Robustify ML-Based Lithography Hotspot Detectors

Jingyu Pan[1], Chen-Chia Chang[1], Zhiyao Xie[2], Jiang Hu[3], and Yiran Chen[1]
Duke University[1] Hong Kong University of Science and Technology[2] Texas A&M University[3]
{jingyu.pan, chenchia.chang, yiran.chen}@duke.edu, eezhiyao@ust.hk, jianghu@tamu.edu

ABSTRACT

Deep learning has been widely applied in various VLSI design automation tasks, from layout quality estimation to design optimization. Though deep learning has shown state-of-the-art performance in several applications, recent studies reveal that deep neural networks exhibit intrinsic vulnerability to adversarial perturbations, which pose risks in the ML-aided VLSI design flow. One of the most effective strategies to improve robustness is regularization approaches, which adjust the optimization objective to make the deep neural network generalize better. In this paper, we examine several adversarial defense methods to improve the robustness of ML-based lithography hotspot detectors. We present an innovative design rule checking (DRC)-guided curvature regularization (CURE) approach, which is customized to robustify ML-based lithography hotspot detectors against white-box attacks. Our approach allows for improvements in both the robustness and the accuracy of the model. Experiments show that the model optimized by DRC-guided CURE achieves the highest robustness and accuracy compared with those trained using the baseline defense methods. Compared with the vanilla model, DRC-guided CURE decreases the average attack success rate by 53.9% and increases the average ROC-AUC by 12.1%. Compared with the best of the defense baselines, DRC-guided CURE reduces the average attack success rate by 18.6% and improves the average ROC-AUC by 4.3%.

1 INTRODUCTION

Machine learning (ML) techniques, especially those based on deep learning, have been widely employed in electronic computer-aided design (CAD) domains ranging from logic synthesis [24] to physical design [8] and design for manufacturability (DFM) [18]. In DFM, ML-based lithography hotspot detectors are well studied and serve as a successful example of ML techniques applied to accelerate the DFM development cycle. For advanced technology nodes, since the transistor feature sizes are reaching the limit of conventional optical lithography systems, lithographic process variations can drastically affect the manufacturing yield. Therefore, detection of lithography hotspots - layout patterns that can potentially cause manufacturing defects - is very important. Conventional lithography hotspot detection approaches involve optical proximity correction (OPC) [16] and lithographic simulation, which suffers from high runtime overhead. To enable faster and accurate lithography hotspot detection, recent studies has shown that convolutional neural networks (CNN)-based lithography hotspot detectors can bypass the time-consuming simulation and achieve state-of-the-art accuracy [22].

However, deep neural networks like CNNs exhibit intrinsic susceptibility to *adversarial perturbations* [5, 10, 12]. Adversarial perturbations are small but deliberate alterations to the inputs of the deep neural network, resulting in incorrect outputs. And such susceptibility to adversarial perturbations *poses risks in the VLSI design flow* [19, 20]. Due to the global trends in VLSI design and manufacturing, it is common for designers to procure intellectual property (IP) designs from third-party design vendors and combine them with components designed in-house to generate the chip layout. A blueprint of the chip layout can be sent to the foundry for further mask synthesis and manufacturing. However, third-party design vendors cannot always be trusted. Before tapeout, the designer can quickly verify the printability of the purchased designs using a CNN-based lithography hotspot detector, which can be integrated into a commercial tool [11].

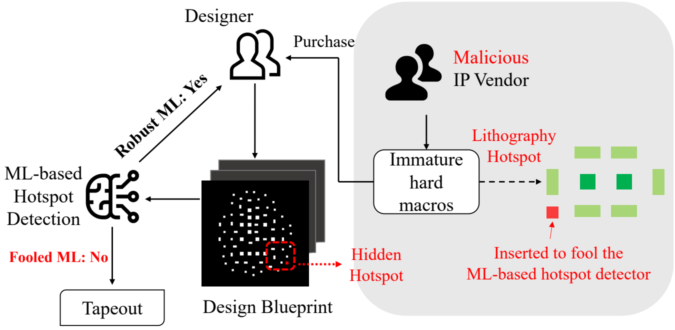


Figure 1: Malicious IP vendors may use adversarial perturbations to hide hotspots in immature designs. Designers using an unrobust ML-based hotspot detector for printability verification may suffer from great loss at the tapeout stage due to hidden hotspots.

Figure 1 shows an adversarial scenario where a malicious design vendor may seek short-cut profits by selling immature IP designs. The malicious vendor can add adversarial perturbations to lithography hotspots to hide these defects, instead of really correcting them. In this way, CNN-based lithography hotspot detectors may be fooled and unable to recognize the perturbed hotspots. And if the perturbed layout is sent to a foundry for tapeout, it can cause a great loss to downstream chip yield, wasting the designer's effort. Another potential risk is that the malicious design vendor may simply sell bad IP designs to sabotage the design under development, wasting the designer's time and resources on design recycling and fixing poor designs. In summary, the risks posed by adversarial perturbations can *fundamentally undermine the trust in the ML-based lithography hotspot detectors.* Ensuring the reliability of the ML-based lithography hotspot detectors is a critical step toward the feasibility of ML's integration into the VLSI design flow. However, methods of robustifying ML-based lithography hotspot detectors are rarely discussed.

Recent ML research has demonstrated that adversarial perturbations pose risks in practically every application where deep neural networks are used [5, 10, 12]. Borrowing ideas from computer vision, prior works have also studied CAD-related tasks like lithography hotspot detection, proposing approaches of generating adversarial perturbations on via layout hotspots [11, 23]. It is worth noting that there are some important differences of adversarial perturbations between computer vision and lithography. In computer vision, typical perturbation methods make an imperceptible perturbation to each pixel [1], inducing incorrect model output. In lithography, however, such a method is infeasible because (1) both the layout patterns and the perturbations are binary (e.g., insertion or removal of pattern components), and (2) perturbations should pass design rule checking (DRC) and thus are constrained in their sizes, shapes and locations. Prior works typically make perturbations to the layout iteratively to ensure DRC-clean at each step. [11] focus on the insertion of fraudulent sub-resolution assistant features (SRAFs) and use pixel-based gradient method to find the best SRAF combination. Following [11], [23] considers removal of preexisting SRAFs and proposes an efficient group gradient method to optimize the perturbation, yielding better performance in successful perturbation generation. However, these works focus on the perturbation methods and rarely discuss defense methods. To the best of our knowledge, little systematic research on the methodologies of enhancing the robustness of ML for CAD has been found.

Motivated by the risks of adversarial perturbations, in this paper, we propose a customized regularization-based defense method, called **DRC-guided CURE**, to robustify ML-based lithography hotspot detectors[1]. We design the regularizer based on DRC constraints of the adversarial perturbations. In our experiments, we focus on the white-box attacker with full knowledge about the model, since this is the strongest attacker setting. Experiment results show that DRC-guided CURE achieves superior performance in both robustness and accuracy compared with the vanilla model (no defense applied), adversarial training and CURE. Our main contributions are summarized as follows:

- We analyze the effectiveness of existing defense techniques, including adversarial training and regularization-based approaches, in a case study of lithography hotspot detection. We demonstrate their limitations in robustifying the ML-based lithography hotspot detector.
- We propose an innovative regularization-based defense method, called DRC-guided CURE, which is customized for robustifying ML-based lithography hotspot detectors. Our proposed DRC-guided CURE outperforms all baseline defense techniques, decreasing the perturbation success rate by 53.9% compared with the vanilla model. Moreover, this robustness improvement costs no accuracy loss. Instead, it even improves the area under the ROC curve of the model by 12.1%.
- We provide an ablated analysis on the effectiveness of our proposed DRC-guided CURE in the white-box attacker setting. We give intuitive explanations of DRC-guided CURE's capability of improving both robustness and accuracy of the ML-based lithography hotspot detector.

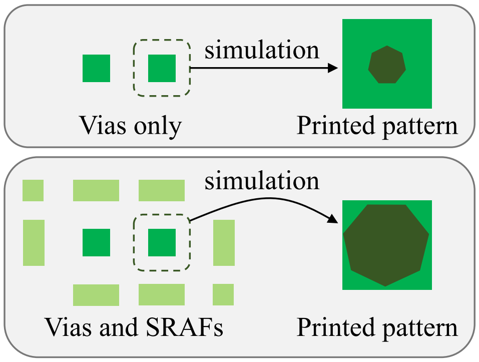


Figure 2: Illustration of simulated printed patterns of vias layouts without SRAFs and with SRAFs.

2 PRELIMINARIES

In this section, we will introduce backgrounds related to adversarial defenses for ML-based lithography hotspot detectors.

2.1 ML-based Lithography Hotspot Detection

Lithography hotspot detection facilitates the VLSI back-end design and sign-off flow by early detection of the lithography hotspots. The conventional approach to lithography hotspot detection is based on simulation using physical models of optical lithography. Despite its accuracy, such simulation costs lots of computational resources and is very time-consuming, especially for modern VLSI designs.

When a lithography hotspot is detected, resolution enhancement techniques (RETs) such as SRAF insertion [2, 4, 21] and OPC [16] are applied to compensate for lithography distortion and thus fix the hotspot. Figure 2 shows illustration of simulated printed patterns of vias layouts without SRAFs and with SRAFs. If the vias are printed as is, the resulting printed output would be only a small region of the vias pattern, which is far from the desired result. With SRAF insertion, the printed pattern can more accurately reflect the desired vias pattern.

Due to the prohibitive run-time of conventional lithography simulation, recent studies have proposed alternative methods to speed up the hotspot detection process using machine learning [14, 22]. Machine learning solutions seek to statistically model the underlying relationships between lithographic features and the corresponding layout's printability. Note that, by transforming the lithographic features into images, one can pose the lithography hotspot detection problem as a binary image classification problem. Borrowing ideas from computer vision, recent work has proposed convolutional neural networks (CNN) for this problem, achieving state-of-the-art accuracy [22].

2.2 Adversarial Perturbations on Lithography Hotspot Patterns

To the best of our knowledge, two gradient-based white-box adversarial perturbation methods have been proposed for CNN-based lithography hotspot detectors [11, 23]. The first method is the pixel-based gradient method [11] which generates DRC-clean fraudulent

[1] In order to contribute to the reproducibility and fair comparison in ML for CAD, the framework and the dataset used in our experiment are anonymously open-sourced at https://github.com/panjingyu/Robustify-ML-Based-Lithography-Hotspot-Detectors.

SRAFs and makes attempts based on pixel gradients to add fraudulent SRAFs to the original layout to fool the lithography hotspot detector. In [11], hotspot clips are converted to images, and each valid fraudulent SRAF region is a block of pixels. In the pixel-based gradient method, given a set of valid SRAF shapes, the algorithm iterates the summation of gradients of all pixels in each possible location. Fraudulent SRAFs with negative gradient sums are iteratively inserted into the layout in the ascending order of their gradient sums, until either target neural network gives incorrect output or the attack constraint is met.

The second method, which is known as the group gradient method [23], makes improvements based on the pixel-based gradient method [11]. The group gradient method randomly generates a set of DRC-clean SRAF candidates and iteratively optimizes the weight of each candidate to minimize the perturbation given the constraint of successfully fooling the deep neural network. [23] also expands the adversarial perturbation space by considering removal of preexisting SRAFs. As a result, the group gradient method reports better attack success rate than the pixel-based gradient method. Therefore, we will focus on the group gradient method as our attacker model in our experiments.

2.3 Threat Model

2.3.1 Setting. We explore the scenario of a designer considering the purchase of a macro from a third-party intellectual property (IP) vendor, as posed in previous studies of threats to the VLSI design flow [19]. The IP vendor distributes hard macros in GDS-II format, where the layout is allegedly enhanced for lithography using RETs. As part of the validation process, the designer checks the macro to establish its quality by using a CNN-based hotspot detector. And the attacker's goal is to generate minimal perturbations on a lithography hotspot to fool the target ML-based lithography hotspot detector into misclassifying the perturbed hotspot as a non-hotspot.

2.3.2 Attacker Capabilities. Following research on adversarial attacks in deep learning, the attackers can be roughly categorized into *white-box* attackers and *black-box* attackers according to the constraints on their access to information about the target model. For example, given a CNN-based target model, a black-box attacker can only make queries to the model and access the input-output pairs. In contrast, white-box attackers posses access to all the information including the hyperparameters of the target model (e.g., its architecture, its training algorithm), the parameters (e.g., the weights and bias) of each layer, the training data, and so on. Therefore, a white-box attacker is a stronger attacker model than a black-box attacker [1, 5, 12]. In this work, we consider defense techniques against white-box attackers.

3 PROBLEM FORMULATION

Consider a white-box attacker $\mathcal{A}$ with full knowledge about the architecture, training algorithms, and weights and bias of each layer of the target ML-based lithography hotspot detector f, $\mathcal{A}$ crafts adversarial perturbations to a set of lithography layout patterns. The objective of robustifying an ML-based lithography hotspot detector f is to maximize the accuracy of f on the dataset containing adversarially perturbed lithography layout patterns.

Algorithm 1 Adversarial Training

Input: Training hotspot data D_H, training non-hotspot data D_N, a trained ML-based lithography hotspot detector f, a training algorithm T, attacker function $\mathcal{A}(d, f)$ which adds adversarial perturbations to hotspot data d in order to fool model f and outputs the perturbed data, and the number N of perturbed data samples.

Output: Robustified model f.

```
$D'_H \leftarrow \{\}$                ▷ Initialize a set of adversarial hotspot data.
for $d_H \in D_H$ do
    $D'_H = D'_H \cup \mathcal{A}(d_H, f)$
    if $|D'_H| \geq N$ then
        End this for loop.
$D = D'_H \cup D_H \cup D_N$           ▷ Construct the training dataset.
$f \leftarrow T(f, D)$                  ▷ Retrain the model.
return $f$
```

For a fair evaluation on the overall accuracy of the target hotspot detector, we use the **area under the receiver operating characteristic curve (ROC-AUC)** as the metric for accuracy. Besides, to directly analyze the robustness against adversarial perturbations on layout hotspots, we define **hotspot accuracy** as *the ratio of the number of correctly detected hotspots and the total number of hotspots*. We also define **attack success rate** r as

$$r = \frac{|\{\text{True Positives}\} \cap \{\text{Successful attacks}\}|}{|\{\text{True Positives}\}|},$$

where {Positives} denotes the set of hotspots that can be correctly detected by the detector before any adversarial perturbation, and {Successful attacks} denotes the set of hotspots that are incorrectly undetected by the detector after adding adversarial perturbations by the adversary $\mathcal{A}$.

4 DEFENSE METHODOLOGIES

In this section, we first introduce two baseline defense methods, adversarial training and curvature regularization. Then we present our innovative DRC-guided CURE method.

4.1 Adversarial training

Assuming that the defender is aware of the risks of adversarial perturbations on hotspot detectors, it can robustify the model by including adversarial layouts into the training dataset but with true hotspot labels and then retraining the model. This method is known as adversarial training [13]. Algorithm 1 illustrates the procedure of adversarial training.

Adversarial training has recently been shown to be one of the most effective methods for increasing the robustness of a deep neural network against adversarial perturbations. Recent studies showed that adversarial training correlates to decreased curvature of the loss [15]. Intuitively, a smaller curvature of the loss reflects a smoother decision boundary of the classifier, which is more robust against small random perturbations to the inputs.

4.2 Curvature Regularization (CURE)

The CURE method robustify the model by adjusting its optimization objective. CURE aims at directly minimizing the curvature of the loss to achieve robustness that is comparable to adversarial training [15]. In order to minimize the curvature of the loss, the CURE regularizer should penalize large eigenvalues of the Hessian H of the loss ℓ at input point x, since the eigenvalues correspond to the amount of curvature at the direction of their corresponding eigenvectors. Let $\lambda_1, \ldots, \lambda_d$ denote the eigenvalues of H. To encourage all eigenvalues to be small, the CURE regularizer L_{CURE} can be formulated as $L_{\text{CURE}} = \sum_i \lambda_i^2$, which corresponds to the Frobenius norm of the Hessian H. With function $p(\lambda) = \lambda^2$, we have

$$L_{\text{CURE}} = \sum_i p(\lambda) = \text{trace}(p(H)) = \mathbb{E}(z^T p(H) z) = \mathbb{E}\, ||Hz||^2\,,$$

where the expectation is taken over $z \sim \mathcal{N}(0, I_d)$. By using a finite difference approximation of the Hessian H, we have $Hz \approx \frac{\nabla\ell(x+hz)-\nabla\ell(x)}{h}$, where h denotes the discretization step. Therefore, the regularizer becomes

$$L_{\text{CURE}}(x) = \frac{1}{h^2}\mathbb{E}\,||\nabla\ell(x+hz) - \nabla\ell(x)||\,,$$

which involves computing an expectation over z and penalizes large curvatures over all directions. Since prior works [3, 7] have shown that the direction of the gradients indicates the directions of high curvature, it is usually a natural choice to make z in line with the gradient direction. Besides, in practice, common image-based adversarial problems usually involve ℓ_∞ norm constraints on the adversarial perturbations. Hence, [15] set the step z to the sign of the gradient multiplied with a normalization factor, written as $z = \frac{\text{sign}(\nabla\ell(x))}{||\text{sign}(\nabla\ell(x))||}$. Finally, neglecting the $\frac{1}{h^2}$ factor, the CURE regularizer is formulated as

$$L_{\text{CURE}}(x) = ||\nabla\ell(x+hz) - \nabla\ell(x)||^2,$$

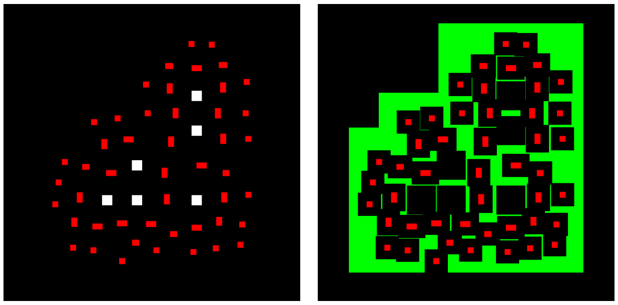
where h controls the scale of the discretization step. And the overall optimization objective of the model becomes $\ell(x) + \eta L_{\text{CURE}}(x)$.

4.3 DRC-guided CURE

Utilizing the DRC constraints on the adversarial lithography perturbations, we propose the DRC-guided curvature regularization method (DRC-guided CURE) to robustify ML-based lithography hotspot detectors. Regarding the DRC constraints, we consider both fraudulent SRAF insertion and preexisting SRAF removal. DRC-clean fraudulent SRAF insertion must satisfy the following constraints:

- Fraudulent SRAFs can only be inserted to the SRAF layer.
- Fraudulent SRAFs should be rectangles with a fixed width of 40nm and a variable height between 40-90nm, at a resolution of 1nm. The SRAF can be placed either horizontally or vertically.
- The Euclidean distance between any two SRAFs should be at least 40nm.
- Fraudulent SRAFs should not overlap with the forbidden region surrounding the vias in a layout.

According to these constraints, we can calculate the valid region for fraudulent SRAF insertion given a layout with preexisting vias and SRAFs. The valid removal candidates of adversarial perturbations are simply the preexisting SRAFs. Figure 3(a) shows an example of vias layout hotspot clip, where the red rectangles denote existing SRAFs and the white squares denote the vias. And Figure 3(b) shows the corresponding regions for potential adversarial perturbations. In Figure 3(b), the green region marks where DRC-clean fraudulent SRAF insertion is possible. The red rectangles denote removal candidates of adversarial perturbations, which are simply preexisting SRAFs. To ease the calculation of such regions for each layout, we consider the Chebyshev distance (i.e., the L_∞ distance) as an approximation of the Euclidean distance in the spacing constraints. Besides, we also assume the inserted fraudulent SRAFs should not be too far from the vias, and thus adding a limit to the outer boundaries of the possible adversarial insertion region. And the inner boundaries of the possible adversarial insertion region is determined by the spacing constraints regarding the preexisting SRAFs and the forbidden zone around the vias.



(a) Via and SRAF patterns.

(b) The corresponding regions for possible adversarial perturbations.

Figure 3: Illustration of via and SRAF patterns of a given layout and its corresponding regions for possible adversarial perturbations. In (a), the red rectangles denote existing SRAFs, and the white squares denote vias. In (b), the green region denotes the DRC-clean fraudulent SRAF insertion spots. And the red rectangles denote the adversarial removal candidates, which correspond to the preexisting SRAFs.

Based on the calculation of the regions of possible adversarial perturbations, we re-design the regularization term in CURE and propose the **DRC-guided CURE**. It is worth noting that, in CURE, the rationale behind the choice of step $z = \frac{\text{sign}(\nabla\ell(x))}{||\text{sign}(\nabla\ell(x))||}$ is that it targets robustness against ℓ_∞-constrained perturbations. However, the ℓ_∞ constraint does not hold in adversarial perturbations to lithography hotspot layouts. Instead, the adversarial perturbations that DRC-guided CURE targets is constrained by the DRC-clean fraudulent SRAF insertion region and the locations of preexisting SRAFs. Therefore, we propose a DRC-guided step $z' = \frac{m_i - m_r}{||m_i - m_r||}$, where m_i is the binary mask of the region for potential SRAF insertion (e.g., the green region in Figure 3) and m_r is the binary mask of the region for potential SRAF removal (e.g., the red rectangles in Figure 3). Here, $m_i, m_r \in \{0, 1\}^{H \times W}$, where H and W denote the height and width of the via layout, respectively. Therefore, we formulate the DRC-guided CURE regularizer as

$$L_{\text{DRC}}(x) = ||\nabla\ell(x + hz') - \nabla\ell(x)||^2,$$

Layer	Kernel size	Stride	Activation	Output Size
conv_DCT	128 × 128	128	-	(20, 20, 32)
conv_1	3 × 3	1	ReLU	(20, 20, 16)
conv_2	3 × 3	1	ReLU	(20, 20, 16)
pool_1	2 × 2	2	-	(10, 10, 16)
conv_3	3 × 3	1	ReLU	(10, 10, 32)
conv_4	3 × 3	1	ReLU	(10, 10, 32)
pool_2	2 × 2	2	-	(5, 5, 32)
linear_1	-	-	ReLU	256
linear_2	-	-	-	2

Table 1: Model architecture.

where h controls the scale of the DRC-guided step z'. The overall optimization objective then become the regularized loss function $\ell(x) + \lambda L_{\mathrm{DRC}}(x)$, where λ controls the strength of DRC-guided CURE.

In summary, our proposed DRC-guided CURE improves CURE by penalizing the curvature in the direction z' determined by the DRC constraints, rather than the gradient-guided direction z as proposed in CURE.

5 EXPERIMENTS

5.1 Experiment Setup

To investigate the effectiveness of our proposed DRC-guided CURE method, we compare the accuracy and robustness of the model trained using the DRC-guided CURE method with several baseline defense algorithms. Our baselines include the vanilla method (i.e., normal training without defense techniques), adversarial training and CURE. Our attacker adopts the state-of-the-art group gradient method [23]. We implement the all the defense methods and the attack method in Python using the PyTorch framework [17]. Our experiments run on a NVIDIA TITAN RTX GPU with Intel® Xeon® Gold 6136 CPU.

Regarding the via layout hotspot dataset, we use legacy node via designs that are verified and simulated using Mentor Graphics `Calibre` Design For Manufacturability tool suite [6]. We construct our dataset based on the raw via layout data in GDS-II format from [23]. We transform the raw via layout clips of $2\mu m \times 2\mu m$ size to images with a resolution of 2048×2048 pixels. The image-based via layout dataset is split randomly into a training dataset and a testing dataset. The training dataset includes 68565 via layout clips with 5012 hotspots. The testing dataset is composed of four groups, each of which is composed of 100 non-hotspot clips and 100 hotspot clips. Given a trained target model, the attacker model generates adversarial perturbations at its best effort based on the hotspot clips of the four testing groups. Besides, to ensure a strong attacker model, we relax the constraint on the maximal number of adversarial perturbations in each via layout to 20, posing a challenging setting to the defense methods.

ID	Original Hotspot Accuracy			
	Vanilla	AT	CURE	DRC-guided
1	0.79	0.69	**0.83**	**0.83**
2	0.71	0.66	0.81	**0.86**
3	0.76	0.75	**0.84**	0.83
4	0.71	0.66	0.80	**0.88**
Average	0.743	0.690	0.820	**0.850**

Table 2: Comparison of hotspot accuracy before perturbation attack when different defense methods are applied. Here, AT denotes adversarial training.

Table 1 gives the configuration of our model architecture, which directly follows [22]. The *conv_DCT* layer is equivalent with the Discrete Cosine Transform (DCT)-based feature tensor extraction in [22], transforming a layout clip image to a tensor of DCT frequency components. For both the vanilla method and the defense methods, we train the model for 40000 steps using the Adam optimizer [9] and set the initial learning rate to 0.01, with a batch size of 128, and L2 regularization strength of 0.00005. During training, to combat over-fitting caused by the imbalance in the number of hotspot clips and non-hotspot clips in the training set, we re-sample the hotspots to ensure the numbers of hotspot clips and non-hotspot clips are balanced in each batch of data. For adversarial training, we empirically set the number of adversarially perturbed layouts to 750, which is around 15% of the hotspot clips in the training set. For a fair comparison, we retrain the model from scratch on the training set with adversarial perturbations. For CURE, we set the step scale $h = 6$ and the strength of CURE $\lambda = 0.33$. For DRC-guided CURE, we set the step scale $h = 1$ and the strength $\lambda = 0.2$. For the group gradient method attack, we follow the hyperparameters in [23].

5.2 Evaluation of Robustified Models

Figure 4 compares the ROC-AUC of the model over the four testing groups and the average ROC-AUC, with the presence of adversarial perturbations. In each of the four testing group, our proposed DRC-guided CURE outperforms the vanilla model, adversarial training and CURE. Compared with the vanilla model, the DRC-guided CURE increases the average ROC-AUC from 0.786 to 0.881, which is a 12.1% improvement. As for the baseline defense methods, the model trained using adversarial training suffer from accuracy degradation. This is because the adversarial examples seen by the model in the adversarial training has limited coverage of the whole perturbation space. Therefore, adversarial training only provides robustness against a small subset of all the possible adversarial perturbations. Furthermore, the white-box attacker is actually highly flexible, since it can adjust the generated adversarial perturbations

ID	Hotspot Accuracy After Attack			
	Vanilla	AT	CURE	DRC-guided
1	0.43	0.60	0.66	**0.70**
2	0.47	0.60	0.62	**0.68**
3	0.38	0.57	0.60	**0.68**
4	0.47	0.61	0.64	**0.70**
Average	0.438	0.595	0.630	**0.690**

Table 3: Comparison of hotspot accuracy after perturbation attack when different defense methods are applied.

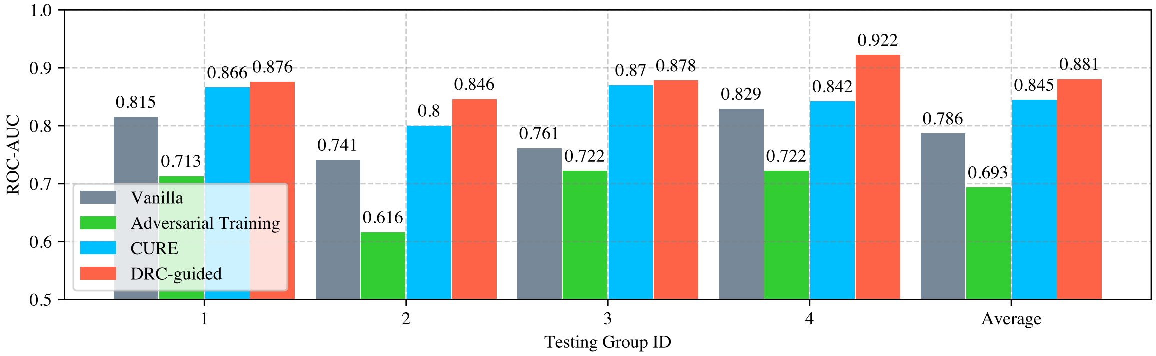


Figure 4: Comparison of ROC-AUC of different defense methods at the presence of adversarial perturbations.

according to the parameters of the model. Thus, adversarial training may finally provide little robustness to the target model. On the other hand, training on perturbed layouts can mislead the model to learn some trivial features and neglect important features from unperturbed layouts, thus hampering the accuracy of the model. And compared with CURE, our proposed DRC-guided CURE still shows an advantage of 4.3% higher ROC-AUC. This result proves the effectiveness of the DRC-guided step z' which utilizes information of the via layout patterns and regions of potential adversarial perturbations.

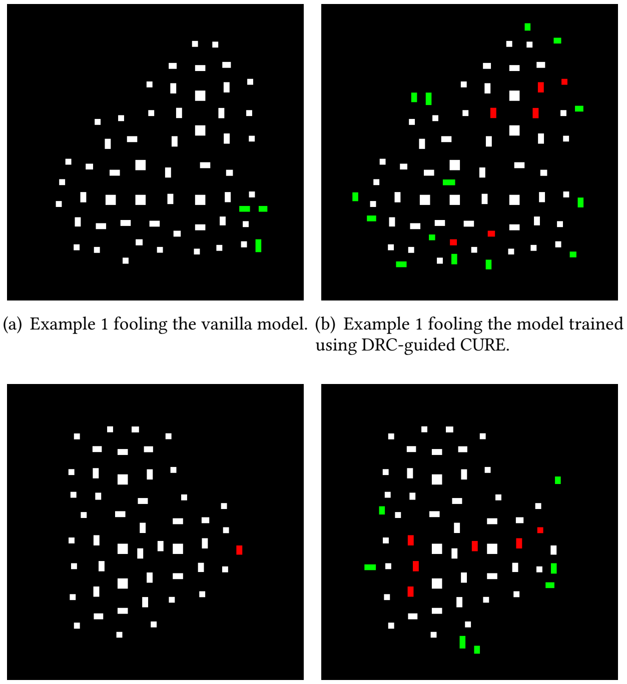
To directly analyze the robustness of the target model on the perturbed hotspot data, we compare the variation of hotspot accuracy before and after adversarial perturbations for different defense methods, along with comparison of their attack success rates. Table 2 and Table 3 shows the hotspot accuracy of the target model on the four testing groups before and after perturbations attack, respectively. Table 4 shows the corresponding attack success rate of the group gradient method on the four testing groups. AT in the tables denotes adversarial training. The vanilla model suffers from a high attack success rate of 40.8%, which indicates a drastic drop of average hotspot accuracy from 74.3% to 43.8%. For defense methods, adversarial training shows an even larger accuracy degradation than the vanilla model, failing to robustify the model. This result demonstrates that adversarial training yields poor convergence for the target model. CURE provides much better robustness than adversarial training, showing a much lower average attack success rate of 23.1% and a higher average hotspot accuracy of 63.0% after attack. Furthermore, our proposed DRC-guided CURE outperforms both adversarial training and CURE, showing a 53.9% lower average attack success rate of 18.8% than the vanilla model. Even compared with CURE, the attack success rate of DRC-guided CURE is still 18.6% lower, which proves the effectiveness of our customized regularizer. DRC-guided CURE also shows a superior hotspot accuracy after attack of 69.0%, which is 57.5% higher than the vanilla model The DRC-guided CURE can improve the hotspot accuracy of the model even without the presence of adversarial perturbations because it co-optimize both the value and the sharpness (i.e., curvature) of the loss simultaneously, helping the loss function converges at a smooth surface. Therefore, the model trained by DRC-guided CURE achieves a lower inference loss, which corresponds to higher accuracy. Besides, such regularization on curvature also help prevent over-fitting to the training data, improving the generality of the model.

ID	Attack Success Rate			
	Vanilla	AT	CURE	DRC-guided
1	0.456	0.377	0.205	**0.157**
2	0.338	0.424	0.235	**0.209**
3	0.500	0.427	0.286	**0.181**
4	0.338	0.530	**0.200**	0.205
Average	0.408	0.440	0.231	**0.188**

Table 4: Comparison of attack success rates when different defense methods are applied.

5.3 Visualization of Adversarial Perturbations after Defense

In the following, we intuitively analyze the robustness of the ML-based lithography hotspot detector trained using our proposed DRC-guided CURE. Figure 5 shows two examples of successful adversarial perturbations before and after DRC-guided CURE defense. The white blocks denote preexisting SRAFs and vias in the layout, while the colored blocks denote adversarial perturbations. To be specific, the green blocks denote inserted fraudulent SRAFs, and the red blocks denote removed preexisting SRAFs. In the group gradient method, adversarial perturbations are added to the layout iteratively. Therefore, when performing group gradient method, if fewer adversarial perturbations are needed to flip the output of hotspot detector, the hotspot detector is less robust. For the vanilla model, successful adversarial perturbations require insertion of merely three fraudulent SRAFs in example 1 and removal of one preexisting SRAF in example 2. However, after defense using DRC-guided CURE, successful adversarial perturbations require a greater number of modifications and a wider range of locations than for the vanilla model, in both examples. These two examples demonstrate that with the defense of DRC-guided CURE, the difficulty of generating successful adversarial perturbations is significantly higher.



(a) Example 1 fooling the vanilla model.

(b) Example 1 fooling the model trained using DRC-guided CURE.

(c) Example 2 fooling the vanilla model.

(d) Example 2 fooling the model trained using DRC-guided CURE.

Figure 5: Two examples of adversarial perturbations required to fool the ML-based lithography hotspot detector before and after DRC-guided CURE defense. Here, green blocks denote inserted fraudulent SRAFs, and red blocks denote deleted preexisting SRAFs.

6 CONCLUSIONS

In this work, we propose a customized DRC-guided CURE method to robustify ML-based lithography hotspot detectors in order to combat the risks posed by adversarial perturbations. We compare the robustness of the target model trained using DRC-guided CURE with the models trained using the vanilla method, adversarial training and CURE. Our proposed DRC-guided CURE proves to provide the most robustness to the model compared with the baseline methods, decreasing the attack success rate by 53.9%. Furthermore, DRC-guided CURE also increases the accuracy of the model, showing an improvement of the ROC-AUC by 12.1%. Compared with the best result among the baseline defense methods, DRC-guided CURE decreases the average attack success rate by 18.6% and improves the average ROC-AUC by 4.3%.

ACKNOWLEDGMENTS

This work is supported by SRC GRC-CADT 3103.001/3104.001 and NSF CCF-2106725/2106828. We also give special thanks to Haoyu Yang for sharing the vias layout hotspot dataset.

REFERENCES

[1] Battista Biggio and Fabio Roli. 2018. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition* 84 (2018), 317–331.

[2] Liang Deng, Martin DF Wong, Kai-Yuan Chao, and Hua Xiang. 2007. Coupling-aware dummy metal insertion for lithography. In *2007 Asia and South Pacific Design Automation Conference*. IEEE, 13–18.

[3] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, Pascal Frossard, and Stefano Soatto. 2018. Empirical study of the topology and geometry of deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3762–3770.

[4] Hao Geng, Wei Zhong, Haoyu Yang, Yuzhe Ma, Joydeep Mitra, and Bei Yu. 2019. SRAF insertion via supervised dictionary learning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39, 10 (2019), 2849–2859.

[5] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[6] Mentor Graphics. 2008. Calibre verification user's manual.

[7] Saumya Jetley, Nicholas Lord, and Philip Torr. 2018. With friends like these, who needs adversaries? *Advances in neural information processing systems* 31 (2018).

[8] Andrew B Kahng. 2018. Machine learning applications in physical design: Recent results and directions. In *Proceedings of the 2018 International Symposium on Physical Design*. 68–73.

[9] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[10] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. 2018. Adversarial examples in the physical world. In *Artificial intelligence safety and security*. Chapman and Hall/CRC, 99–112.

[11] Kang Liu, Haoyu Yang, Yuzhe Ma, Benjamin Tan, Bei Yu, Evangeline FY Young, Ramesh Karri, and Siddharth Garg. 2020. Adversarial perturbation attacks on ML-based CAD: A case study on CNN-based lithographic hotspot detection. *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 25, 5 (2020), 1–31.

[12] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. 2016. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770* (2016).

[13] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083* (2017).

[14] Tetsuaki Matsunawa, Jhih-Rong Gao, Bei Yu, and David Z Pan. 2015. A new lithography hotspot detection framework based on AdaBoost classifier and simplified feature extraction. In *Design-Process-Technology Co-optimization for Manufacturability IX*, Vol. 9427. SPIE, 201–211.

[15] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. 2019. Robustness via curvature regularization, and vice versa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9078–9086.

[16] Oberdan W Otto, Joseph G Garofalo, KK Low, Chi-Min Yuan, Richard C Henderson, Christophe Pierrat, Robert L Kostelak, Sheila Vaidya, and PK Vasudev. 1994. Automated optical proximity correction: a rules-based approach. In *Optical/Laser Microlithography VII*, Vol. 2197. International Society for Optics and Photonics, 278–293.

[17] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019).

[18] Martin Rapp, Hussam Amrouch, Yibo Lin, Bei Yu, David Z Pan, Marilyn Wolf, and Jörg Henkel. 2021. MLCAD: A Survey of Research in Machine Learning for CAD Keynote Paper. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2021).

[19] Masoud Rostami, Farinaz Koushanfar, and Ramesh Karri. 2014. A primer on hardware security: Models, methods, and metrics. *Proc. IEEE* 102, 8 (2014), 1283–1295.

[20] Zhiyao Xie, Jingyu Pan, Chen-Chia Chang, and Yiran Chen. 2022. The Dark Side: Security Concerns in Machine Learning for EDA. *arXiv preprint arXiv:2203.10597* (2022).

[21] Xiaoqing Xu, Tetsuaki Matsunawa, Shigeki Nojima, Chikaaki Kodama, Toshiya Kotani, and David Z Pan. 2016. A machine learning based framework for sub-resolution assist feature generation. In *Proceedings of the 2016 on International Symposium on Physical Design*. 161–168.

[22] Haoyu Yang, Jing Su, Yi Zou, Yuzhe Ma, Bei Yu, and Evangeline FY Young. 2018. Layout hotspot detection with feature tensor generation and deep biased learning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 38, 6 (2018), 1175–1187.

[23] Haoyu Yang, Shifan Zhang, Kang Liu, Siting Liu, Benjamin Tan, Ramesh Karri, Siddharth Garg, Bei Yu, and Evangeline FY Young. 2021. Attacking a CNN-based Layout Hotspot Detector Using Group Gradient Method. In *2021 26th Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 885–891.

[24] Cunxi Yu, Houping Xiao, and Giovanni De Micheli. 2018. Developing synthesis flows without human knowledge. In *Proceedings of the 55th Annual Design Automation Conference*. 1–6.