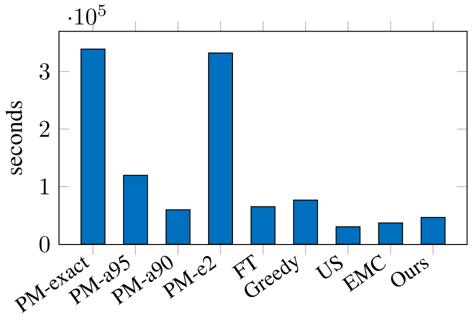


Fig. 9. Runtime comparison among different solutions. "PM-xx"s are conducted on 10-core Intel E7-4830v2 with 512-GB memory. "FT," "Greedy," "US," "EMC," and "Ours" are deep learning-based flows that are tested on a GPU platform with one GeForce GTX 1080Ti, one Intel i9-7900X, and 64-GB memory.

EMC, and the weak accuracy behavior makes the lithography overhead advantage of US and EMC trivial. We can also observe that the performance gap is narrowed on the `ICCAD12`. It can be explained by that the `ICCAD12` benchmark is not a full-chip design and are delivered with clips that are diverse in hotspot and nonhotspot patterns. Thus, most sampling methods will work and the deep learning model dominates the performance. Different tradeoffs of US and EMC on `ICCAD12` is a reflection of sampling mechanisms in two methods. Although US and EMC behave reasonably good on `ICCAD12-28`, their drawbacks can be easily seen when patterns are becoming complicated, which is consistent with the results on `ICCAD19` (72.4% of US and 69.14% of EMC compared to 96.3% of our approach). We also conduct conservative runs of US and EMC on `ICCAD19` with results listed in `ICCAD19-C` of Table IV. We can observe that at conservative condition, US and EMC still exhibit weaker detection accuracy than ours even with larger lithography overhead. We did not implement QBC [23] in this work, because QBC requires ensemble learning engines which is not directly compatible with our deep learning-based framework.

C. Some Discussions

Intuitively, the final prediction results are also sensitive to the number of instances sampled in the initial sampling stage. In this experiment, we study the influence of different initial sampling sizes. Here, we use `ICCAD16-3` as an example and conduct our PSHD flow with different initial sampling size, as shown in Fig. 8. It can be seen that initial sampling size does not cause much variations in prediction accuracy due to our diversity-aware sampling strategy. The only difference comes from lithography overhead that a reasonably large initial sampling size will efficiently reduce the lithography overhead.

In real backend layout verification flow, almost all the computation and runtime overhead come from lithography simulation. Here, we assume a 10-s penalty on each litho-clip in our framework as in [17]. The overall runtime is then evaluated by the summation of simulation penalty and PSHD overhead. As shown in Fig. 9, the proposed framework is much more efficient than existing solutions without any performance degradation.

It can be observed from the result table that for some test cases especially for these EUV-specific designs in `ICCAD16`, the active sampling approach comes with much larger lithography overhead compared to the results in `ICCAD12` series. One explanation is that behavior prediction of EUV lithography (i.e., main objective of general lithography hotspot detection) is more complicated than the legacy DUV technology node. Because there are much more aspects to be considered in EUV design-to-hotspot mapping, such as mask 3-D effects, new resist models, stochastic lithography, etc. [36], which are clearly more challenging for learning models to capture. Such challenges also affect other hotspot detection methodologies that include pattern matching and clustering, which can be seen in the result table that most solutions (ours and others) are exhibiting larger lithography overhead on `ICCAD16` than `ICCAD12` when similar detection accuracies are achieved. As for legacy designs, pattern complexity will be responsible for weak behaviors of hotspot detectors. Evidence comes with experimental results on the dataset of `ICCAD12-1`, `ICCAD12-3`, and `ICCAD19`, which include large fraction of complicated hotspot patterns and hence draw challenges on learning-based hotspot detectors. As a result, all learning-based solutions (Greedy, US, EMC, and Ours) present higher lithography overhead on these three test cases. *In Summary, these challenging designs pose threats of model reliability, which we believe is one of the key considerations in efficient hotspot detector design in the future.*

V. Conclusion

A layout PSHD flow was proposed to adaptively sample layout patterns into a pattern library that is used to train a machine learning model for layout hotspot detection. The diversity-aware batch sampling and the interactive optimization of learning model can efficiently select interesting patterns and ensure a better model generality. Experiments show that the proposed framework is able to achieve similar detection accuracy requiring much smaller number of labeled patterns, which reduces lithography simulation overhead by a significant amount. Continuing study on model robustness, feature extraction, and training set initialization are also interesting to fit the proposed framework better on modern chip/circuit design requirements.

References

[1] E. Roseboom, M. Rossman, F.-C. Chang, and P. Hurat, "Automated full-chip hotspot detection and removal flow for interconnect layers of cell-based designs," in *Proc. SPIE*, vol. 6521, 2007, Art. no. 65210C.

[2] T. Matsunawa, J.-R. Gao, B. Yu, and D. Z. Pan, "A new lithography hotspot detection framework based on AdaBoost classifier and simplified feature extraction," in *Proc. SPIE*, vol. 9427, 2015, Art. no. 94270S.

[3] W.-Y. Wen, J.-C. Li, S.-Y. Lin, J.-Y. Chen, and S.-C. Chang, "A fuzzy-matching model with grid reduction for lithography hotspot detection," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 33, no. 11, pp. 1671–1680, Oct. 2014.

[4] W. Zhang, X. Li, S. Saxena, A. Strojwas, and R. Rutenbar, "Automatic clustering of wafer spatial signatures," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, 2013, pp. 1–6.

[5] H. Yang, J. Su, Y. Zou, B. Yu, and E. F. Y. Young, "Layout hotspot detection with feature tensor generation and deep biased learning," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, 2017, pp. 1–6.

[6] Y.-T. Yu, G.-H. Lin, I. H.-R. Jiang, and C. Chiang, "Machine-learning-based hotspot detection using topological classification and critical feature extraction," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 3, pp. 460–470, Mar. 2015.

[7] H. Yang, P. Pathak, F. Gennari, Y.-C. Lai, and B. Yu, "DeePattern: Layout pattern generation with transforming convolutional auto-encoder," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, 2019, pp. 1–6.

[8] H. Yang, P. Pathak, F. Gennari, Y.-C. Lai, and B. Yu, "Detecting multi-layer layout hotspots with adaptive squish patterns," in *Proc. IEEE/ACM Asia South Pac. Design Autom. Conf. (ASPDAC)*, 2019, pp. 299–304.

[9] H. Yang, L. Luo, J. Su, C. Lin, and B. Yu, "Imbalance aware lithography hotspot detection: A deep learning approach," in *Proc. SPIE Adv. Lithography*, vol. 10148, 2017, Art. no. 1014808.

[10] T. Matsunawa, S. Nojima, and T. Kotani, "Automatic layout feature extraction for lithography hotspot detection based on deep neural network," in *Proc. SPIE Adv. Lithography*, vol. 9781, 2016, Art. no. 97810H.

[11] M. Shin and J.-H. Lee, "Accurate lithography hotspot detection using deep convolutional neural networks," *J. Micro/Nanolithography MEMS MOEMS*, vol. 15, no. 4, 2016, Art. no. 043507.

[12] R. Chen, W. Zhong, H. Yang, H. Geng, X. Zeng, and B. Yu, "Faster region-based hotspot detection," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, 2019, pp. 1–6.

[13] H. Li *et al.*, "Attacking split manufacturing from a deep learning perspective," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, 2019, pp. 1–6.

[14] W.-C. Chang, I. H.-R. Jiang, Y.-T. Yu, and W.-F. Liu, "iClaire: A fast and general layout pattern classification algorithm," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, 2017, pp. 1–6.

[15] K.-J. Chen, Y.-K. Chuang, B.-Y. Yu, and S.-Y. Fang, "Minimizing cluster number with clip shifting in hotspot pattern classification," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, 2017, pp. 1–6.

[16] D. Ding, B. Yu, J. Ghosh, and D. Z. Pan, "EPIC: Efficient prediction of IC manufacturing hotspots with a unified meta-classification formulation," in *Proc. IEEE/ACM Asia South Pac. Design Autom. Conf. (ASPDAC)*, 2012, pp. 263–270.

[17] H. Zhang, B. Yu, and E. F. Y. Young, "Enabling online learning in lithography hotspot detection with information-theoretic feature optimization," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, 2016, pp. 1–8.

[18] S. Shim and Y. Shin, "Topology-oriented pattern extraction and classification for synthesizing lithography test patterns," *J. Micro/Nanolithography MEMS MOEMS*, vol. 14, no. 1, 2015, Art. no. 013503.

[19] T. Matsunawa, B. Yu, and D. Z. Pan, "Laplacian eigenmaps-and bayesian clustering-based layout pattern sampling and its applications to hotspot detection and optical proximity correction," *J. Micro/Nanolithography MEMS MOEMS*, vol. 15, no. 4, 2016, Art. no. 043504.

[20] J. Guo, F. Yang, S. Sinha, C. Chiang, and X. Zeng, "Improved tangent space based distance metric for accurate lithographic hotspot classification," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, 2012, pp. 1173–1178.

[21] B. Settles, "Active learning literature survey," Dept. Comput. Sci., Univ. Wisconsin Madison, Madison, WI, USA, Rep. 1648, 2010.

[22] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *Proc. ACM SIGIR Conf.*, 1994, pp. 3–12.

[23] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby, "Selective sampling using the query by committee algorithm," *Mach. Learn.*, vol. 28, no. 2, pp. 133–168, 1997.

[24] W. Cai, Y. Zhang, and J. Zhou, "Maximizing expected model change for active learning in regression," in *Proc. IEEE Int. Conf. Data Min. (ICDM)*, 2013, pp. 51–60.

[25] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *J. Mach. Learn. Res.*, vol. 2, pp. 45–66, Nov. 2001.

[26] G. Schohn and D. Cohn, "Less is more: Active learning with support vector machines," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2000, pp. 839–846.

[27] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Stat.*, vol. 22, no. 1, pp. 79–86, 1951.

[28] S. Chakraborty, V. Balasubramanian, Q. Sun, S. Panchanathan, and J. Ye, "Active batch selection via convex relaxations with guaranteed solution bounds," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 10, pp. 1945–1958, Jan. 2015.

[29] C. Mack, *Fundamental Principles of Optical Lithography: The Science of Microfabrication*. New York, NY, USA: Wiley, 2008.

[30] J. E. Greivenkamp, *Field Guide to Geometrical Optics*. Bellingham, WA, USA: SPIE Press, 2004.

[31] G. H. Golub and C. F. Van Loan, *Matrix Computations*, vol. 3. New York, NY, USA: JHU Press, 2012.

[32] A. J. Torres, "ICCAD-2012 CAD contest in fuzzy pattern matching for physical verification and benchmark suite," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, 2012, pp. 349–350.

[33] R. O. Topaloglu, "ICCAD-2016 CAD contest in pattern classification for integrated circuit design space analysis and benchmark suite," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, 2016, pp. 1–4.

[34] G. R. Reddy, K. Madkour, and Y. Makris, "Machine learning-based hotspot detection: Fallacies, pitfalls and marching orders," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, 2019, pp. 1–8.

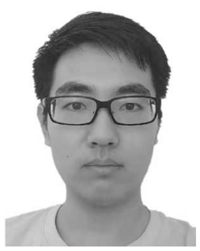
[35] S. Shim, W. Chung, and Y. Shin, "Synthesis of lithography test patterns through topology-oriented pattern extraction and classification," in *Proc. SPIE*, vol. 9053, 2014, Art. no. 905305.

[36] H. J. Levinson and T. A. Brunner, "Current challenges and opportunities for euv lithography," in *Proc. SPIE*, vol. 10809, 2018, Art. no. 1080903.

Haoyu Yang received the B.E. degree from Qiushi Honors College, Tianjin University, Tianjin, China, in 2015. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong.

He has interned with ASML, San Jose, CA, USA, and Cadence Design Systems, San Jose. His research interests include machine learning and very large-scale integration design and sign-off.

Mr. Yang received the 2019 Nick Cobb Scholarship by SPIE and Mentor Graphics.



Shuhe Li received the B.Sc. and M.Sc. degrees from the Department of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong, in 2018 and 2019, respectively.

His research interests include machine learning for very large-scale integration EDA.

Cyrus Tabery was born in Austin, TX, USA. He is currently pursuing the graduation degree in chemical engineering with the University of Texas at Austin, Austin, where he studied lithography with the Willson Research Group.

In 2000, he was with AMD, Silicon Valley, CA, USA, where he researched on lithography, mask making, and OPC. He worked with Globalfoundries, Santa Clara, CA, USA, until August 2011 and Intel Custom Foundry from 2011 to 2016 on process design kit development with focus on physical verification and fill and DFM technology. In 2016, he joined ASML, Veldhoven, The Netherlands, to work on design and patterning technology in Brion ATD. He is an Expert in yield, DFT, statistics, design technology, lithography, metrology, reticle technology, and OPC.

Bingqing Lin received the Ph.D. degree from Nanyang Technological University, Singapore, in 2014.

He is currently an Assistant Professor with the College of Mathematics and Statistics, Shenzhen University, Shenzhen, China. His current research interests include machine learning and model selection.

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

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

Dr. Yu received six Best Paper Awards from International Conference on Tools with Artificial Intelligence 2019, *Integration, the VLSI Journal* in 2018, International Symposium on Physical Design 2017, SPIE Advanced Lithography Conference 2016, International Conference on Computer-Aided Design 2013, and Asia and South Pacific Design Automation Conference 2012, 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*.