GAN-SRAF: Sub-Resolution Assist Feature Generation Using Conditional Generative Adversarial Networks

Mohamed Baker Alawieh[1], Yibo Lin[1], Zaiwei Zhang[2], Meng Li[1], Qixing Huang[2], and David Z. Pan[1]

[1]ECE Department, University of Texas at Austin
[2]Computer Science Department, University of Texas at Austin

***Abstract*—As the integrated circuits (IC) technology continues to scale, resolution enhancement techniques (RETs) are mandatory to obtain high manufacturing quality and yield. Among various RETs, sub-resolution assist feature (SRAF) generation is a key technique to improve the target pattern quality and lithographic process window. While model-based SRAF insertion techniques have demonstrated high accuracy, they usually suffer from high computational cost. Therefore, more efficient techniques that can achieve high accuracy while reducing runtime are in strong demand. In this work, we leverage the recent advancement in machine learning for image generation to tackle the SRAF insertion problem. In particular, we propose a new SRAF insertion framework, GAN-SRAF, which uses conditional generative adversarial networks (CGANs) to generate SRAFs directly for any given layout. Our proposed approach incorporates a novel layout to image encoding using multi-channel heatmaps to preserve the layout information and facilitate layout reconstruction. Our experimental results demonstrate $\sim$14.6$\times$ reduction in runtime when compared to the previous best machine learning approach for SRAF generation, and $\sim$144$\times$ reduction compared to model-based approach, while achieving comparable quality of results.**

I. Introduction

While the integrated circuits technology node continues to scale, the photolithography techniques are supposed to keep up the pace and cope with the ever shrinking feature size. In fact, low image contrast and complex target pattern shapes make it extremely difficult for low$-k1$ lithography, the mainstream technique, to achieve desired lithographic process windows [1], [2]. Hence, resolution enhancement techniques have been the major strategy to improve lithographic process window.

Among these techniques, Sub-Resolution Assist Feature (SRAF) generation is used to improve the lithographic process window of target patterns. These assist features are not actually printed; instead, the SRAF patterns would deliver light to the positions of target patterns at proper phase which can improve the robustness of target printing to lithographic variations [3]. In practice, the process variation band is typically used as a measure of such robustness, where the goal is to achieve the minimum possible band [3].

In literature, different SRAF generation approaches have been proposed and adopted. The rule-based approach can achieve acceptable accuracy within short execution time for simple designs and regular target patterns [1], [4], [5]. However, the rule-based approach cannot handle complex shapes as it requires significant pre-processing engineering efforts. On the other hand, two trends of model-based SRAF generation methods have been proposed and these can be categorized based on the lithographic computations involved. The first trend uses simulated aerial images to seed the SRAF generation [1], [2], while the other applies inverse lithography technology (ILT) and computes the image contour to guide the SRAF generation [6]. Despite better lithographic performance compared to the rule-based approach, the model-based SRAF generation is very time-consuming and it is difficult to achieve the same SRAFs around the same layout configuration, i.e., not consistent [3].

Recently, machine learning has been proposed to tackle the problem of SRAF insertion to reduce the computational cost associated with model-based methods [3], [7]. The proposed method relies on SRAF features extraction with local sampling scheme to obtain the optimal SRAF map. The key idea is to use a 2D grid plane where a classification model is trained to predict the probability of existence of SRAF in each grid based on the extracted features. Although this approach has demonstrated significant speedup compared to model-based approaches while achieving comparable results in terms of process variation band, there is still significant room for improvement as we will show in this paper.

Motivated by the recent advancement in the field of image processing in computer vision [8]–[12], we elect in this work to address the SRAF generation problem from a new perspective. In fact, a layout in its essence can be simply viewed as an image; hence, machine learning techniques developed for image related task can come in handy. Moreover, it has been shown that using convolutional neural networks has demonstrated superior runtime and performance compared to local decision based approaches when dealing with visual data. Specifically, Conditional Generative Adversarial Network (CGAN) has been leveraged to perform image translation tasks. In other words, given related images in two different domains, CGAN can be trained to translate images from one domain to another [8], [9]. Examples of such applications include, among others, image colorization and aerial to map and edge to photo translations [9]. In addition, CGAN has been recently adopted as a means to enhance the optical proximity correction in IC manufacturing [13].

In this work, we propose to use CGAN for SRAF generation by casting the problem into an image translation task where the two images domains are: (i) original layout and (ii) layout with SRAFs. Hence, generating an SRAF scheme for a particular layout can be seen as translating the layout image from the first domain (i.e., original layout) to the second domain (i.e., layout

DAC '19, June 2-6, 2019, Las Vegas, NV, USA

ACM ISBN 978-1-4503-6725-7/19/06...$15.00
https://doi.org/10.1145/3316781.3317832

with SRAFs). Towards this end, layout files are mapped into images in a novel encoding scheme that captures the layout details. This scheme incorporates a multi-channel heatmap encoding of different layout objects into different layers of an image [14]–[17]. Additionally, this encoding is accompanied by a fast GPU-accelerated decoding scheme to recover layout schemes from images generated by CGAN. With our proposed encoding/decoding framework, a CGAN is trained to generate layouts with SRAF inserted using a labeled data set. Once trained, the CGAN can take an original layout image as an input and generate a new image with SRAFs inserted. These generated images can be eventually get mapped back to layout files.

In this SRAF generation framework, our main contributions are summarized as follows:

- A conditional generative adversarial network is used for the first time for SRAF generation.
- We cast the SRAF generation problem as an image-to-image translation task where the layout is translated from its *original* domain to *layout with SRAFs* domain.
- A novel multi-channel heatmap encoding/decoding scheme is used to map layouts to images suitable for CGAN training while preserving the layout details.
- Our proposed framework achieves $\sim 14.6\times$ speed-up with comparable lithographic performance when compared with state-of-art machine learning based approach and $\sim 144\times$ speed-up over the model-based approach in commercial tool Mentor/Calibre [3] while achieving comparable results.

The remainder of this paper is organized as follows. In Section II we review the technical background and then present the proposed approach in Section III. Section IV presents numerical results demonstrating the efficacy of our method, and conclusions are presented in Section V.

II. Problem Formulation

The objective of the SRAF generation framework is to insert SRAFs on any given layout in a manner that mimics the SRAF scheme generated from model-based techniques. Practically, the input is a layout clip with target patterns only as shown in Fig. 1a, and the expected output is a new layout clip similar to the one shown in Fig. 1b where SRAFs are generated to aid the printing of target patterns. In other words, the objective is to train a CGAN to translate images from the target domain, D_{Trgt}, (Fig. 1a) to the SRAF domain, D_{SRAF}, (Fig. 1b).

In the training phase, each training sample consists of a pair of images representing the original layout in D_{Trgt} and its corresponding layout in D_{SRAF}. Based on the training data, the CGAN model is trained to map images from D_{Trgt} to D_{SRAF}. Then, the trained model can be used to generate SRAFs from layouts with target patterns. However, two challenges should be addressed here. The first is that proper image encoding/decoding is needed to aid the CGAN training scheme. Secondly, the generated SRAF scheme may violate some of the manufacturing rules; hence, a post-processing step is needed to generate a final layout with SRAFs while abiding by the specified rules.

To evaluate our proposed SRAF generation method, we use two metrics to assess the performance of the mask optimization results: (i) process variation (PV) band and (ii) edge placement error (EPE). These metrics are defined in a way analogous to the definitions used in [3].

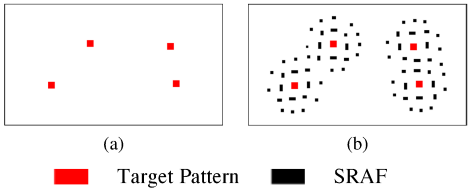


Fig. 1 SRAF generation task can be cast as an image translation problem where layout with target contacts (a) are translated to ones with SRAF generated (b) .

III. SRAF Insertion Using CGAN

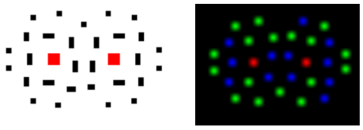
A. Data Preparation using Heatmap Encoding

As shown in Fig 1, the layouts from both domains D_{Trgt} and D_{SRAF} can be treated directly as images. However, this direct image representation is not suitable for the SRAF generation using CGAN because the expected output cannot be directly mapped to layout files due to two major limitations. First, the trained CGAN is not guaranteed to generate 'clean' rectangular shapes for the SRAFs. In practice, images generated from generative adversarial networks (GANs) tend to be blurry and GANs exhibit inherent limitation in detecting sharp edges [12]. In addition, and even under the assumption that the CGAN model can generate sharp-edged rectangles for the SRAFs, extracting the SRAF information from the image to be mapped back to the layout file can be prohibitively expensive. Such mapping requires obtaining both SRAF locations and sizes from the image generated by the CGAN model. Hence, the direct image representation similar to that shown is Fig. 1 is ill-equipped for SRAF generation using CGAN.

With this in mind, we propose using a special encoding scheme, typically used in keypoint estimation [14]–[17], that can overcome the aforementioned limitations. The proposed scheme is based on multi-channel heatmaps which associates each object type with one channel in the image [16], [17]. Specifically, a multi-channel image is a simple representation where the number of channels is equal to that of the object types in the problem. On each particular channel, the first step is to obtain the locations of its corresponding objects in the original image. Next, a Gaussian noise circle is centered at the obtained locations on the channel [16], [17].

To elaborate on this, we consider the example shown in Fig. 2 where an original layout is shown in Fig. 2a and the multi-channel heatmap representation is shown in Fig. 2b. In this example, we limit the number of channels to 3 to visualize the encoded representation through a red-green-blue (RGB) image. These three types are : (i) target patterns (in red), (ii) horizontal SRAFs (in green) and (ii) vertical SRAFs (in blue). Similar encoding can be done for images in D_{Trgt} where only one non-empty channel contains the target patterns.

The representation shown in Fig. 2 has two main advantages. First, learning sharp edges, which is a hard task in GANs, is not needed. Instead, training-friendly Guassian objects are used to encode the objects in the original image. Secondly, and most importantly, with this representation the images generated by the CGAN model can be easily mapped back to layout files. In practice, since each channel represents a well-defined type of SRAFs, it suffices to detect the location



(a) (b)

Fig. 2 The multi-channel heatmaps encoding process is demonstrated where (a) shows an original layout representation and (b) shows the encoded representation.

of excitations in the channel to get the locations of SRAFs of this particular type in the layout file.

Therefore, prior to the CGAN training step, images in the training set are all encoded into multi-channel heatmap representation. In such mapping, the number of channels is equal to $M+1$ where M is the number of SRAF types and one additional channel is used to encode the target patterns.

B. CGAN Training

In their essence, GANs were proposed as generative models that learn a mapping from a random noise vector z to output image y, $G : z \rightarrow y$ [11]. Later, different versions of GANs, tailored towards specific domain and applications, were proposed. Among those are the CGANs which, in contrast with original GANs, learn a mapping from an observed image x and random noise vector z, to y, $G : \{x, z\} \rightarrow y$. As in the case of most GAN structure, the architecture of a CGAN is composed of two main components: the *generator* and the *discriminator*. The generator G is trained to produce images in D_{SRAF}, based on an input image in D_{Trgt}, that cannot be distinguished from "real" images by an adversarially trained discriminator, D, which is trained to do as well as possible at detecting the generator "fakes". The overview of the training procedure is described in Fig. 3 [8], [9]. In this work we adopt the CGAN structure proposed in [9] for the image translation task.

Mathematically, the loss function used for training the CGAN can be given as [8], [9]:

$$\begin{aligned}\mathcal{L}_{CGAN}(G,D) &= \mathbb{E}_{x,y}[\log D(x,y)] \\ &+ \mathbb{E}_{x,z}[\log(1 - D(x, G(x,z)))] \\ &+ \lambda_{L1}\mathbb{E}_{x,z,y}[||y - G(x,z)||_1].\end{aligned} \tag{1}$$

In (1), the first two terms represent the traditional GAN loss function where G tries to minimize the objective against an adversarial D that tries to maximize it [8], [9], [11]. The third term in the equation affects only the generator whose objective is, not only to fool the discriminator, but also to generate images close to the ground truth. Here, L_1-norm is used because it encourages less blurring when compared to L_2-norm [9].

Hence, the optimal Generator can be obtained by solving for the following objective:

$$G^* = \arg\min_G \max_D \mathcal{L}_{CGAN}(G,D). \tag{2}$$

In practice, experiments shown in [9], [18] have demonstrated that the noise z is typically ignored by the generator. Hence, noise is instead introduced through dropout on several layers of the generator in both the training and inference stages.

In the next subsections, the details of both the generator and discriminator used in image to image translation CGAN are

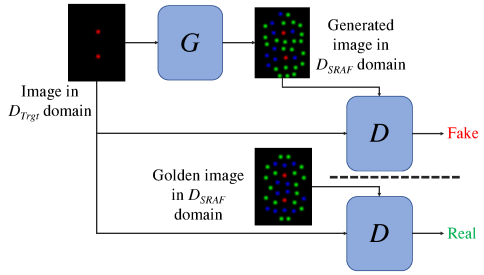


Fig. 3 An overview of the CGAN functionality is shown.

Layer	Output Dimension	Channels
Input	256x256	3
Conv1	128x128	64
Conv2	64x64	128
Conv3	32x32	256
Conv4	16x16	512
Conv5	8x8	512
Conv6	4x4	512
Conv7	2x2	512
Conv8	1x1	512

TABLE I The details of the encoder network in the generator are presented.

Layer	Output Dimension	Channels
Input	256x256	6
Conv1	128x128	64
Conv2	64x64	128
Conv3	32x32	256
Conv4	16x16	512
FC	1	1

TABLE II The details of the discriminator network are presented.

shown in addition to the training and inference process. These implementations are adapted from the deep convolutional generative adversarial networks framework proposed in [19].

1) Generator: The conventional generator in a GAN is basically an encoder-decoder scheme where the input is passed through a series of layers that progressively downsample the input (i.e, encoding), until a bottleneck layer, at which point the process is reversed (i.e, decoding) [8], [9], [11], [19]. This process is shown in Fig. 2a where the gray (white) layers form the encoder (decoder) respectively.

For image translation tasks using CGAN, a significant amount of information is shared between the input and the output, and it would be desirable to shuttle this information directly across the net without passing through the bottleneck layer. Towards this goal, skip connections are added following the general shape of a U-Net [9], [20]. As shown in Fig. 2a, skip connections are added between each layer i and layer $L-i$, by simply concatenating all channels at layer i with those at layer $L-i$, where L is the total number of layers.

Table I lists the characteristics of the layers in the encoder [9]. In all convolutions layers (Conv1-8), (5×5) filters are used with stride 2, and leaky relu is used as an activation function [8], [9], [19]. On the other hand, the decoder is simply a mirrored image of the encoder with deconvolutional layers replacing the convolutional layers.

2) Discriminator: Practically, the discriminator is a convolutional neural network whose objective is to classify "fake" and "real" images. Hence, its structure differs from that of the generator. Table II summarizes the different layers of the discriminator which constitutes of 4 convolutional layers (Conv1-4) and one fully connected layer (FC) whose output is the binary classification results [9]. Similar to the generator, all convolutional filters are of size (5×5) with stride 2.

3) CGAN Training and Inference: Training a CGAN model follows the typical procedure used for training GAN with mini-batch Stochastic Gradient Descent (SGD) and Adam solver [11], [21]. The training alternates between one gradient

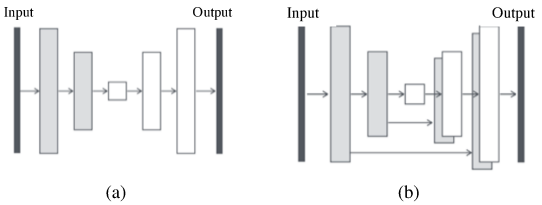


Fig. 4 Two generator network schemes are shown: (a) shows the conventional approach while (b) shows the Unet scheme with skips [9], [20]

descent step on the discriminator, and then one step on the generator. During inference, the generator net is used in the same manner as during the training phase. Indeed, the dropout steps introduced in the layers of the decoder are also used during the inference time. Moreover, batch normalization is applied using the statistics of the test batch where a batch size of 4 is used for the SRAF generation task.

C. CGAN Results Decoding: Heatmap to Layout

The output of the CGAN for SRAF generation is a layout image in a multi-channel heatmap representation as in the example of Fig. 2b. Hence, a decoding step is required to extract the SRAF information from the encoded image. Here, it is important to note that the choice of the encoding scheme was made with this SRAF extraction task in mind. The objective of this step is to extract both the types (i.e, sizes) and locations of the SRAF to be generated on the layout. As described in Section III-A, the SRAF scheme in encoded such that each channel of the image contains one type of SRAFs. Hence, for each channel, it suffices to get the locations of excitations on the heatmap to get locations of the SRAFs of a particular type. Therefore, the task reduces to detecting the excitations on each channel.

Towards the goal of parsing the heatmap, we start from conventional methods used in parsing heatmaps in the field of keypoint estimation [14], then tailor these methods to the SRAF generation task at hand. Knowing that the encoding scheme uses a Gaussian circle centered at each keypoint location, it is expected that the exact SRAF location possess the highest magnitude compared to its neighbors. Therefore, on each channel, a fixed size window is swept over all non-zero pixels on the map where the value of the center pixel is compared to that of all other pixels in the window. The center pixel is considered an SRAF location only if it possess the highest magnitude among all pixels in the window. This step constitutes the core of the decoding procedure. In addition, two filtering stages are used to reduce the effect of noise and false alarms.

In the first step, and before performing the window sweeping operation, the image generated from the CGAN is passed through a filtering stage with a fixed threshold that sets all pixels with values below the threshold to zero. This step helps reduce the effect of noise present in the generated image. On the other hand, a checking step accompanies the core window screening step mentioned above to discard *isolated* pixels. In other words, if a single pixel in a window has a non-zero value, it is more likely a noise pixel than a legit SRAF location. This is mainly because SRAF locations are expected to be encoded through a Gaussian circle, not a single pixel excitation. When the center pixel in a window possess the highest value compared to other pixels, the number of its immediate non-zero neighbors is examined. If a majority of the neighboring pixels are non-zeros, the pixel is considered a legitimate SRAF; otherwise, the pixel is considered an *isolated* pixel and hence a false alarm.

Algorithm 1 CGAN Results Decoding

Require: An image I with dimensions $(N \times N \times M)$, a widow size w, ϵ_1 and ϵ_2

1: Initialize $\{\Omega_m \leftarrow \emptyset : m = 1, 2, \ldots, M\}$
2: Set all pixels less than ϵ_1 in I to 0
3: **for** $m = 1, 2, \ldots, M$ **do**
4: **for** $i = w, w+1, \ldots, N-w$ **do**
5: **for** $j = w, w+1, \ldots, N-w$ **do**
6: **if** $I_{i,j,m} > 0$ and nnz$(I_{i\mp1,j\mp1,m}) > \epsilon_2$ **then**
7: **if** $I_{i,j,m} = \max I_{i\mp w,j\mp w,m}$ **then**
8: $\Omega_m \leftarrow \Omega_m \cup (i, j)$
9: **end if**
10: **end if**
11: **end for**
12: **end for**
13: **end for**
14: **return** $\{\Omega_m : m = 1, 2, \ldots, M\}$.

The decoding scheme is summarized in Algorithm 1. The input is a multi-channel heatmap encoded image I generated from the CGAN model with the channel carrying the target pattern dropped. Algorithm 1 also requires the size of the scanning window as an input in addition to the two threshold values ϵ_1 and ϵ_2 that are used for the filtering stage and the neighborhood check respectively. As a first step, thresholding is done using ϵ_1 (line 2). Next, for each channel in the image, all pixels are scanned, and at each location three conditions are checked: (i) if the value of the pixel is nonzero, (ii) if the number of its non-zero (nnz) neighbors is greater than ϵ_2 (line 6) and if its value is the maximum in the window (line 7). If all three conditions are satisfied at a particular (i, j, m) location, the coordinates (i, j) are added to Ω_m. The Algorithm returns the sets $\{\Omega_m : m = 1, \ldots, M\}$, where each set Ω_m contains the locations of SRAFs of type m. These sets contain all necessary information to generate a layout clip similar to the one in Fig. 2a. In practice, Algorithm 1 can be significantly accelerated with massive parallelization, since each pixel can be checked independently. Therefore, we develop a custom CUDA accelerator for this process and integrate it to GAN-SRAF.

D. Post Processing

The decoding procedure in Algorithm 1 generates a properly formatted layout clip. However, this clip is not guaranteed to follow all SRAF manufacturing rules such as minimum spacing [3]. Hence, a final legalization step is employed to ensure that all design rules are satisfied. In this work, we adopt the same greedy simplification scheme proposed in [3] to accommodate the mask manufacturing rules.

IV. Experimental Results

A. Data Description and CGAN Training

To train the CGAN model, a training dataset containing 1620 layout clips is used. From each clip, two images are created using the multi-channel heatmap encoding presented

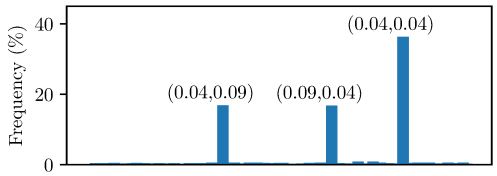


Fig. 5 The distribution of SRAF sizes in the dataset is shown.

Parameter	Learning rate	β_{ADAM}	# of epochs	Batch size	λ_{L1} in (1)
value	0.0002	0.5	100	4	100

TABLE III A summary of CGAN training parameters is shown.

in Section III-A. The first image is in D_T, where only target patterns are present, while the second is in D_S where both target patterns and SRAFs are present. The SRAFs in the layout clips are generated using model-based SRAF generation in Mentor/Calibre with the same setup as [3] and are considered the golden solution when training the CGAN model. This setup corresponds to memory design and is used for contact generation. In practice, it matches closely intels 14nm process (for P1272-CPU) with minimum spacing and minimum width for contacts set to 70nm (pitch=140nm).

It is shown in Fig. 5 that a few SRAF shapes have significantly higher frequency than the remaining ones. In fact, SRAFs with sizes $(0.04, 0.09)$, $(0.09, 0.04)$, and $(0.04, 0.04)$ are evidently dominant. Therefore, assigning a separate channel to encode each of the total 101 unique shapes will be too expensive and of small return. Hence, it makes more sense to map all shapes to a small set of categories based on the most dominant ones. By examining the distribution in Fig. 5, three major categories can be intuitively obtained based on the three dominant shapes.

However, keeping in mind that the target contact will occupy one channel in the image, we desire to restrict the SRAF types to two for the evident reason that a 3-channel image can be easily visualized. In fact, we observe that even with only $(0.04, 0.09)$ and $(0.09, 0.04)$ SRAF shapes, the final SRAF solutions after post-processing could still contain SRAFs of size $(0.04, 0.04)$ to satisfy the mask constraints. Hence, we choose to map SRAFs in the layout clips to two types (shapes) only: (i) $(0.09, 0.04)$ and (ii) $(0.04, 0.09)$. All other shapes are mapped to one of these values based on their similarity. For the square patterns, they are mapped to one of the two categories randomly. In total, three channels are used to encode each clip in the given dataset where one channel is used to encode target patterns and the other two represent the two types of SRAFs. However, note that the proposed approach is general, and the number of channels to be used can be set by the user depending on the data. With the prepared dataset, CGAN is trained with the setup in Table III.

B. SRAF Generation

To demonstrate the efficacy of our proposed approach, we compare the layouts generated from our proposed approach (denoted CGAN) with (i) those obtained from the local sampling scheme approach presented in [3] where Support Vector Machine is used as the classification model (denoted LS_SVM) and (ii) those obtained from model-based SRAF generation (denoted MB). For comparison, a separate testing dataset with 404 clips is used. The performance of the three different methods, under the same setup, can be visualized

	No SRAF	MB	LS_SVM [3]	CGAN
PV band (*0.001 um^2)	3.354	2.845	3.009	2.916
EPE (nm)	3.9287	0.5270	0.50669	0.5410
Runtime (s)	-	6910	700	48

TABLE IV The comparison of evaluation metrics and run time across different SRAF generation schemes is shown.

in Fig. 6 which shows the result for SRAF generation using MB (Fig. 6a), CGAN (Fig. 6b), and LS_SVM (Fig. 6c) for two layout clips in the testing dataset. For both CGAN and LS_SVM the "prediction" label denotes the results of SRAF generation prior to the post processing step described in III-D. Particularly, for the case of CGAN, these predictions are the results of Algorithm 1. The final legal layout scheme after post processing is labeled "Final".

By examining the results in Fig. 6, and recalling that both CGAN and LS_SVM are trained against the model-based SRAF results as the golden solution, one can easily conclude that the "predicted" SRAFs from CGAN mimic the MB results much better than the "predicted" SRAFs from LS_SVM. Moreover, comparing the "Final" SRAFs obtained from CGAN and the MB results shows that, although the two do not match exactly, the CGAN has captured the systematic way of generating the SRAFs in a model-based approach. In addition, one can notice a relatively small difference between the predicted SRAFs and final SRAFs for the case of CGAN when compared to those of LV_SVM. This is due to the fact that the CGAN generated SRAF scheme is very close to the legal layout; hence, minimal changes are needed in the post processing stage. On the other hand, LS_SVM generates a clip of clustered SRAFs which requires intensive post-processing before obtaining a legal layout.

C. Lithography Compliance Check and Run Time

To quantify the quality of our proposed approach, we integrate the SRAF generation with a complete mask optimization flow using Mentor Calibre. The three different generation schemes are compared in terms of PV band and EPE. For each contact, the PV band value is measured, and the EPE value at the center of the edges at nominal conditions is considered. Table IV summarizes the the mean absolute values of the two metrics. The table also includes the PV band and EPE evaluations with no SRAFs to better demonstrate the performance gain achieved through SRAF. In practice, the most important metric of evaluation is the PV band (smaller is better) [3], and as demonstrated by the results, CGAN can achieve better PV band when compared to LS_SVM [3] which demonstrates a superior performance in terms of SRAF insertion quality. This is in fact consistent with our observation in Section IV-B where SRAF schemes generated from CGAN can better mimic the MB generated schemes. Hence, the PV band obtained from CGAN is closer to that of MB. On the other hand, despite that fact that LS_SVM can achieve better results in terms of EPE, EPE can be further improved with better OPC [22]; hence, it is not the best metric used to judge upon the quality of SRAF generation [3]. In general, the three SRAF generation schemes -MB, LS_SVM, and CGAN- achieve comparable results in terms of lithography evaluation metrics with CGAN demonstrating suporior results compared to LS_SVM. This can be better seen when examining the histograms in Figs. 7 and 8 showing the distribution of EPE and PV band respectively across all clips

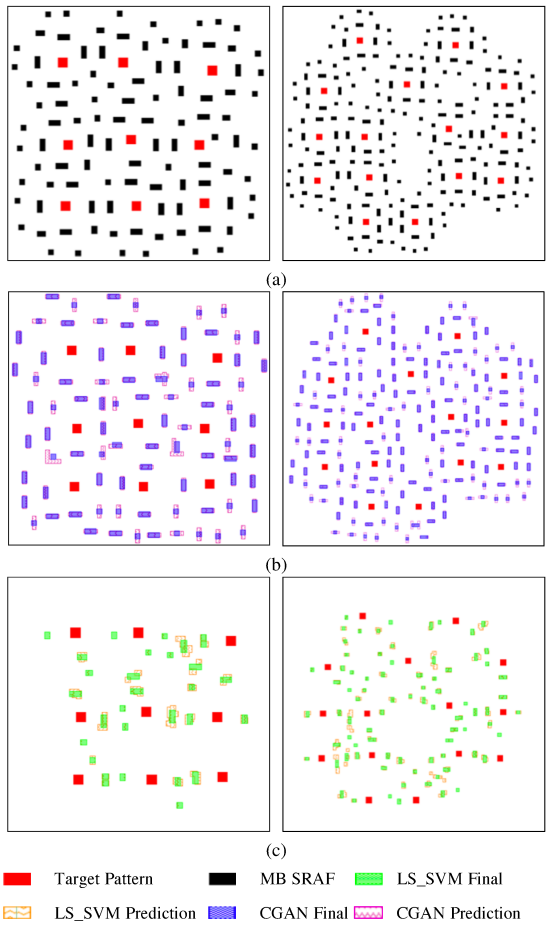


Fig. 6 The results of SRAF generation for two clips in the test data using MB (a), CGAN (b) and LS_SVM (c) are shown.

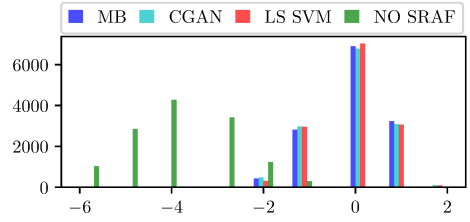


Fig. 7 The comparison of EPE distribution across different SRAF generation schemes is shown.

in the testing dataset. The figures clearly show that the CGAN performance is comparable to that of LS_SVM and MB.

Most importantly, considering the runtime of generating SRAFs for all clips in Table IV, CGAN can achieve $\sim$14.6$\times$ runtime reduction when compared to LS_SVM while achieving better PV band, and $\sim$144$\times$ when compared to model based approach while achieving comparable results.

V. Conclusion

In this paper, a novel SRAF generation framework, *GAN-SRAF*, is presented based on conditional generative adversarial neural networks. We propose an effective encoding scheme to represent the layout information using a multi-channel heatmaps and a GPU-accelerated decoding scheme

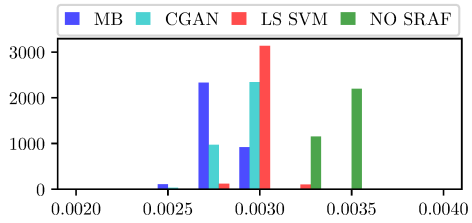


Fig. 8 The comparison of PV band distribution across different SRAF generation schemes is shown.

for extraction of SRAF solutions. The experimental results demonstrate that GAN-SRAF achieves $\sim$14.6$\times$ reduction in computational cost compared to state-of-art machine learning SRAF generation approaches, and $\sim$144$\times$ when compared to model-based approach, while achieving comparable quality.

VI. Acknowledgement

This work is supported in part by NSF under Award No. 1718570 and Toshiba Memory. The authors would like to thank Dr. Xiaoqing Xu for the helpful discussions.

References

[1] R. Viswanathan, J. T. Azpiroz, and P. Selvam, "Process optimization through model based sraf printing prediction," in *Proc. SPIE*, 2012, pp. 83 261A–83 261A.

[2] K. Sakajiri, A. Tritchkov, and Y. Granik, "Model-based sraf insertion through pixel-based mask optimization at 32nm and beyond," in *Proc. SPIE*, 2008, pp. 702 811–702 811.

[3] X. Xu, Y. Lin, M. Li, T. Matsunawa, S. Nojima, C. Kodama, T. Kotani, and D. Z. Pan, "Sub-Resolution Assist Feature Generation with Supervised Data Learning," *IEEE TCAD*, vol. PP, no. 99, 2017.

[4] J.-H. Jun, M. Park, C. Park, H. Yang, D. Yim, M. Do, D. Lee, T. Kim, J. Choi, G. Luk-Pat *et al.*, "Layout optimization with assist features placement by model based rule tables for 2x node random contact," in *Proc. SPIE*, 2015, pp. 94 270D–94 270D.

[5] C. Kodama, T. Kotani, S. Nojima, and S. Mimotogi, "Sub-resolution assist feature arranging method and computer program product and manufacturing method of semiconductor device," Aug. 19 2014, US Patent 8,809,072.

[6] B.-S. Kim, Y.-H. Kim, S.-H. Lee, S.-I. Kim, S.-R. Ha, J. Kim, and A. Tritchkov, "Pixel-based sraf implementation for 32nm lithography process," in *Proc. SPIE*, 2008, pp. 71 220T–71 220T.

[7] Y. Lin, M. B. Alawieh, W. Ye, and D. Pan, "Machine learning for yield learning and optimization," in *Proc. ITC*, 2018.

[8] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[9] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *arxiv*, 2016.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.

[11] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. NIPS*, 2014, pp. 2672–2680.

[12] B. Wu, H. Duan, Z. Liu, and G. Sun, "SRPGAN: perceptual generative adversarial network for single image super resolution," *CoRR*, vol. abs/1712.05927, 2017.

[13] H. Yang, S. Li, Y. Ma, B. Yu, and E. F. Young, "Gan-opc: mask optimization with lithography-guided generative adversarial nets," in *Proceedings of the 55th Annual Design Automation Conference*. ACM, 2018, p. 131.

[14] X. Zhou, A. Karpur, C. Gan, L. Luo, and Q. Huang, "Unsupervised domain adaptation for 3d keypoint prediction from a single depth scan," *CoRR*.

[15] S. Tulsiani and J. Malik, "Viewpoints and keypoints," *Proc. CVPR*, pp. 1510–1519, 2015.

[16] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *ECCV*, 2016.

[17] J. Tompson, A. Jain, Y. LeCun, and C. Bregler, "Joint training of a convolutional network and a graphical model for human pose estimation," in *NIPS*, 2014.

[18] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," *CoRR*, 2015.

[19] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *CoRR*.

[20] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, 2014.

[22] C. Mack, *Fundamental Principles of Optical Lithography: The Science of Microfabrication*. John Wiley & Sons, 2008.