This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3021663, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

Faster Region-based Hotspot Detection

Ran Chen, Wei Zhong, Haoyu Yang, Hao Geng, Fan Yang, Xuan Zeng and Bei Yu

***Abstract*—As the circuit feature size continuously shrinks down, hotspot detection has become a more challenging problem in modern DFM flows. Developed deep learning techniques have recently shown their superiorities on hotspot detection tasks. However, existing hotspot detectors can only handle defect detection from one small layout clip each time, thus may be very time-consuming when dealing with a large full-chip layout. In this paper, we develop a new end-to-end framework that can detect multiple hotspots in a large region at a time and promise a better hotspot detection performance. We design a joint auto-encoder and inception module for efficient feature extraction. A two-stage classification and regression framework is designed to detect hotspot with progressive accurate localization, which provides a promising performance improvement. Experimental results show that our framework enables a significant speed improvement over existing methods with higher accuracy and fewer false alarms.**

I. Introduction

With the development of the semiconductor industry, transistor feature size shrinks rapidly, which significantly challenges manufacturing yield. For instance, low-fidelity pattern on wafer (a.k.a. "hotspot") is one of the emerging issues in the manufacturing [1], [2]. To ensure the printability of layout designs, an efficient and accurate hotspot detector is indispensable. Currently, there are three main classes of methods: lithography simulation, pattern matching and machine learning. By using complicated lithography models to identify hotspots, lithography simulation [3], [4] is accurate but extremely time-consuming. High performance clusters with amounts of nodes are needed in the whole simulation flow and several days are required to complete it.

As good replacements of simulation-based methods, pattern matching and machine learning-based techniques are proposed to accelerate the hotspot detection flow while the detection accuracy is attained as much as possible.

Pattern matching is to set up a collection of hotspot layout patterns to identify any matched patterns in a new design as hotspots [5]–[9]. For example, in [7], critical topological features of hotspots are extracted as design rules in design rule checking to locate the hotspot positions. To handle partially variant layout patterns from the pre-defined hotspots, Wen *et al*. [8] proposed a fuzzy matching model which integrates both pattern-matching and machine-learning techniques to dynamically tune the fuzzy region around a known hotspot. Although the pattern matching overcame the runtime issue, this approach, including fuzzy pattern matching, cannot give a confident result on unseen hotspot patterns.

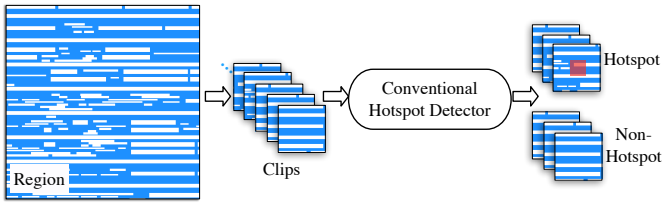


Fig. 1 The conventional hotspot detection flow.

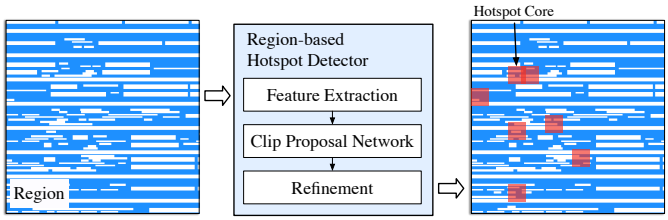


Fig. 2 The proposed region-based hotspot detection flow.

Machine learning-based methods have shown the capability to offer accurate solutions to both known and unknown hotspot patterns with generalized feature extractors [10]–[21]. A learning model is usually trained by features which are extracted from a batch of labeled data and then conducts hotspot prediction on new layout patterns efficiently. Ding *et al*. [12] exploited a meta-classifier which combines pattern matching and machine learning methods into a unified framework. In [15], a detection flow based on critical feature extraction and PCA-SVM classifier is proposed. To update the learning model with newly detected and verified layout patterns, Zhang *et al*. [17] investigated a classifier based on smooth boosting and optimized concentric circle sampling feature extractor. Recently, Ye *et al*. [21] pointed out the uncertainty problem in hotspot detection and presented a Gaussian process assurance to provide confidence in each prediction. Conventional machine learning approaches achieve good performance, but they are limited to manually-crafted feature extractors. Besides, these approaches are challenged by scalability requirements for printability estimation of a large scale layout.

Convolutional neural networks (CNNs) have become a powerful technique to improve hotspot detection performance

The preliminary version has been presented at the ACM/IEEE Design Automation Conference (DAC) in 2019. This work is supported in part by The Research Grants Council of Hong Kong SAR (Project No. CUHK24209017, CUHK14209420), National Key Research and Development Program of China (2016YFB0201304), National Natural Science Foundation of China (NSFC) research projects 61574046, 61774045, and NVIDIA. (*Corresponding author: Bei Yu.*)

R. Chen, H. Yang, H. Geng and B. Yu are with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, NT, Hong Kong SAR.

W. Zhong is with International School of Information Science & Engineering, Dalian University of Technology, China.

F. Yang and X. Zeng are with State Key Lab of ASIC & System, Microelectronics Department, Fudan University, China.

[22]–[31], thanks to its non-linearity and multi-level feature extraction in an automatic manner. For example, Yang *et al*. [24] investigated a deep CNN which considered the data unbalanced issue and achieved high classification accuracy. Additionally, they designed a biased learning technique for an unbalanced dataset and applied discrete-cosine transformation (DCT) to give proper feature expression [25]. To handle the scenario that labeled data are limited, a semi-supervised neural network is built in [29]. In [30], Jiang *et al*. proposed a binarized neural network to further enhance the performance of the detector.

In literature, however, hotspot detectors only work on small clips extracted from a whole chip layout and can only detect one hotspot location at a time that occurs at a center (i.e. core in [32]) of each clip. Fig. 1 illustrates a conventional hotspot detection scheme, which requires repeatedly scanning overlapping regions of a full chip design. Therefore, it could be a waste of computational resources and time-consuming when facing with extremely large layouts. To solve this problem, [33] proposed a new faster region-based hotspot detection framework, which can mark multiple hotspot locations within a region that is much larger than a clip applied in previous works, as shown in Fig. 2. To the best of our knowledge, [33] is the first art to design a hotspot detector on detecting multiple processes weak points within very large scale layout clips in one step feed-forward detection. The framework contains a regression and classification multi-task flow which guide to higher accuracy, higher detection speed and lower false alarm penalty. However, there still exist some defects in our design for region-based hotspot detector. For example, due to the single-scale description of layouts, the encoder-decoder structure limits the feature expression of our extractor. Additionally, the regression loss in [33] handles the coordinates individually where the geometric constraint is not considered. Consequently, multi-branch design for encoder-decoder and IoU regularizer are proposed to enhance the preliminary region-based hotspot detector. The main contributions of this paper are listed as follows:

- We construct a deep neural network specifically for region-based hotspot detection task and our network framework can be efficiently trained end-to-end.
- A clip proposal network and a refinement stage are built to further improve accuracy and reduce false alarm.
- We apply a novel classification and regression strategy to reduce the detection region and make the multiple hotspot detection become realizable in large scales.
- Multi-branch design for encoder-decoder and IoU regularization is introduced to further strengthen the proposed region-based hotspot detector.
- Experimental results show that our proposed framework has great advantages over the state-of-the-art detectors. It can achieve 7.40% accuracy improvement and 13× speedup on average.

The rest of the paper is organized as follows. Section II introduces basic concepts and gives problem formulation. Section III discusses the details of the proposed end-to-end neural framework. Section IV introduces the techniques to raise the performance of the proposed detector. Section V lists experimental comparisons with state-of-the-art methods, followed by conclusion in Section VI.

II. Preliminaries

Due to the manufacturing process variation, designed layout patterns stochastically cause defects on wafers during the lithographic process. These sensitive patterns may cause reduction of manufacturing yield or even potential circuit failures. Layout patterns that are sensitive to process variations are defined as *hotspots*. We also define a *hotspot clip* as a clip that contains at least one hotspot at its core region [32]. Here the core region is the middle area in the clip. In this paper, the following definitions and metrics are used to evaluate the performance of a hotspot detector.

Definition 1 (Accuracy)**.** *The ratio between the number of correctly detected hotspots and the number of ground truth hotspots.*

Definition 2 (False Alarm)**.** *The number of non-hotspot clips that are detected as hotspots by the classifier.*

Definition 3 (F1 score)**.** *The weighted harmonic mean of the test's precision and recall. The score is calculated according to*

$$F1 = \left(\frac{2}{recall^{-1} + precision^{-1}} \right). \tag{1}$$

It should be noted that the accuracy is also equivalent to the *true positive rate* and the false alarm corresponds to the number of *false positives*. Because a good hotspot detector aims to recognize as many real hotspots as possible and avoids incorrect predictions on non-hotspot patterns, with the evaluation metrics above, we define the region-based hotspot detection (R-HSD) problem as follows.

Problem 1 (R-HSD)**.** *Given a layout region that consists of hotspot and non-hotspot patterns, the objective of region-based hotspot detection is training a model to locate and classify all the hotspot and the non-hotspot within the region, such that the detection accuracy is maximized with minimum false alarm penalty.*

III. R-HSD Neural Network

Our proposed region-based hotspot detection (R-HSD) neural network, as illustrated in Fig. 2, is composed of three steps: (1) feature extraction, (2) clip proposal network, and (3) refinement. In this section, we will discuss each step with details. At first glance, the R-HSD problem is similar to object detection problem, which is a hot topic in computer vision domain recently. In object detection problem, objects with different shapes, types and patterns are the instances to be detected. However, as we will discuss, there is a gap between hotspot detection and object detection, e.g. the hotspot pattern features are quite different from the objects in real scenes, thus typical strategies and framework utilized in object detection cannot be applied here directly.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3021663, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

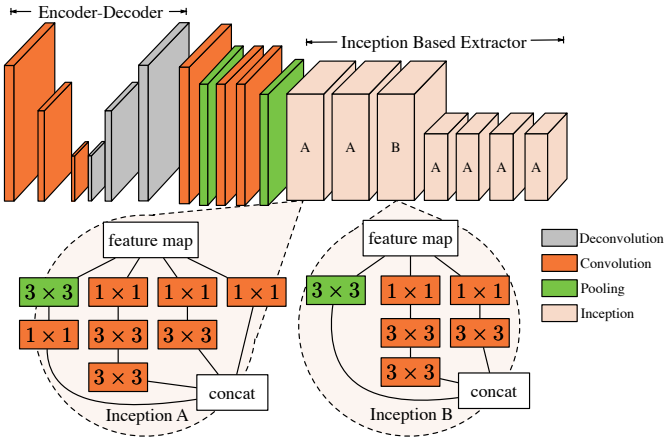


Fig. 3 The tensor structure of feature extractor.

A. Feature Extraction

Because of wide variations between traditional objects in real scenarios and VLSI layout patterns, it is extremely crucial to design an appropriate feature extractor in our neural network framework. Our feature extractor aims at transforming original layout features non-linearly while reducing computation overhead. Besides, we also tend to enrich the feature diversity with fewer parameters. Based on these two major principles, a feature extractor based on encoder-decoder structure and inception-based modules for efficient extraction is designed. Three convolution layers and two max-pooling layers connect the encoder-decoder structure and inception-based modules. This connection is applied to compress the feature map size from 224×224 to 56×56 which can bring speed-up at the training stage and inference stage.

Yang *et al*. [24] successfully applied DCT to manually extract layout pattern features. Although DCT keeps the spatial information, inevitably, this manual design of the feature expression ignores some key features thus may not give a comprehensive expression. Furthermore, the processing of the DCT is very time-consuming. Compared to the manually rule-based DCT, our proposed feature extractor can transform the origin layout into a network-compatible expression automatically. As the feature extractor is a part of the whole convolutional neural network, the training procedure is more flexible and efficient.

The convolutional network structure designed by [24] performs well feature extraction, but its structure is too simple thus is limited to the single clip hotspot detection problem. The naive replacement on DCT without redesign on further extractor is not available to the region-based task. There are two metrics for us to think about how to design a new structure in our work. One is going deeper with more layers, while the other is going wider with multiple branches.

1) Encoder-Decoder Structure: The encoder-decoder structure has been successfully applied to many computer vision tasks, including object detection [34]–[36]. The vanilla encoder consists of several convolution layers and the decoder includes the same number of deconvolution layers. The encoder gradually extracts the features from the origin image space to high dimensions latent space by increasing the number of the convolution kernels, then the decoder gradually downsamples the features from high dimensions to origin image space with the symmetrical kernel settings.

Convolution Layer. The convolution layer is the major part of the convolution neural network which has been widely used. The operation between tensor and kernel can be expressed as:

$$\mathbf{F} \otimes \mathbf{K}(j,k) = \sum_{i=1}^{c} \sum_{m_0=1}^{m} \sum_{n_0=1}^{m} \mathbf{F}(i, j-m_0, k-n_0)\mathbf{K}(m_0, n_0), \tag{2}$$

with tensor $\mathbf{F} \in \mathbb{R}^{c \times p \times p}$ and kernel $\mathbf{K} \in \mathbb{R}^{c \times m \times m}$.

Deconvolution Layer. In contrast to convolutional layers, deconvolution layers do the inverse operation which maps the single input feature point to multiple outputs, which can be considered as a feature generation. The expression can be written as:

$$\mathbf{T} \otimes \mathbf{K}(j,k) = \sum_{i=1}^{c} \sum_{m_0=1}^{m} \sum_{n_0=1}^{m} \mathbf{T}(i, j-m_0, k-n_0) \mathbf{K}(m-m_0, n-n_0), \tag{3}$$

where $\mathbf{T} \in \mathbb{R}^{c \times n \times n}$ is the tensor $\mathbf{F} \in \mathbb{R}^{c \times p \times p}$ padded with zero and $n = (m-1) \times 2 + p$, kernel $\mathbf{K} \in \mathbb{R}^{c \times m \times m}$. Here padding size is the number of pixels we fill on the border of the origin feature maps. In our experiments, the size of a padded feature map is equal to the size of output. Kernel size is the size of the deconvolution kernel. We use 3×3 kernel size which is same as the encoder part. During training, the feature map of the deconvolution layer is updated with the back-propagation.

2) Inception-based Structure: Empirically, a deeper neural network can give a much more robust feature expression and get higher accuracy compared to a shallow structure as it increases the model complexity. However, deeper networks are prone to overfitting, and gradient vanishing attaches to it. Even worse, it brings sacrifice on speed at both the inference stage and the training stage. Another point we need to take into consideration is that features extracted from the encoder-decoder structure are still in low dimension space. In other words, more convolution kernels are needed. Additionally, salient parts in images (i.e. hotspots in our case) may have pretty large variations in locations and sizes. According to these issues, we propose an inception-based structure. The following three points are the main rules of our design:

- Increase the number of filters in width at each stage. For each stage, multiple filters do the convolution operation with different convolution kernel size and then concatenate them in channel direction as feature fusion.
- Prune the depth of the output channel for each stage.
- Downsample the feature map size in height and width direction.

With the above rules, the inception structure [37] can take a good balance between the accuracy and the time. The blobs showed in Fig. 3 are what we apply in our framework. We construct the module A with the operation stride one and four

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3021663, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

branches. The aim of module A is to extract multiple features without downsampling the feature map. The operation stride of each layer in module B is two. Here the stride denotes the convolution operation step of kernels on feature maps, the larger strides can decrease the tensor size and reduce the number of operations in subsequent layers. Note that Module B has the same design principles as Module A, the bonus of Module B is to reduce the spatial size of features. We only use one Module B here, because the feature map size should not be too small, while the low dimension of feature expression in final layers may bring negative affects to the final result. The output feature size of final module A in Fig. 3 is 14×14, which is the input feature map of Clip Proposal Network in Fig. 4. It can be seen that 1×1 convolution kernel has been applied in both modules. The exploited 1×1 convolution kernel with low channel numbers offers a channel-wise pooling, which brings the dimension reduction by decreasing the number of feature maps whilst retaining their salient features. This technique successfully controls the number of the parameters and convolutional operations, and thus reduces the computational overhead.

In summary, the inception structure brings more abundant feature expressions, which gives the network the ability to do the kernel selection with no operation penalty.

B. Clip Proposal Network

Given the extracted features, a clip proposal network is developed here to detect potential hotspot clips. For both feature maps and convolutional filters, the tensor structures of the clip proposal network are illustrated in Fig. 4. Per preliminary experiments, clips with single aspect ratio and scale (e.g. square equal to the ground truth) may lead to bad performance. Therefore, for each pixel in a feature map, a group of 12 clips with different aspect ratios is generated.

The network is split into two branches: one is for classification and the other is for regression. In classification branch, for each clip, a prediction of hotspot and non-hotspot is calculated through softmax function. The basic sampling strategy to train the classifier is that the clips highly overlapped with ground-truth are regarded as positive samples and the ones with lower overlaps are considered as negative samples. Apparently, it needs some tweaks and compromises to separate hotspots and non-hotspots. In regression branch, the location and the shape of each clip are determined by a vector $[x, y, w, h]$, where x and y refer to the location of a clip center while w and h respectively record the width and the height of a clip. One criterion for the regressor during training is that clips labeled as non-hotspots are not fed into the regression branch since there are no ground-truth clips for them. The output of our clip proposal network is a bunch of proposals that will be examined by the above-mentioned classifier and regressor to eventually check the occurrence of hotspots. More precisely, it predicts the possibility of a clip being a hotspot or not, and refines the clip.

1) Clip Pruning: While the number of clips is extremely large during training, high-quality clips should be reserved to train the classifier and the low-quality clips which have

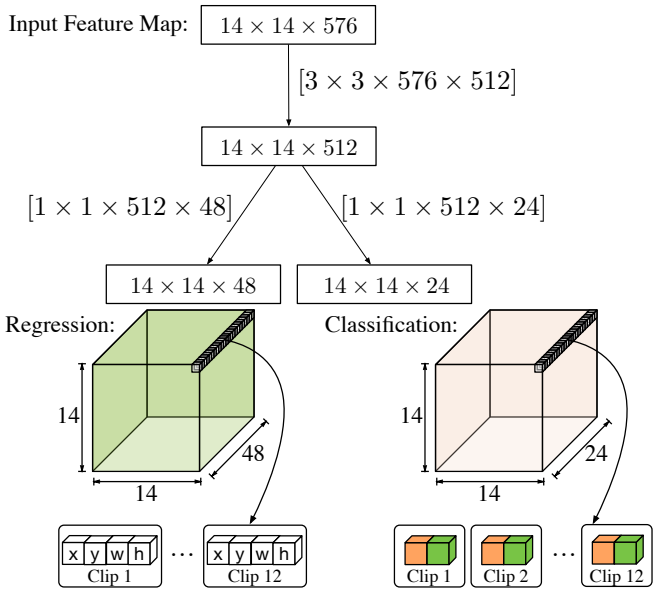


Fig. 4 The kernel work flow of clip proposal network.

medium intersection area to the ground truth should be removed as they are the noises to the classifier. For the clip regression task, it is not reasonable to consider linear regression on these clips with large offset to the ground truth clips. To overcome this problem, we consider automatic clip pruning in our neural network.

We first define intersection of union (IoU) as follows:

$$\text{IoU} = \frac{clip_{groundtruth} \bigcap clip_{generated}}{clip_{groundtruth} \bigcup clip_{generated}}. \tag{4}$$

Then the following clip pruning rules are established:

- A clip's IoU with ground truth clip higher than 0.7 should be reserved as a positive sample;
- The clip's IoU with any ground truth highest score should be reserved as a positive sample;
- A clip's IoU with ground truth clip lower than 0.3 should be reserved as a negative sample;
- Rest of clips do no contribution to the network training.

2) Hotspot Non-Maximum Suppression: After the classification and regression, the distance between some neighbor hotspots is quite close to each other, there exists a set of overlapped clips which have the same regression target. To avoid these redundant calculations at training and inference stages, we develop a hotspot non-maximum suppression (H-NMS) strategy to remove these clips. The H-NMS strategy is shown in Algorithm 1.

The elements of $sorted\_ws$ are arranged in descending order according to the classification score (line 1). `Centre_IoU` is a function returning the IoU score which focus on overlap of cores (line 7). If the IoU is larger than the threshold, we remove the clip with the lower score from the list (lines 8–10). The removed clips will not contribute to further operation. Note that applying H-NMS with higher threshold could lead to a drop on accuracy due to aggressive suppression, while suppressing nearby clips with a lower

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3021663, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

Algorithm 1 hotspot non-maximum suppression

```
sorted_ws ← sorted clip set;
k ← size of clip set;
for i ← 1, 2, ..., k do
    current_w ← sorted_ws[i];
    for j ← i, i + 1, ..., k do
        compared_w ← sorted_ws[j];
        Overlap ← Centre_IoU(current_w, compared_w);
        if Overlap > threshold then
            Remove compared_w; k ← k − 1;
        end if
    end for
end for
return sorted_ws;
```

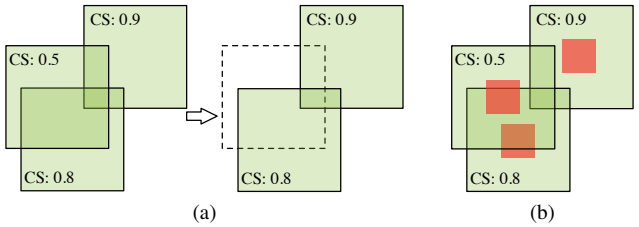


Fig. 5 Examples of (a) conventional non-maximum suppression, and (b) the proposed hotspot non-maximum suppression.

threshold would increase the false alarm since the less confident proposals are less likely to be suppressed. To some extents, the threshold value makes a tradeoff between two conflicting needs. In our experiment, the IoU threshold value is set to 0.7.

Compared to conventional non-maximum suppression method, our proposed method takes advantage of the structural relation between the core region and clips, thus can avoid error dropout during the training. More importantly, the H-NMS does not harm the ultimate detection accuracy but substantially reduces the number of proposals. An example is shown in Fig. 5, the clip with 0.5 classification score (CS) is removed in conventional methods, while saved in our method if we consider the core within each clip.

C. Refinement

After the prediction of the first classification and regression in the clip proposal network stage, we get a rough prediction on hotspot localization and filtered region which is classified as non-hotspot. While the greedy method of clip filtering cannot guarantee all the reserved clips are classified correctly, the false alarm may be too high. To bring a robust detection with lower false alarms, we further construct refinement stage in the whole neural network, which includes a region of interests (RoI) pooling layer, three inception modules, as well as another classification and regression. The structure of Refinement is shown in Fig. 6.

RoI Pooling. The coordinates of each clip are the actual location from the original input image. We scale down the coordinates to conform with the spatial extent of the

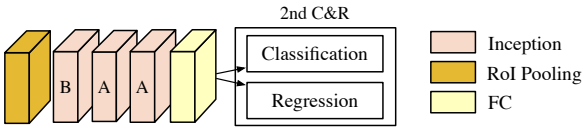


Fig. 6 Tensor structure of Refinement.

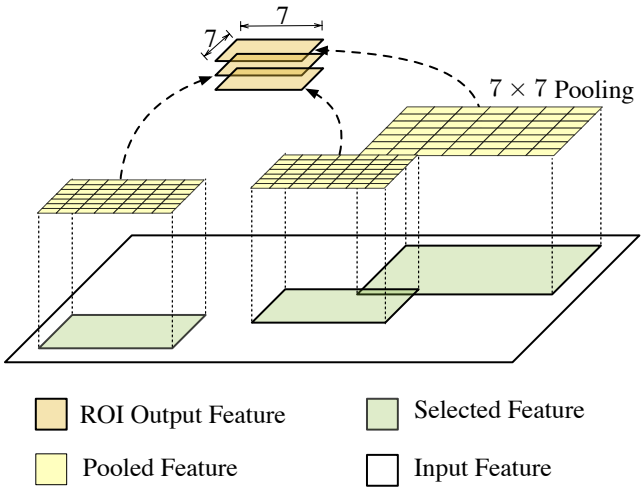


Fig. 7 Visualized 7×7 RoI pooling.

last feature map before the refinement. In traditional image processing, the most common ways to resize the image are cropping and warping. The crop method cuts the pattern boundary to fix the target size which leads to information loss. The warping operation will change the shape of origin features. Here we apply region of interests (RoI) pooling to transform the selected feature region $h \times w$ to a fixed spatial size of $H \times W$ (H and W are the hyper-parameters, and we use 7×7 in this work). For each pooled feature region $\lfloor h/H \times w/W \rfloor$, the max-pooling is applied independently. More specifically, the scaling is done by the following two steps. Firstly, each clip proposal is divided into equal-sized sections, the number of which is the same as the dimension of the output. Afterward, the largest value is found and output in each section. Consequently, the dimension of the output does not depend on the size of the input feature map nor on the size of the clips. On the contrary, it is determined solely by the number of sections we divide the proposal into (i.e. $H \times W$ in our algorithm). The RoI pooling transforms clips with different sizes into a fixed size which reserves the whole feature information and makes further hotspot classification and regression feasible. It bridges the two stages of our region-based hotspot detector, thus training object detection systems in an end-to-end manner is allowed. Additionally, it also benefits the processing speed in both the training and testing stage. Fig. 7 gives an example of RoI pooling operations.

Besides classification and regression in clip proposal network, here additional classification and regression are designed to finetune the clip location and give a more reliable classification result. At this stage, most non-hotspot clips have been removed, thus two stage of hotspot classification can efficiently reduce false alarm. Fig. 8 illustrates an example of

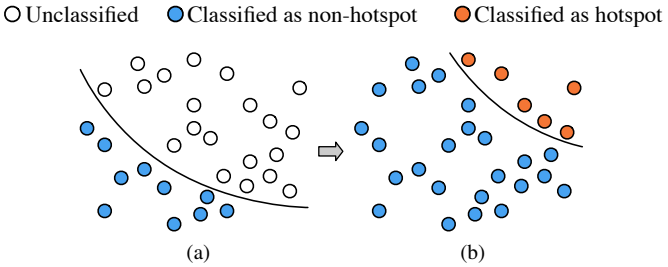


Fig. 8 (a) 1st hotspot classification in clip proposal network; (b) The labelled hotspots are fed into 2nd hotspot classification in refinement stage to reduce false alarm.

the two stage hotspot classification.

D. Loss Function Design

We design a multi-task loss function called classification and regression (C&R) to calibrate our model. As shown in Fig. 4 and Fig. 6, C&R is applied both in the clip proposal network stage and refinement stage. The input tensors of 1st C&R are boxes in Fig. 4. W, H and C are width, height and channel respectively. The probability score of the hotspot, non-hotspot and prediction of clip coordinates are grouped in the channel direction. As aforementioned, x, y are the coordinates of the hotspot, which means the centre of the clip area. w, h are the width and height of the clip. In 2nd C&R, the tensor flow of the classification and regression is the same as [38] using fully-connected layers.

In the task of region-based hotspot detection, h_i is the predicted probability of clip i being a hotspot. h_i' is the ground truth of clip i, which equals to 1 if a hotspot is located in the centre and 0 vice versa. $\boldsymbol{l}_i = \{l_x, l_y, l_w, l_h\} \in \mathbb{R}^4$ and $\boldsymbol{l}_i' = \{l_x', l_y', l_w', l_h'\} \in \mathbb{R}^4$ are assigned as coordinates of clips with index i representing the encoded coordinates of the prediction and ground truth respectively. The encoded coordinates can be expressed as:

$$\begin{aligned} l_x &= (x - x_g)/w_g, \quad & l_y &= (y - y_g)/h_g, \\ l_x' &= (x' - x_g)/w_g, \quad & l_y' &= (y' - y_g)/w_g, \\ l_w &= \log(w/w_g), & l_h &= \log(h/h_g), \\ l_w' &= \log(w'/w_g), & l_h' &= \log(h'/h_g), \end{aligned} \tag{5}$$

where variables x, x_g and x' are for the prediction of clip, g-clip and ground truth clip respectively (same as the y, w and h).

The classification and regression loss function for clips can be expressed as:

$$\begin{aligned} L_{C\&R}(h_i, \boldsymbol{l}_i) = \alpha_{loc} \sum_i h_i^{'} l_{loc}(\boldsymbol{l}_i, \boldsymbol{l}_i') + \sum_i l_{hotspot}(h_i, h_i^{'}) \\ + \frac{1}{2}\beta(\|\boldsymbol{T}_{loc}\|_2^2 + \|\boldsymbol{T}_{hotspot}\|_2^2), \end{aligned} \tag{6}$$

where β is a hyper-parameter which controls the regularization strength. α_{loc} is the hyper-parameter which controls the balance between two tasks. The term $h_i^{'} l_{loc}(\boldsymbol{l}_i, \boldsymbol{l}_i')$ indicates that regression loss is only activated for clips labeled as hotspots. $\boldsymbol{T}_{loc}$ and $\boldsymbol{T}_{hotspot}$ are the weights of the neural network. For elements $l_i[j]$ and $l_i'[j]$ ($j \in [1, 4]$) in $\boldsymbol{l}_i, \boldsymbol{l}_i'$ respectively, l_{loc} can be expressed as

$$l_{loc}(l_i[j], l_i'[j]) = \begin{cases} \frac{1}{2}(l_i[j] - l_i'[j])^2, & \text{if } |l_i[j] - l_i'[j]| < 1, \\ |l_i[j] - l_i'[j]| - 0.5, & \text{otherwise}, \end{cases} \tag{7}$$

which is the so-called robust loss or smooth L_1 loss (defined in [39]) applied to avoid the exploding gradients problem at training stage. $l_{hotspot}$ is the cross-entropy loss which is calculated as:

$$l_{hotspot}(h_i, h_i^{'}) = -(h_i \log h_i^{'} + h_i^{'} \log h_i). \tag{8}$$

In Equation (6), we apply the L_2 regularization to the loss function, which is the sum of the squares of all the weights in the network. The L_2 regularization penalizes peaky weight vectors and prefers diffuse weight vectors. Due to multiplicative interactions between weights and features, the L_2 regularization term has appealing property of encouraging the network to use all of its inputs rather than skewed on partial of its inputs.

E. Example of Detection Flow

An example of R-HSD flow is illustrated in Fig. 9. We first extract output tensors of each stage and sum up in channel-wise for visualization. Note that we visualize the features in grayscale, where the locations with lighter colors have higher values and darker locations have lower values vice versa. With feature extraction going deeper, values of features at hotspot regions have higher response comparing to the non-hotspot regions. The hotspot and non-hotspot areas presented as rectangles (for a clear explanation, not all rectangles are shown in the figures) in clip proposal network are cropped and downsampled to the same size with RoI pooling at the refinement stage. After the second stage classification and regression, a more accurate result is given.

IV. Enhanced R-HSD Neural Network

In previous sections, the preliminary R-HSD neural network has been proposed. However, there still exists some room to improve the performance of our region based hotspot detector. For example, the encoder-decoder structure in feature extractor becomes a bottleneck since it lacks a multi-level description of an input layout. Therefore, based on the preliminary design for R-HSD Neural Network, two new concepts are introduced for further enhancement.

A. Multi-branch Design for Encoder

The encoder-decoder in prior arts is in a single-branch structure, which may bring the following issues into the learning process. One is a fixed-size kernel cannot capture multi-scale information. The other is naively stacking large convolution operations is computationally expensive.

To alleviate the above issues, we propose our multi-branch design based on the basic idea of Inception network [37] and atrous convolution [40]–[42]. The core of the proposed design

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3021663, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

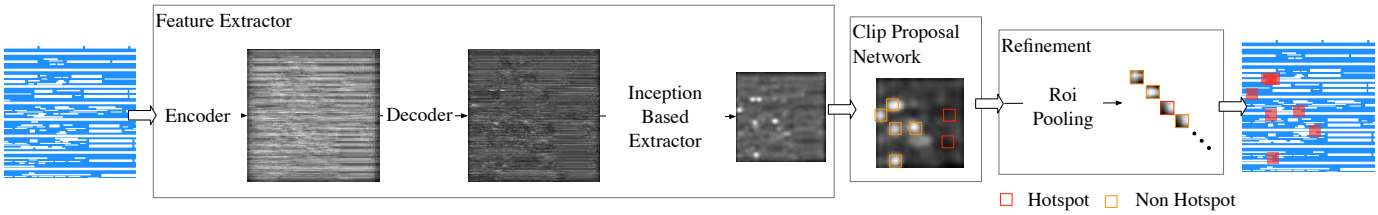


Fig. 9 An example of features presentation in our framework.

is that convolutional kernels with multiple sizes operate on the same level simultaneously. By aggregating the feature maps on different scales, the encoder-decoder structure has the potential to surpass other feature extractors. Furthermore, to reduce the computational complexity, we exploit atrous convolution as an alternative to the standard convolution. As a generalization of standard convolution operation, atrous convolution is a powerful technique that explicitly controls the resolution of features computed by CNNs and adjust kernel's field of view to handle multi-scale information. Except for the dilated rate, it works in a similar way as standard convolution which moves across the whole image with a stride-size column change on the horizontal movements, and a stride-size row change on the vertical movements. We visualize the computation process of atrous convolution in Fig. 10, where only grids filled with dashed and oblique lines need to compute. It can be seen that the dilated rate controls the field-of-view scope of a kernel and affects the resolution of the output feature map. Note that when dilated rate equals to 1, the atrous convolution degrades to standard convolution. Hence, we can rewrite Equation (2) as following:

$$\begin{aligned}&\mathbf{F} \otimes \mathbf{K}(j,k)\\ &= \sum_{i=1}^{c}\sum_{m_0=1}^{m}\sum_{n_0=1}^{m}\mathbf{F}(i, j - r * m_0, k - r * n_0)\mathbf{K}(m_0, n_0),\end{aligned} \tag{9}$$

where r refers to the dilated rate. Implicitly expressed by Equation (9), the advantage of atrous convolution is that it expands kernel size without introducing additional computational complexity.

Building on top of the aforementioned ideas, our multi-branch framework is designed as in Fig. 11. Three branches with different dilated rates (e.g. 1, 3, 5 as fine-tuned configurations) work collaboratively, and then all output tensors concatenate as a fusion feature map via channel dimension.

B. IoU Regularizer for Loss Function Design

To accelerate bounding box prediction, we leverage a novel IoU [43], [44] regularization term in our loss function. Hence, our loss function is redesigned as:

$$\begin{aligned}L_{C\&R}(h_i, \boldsymbol{l}_i) = \alpha_{loc}\sum_i h_i' l_{loc}(\boldsymbol{l}_i, \boldsymbol{l}_i') + \sum_i l_{hotspot}(h_i, h_i')\\ + \frac{1}{2}\beta(\|\boldsymbol{T}_{loc}\|_2^2 + \|\boldsymbol{T}_{hotspot}\|_2^2 + R_{IoU}),\end{aligned} \tag{10}$$

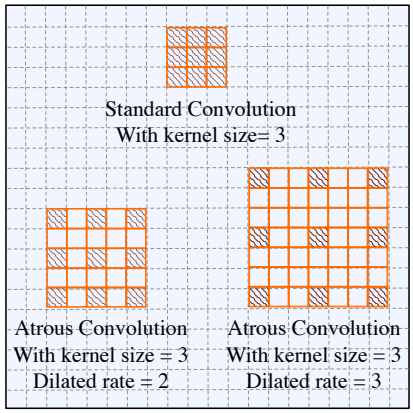


Fig. 10 The illustration of atrous convolution.

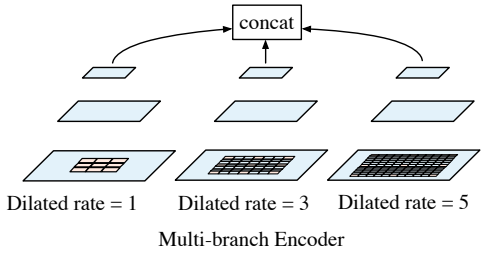


Fig. 11 The Illustration of proposed multi-branch design.

where R_{IoU} is the proposed IoU regularizer.

IoU, also known as the Jaccard index, is a widely exploited metric for comparing the similarity between two geometric shapes. IoU encodes the shape and position properties of the objects under comparison, e.g. the indices of left upper corner and right bottom corner of two clips in our case, into the region property, and then calculates a normalized measure that focuses on their areas. This property makes IoU robust to the scale of the problem under consideration. Thanks to this appealing property, this metric is the foundation of all performance measures in segmentation, object detection, and tracking tasks. The exploited IoU regularizer, shown in Fig. 12, directly enforces the maximal overlap between the predicted clip and the ground truth, and jointly regresses all the bound variables as a whole unit.

To give a more mathematical understanding of the proposed IoU regularizer, the deduction of back-propagated information of itself is shown as follows. Assume the predicted clip and corresponding ground truth are located as shown in Fig. 12. Firstly, the partial derivatives of the area of the predicated clip

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3021663, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

TABLE I Benchmark information. `case2`, `case3` and `case4` are three cases from ICCAD CAD contest 2016 benchmark suite [45]. `Via` is generated by open source layout generator and simulated using Mentor Calibre. Clips for training are generated by random cropping on layouts.

Bench	Train #HS	Test #HS	Train #Clips†	Test #Clips	Training Set Size ($\mu m \times \mu m$)	Testing Set Size ($\mu m \times \mu m$)
`case2`	40	39	1000	8	6.95 × 3.75	6.95 × 3.75
`case3`	1388	1433	1000	33	12.91 × 10.07	12.91 × 10.07
`case4`	90	72	1000	55	79.95 × 42.13	79.95 × 42.13
`Via`	2184	2184	1000	1947	53.82 × 53.82	53.82 × 53.82

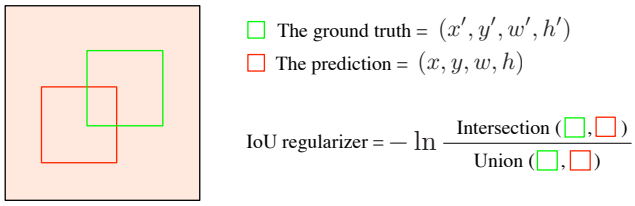


Fig. 12 The illustration of the IoU regularizer.

with respect to y_l, y_r, x_l, x_r, are computed as:

$$\frac{\partial X}{\partial y_l\ (\textbf{or}\ \partial y_r)} = x_r - x_l, \tag{11}$$

and

$$\frac{\partial X}{\partial x_l\ (\textbf{or}\ \partial x_r)} = y_r - y_l. \tag{12}$$

For simplicity, we use ∇X refer to any derivatives w.r.t y_l, y_r, x_l, x_r. Next, the partial derivatives of the intersection area I w.r.t any y_l, y_r, x_l, x_r, marked as ∇I, is deduced as:

$$\frac{\partial I}{\partial y_l} = x_r - x_l', \tag{13}$$

while $\frac{\partial I}{\partial y_r} = 0$ and

$$\frac{\partial I}{\partial x_r} = y_r' - y_l, \tag{14}$$

while $\frac{\partial I}{\partial x_l} = 0$. Eventually, the back-propagated information of proposed regularization w.r.t p is

$$\begin{aligned} \nabla R_{IoU} &= \frac{I\left(\nabla X - \nabla I\right) - U\nabla I}{U^2 IoU} \\ &= \frac{1}{U}\nabla X - \frac{U+I}{UI}\nabla I, \end{aligned} \tag{15}$$

where the union area U equals to $(X + X')$. According to Equation (15), the first term $\frac{1}{U}\nabla X$ penalizes the predicted clip, whilst the second term is a soft constraint on the intersection area. When the gradient equals to zero, the limit case which means predicted clip exactly matches the ground truth are attained.

V. Experimental Results

Our region-based hotspot detection flow is evaluated on ICCAD CAD Contest 2016 benchmark suite [45], which contains four designs that are shrunk to match EUV metal layer design rules. Ground truth hotspot locations are labelled according to the results of industrial $7nm$ metal layer EUV lithography simulation under a given process window[1]. As there are limited defects found with lithography simulation on the first benchmark, all our experiments are conducted on rest three designs. Each layout is split into two equal halves with one part used for training and the other one used for testing. Besides these three cases, we generate a much larger benchmark called `Via` to present a more comprehensive comparison with previous related works. `Via` benchmark is generated following an open source layout generator[2], and simulated using Mentor Calibre. More details about benchmarks are shown in TABLE I. We implement our region based hotspot detection framework with Tensorflow [48] in Python, and test it on a platform with a Xeon Silver 4114 processor and a Nvidia GTX Titan graphic card. Nvidia GTX Titan has a comparable computational capacity as the high performance clusters with 24 Maxwell stream processors. *Note that three training layouts are merged together to train one model that will be used in the inference stage.* In the following experiments, the neural network is trained with following parameter settings: $input\ size = 256 \times 256$ (corresponding to $2.56\mu m \times 2.56\mu m$), $batch\ size = 12$, $initial\ learning\ rate = 0.002$ (decay ten times for each 30000 steps), $aspect\ ratio$ = [0.5, 1.0, 2.0] and $scales$ = [0.25, 0.5, 1.0, 2.0]. The parameters of the loss function are heuristically chosen, what we use are $\beta = 0.2, \alpha_{loc} = 2.0$. At the inference stage, we follow the same data generation rule applied in [25].

We list the detailed result comparison in TABLE II. Column "Bench" lists three benchmarks used in our experiments. Columns "Accu", "FA", "Time" denote hotspot detection accuracy, false alarm count and detection runtime respectively. Column "TCAD'19" lists the result of a deep learning-based hotspot detector proposed in [25] that adopts frequency domain feature extraction and biased learning strategy. We also implement two baseline frameworks that employ Faster R-CNN [46] and SSD [47], respectively, which are two classic techniques match our region-based hotspot detection objectives well. Note that we do not apply the pre-trained model in this work, all the models are trained from scratch. The corresponding results are listed in columns "Faster R-CNN [46]" and "SSD [47]". The rest two columns, "R-HSD" and "Enhanced R-HSD", denote the methods proposed in [33] and the framework presented in this work. The results in R-HSD surpass [25] with average of 6.11% improvement on hotspot detection accuracy and $\sim$ 170 less false alarm penalty,

[1]Shrunk benchmarks and their hotspot information is available at https://github.com/phdyang007/ICCAD16-N7M2EUV

[2]https://github.com/phdyang007/layout-generator

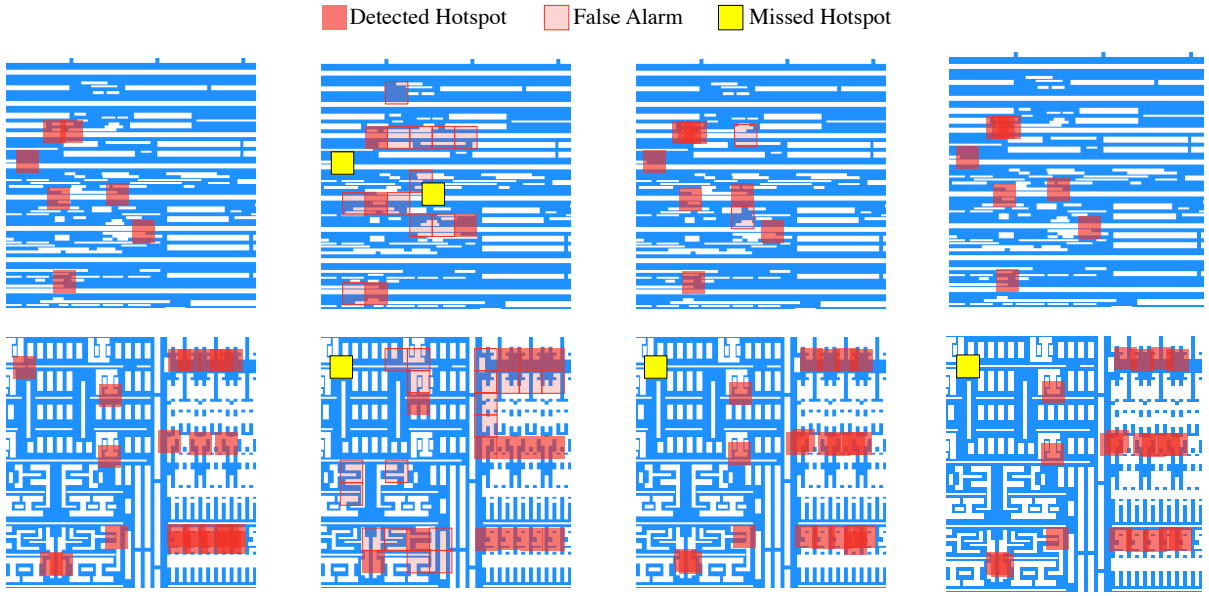
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3021663, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

TABLE II Performance comparison with state-of-the-art methods. Faster R-CNN [46] and SSD [47] are two classical techniques match to the region-based hotspot detection objectives. TCAD'19 [25] is a deep learning based hotspot detector. R-HSD and Enhanced R-HSD are the methods proposed in [33] and in this work, separately.

Bench	TCAD'19 [25]			Faster R-CNN [46]			SSD [47]			R-HSD [33]			Enhanced R-HSD		
	Accu (%)	FA	F1	Accu (%)	FA	F1	Accu (%)	FA	F1	Accu (%)	FA	F1	Accu (%)	FA	F1
case2	77.78	48	0.51	1.80	3	0.05	71.90	519	0.53	93.02	17	0.78	95.74	15	0.81
case3	91.20	263	0.87	57.10	74	0.11	57.40	1730	0.61	94.50	34	0.96	94.72	78	0.94
case4	100	511	0.22	6.90	69	0.07	77.80	275	0.03	100	201	0.38	100	92	0.61
Via	81.30	2672	0.54	86.52	25551	0.12	64.21	65319	0.38	87.20	2577	0.57	89.40	2435	0.58
Average	87.57	873.5	0.54	38.08	6424.5	0.24	67.83	16960	0.39	93.68	707.3	0.67	94.97	655	0.74
Ratio	1.00	1.00	1.00	0.43	7.35	0.44	0.77	19.42	0.72	1.07	0.96	1.24	**1.08**	**0.91**	**1.37**

TABLE III Runtime Comparison with State-of-the-art methods.

Bench	TCAD'19 [25]	Faster R-CNN [46]	SSD [47]	R-HSD [33]	Enhanced R-HSD
case2	60.0	1.0	1.0	2.0	2.3
case3	265.0	11.0	3.0	10.0	10.8
case4	428.0	8.0	2.0	6.0	6.6
Via	87.8	57.1	21.3	43.5	46.3
Average	210.2	19.3	6.8	15.4	16.5



(a) Ground-truth (b) TCAD'19 [25] (c) R-HSD [33] (d) Enhanced R-HSD

Fig. 13 Visualization of different hotspot detection results.

TABLE IV Comparison with ASPP Module. ASPP module is approached in [42], which is designed for general objects.

Bench	ASPP [42] + R-HSD [33]				Enhanced R-HSD			
	Accu (%)	FA	F1	Time (s)	Accu (%)	FA	F1	Time (s)
case2	82.05	22	0.68	3.6	95.74	15	0.81	2.3
case3	90.95	96	0.92	11.0	94.72	78	0.94	10.8
case4	100	143	0.50	4.45	100	92	0.61	6.6
Via	87.61	2558	0.58	44.7	89.4	2435	0.58	46.3
Average	90.39	704.8	0.67	15.38	94.97	655	0.74	16.5
Ratio	1.00	1.00	1.00	1.00	**1.05**	**0.97**	**1.10**	1.07

while our enhanced R-HSD behaves even better with an average accuracy of 94.97 % and further decrease on the false alarm compared to the R-HSD. Especially, our framework gets much better on case2 with 95.74% detection accuracy compared to 77.78%, 1.8% and 71.9% for [25], Faster R-CNN, and SSD respectively.

The advantage of the proposed two-stage classification and regression flow can also be seen here that [25] achieves similar hotspot detection accuracy compared to our frameworks but has extremely large false alarms that will introduce additional issue. As shown in TABLE III, the detection runtime for the proposed region-based framework is much faster than [25] thanks to the region-based detection scheme. We can also observe that although Faster R-CNN and SSD are originally designed for large region object detection, they perform very poor on hotspot detection tasks which reflects the effectiveness and efficiency of our customized frameworks. Different from

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3021663, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

TABLE V Ablation study on each proposed strategy.

Bench	w/o. ED		w/o. MB ED		w/o. L_2		w/o. IoU		w/o. Refine		Full	
	Accu (%)	FA	Accu (%)	FA	Accu (%)	FA	Accu (%)	FA	Accu (%)	FA	Accu (%)	FA
case2	75	31	93.32	20	91.9	1	94.11	21	75.76	394	95.74	15
case3	92.8	27	94.72	31	91.7	64	94.91	23	94	23	94.72	78
case4	98.6	166	100	220	97.2	127	100	240	100	201	100	92
Via	85.12	2762	84.57	2812	86.5	2697	88.7	2516	89.1	2516	89.4	2435
Average	87.8	746.5	93.15	770.8	91.82	722.3	94.43	700	89.72	783.5	94.97	655
Ratio	0.925	1.06	0.981	1.07	0.967	1.04	0.994	1.07	0.94	1.20	**1.00**	**1.00**

TABLE VI Ablation study on anchor generation.

Anchor number	Scales	Aspect Ratios	Average Accu (%)	Average FA
3	[0.25, 0.5, 1.0]	[0.5]	89.13	3244.5
6	[0.25, 0.5, 1.0]	[0.5, 1.0]	92.31	1820
9	[0.25, 0.5, 1.0]	[0.5, 1.0, 2.0]	90.12	1725.5
12 (final result)	[0.25, 0.5, 1.0, 2.0]	[0.5, 1.0, 2.0]	94.97	1655

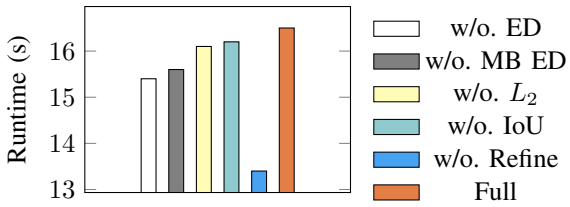


Fig. 14 Runtime comparison among different settings.

the atrous spatial pyramid pooling (ASPP) proposed in [40], [42] which extracts features in multiple scales. Multi-branch encoder with atrous convolution layer designed for clip-wise feature extractions has much lower dilation rate than ASPP, because the size and scale of clips are much regular than objects in real life. The setting of ASPP is not compatible with the hotspot detection task. The experiments in TABLE IV show that our proposed design in hotspot detection task outperforms ASPP by a large margin.

We also study how different configurations of our framework affect performance. TABLE V and Fig. 14 summarize the contributions of encoder-decoder structure, multi-branch encoder-decoder, L_2 regularization, IoU regularizer and refinement stage to our backbone neural network. "w/o. ED" denotes the framework without the encoder-decoder structure, "w/o. MBED" denotes the framework without the multi-branch design for the encoder-decoder structure, "w/o. L_2" stands for the framework without the L_2 regularization, "w/o. IoU" denotes the framework without the IoU regularizer, "w/o. Refine" denotes the framework without the refinement classification and regression, and "Full" is our framework with entire techniques. The ablation study shows that with the encoder-decoder structure, we get 7.17% accuracy improvement on average, which indicates that the encoder-decoder structure gives a more efficient feature expression than the original input. After incorporating multi-branch design for the encoder-decoder structure, the accuracy is improved by 1.82% on average, which demonstrates the effectiveness of this configuration. It can be seen that without IoU regularizer, the performance degrades with 45 additional false alarms. With the L_2 regularization, the framework gets around 3% improvement in all cases, which means under the same experiment settings, the L_2 regularization resolves the overfitting problem effectively. Comparing the whole framework with the model without refinement, the model with refinement reduces around 20% false alarms and achieves 5.25% further improvement on average accuracy. TABLE VI shows an ablative comparison on different anchor generation settings. "Anchor number" denotes the number of anchors we generate for each location. "Scales" denotes the size ratio compared to a standard anchor with the shape of 16×16. "Aspect Ratios" denotes the ratio between anchor width and height. With the increasing of anchors, the number of false alarms can be reduced signficantly, which indicates a sufficient sampling is necessary for the training.

VI. Conclusion

In this paper, we have proposed an innovative end-to-end region-based hotspot detection framework. Our feature extractor based on multi-branch encoder-decoder design and inception module provides a self-adaptive way to perform feature transformation, which is very compatible with convolution neural networks. With pruning and hotspot non-maximum suppression strategies, the clip proposal network locates the potential hotspot in an efficient regression way. We take advantage of L_2 regularization's property to prevent over-fitting and get higher performance. IoU regularizer is leveraged to boost the regression procedure thus attain improvements on both accuracy and false alarm. Additionally, our classification and regression strategy with refinement reduces false alarms and increases accuracy at a remarkable speed. The experimental results show that our framework outperforms the current deep learning based models. The defect results in this paper come from rigorious EUV model simulation model. In our future work, more experiments with compact lithography simulation will be discussed. We hope this paper can give a new perspective on deep learning based hotspot detection and provide a more powerful solution for the advanced design for manufacturability (DFM) research.

REFERENCES

[1] D. Z. Pan, B. Yu, and J.-R. Gao, "Design for manufacturing with emerging nanolithography," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 32, no. 10, pp. 1453–1472, 2013.

[2] B. Yu, X. Xu, S. Roy, Y. Lin, J. Ou, and D. Z. Pan, "Design for manufacturability and reliability in extreme-scaling VLSI," *Science China Information Sciences*, pp. 1–23, 2016.

[3] J. Kim and M. Fan, "Hotspot detection on Post-OPC layout using full chip simulation based verification tool: A case study with aerial image simulation," in *Proceedings of SPIE*, vol. 5256, 2003.

[4] 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 *Proceedings of SPIE*, vol. 6521, 2007.

[5] A. B. Kahng, C.-H. Park, and X. Xu, "Fast dual graph based hotspot detection," in *Proceedings of SPIE*, vol. 6349, 2006.

[6] H. Yao, S. Sinha, C. Chiang, X. Hong, and Y. Cai, "Efficient process-hotspot detection using range pattern matching," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2006, pp. 625–632.

[7] 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.

[8] 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.

[9] S. S.-E. Tseng, W.-C. Chang, I. H.-R. Jiang, J. Zhu, and J. P. Shiely, "Efficient search of layout hotspot patterns for matching SEM images using multilevel pixelation," in *Proceedings of SPIE*, vol. 10961, 2019.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] B. Yu, J.-R. Gao, D. Ding, X. Zeng, and D. Z. Pan, "Accurate lithography hotspot detection based on principal component analysis-support vector machine classifier with hierarchical data clustering," *Journal of Micro/Nanolithography, MEMS, and MOEMS (JM3)*, vol. 14, no. 1, p. 011003, 2015.

[16] 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," *Journal of Micro/Nanolithography, MEMS, and MOEMS (JM3)*, vol. 15, no. 4, pp. 043 504–043 504, 2016.

[17] 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.

[18] 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.

[19] 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.

[20] H. Geng, H. Yang, B. Yu, X. Li, and X. Zeng, "Sparse VLSI layout feature extraction: A dictionary learning approach," in *IEEE Annual Symposium on VLSI (ISVLSI)*, 2018, pp. 488–493.

[21] W. Ye, M. B. Alawieh, M. Li, Y. Lin, and D. Z. Pan, "Litho-GPA: Gaussian process assurance for lithography hotspot detection," in *IEEE/ACM Proceedings Design, Automation and Test in Eurpoe (DATE)*, 2019, pp. 54–59.

[22] 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.

[23] 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.

[24] 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.

[25] H. Yang, J. Su, Y. Zou, Y. Ma, B. Yu, and E. F. Y. Young, "Layout hotspot detection with feature tensor generation and deep biased learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 38, no. 6, pp. 1175–1187, 2019.

[26] H. Yang, S. Li, Y. Ma, B. Yu, and E. F. Young, "GAN-OPC: Mask optimization with lithography-guided generative adversarial nets," in *ACM/IEEE Design Automation Conference (DAC)*, 2018, pp. 131:1–131:6.

[27] W. Ye, Y. Lin, M. Li, Q. Liu, and D. Z. Pan, "LithoROC: lithography hotspot detection with explicit ROC optimization," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2019, pp. 292–298.

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

[29] Y. Chen, Y. Lin, T. Gai, Y. Su, Y. Wei, and D. Z. Pan, "Semi-supervised hotspot detection with self-paced multi-task learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2019.

[30] Y. Jiang, F. Yang, H. Zhu, B. Yu, D. Zhou, and X. Zeng, "Efficient layout hotspot detection via binarized residual neural network," in *ACM/IEEE Design Automation Conference (DAC)*, 2019, pp. 147:1–147:6.

[31] H. Geng, H. Yang, L. Zhang, J. Miao, F. Yang, X. Zeng, and B. Yu, "Hotspot detection via attention-based deep layout metric learning," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2020.

[32] 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, pp. 349–350.

[33] R. Chen, W. Zhong, H. Yang, H. Geng, X. Zeng, and B. Yu, "Faster region-based hotspot detection," in *ACM/IEEE Design Automation Conference (DAC)*, 2019, pp. 146:1–146:6.

[34] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2117–2125.

[35] A. Shrivastava, R. Sukthankar, J. Malik, and A. Gupta, "Beyond skip connections: Top-down modulation for object detection," *arXiv preprint arXiv:1612.06851*, 2016.

[36] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "Dssd: Deconvolutional single shot detector," *arXiv preprint arXiv:1701.06659*, 2017.

[37] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2818–2826.

[38] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Conference on Neural Information Processing Systems (NIPS)*, 2012, pp. 1097–1105.

[39] R. Girshick, "Fast R-CNN," in *IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448.

[40] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[41] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[42] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 801–818.
[43] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang, "Unitbox: An advanced object detection network," in *Proceedings of the 24th ACM international conference on Multimedia*, 2016, pp. 516–520.
[44] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 658–666.
[45] R. O. Topaloglu, "ICCAD-2016 CAD contest in pattern classification for integrated circuit design space analysis and benchmark suite," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2016, pp. 41:1–41:4.
[46] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Conference on Neural Information Processing Systems (NIPS)*, 2015, pp. 91–99.
[47] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *European Conference on Computer Vision (ECCV)*, 2016, pp. 21–37.
[48] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean *et al.*, "TensorFlow: A system for large-scale machine learning," in *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2016, pp. 265–283.

Ran Chen received his B.E. degree from the Department of Information Engineering, The Chinese University of Hong Kong in 2018. He is currently pursuing his Ph.D. degree at the Department of Computer Science and Engineering, The Chinese University of Hong Kong. His research interests include machine learning application in pattern recognition and VLSI design for manufacturing.

Wei Zhong received the B.S. degree from Dalian University of Technology, Dalian, China, in 2008, the M.S. and Ph.D. degrees from Waseda University, Tokyo, Japan, in 2010 and 2014, respectively. He served as a Research Assistant in the Information, Production and Systems Research Center of Waseda University, Kitakyushu, Japan, from 2010 to 2011, an Associate Specialist in the Central Research Laboratory of Ricoh Company, Tokyo, Japan, from 2011 to 2014, a Chief Designer and Director of the Institute of Image Processing Technology in the State Key Laboratory of Digital Multimedia Technology, Hisense Group, Qingdao, China, from 2014 to 2018. From 2015 to 2017, he also served as a Postdoctoral Researcher in the University of Science and Technology of China, Hefei, China. He is currently an Associate Professor in the International School of Information Science and Engineering, Dalian University of Technology, Dalian, China. His research interests include several aspects of computer vision and image processing algorithms, VLSI design automation, networks on chips and hardware-software co-design of embedded systems.

Haoyu Yang received his B.E. degree from Qiushi Honors College, Tianjin University in 2015, and Ph.D. Degree from the Department of Computer Science and Engineering, Chinese University of Hong Kong in 2020. He is currently working as a post-doctoral fellow in the same department at CUHK. His research interests include machine learning in VLSI design for manufacturability, high performance VLSI physical design with parallel computing and machine learning security.

Hao Geng received the M.E. degree from the Department of Electronic Engineering and Information Sciences, University of Science and Technology of China in 2015, and the M.Sc. degree in machine learning from the Department of Computing, the Imperial College London in 2016. He is currently pursuing his Ph.D. degree at the Department of Computer Science and Engineering, The Chinese University of Hong Kong. His research interests include machine learning and deep learning in VLSI design for manufacturing.

Fan Yang (M'08) received the B.S. degree from Xi'an Jiaotong University in 2003, and the Ph.D. degree from Fudan University in 2008. From 2008 to 2011, he was an Assistant Professor with Fudan University. He is currently an Associate Professor with the Microelectronics Department, Fudan University. His research interests include model order reduction, circuit simulation, high-level synthesis, yield analysis, and design for manufacturability.

Xuan Zeng (M'97–SM'17) received the B.S. and Ph.D. degrees in electrical engineering from Fudan University, Shanghai, China, in 1991 and 1997, respectively. She is a Full Professor with the Department of Microelectronics, Fudan University. She was the Director of the State Key Laboratory of ASIC & System from 2008 to 2012. She was a Visiting Professor at the Department of Electrical Engineering, Texas A&M University, College Station, TX, USA, and the Department of Microelectronics, Technische Universiteit Delft, Delft, The Netherlands, in 2002 and 2003, respectively. Her current research interests include analog circuit modeling and synthesis, design for manufacturability, high-speed interconnect analysis and optimization, circuit simulation. Prof. Zeng is an Associate Editor of IEEE Trans. on Circuits and Systems: Part II, IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, and ACM Trans. on Design Automation on Electronics and Systems. Dr. Zeng received the Best Paper Award from the 8th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference 2017. She received Changjiang Distinguished Professor with the Ministry of Education Department of China in 2014, the Chinese National Science Funds for Distinguished Young Scientists in 2011 and the First-Class of Natural Science Prize of Shanghai in 2012, 10th For Women in Science Award in China in 2013, Shanghai Municipal Natural Science Peony award in 2014.

Bei Yu (S'11–M'14) received the Ph.D. degree from The University of Texas at Austin in 2014. He is currently an Assistant Professor in the Department of Computer Science and Engineering, The Chinese University of Hong Kong. He has served as TPC Chair of ACM/IEEE Workshop on Machine Learning for CAD, and in many journal editorial boards and conference committees. He is Editor of IEEE TCCPS Newsletter. He 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.