LithoROC: Lithography Hotspot Detection with Explicit ROC Optimization

Wei Ye
ECE Department, UT Austin
weiye@utexas.edu

Yibo Lin
ECE Department, UT Austin
yibolin@utexas.edu

Meng Li
ECE Department, UT Austin
meng_li@utexas.edu

Qiang Liu
CS Department, UT Austin
lqiang@cs.utexas.edu

David Z. Pan
ECE Department, UT Austin
dpan@ece.utexas.edu

ABSTRACT

As modern integrated circuits scale up with escalating complexity of layout design patterns, lithography hotspot detection, a key stage of physical verification to ensure layout finishing and design closure, has raised a higher demand on its efficiency and accuracy. Among all the hotspot detection approaches, machine learning distinguishes itself for achieving high accuracy while maintaining low false alarms. However, due to the class imbalance problem, the conventional practice which uses the accuracy and false alarm metrics to evaluate different machine learning models is becoming less effective. In this work, we propose the use of the area under the ROC curve (AUC), which provides a more holistic measure for imbalanced datasets compared with the previous methods. To systematically handle class imbalance, we further propose the surrogate loss functions for direct AUC maximization as a substitute for the conventional cross-entropy loss. Experimental results demonstrate that the new surrogate loss functions are promising to outperform the cross-entropy loss when applied to the state-of-the-art neural network model for hotspot detection.

1 INTRODUCTION

With the rapid shrinking of semiconductor process technology nodes, there is a widening gap between design demands and manufacturing capabilities posed by the current mainstream 193nm lithography. Due to the complexity of lithography systems and process variation, the layout patterns that are hard to print become lithography hotspots. Although numerous design for manufacturability techniques have been proposed to improve manufacturing yield, lithography hotspots still exist and need to be identified and eliminated during physical verification. For the purpose of yield improvement, efficient and accurate lithography hotspot detection is desired for layout finishing and design closure in the physical verification stage.

Existing hotspot detection methods mainly fall into three categories: lithography simulation, pattern matching, and machine learning techniques. Conventional lithography simulation locates lithography hotspots using complicated lithography models. Problematic patterns are captured accurately through full-chip simulations; however, it is associated with an expensive computational cost [1]. To this end, pattern matching and machine learning based techniques have been proposed for quick and accurate detection of hotspots. Pattern matching is a direct and fast method for hotspot detection. It forms a predefined library of hotspot layout patterns, and then any new pattern is compared with the patterns in the library [2, 3]. There are some extensions that use fuzzy pattern matching to increase the coverage of the library [4–6]. However, pattern matching, including fuzzy pattern matching, is ill-equipped to handle never-before-seen hotspot patterns.

In contrast, machine learning approaches have demonstrated good generalization capability to recognize unseen hotspot patterns [7–16]. These methods generally perform one-time training on a labeled dataset to build a machine learning model which learns the internal relationships between layout patterns. In order to enhance model scalability and get around spatial information loss induced from feature representation, deep learning techniques have been actively explored to further improve detection accuracy [17–20]. For these methods, the main target is to improve the accuracy of the classifiers while reducing false alarms. Usually, accuracy (i.e., true positive rate) is a major concern at the expense of tolerating a small number of false alarms, as missing any hotspot may result in significant yield degradation.

One special characteristic of lithography hotspot detection tasks is the imbalance in the layout datasets. Despite the fact that the lithography defects are critical, their relative number is significantly small across the whole chip after various resolution enhancement techniques are applied. Ideally, we would like to have a model with a high true positive rate (TPR) and a low false positive rate (FPR), but in real-world scenarios, there is always a trade-off between the two metrics. Assume there are two classifiers at hand. The first classifier successfully detects more hotspots than the second classifier, but it also generates significantly more false alarms. It is hard to conclude which one is better because we cannot tolerate such a high number of non-hotspot clips falsely identified as hotspots. It is a waste of time and efforts to fix those safe clips. A robust performance evaluation and model selection for imbalanced learning problems have been often accomplished with the support of the receiver operating characteristic (ROC) curve which represents the relationship between the true positive rate and the false positive

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

ACM ISBN 978-1-4503-6007-4/19/01…$15.00
https://doi.org/10.1145/3287624.3288746

rate of a family of classifiers resulted from different decision thresholds [21]. Hence, the area under the ROC curve (AUC) is a more proper model evaluation criterion in the sense of being a global metric for all thresholds regardless of class prior probabilities.

Most existing methods still minimize misclassification error such as cross-entropy during training while using certain class balancing techniques. The most straightforward and common approach dealing with imbalance is the use of sampling methods. Undersampling and oversampling methods operate on the training data to improve its balance. Other techniques, including cost-sensitive learning and threshold moving, tackle the class imbalance on the level of the classifier and adjust training or inference algorithms. Since AUC has been widely used to measure performance for binary classification tasks especially on imbalanced datasets, the question then arises: is it possible to use AUC explicitly as the loss function in order to systematically handle the class imbalance problem?

In this work, we examine the effectiveness of directly optimizing a surrogate of AUC to boost the performance of neural network models when facing class imbalance. Our main contributions in the proposed LithoROC framework can be summarized as follows:

- We propose a ROC curve based measure for hotspot detection algorithms, which provides a more holistic view of imbalanced datasets than the conventional measure using accuracy and false alarm.
- We discuss multiple loss functions for neural network models to explicitly optimize the proposed new measure other than the conventional cross-entropy loss.
- Experimental results demonstrate that the new loss functions are promising to outperform the cross-entropy loss when applied to the state-of-the-art neural network model for hotspot detection [20].

The rest of this paper is organized as follows. Section 2 reviews the challenges in hotspot detection and gives the problem formulation. Section 3 provides a detailed explanation of the proposed approach. Section 4 demonstrates the effectiveness of our approaches with comprehensive results, followed by conclusion in Section 5.

2 PRELIMINARIES

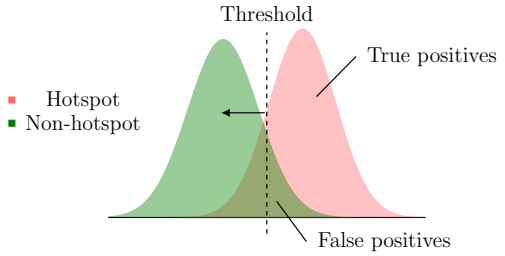
Lithography simulation computes aerial images and then generates the contours of printed patterns; therefore it can accurately detect lithography hotspots even at a high computational cost. The hotspot detection task to be solved by machine learning techniques can be formulated as a two-class image classification problem. In this way, machine learning based hotspot detection can bypass the lithography simulations by associating layout features with hotspot labels through a one-time training process. Then, the trained model can make efficient prediction for new layout clips. Figure 1 gives an example of hotspot and non-hotspot clips.

2.1 ROC Curve and AUC Score

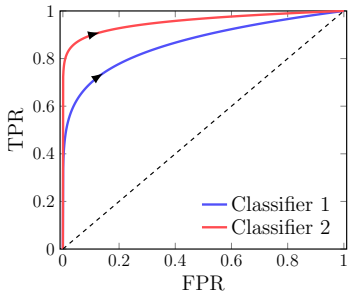
For binary classification tasks, in order to separate the positive class from the negative class, a decision threshold is usually defined to map the continuous predicted score given by the model to a binary category. For each setting of the decision threshold (Figure 2(a)), a pair of true-positive rate and false-positive rate values is obtained. By varying the decision threshold over the range [0, 1], the ROC

(a) Hotspot (b) Non-hotspot

Figure 1: Example of (a) lithography hotspot clip and (b) non-hotspot clip.



(a)



(b)

Figure 2: (a) An overlapping distribution of predicted scores for positive and negative samples and (b) the ROC curves of two example classifiers. As the threshold in (a) moves to the left, both FPR and TPR in (b) go up accordingly.

curve showing the relationship between true positive rate and the false positive rate can be obtained (Figure 2(b)). Moreover, as Figure 2(a) demonstrates, if the predicted score implies the classifier's belief that an sample belongs to the positive class, decreasing the decision threshold (e.g., moving the threshold to the left) will increase both true and false positive rates.

AUC is a threshold-independent metric which measures the fraction of times a positive instance is ranked higher than a negative one [21, 22]. Unlike single point metrics, the ROC curve compares classifier performance across the entire range of class distributions, and therefore, the AUC score is a general measure of classifier discrimination performance. Figure 2(b) presents two ROC curves. The closer the curve is pulled towards the upper left corner, the better is the classifier's ability to discriminate between the two classes. Therefore, in Figure 2(b), classifier 2 has a better performance than classifier 1.

2.2 Partial AUC Score

The AUC metric traces classifier performance across all thresholds. However, it may summarize over regions of the ROC curve in which one would never operate. For hotspot detection tasks, the primary goal is to detect all possible hotspots. Nevertheless, a practical classifier is not allowed to accomplish the goal at the expense of introducing too many false alarms; the time and money costs associated with fixing those false alarm hotspots render the classifier less favorable than the traditional simulation approach. In this case, our interest is to see the classifier's ability to detect hotspots in the region of the ROC curve corresponding only to acceptably low FPRs.

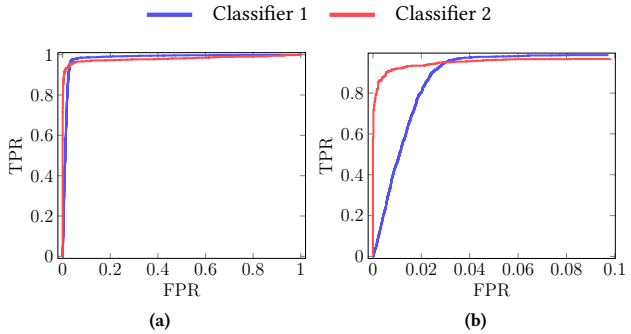


Figure 3: Comparison of the ROC curves over (a) the entire FPR range and (b) the FPR range of interest.

To elaborate on this, consider the two classifiers shown in Figure 3. Classifier 1 has better AUC than classifier 2 according to Figure 3(a). But if we zoom into the region of interest (e.g., FPR less than 2%) in Figure 3(b), classifier 2 has better overall TPR in this region and it outperforms classifier 1. Therefore, besides measuring the overall AUC score of the classifier, we look into the partial AUC defined in the following way [23, 24]:

$$\widetilde{\mathrm{AUC}}(t_0, t_1) = \int_{t_0}^{t_1} \mathrm{ROC}(t)\mathrm{d}t, \tag{1}$$

where the interval (t_0, t_1) denotes the false positive rate region of interest. We can further scale the partial AUC and derive the normalized partial AUC given by [23]

$$\mathrm{AUC}(t_0, t_1) = \frac{1}{t_1 - t_0} \int_{t_0}^{t_1} \mathrm{ROC}(t)\mathrm{d}t. \tag{2}$$

2.3 Handling Class Imbalance

Due to the fact that the lithography hotspots are critical, various resolution enhancement techniques are applied to significantly reduce their relative number. Therefore, when a grid scheme is used to extract images from the design, only a small number of images will encompass lithography hotspots while the majority will correspond to sites in the design with no defects. This poses a major challenge when formulating the task as a learning problem.

The class imbalance problem is encountered in many application domains. It has been established that in certain cases, class imbalance hinders the performance of standard classifiers [25], in terms of training convergence and generalization of the model. Sometimes the classifiers even achieve a low error rate by trivially predicting each sample to be negative when the dataset is biased towards the negative class.

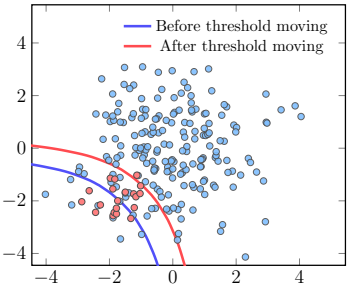


Figure 4: Example of threshold moving.

Various methods have been proposed to deal with the class imbalance problem. Among them, oversampling and undersampling alter the distribution of training data to make it more balanced. **Undersampling** removes samples from the majority class until all classes have the same amount of data. For example, one-side selection carefully identifies and removes redundant examples close to the boundary between classes [26]. A major disadvantage of undersampling is that it discards potentially useful training samples. Therefore, undersampling is rarely adopted for hotspot detection tasks because those training datasets are highly imbalanced but far from abundance. **Oversampling** is one of the most commonly used methods. It simply replicates randomly selected samples from minority classes, but this approach can increase the time necessary to build a classifier, and may even lead to overfitting [27]. Advanced sampling methods such as SMOTE [28] and its variant [29] create artificial examples by interpolating neighboring data points. In addition, cluster-based oversampling first clusters the dataset and then oversamples each cluster separately [30]. In this way, both between-class and within-class imbalances are reduced. Since the input data samples of hotspot detection tasks are images and optical sources are symmetric, [31, 32] augment the training data with rotation and flipping; besides, although general convolutional neural networks (CNNs) are not rotate invariant, data augmentation by rotation and flipping can help obtain some rotation invariance.

Cost sensitive learning assigns different cost to the misclassification of samples from different classes [33, 34]. For hotspot detection tasks, this is done by associating a greater cost with false negatives than with false positives. [35, 36] study cost sensitive learning of deep neural networks. [37] proposes a new loss function for neural network training to make the networks more sensitive to the minority class. To incorporate the cost sensitivity into neural networks, one can place a heavier penalty on misclassifying the minority class in the loss function such that minority class contributes more to the update of weights. And then, we can train the network by minimizing the misclassification cost instead of the standard loss function.

Threshold moving adjusts the decision threshold of a classifier to cope with the class imbalance problem. This approach is usually applied in the test phase. As demonstrated in Figure 4, it moves the threshold toward the majority class such that samples from the minority class become harder to be misclassified. For traditional machine learning methods, adjustment of the decision boundary is straightforward. For example, it can be done by shifting the bias in

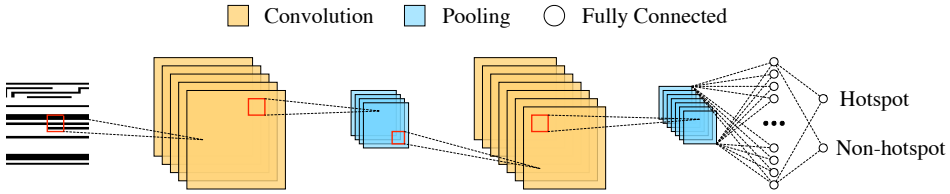


Figure 5: Example illustration of convolutional neural network architecture for hotspot detection.

a support vector machine (SVM) model. However, it is less practical to move the decision threshold directly when using neural network based classifiers because these networks tend to be overconfident in their prediction; the softmax outputs of the two neurons in last fully-connected layer shown in Figure 5 are usually very close to 1 and 0. As it is hard to control the appropriate shift amount, this method may take effect at cost of a large number of false alarms. Instead, [20] biases the ground truth for negative samples from 0 to ϵ during the training phase.

Other approaches explore different training methods specific to neural networks. [38] proposes a two-phase training method which first trains the network on the balanced set and then fine-tunes the output layers. The aforementioned approaches to tackle class imbalance either operate on training data or adjust training or inference methods. As we will demonstrate in the next section, AUC can be interpreted as a ranking measure; that is, the AUC is equal to the probability of ranking a random positive sample over a random negative sample. Therefore, orderings of data samples by the predicted probabilities is consistent even in the face of class imbalance. In this sense, both the shape of the ROC curve and AUC are insensitive to the class distribution. The question then arises, given that AUC is a robust measure of classification performances especially for imbalanced problems, is it possible to develop algorithms that directly optimize this metric during the training phase? In other words, can we optimize the ROC curve explicitly?

2.4 Problem Formulation

Traditionally, accuracy (i.e., true positive rate [39]) and the number of false alarms (i.e., false positives) are the two prevailing metrics used for detection evaluation. Hence, the traditional hotspot detection problem is usually defined as:

Problem 1 (Hotspot detection for accuracy optimization)**.** Given a set of layout clips consisting of hotspot and non-hotspot patterns, the object of hotspot detection is to train a classifier that maximizes the accuracy and minimizes the number of false alarms on the testing dataset.

As we demonstrated in Section 2.1, evaluation of hotspot detection models using accuracy and false alarms separately is not effective, because it is hard to find a good trade-off between the two metrics. Therefore, we propose to assess hotspot detection models using the holistic metric, AUC. Furthermore, the model is trained with the goal of optimizing the ROC curve in the form of maximizing the normalized partial AUC score.

Problem 2 (Hotspot detection for ROC optimization)**.** Given a set of layout clips consisting of hotspot and non-hotspot patterns, the object of hotspot detection is to train a classifier that maximizes the normalized partial AUC score on the testing dataset.

3 ROC OPTIMIZATION

In this section, we derive the AUC with dedicated loss functions for AUC optimization, and compare them with the cross entropy loss.

3.1 AUC Objective and Loss Functions

Given a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathbb{R}^d$ is i-th data sample in the feature space and $y_i \in \{-1, +1\}$ is the true class label of $\mathbf{x}_i$, we can further divide the dataset $\mathcal{D}$ into two sets: the set of positive samples $\mathcal{D}_+ = \{(\mathbf{x}_i^+, +1)\}_{i=1}^{N^+}$ and the set of negative samples $\mathcal{D}_- = \{(\mathbf{x}_i^-, -1)\}_{i=1}^{N^-}$, where $N+$ and $N-$ denote the number of positive and negative samples respectively, and $N = N_+ + N_-$. Let $f(\mathbf{x})$ denote the prediction model. It has been proven that AUC is equivalent to the Wilcoxon-Mann-Whitney (WMW) statistic test of ranks in the following sense [40–42]:

$$\text{AUC} = \frac{1}{N_+ N_-} \sum_{i=1}^{N_+} \sum_{j=1}^{N_-} I(f(\mathbf{x}_i^+) > f(\mathbf{x}_j^-)), \tag{3}$$

where $I(f(\mathbf{x}_i^+) > f(\mathbf{x}_j^-))$ is the indicator function given by

$$I(f(\mathbf{x}_i^+) > f(\mathbf{x}_j^-)) = \begin{cases} 1, & \text{if } f(\mathbf{x}_i^+) > f(\mathbf{x}_j^-), \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

AUC averages the score of a positive sample having a higher probability than a negative sample for all between-class pairs; it can also be viewed as the probability that a positive sample is ranked higher than a negative sample. This statistical interpretation led to the capability of computing AUC without building the ROC curve itself, by counting the number of positive-negative example misorderings in the ranking produced by classifier scores [43]. However, AUC defined in Equation (3) is a sum of indicator functions which is non-differentiable, to which gradient-based optimization methods cannot be applied. In order to make the problem tractable, it is necessary to apply convex relaxation to the AUC. By replacing $I(f(\mathbf{x}_i^+) > f(\mathbf{x}_j^-))$ in Equation (3) with pairwise convex surrogate loss $\Phi(f(\mathbf{x}_i^+) - f(\mathbf{x}_j^-))$, we can minimize the loss defined below as a way to maximize the AUC score:

$$\mathcal{L}_\Phi(f) = \frac{1}{N_+ N_-} \sum_{i=1}^{N_+} \sum_{j=1}^{N_-} \Phi(f(\mathbf{x}_i^+) - f(\mathbf{x}_j^-)). \tag{5}$$

Various surrogate loss functions can be chosen here. Let $z = f(\mathbf{x}_i^+) - f(\mathbf{x}_j^-)$, then the pairwise squared loss (PSL), one of the most commonly used surrogate loss functions, is given by [44, 45]

$$\Phi_{\text{PSL}}(z) = (1 - z)^2. \tag{6}$$

In [46, 47], pairwise hinge loss (PHL) is used as a surrogate function:

$$\Phi_{\text{PHL}}(z) = \max(1 - z, 0). \tag{7}$$

Similarly, [48] utilizes the pairwise logistic loss (PLL) to replace the indicator function:

$$\Phi_{\text{PLL}}(z) = \log(1 + \exp(-\beta z)). \tag{8}$$

[49] proposes the differentiable function given by the following expression as the surrogate loss:

$$\Phi_{\text{R}^*}(z) = \begin{cases} -(z-\gamma)^p, & \text{if } z > \gamma, \\ 0, & \text{otherwise}, \end{cases} \tag{9}$$

where $0 < \gamma \leq 1$ and $p > 1$, and suggests that $p = 2$ or 3 generally achieves the best results. Based on the observation that maximizing the objective with Φ in the form of Equation (9) is ineffective to maximize the WMW statistic because it focuses on maximizing the difference between $f(\mathbf{x}_i^+)$ and $f(\mathbf{x}_i^-)$ instead of moving more pairs of $f(\mathbf{x}_i^+)$ and $f(\mathbf{x}_i^-)$ to satisfy $f(\mathbf{x}_i^+) - f(\mathbf{x}_i^-) > \gamma$, the authors further propose another function,

$$\Phi_{\text{R}}(z) = \begin{cases} (-(z-\gamma))^p, & \text{if } z < \gamma, \\ 0, & \text{otherwise}. \end{cases} \tag{10}$$

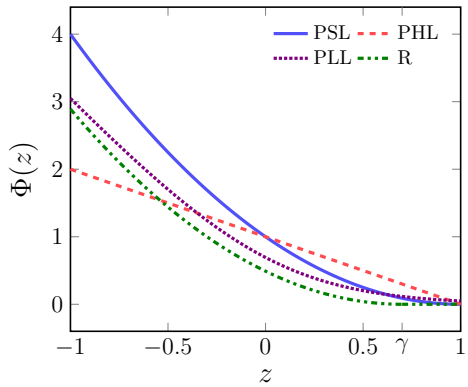


Figure 6: Comparison of the four surrogate functions, where $\beta = 3$ in PLL, and $\gamma = 0.7$ and $p = 2$ in R.

Figure 6 demonstrates the comparison of the four surrogate functions. One can notice that the curve of function R is flat in the region $[\gamma, 1]$, which differentiates it from other three curves. The key idea is, during the process of minimizing $\mathcal{L}$ in Equation (5), if a positive sample has a higher output than a negative sample by margin γ, this pair of samples will not contribute to the objective.

3.2 Comparison with Cross-Entropy Loss

Classifiers such as neural networks typically use cross-entropy (or log-loss) as the cost function. Cross-entropy (CE) loss for binary classifiers is defined as:

$$\text{CE} = -\frac{1}{N}\sum_{i=1}^{N} y_i \log f(\mathbf{x}_i) + (1 - y_i)\log(1 - f(\mathbf{x}_i)). \tag{11}$$

During the optimization process, CE in Equation (11) moves $f(\mathbf{x}_i^+)$ closer to 1 and $f(\mathbf{x}_i^-)$ to 0, while AUC in Equation (3) tries to force $f(\mathbf{x}_i^+) > f(\mathbf{x}_i^-)$. One might consider a weak relationship between CE and AUC, but in general the two objectives are quite different. Cross-entropy takes into account the uncertainty of the prediction based on how much the probability estimates vary from the actual labels, and it has been used when calibration is important [50]. Whereas, AUC is a rank statistic and is only affected by the ranking of the samples induced by the predicted probabilities. The order of the samples can be maintained while changing their probability values.

For the hotspot detection problems where positive labels are few but significant, we seek models that are able to predict positive classes more correctly. Table 1 displays an example dataset containing ten data samples with only two positive labels, and two models provide their predicted scores for each sample. As one can see, the two models only behave differently on sample 8 and 9. Model 1 correctly classifies sample 9 as positive, and model 2 correctly classifies sample 8 as negative. Model 1 is better than model 2 for hotspot detection tasks in the sense that it captures all the hotspots correctly even with one false alarm, while model 2 achieves zero false alarms but misses one hotspot.

Here we compare AUC with CE to see how differently they distinguish the two corresponding models when facing class imbalance. The CE scores for the two models are both 0.36. Clearly, cross-entropy believes the two models are performing equally. However, the AUC scores of the two models are 0.94 and 0.75 respectively, and hence, the AUC metric prefers model 1 over model 2. Cross-entropy fails in this case because the loss function in Equation (11) is symmetric and does not differentiate between classes. AUC captures the difference in classifying the imbalanced class and thus suits better for class imbalance.

Table 1: Comparison of cross-entropy and AUC for model selection on imbalanced dataset.

Sample No.	1	2	3	4	5	6	7	8	9	10
Label	0	0	0	0	0	0	0	0	1	1
Model 1	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.8	0.8	0.8
Model 2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.8

4 EXPERIMENTAL RESULTS

We implement the LithoROC framework in Python with the TensorFlow library [51]. The effectiveness of AUC as the optimization objective for neural networks is validated on the ICCAD 2012 CAD contest benchmark set [52]. Table 2 summarizes the benchmark information, the number of all the clips (#All) and the number of hotspot clips (#H) in the training set (Train) and testing set (Test). We configure the CNN architecture in a way similar to [20], which gives the state-from-the-art performance for hotspot detection. Each training process is repeated five times on the same dataset with different random seeds, and the average results on the testing set are shown in this section.

Table 3 demonstrates the impact of different loss functions on classification performance. The CNN model in [20] is updated at each step using the mini-batch gradient descent method which randomly picks a group of instances. To overcome the bias towards the majority class during the training process, [20] fixes the batch size to 32 and ensures that the number of positive samples and negative samples are the same in each mini-batch. In addition to following this mini-batch configuration, we explore the impact of imbalanced mini-batches by setting the the class ratio of positive

Table 2: ICCAD 2012 contest benchmark statistics [20].

Design	Train		Test	
	#All	#H	#All	#H
ICCAD1	439	99	4,095	226
ICCAD2	5,459	174	41,796	498
ICCAD3	5,552	909	48,141	1,808
ICCAD4	4,547	95	32,067	177
ICCAD5	2,742	26	19,368	41

samples per mini-batch to 0.1 and 0.4 respectively. To ensure the number of hotspots is not too small in each batch, the batch size is increased to 64.

There are four convex surrogate loss functions discussed in Section 3 and we choose the two representative loss functions, the pairwise square loss for AUC maximization (AUC-PSL) in Equation (6), and the R loss for AUC maximization (AUC-R) in Equation (10). We compare the two losses with the traditional cross-entropy loss (CE). Here we set $\gamma = 0.7$ and $p = 2$ in Equation (10). Table 3 shows the normalized partial AUC score on the testing data using different loss functions and different mini-batch configurations, where F(α) denote the the normalized partial AUC score given by Equation (2) over the FPR range $[0, \alpha]$. We consider $\alpha = 0.01$, 0.02 and 1, because the FPR reported in the recent literature is around 0.01 to 0.02 [20]. Reporting the results for $\alpha = 1$ is to show the difference in the AUC score and the partial AUC score. In Table 3, the state-of-the-art classifier from [20] uses CE as the objective function, and sets the batch size to 32 and the ratio of positive examples to 0.5. Its performance for hotspot detection is near saturation, but we can still observe utilizing AUC as the objective function for training the CNN model helps advance the performance of the model under low false positive rates, especially on design ICCAD3.

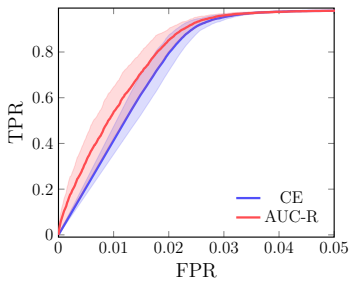
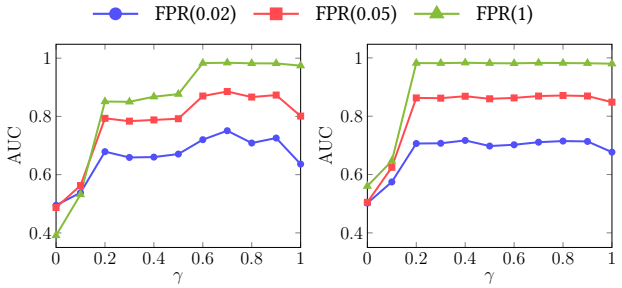


Figure 7: Comparison of ROC curves with different loss functions.

Figure 7 presents the ROC curves for design ICCAD3. The mean ROC curve of the five runs and the corresponding variance of the curve within ±1 standard deviation are shown. One can see that the objective function AUC-R generates a significantly better ROC curve than that of CE. A natural question is then how to choose the margin parameter γ in Equation (10). Figure 8 plots the AUC score versus the γ for various FPR ranges. To show the difference between curves, instead of using FPR(0.01), FPR(0.02), FPR(1), we use FPR(0.02), FPR(0.05), FPR(1), as the curves for FPR(0.01) and FPR(0.02) are very close. As noted in Figure 8, when γ increases from 0 to 0.5, the three AUC scores rise as well. That is because CNN is typically overconfident in its predictions in the sense that the output of the last fully-connected layer after softmax is very close to 0 or 1. In this way, it is over-simple for the between-class sample pairs to satisfy the constraint that a positive sample has as higher output than a negative sample by γ, which actually does not help guide the model to a good optimum. When γ is large enough, the AUC scores for the test data are relatively insensitive.



(a) Hotspot class ratio = 0.1 **(b) Hotspot class ratio = 0.4**

Figure 8: The normalized partial AUC scores at different γ on design ICCAD3 testing data.

5 CONCLUSION

In this work, we propose to use AUC as a robust measure of classifier discrimination performance for hotspot detection tasks. Different surrogate loss functions for AUC maximization are proposed to be used during training to systematically handle the class imbalance problem. Experimental results demonstrate that the new loss functions are promising to outperform the traditional cross-entropy loss when applied to the state-of-the-art neural network model for hotspot detection.

REFERENCES

[1] C. A. Mack, "Thirty years of lithography simulation," in *Optical Microlithography XVIII*, vol. 5754. International Society for Optics and Photonics, 2004, pp. 1–13.

[2] J. Xu, S. Sinha, and C. C. Chiang, "Accurate detection for process-hotspots with vias and incomplete specification," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2007, pp. 839–846.

[3] Y.-T. Yu, Y.-C. Chan, S. Sinha, I. H.-R. Jiang, and C. Chiang, "Accurate process-hotspot detection using critical design rule extraction," in *ACM/IEEE Design Automation Conference (DAC)*, 2012, pp. 1167–1172.

[4] S.-Y. Lin, J.-Y. Chen, J.-C. Li, W.-Y. Wen, and S.-C. Chang, "A novel fuzzy matching model for lithography hotspot detection," in *ACM/IEEE Design Automation Conference (DAC)*, 2013, pp. 68:1–68:6.

[5] 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 Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 33, no. 11, pp. 1671–1680, 2014.

[6] I. Nitta, Y. Kanazawa, T. Ishida, and K. Banno, "A fuzzy pattern matching method based on graph kernel for lithography hotspot detection," in *Design-Process-Technology Co-optimization for Manufacturability XI*, vol. 10148. International Society for Optics and Photonics, 2017.

[7] D. G. Drmanac, F. Liu, and L.-C. Wang, "Predicting variability in nanoscale lithography processes," in *ACM/IEEE Design Automation Conference (DAC)*, 2009, pp. 545–550.

[8] D. Ding, J. A. Torres, and D. Z. Pan, "High performance lithography hotspot detection with successively refined pattern identifications and machine learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 30, no. 11, pp. 1621–1634, 2011.

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

[10] 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," in *ACM/IEEE Design Automation Conference (DAC)*, 2013, pp. 671–676.

Table 3: Comparison between different loss functions for AUC objectives.

Loss	Batch		ICCAD1			ICCAD2			ICCAD3			ICCAD4			ICCAD5		
	Size	Ratio	F(0.01)	F(0.02)	F(1)	F(0.01)	F(0.02)	F(1)	F(0.01)	F(0.02)	F(1)	F(0.01)	F(0.02)	F(1)	F(0.01)	F(0.02)	F(1)
CE	32	0.5	51.0	52.0	88.9	96.6	97.8	99.7	59.9	69.7	98.2	89.9	92.1	98.8	93.0	94.2	98.3
	64	0.1	50.8	51.7	88.1	95.9	97.4	99.7	60.1	70.2	98.2	89.4	92.2	98.9	91.7	92.9	98.2
	64	0.4	50.9	51.8	88.6	95.8	97.3	99.8	61.1	71.7	98.3	90.0	92.7	98.8	92.6	93.7	98.2
AUC-PSL	32	0.5	51.2	52.5	89.2	93.7	95.8	99.6	59.3	68.7	98.0	**90.5**	92.5	98.7	**93.7**	**94.9**	98.6
	64	0.1	51.6	53.3	91.6	88.0	93.4	99.5	58.5	67.0	97.9	86.3	90.8	98.4	92.6	94.7	98.8
	64	0.4	51.6	53.2	91.8	95.6	97.1	99.7	59.9	69.8	98.1	90.2	92.1	98.3	92.3	93.4	98.5
AUC-R	32	0.5	52.9	**55.5**	91.1	96.4	97.7	99.7	60.6	71.0	98.2	**90.5**	**93.0**	98.9	93.4	94.8	98.1
	64	0.1	52.1	53.8	88.9	96.1	97.7	99.7	**64.4**	**74.4**	98.4	89.6	92.2	98.7	93.1	94.6	98.4
	64	0.4	**53.0**	55.1	90.2	**96.9**	**98.0**	99.7	60.1	71.1	98.3	90.0	92.4	98.9	93.1	94.4	98.4

[11] 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 *Proceedings of SPIE*, vol. 9427, 2015.

[12] 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 Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 34, no. 3, pp. 460–470, 2015.

[13] H. Zhang, B. Yu, and E. F. Y. Young, "Enabling online learning in lithography hotspot detection with information-theoretic feature optimization," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2016, pp. 47:1–47:8.

[14] Y. Tomioka, T. Matsunawa, C. Kodama, and S. Nojima, "Lithography hotspot detection by two-stage cascade classifier using histogram of oriented light propagation," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2017, pp. 81–86.

[15] H. Zhang, F. Zhu, H. Li, E. F. Y. Young, and B. Yu, "Bilinear lithography hotspot detection," in *ACM International Symposium on Physical Design (ISPD)*, 2017, pp. 7–14.

[16] J. W. Park, A. Torres, and X. Song, "Litho-aware machine learning for hotspot detection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 37, no. 7, pp. 1510–1514, 2018.

[17] T. Matsunawa, S. Nojima, and T. Kotani, "Automatic layout feature extraction for lithography hotspot detection based on deep neural network," in *SPIE Advanced Lithography*, vol. 9781, 2016.

[18] M. Shin and J.-H. Lee, "Accurate lithography hotspot detection using deep convolutional neural networks," *Journal of Micro/Nanolithography, MEMS, and MOEMS (JM3)*, vol. 15, no. 4, p. 043507, 2016.

[19] H. Yang, L. Luo, J. Su, C. Lin, and B. Yu, "Imbalance aware lithography hotspot detection: a deep learning approach," *Journal of Micro/Nanolithography, MEMS, and MOEMS (JM3)*, vol. 16, no. 3, p. 033504, 2017.

[20] 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 *ACM/IEEE Design Automation Conference (DAC)*, 2017, pp. 62:1–62:6.

[21] J. A. Swets and R. M. Pickett, *Evaluation of diagnostic systems: methods from signal detection theory*. New York : Academic Press, 1982.

[22] D. M. Green and J. A. Swets, *Signal detection theory and psychophysics*. New York : Wiley, 1966.

[23] D. K. McClish, "Analyzing a portion of the roc curve," *Medical Decision Making*, vol. 9, no. 3, pp. 190–195, 1989.

[24] L. E. Dodd and M. S. Pepe, "Partial auc estimation and regression," *Biometrics*, vol. 59, no. 3, pp. 614–623, 2003.

[25] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intelligent data analysis*, vol. 6, no. 5, pp. 429–449, 2002.

[26] M. Kubat and S. Matwin, "Addressing the curse of imbalanced training sets: One-sided selection," in *International Conference on Machine Learning (ICML)*, 1997, pp. 179–186.

[27] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, vol. 106, pp. 249–259, 2018.

[28] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[29] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-smote: a new over-sampling method in imbalanced data sets learning," in *International Conference on Intelligent Computing*, 2005, pp. 878–887.

[30] T. Jo and N. Japkowicz, "Class imbalances versus small disjuncts," *ACM Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 40–49, 2004.

[31] H. Yang, L. Luo, J. Su, C. Lin, and B. Yu, "Imbalance aware lithography hotspot detection: A deep learning approach," in *SPIE Advanced Lithography*, vol. 10148, 2017.

[32] Y. Lin, M. Li, Y. Watanabe, T. Kimura, T. Matsunawa, S. Nojima, and D. Z. Pan, "Data efficient lithography modeling with transfer learning and active data selection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2018.

[33] C. Elkan, "The foundations of cost-sensitive learning," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2001, pp. 973–978.

[34] G. M. Weiss, "Mining with rarity: a unifying framework," *ACM Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 7–19, 2004.

[35] Y.-A. Chung, H.-T. Lin, and S.-W. Yang, "Cost-aware pre-training for multiclass cost-sensitive deep learning," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2016, pp. 1411–1417.

[36] S. H. Khan, M. Hayat, M. Bennamoun, F. A. Sohel, and R. Togneri, "Cost-sensitive learning of deep feature representations from imbalanced data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 8, pp. 3573–3587, 2018.

[37] S. Wang, W. Liu, J. Wu, L. Cao, Q. Meng, and P. J. Kennedy, "Training deep neural networks on imbalanced data sets," in *International Joint Conference on Neural Networks (IJCNN)*, 2016, pp. 4368–4374.

[38] M. Havaei, A. Davy, D. Warde-Farley, A. Biard, A. Courville, Y. Bengio, C. Pal, P.-M. Jodoin, and H. Larochelle, "Brain tumor segmentation with deep neural networks," *Medical image analysis*, vol. 35, pp. 18–31, 2017.

[39] C. M. Bishop *et al.*, *Pattern Recognition and Machine Learning*. Springer New York, 2006, vol. 4, no. 4.

[40] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *Ann. Math. Statist.*, vol. 18, no. 1, pp. 50–60, 1947.

[41] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

[42] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under a receiver operating characteristic (roc) curve." *Radiology*, vol. 143, no. 1, pp. 29–36, 1982.

[43] S. Wu, P. Flach, and C. Ferri, "An improved model selection heuristic for auc," in *European Conference on Machine Learning*, 2007, pp. 478–489.

[44] W. Gao, R. Jin, S. Zhu, and Z.-H. Zhou, "One-pass auc optimization," in *International Conference on Machine Learning (ICML)*, 2013, pp. III–906–III–914.

[45] Y. Ding, P. Zhao, S. C. H. Hoi, and Y.-S. Ong, "An adaptive gradient method for online auc maximization," in *AAAI Conference on Artificial Intelligence*, 2015, pp. 2568–2574.

[46] H. Steck, "Hinge rank loss and the area under the roc curve," in *European Conference on Machine Learning*. Springer Berlin Heidelberg, 2007, pp. 347–358.

[47] P. Zhao, S. C. H. Hoi, R. Jin, and T. Yang, "Online auc maximization," in *International Conference on Machine Learning (ICML)*, 2011, pp. 233–240.

[48] C. Rudin and R. E. Schapire, "Margin-based ranking and an equivalence between adaboost and rankboost," *Journal of Machine Learning Research*, vol. 10, no. Oct, pp. 2193–2232, 2009.

[49] L. Yan, R. H. Dodier, M. Mozer, and R. H. Wolniewicz, "Optimizing classifier performance via an approximation to the wilcoxon-mann-whitney statistic," in *International Conference on Machine Learning (ICML)*, 2003, pp. 848–855.

[50] I. J. Good, "Rational decisions," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 107–114, 1952.

[51] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: a system for large-scale machine learning." in *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, vol. 16, 2016, pp. 265–283.

[52] A. J. Torres, "ICCAD-2012 CAD contest in fuzzy pattern matching for physical verification and benchmark suite," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2012.