2025 International Symposium of Electronics Design Automation (ISEDA) | 979-8-3315-3696-1/25/$31.00 ©2025 IEEE | DOI: 10.1109/ISEDA65950.2025.11100384

Efficient SRAF Generation via Diffusion Models

Minjie Bi[1], Yang Luo[1], Xiaoxiao Liang[1], Zhengzhou Gu[2*], Jinbin Deng[2], Binwu Zhu[3], Yuzhe Ma[1*]
[1]The Hong Kong University of Science and Technology (Guangzhou)
[2]Shenzhen GWX Technology Co., Ltd. [3]Southeast University
[*]*Corresponding author*
Email: {mbi218,yluo208,xliang603}@connect.hkust-gz.edu.cn
{keven,dengjb}@gwxeda.com, bwzhu@seu.edu.cn, yuzhema@hkust-gz.edu.cn

***Abstract*—In semiconductor manufacturing, optical proximity correction and sub-resolution assist features (SRAF) are critical techniques for achieving high-fidelity wafer images, especially as semiconductor device critical dimensions shrink. However, traditional SRAF generation methods often face challenges in scalability, adaptability, and efficiency. This paper introduces a novel method that uses a conditional generative diffusion model for SRAF generation to improve efficiency and flexibility. We treat the SRAF generation task as an image-to-image translation problem, converting the input layout to include optimal assist features. Experimental results show that our proposed approach achieves a 5.57× speed-up over commercial tools while maintaining comparable accuracy in terms of edge placement error and process variation band.**

***Index Terms*—mask optimization, sub-resolution assist feature, EDA, diffusion model**

I. Introduction

Lithography is a cornerstone of microelectronics manufacturing, with its precision and resolution directly impacting chip performance and integration density. However, as the advanced semiconductor critical dimensions (CD) approach the wavelength of the light, optical diffraction becomes more pronounced. Interactions between smaller and closer features can also distort the intended pattern due to the proximity effect [1]. Optical proximity correction (OPC) is a resolution enhancement technique (RET) used to address these issues. By adjusting the shapes and positions of features on the photomask, OPC helps ensure that the final pattern on the wafer closely matches the intended design [2], [3]. Besides, sub-resolution assist feature (SRAF) insertion is also one representative technique to enhance the image quality, which adds SRAFs surrounding the primary pattern, allowing isolated and sparse patterns to receive spatial frequency augmentation. As shown in Fig. 1, compared to basic OPC results that do not incorporate SRAFs, these inserted features significantly enhance the robustness of the target patterns under various process variations, reducing the process variation (PV) band.

In general, SRAF insertion methodologies are categorized into three classes: the rule-based approach [4], [5], the model-based [6]–[8], and machine learning (ML)-based method [9]–[12]. The traditional rule-based SRAF approach relies on a set of predefined guidelines and heuristics to determine the placement of assist features, offering simplicity and ease of implementation. However, rule-based methods may lack the flexibility to adapt to novel patterns or process variations, potentially leading to suboptimal results as technology nodes advance. In contrast, model-based SRAF methodologies utilize detailed simulations of the lithographic process to optimize assist feature insertion. By employing optical models that take into account factors such as diffraction and interference, model-based approaches can achieve higher accuracy in predicting and mitigating proximity effects. While more robust than rule-based techniques, model-based methods often involve complex computations and longer runtimes, posing challenges in terms of scalability and computational efficiency. ML-based approaches have been widely explored for SRAF generation by leveraging historical pattern features to train classification or generative models. For instance, [9] proposed a framework using decision trees and logistic regression to classify SRAF location based on constrained constraint circle with area sampling (CCAS). The predicted probability maps were then used to determine the SRAF locations. As one of the popular generative models, generative adversarial network (GAN) [13] has also been applied to treat SRAF insertion as a keypoint detection problem [11], utilizing heatmap-based feature encoding. Additionally, dictionary learning [14] has been employed to compress features extracted via CCAS [15] and improve their discriminability, combined with integer linear programming (ILP) to optimize SRAF patterns under predefined constraints [14]. While these approaches have demonstrated promising results, they often face limitations in terms of generalization, adaptability, and the need for extensive feature engineering.

In this work, we introduce a novel diffusion model-based approach for SRAF generation, without extra image preprocessing, by formulating the problem as image-to-image translation. By leveraging the generative capabilities of diffusion models [16], our method captures complex spatial correlations in SRAF patterns while achieving improved robustness and generalization across different designs. Specifically, we construct the SRAF generation task as a paired learning problem, where each target pattern is associated with a corresponding SRAF pattern. During training, given a target pattern as input, we progressively add noise to the pattern until it becomes pure Gaussian noise. Conversely, during

This work is supported by the National Natural Science Foundation of China (No. 62204066).

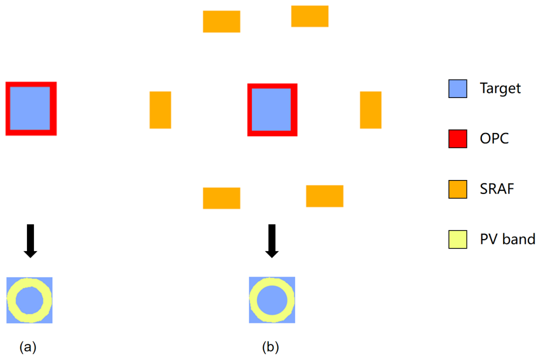

Fig. 1 (a) Printing with OPC only; (b) Better printing with both OPC and SRAF.

inference, we sample from Gaussian noise and utilize the target pattern as a conditional input to guide the denoising process. Through iterative denoising, the model ultimately generates an optimal SRAF pattern that adheres to the required lithographic constraints. Our main contributions are summarized as follows:

- We formulate the SRAF insertion problem as an image-to-image translation task where the layout is transformed to the SRAF domain.
- We introduce a new method by using a conditional generative diffusion model to address the trade-off between local sensitivity and global consistency in SRAF insertion.
- Compared to denoising diffusion probabilistic model (DDPM) [16], the experimental results show that our proposed framework achieves over $10\times$ speed-up.
- Compared to a commercial tool, the experimental results show that our proposed framework achieves $5.57\times$ speed-up while achieving comparable results.

The rest of this paper is organized as follows. Section II shows the preliminaries. Section III gives the details of our method, and Section IV presents experimental results, followed by a conclusion in Section V.

II. Preliminary

A. Diffusion Model

The diffusion model is a probabilistic generative model, which is mainly divided into the forward diffusion stage and the reverse denoising stage. The training process of diffusion model is much more stable than GANs [17], which often suffer from mode collapse and sensitivity to hyperparameters. Compared to variational autoencoder(VAE) [18], diffusion model can generate more diverse and higher fidelity samples. DDPM [16] is one of the mainstream diffusion models. The forward diffusion process of DDPM is to generate a series of noise samples $\boldsymbol{x}_0$ following Gaussian distribution by giving an undamaged data sample $\boldsymbol{x}_0 \sim q(\boldsymbol{x}_0)$ sampled from the real data distribution, and progressively add it to the input sample to get the following Markov process:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I}) \quad \forall t \in (1, ..., T) \tag{1}$$

Where T is the total diffusion step, $\beta_t \in (0, 1)$ is the predefined noise schedule, $\boldsymbol{I}$ represents the identity matrix with the same dimension as the input sample $\boldsymbol{x}_0$, $\mathcal{N}$ represents the normal distribution of the mean and variance of $\boldsymbol{x}_t$ generated, and the samples of each step are only the samples at time $t-1$.

The forward process variance parameter is chosen to be a linearly increasing constant from $\beta_1 = 10^{-4}$ to $\beta_T = 0.02$, formally:

$$\beta_t = \frac{10^{-4}(T-t) + 0.02(t-1)}{T-1} \tag{2}$$

The reverse propagation process starts from $\boldsymbol{x}_T$ sampled from Gaussian distribution and denoises it through the opposite steps. It is parametrized by θ and the joint distribution can be defined by:

$$p_\theta(\boldsymbol{x}_{0:T}) = p(\boldsymbol{x}_T) \prod_{t=1}^{T} p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) \tag{3}$$

Starting from $p(\boldsymbol{x}_T) = \mathcal{N}(\boldsymbol{x}_T; 0, \boldsymbol{I})$, the reverse process transforms the latent variable distribution $p_\theta(\boldsymbol{x}_T)$ to the data distribution $p_\theta(\boldsymbol{x}_0)$. And the reverse process works by progressively removing the noise from the noisy data $\boldsymbol{x}_t$ at each timestep t, using these learned parameters to estimate the clean data $\boldsymbol{x}_{t-1}$:

$$p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x}_{t-1}; \mu_\theta(\boldsymbol{x}_t, t), \sigma_\theta(\boldsymbol{x}_t, t)) \tag{4}$$

Where $\mu_\theta(\boldsymbol{x}_t, t)$ is the learnable mean and $\sigma_\theta(\boldsymbol{x}_t, t)$ is the learnable variance. In order to make the reverse Markov chain match the forward process as well as possible, θ needs to be continuously adjusted so that the joint distribution of the reverse process gradually approaches the forward process.

B. Evaluation Metric

We introduce several metrics to evaluate the performance of mask optimization results. Different generation schemes are compared in terms of process variation (PV) band, edge placement error (EPE), and runtime. PV band means the area between the outer contour and inner contour for the lithography simulation contours at a set of focus, dose conditions. EPE means the distance between the target pattern contour and nominal contour for the lithography simulation contour at the best focus, dose condition.

C. Problem Formulation

When the diffusion model is combined with SRAF generation, the input to the model will be a layout clip containing only the target pattern. This input clip represents the initial design that needs to be optimized for lithography. We want to output a new layout clip from the processing of the already trained diffusion model, where the SRAF has been generated for the target pattern. Our diffusion model is trained to insert SRAF into images based on the image format training data.

III. Proposed Method

Our proposed diffusion model for SRAF generation encodes the input image with the U-Net encoder [19], [20]. The extracted features are summed with the feature map of the condition image. At the same time, the U-Net encoder and

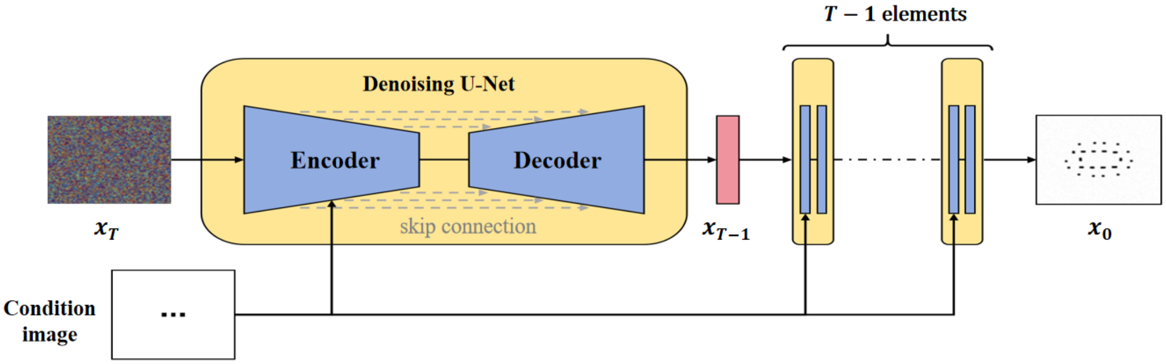

Fig. 2 The proposed workflow to generate SRAF by diffusion model

decoder are used to refine the estimated SRAF layout. The workflow of our diffusion model is shown in Fig. 2.

A. Data Preparation

In the process of data preparation, since there is no open source image dataset with both target pattern and SRAF, we used commercial software to generate SRAF for our target pattern first, so as to obtain a set of training datasets. To address this, we choose the Lithobench dataset [21] as the basis for our research. The Lithobench dataset is a high-quality dataset designed for lithography processes that provide detailed vertex coordinate information on a variety of target patterns. In our study, we first extract the vertex coordinates of the target pattern from the Lithobench dataset. These coordinates are from the GLP file in the dataset. Each line in the file corresponds to the four vertices of a rectangular target pattern. We can use commercial tool to read these files and output the GDSII files with the target patterns corresponding to them. Through these steps, we are able to accurately reconstruct the target pattern dataset, ensuring that the geometric properties of the pattern are consistent with the design in the actual lithography process. Next, we use the commercial tool to generate SRAF of these target patterns. In this step, each target layout gets a corresponding SRAF generation. Because the commercial tool can only read GDSII files and output the corresponding generated layout, while DDPM networks generally receive image information, we need to convert the target pattern and the corresponding SRAF (Fig. 3) into an image after visualization with a unified standard. Therefore, prior to the DDPM training step, we have two image datasets for DDPM training, D_{tar} and its corresponding D_{SRAF}.

B. Training

In essence, DDPM is a type of generative model that aims to generate realistic data samples by progressive denoising from a noise distribution. It is initially an unconditional model and is relatively simple to apply because it does not require any additional information for conditionality. However, this also means that the generated samples are entirely dependent on randomness within the model and cannot be generated for orientation or specific features. Conditional DDPM builds upon the foundation of unconditional models by incorporating additional input information, such as class labels, textual descriptions, or other auxiliary data, into the denoising process. In practice, conditional DDPM explicitly incorporates conditional variables into the denoising network, allowing the model to consider these conditions during the generation process and produce samples that meet specific criteria. The advantage of conditional models is their ability to generate more guided samples. This feature makes conditional DDPM particularly valuable in tasks that require precise control over the generated content. In existing image generation tasks, the condition input to DDPM is generally set to the class label. However, in our experiment, since the generation of SRAF is highly related to the location of the target pattern, it is impossible to set separate label information for each layout, so we decided to modify the condition input interface of DDPM to accept a single binary image, which means a two-dimensional matrix, as condition input. Algorithm 1 displays the complete training procedure with DDPM learning SRAF generation. In our training process, the input to DDPM is the images in D_{SRAF}, and D_{tar} will be the condition to control the direction of image generation. In step $2 \sim 4$, $\boldsymbol{x}_k$ is randomly sampled from D_{SRAF}, and the corresponding target pattern is extracted from D_{tar}. A time t is randomly sampled from 1 to T to represent the level of added noise. Step 5 random sampling of a two-dimensional Gaussian noise. Step $6 \sim 7$ calculate the noise at time t and combine it with $\boldsymbol{x}_k$; The neural network is trained to predict the noise added to $\boldsymbol{x}_k$ based on the information in $\boldsymbol{x}_t$ and $\boldsymbol{C}_k$ after the noise

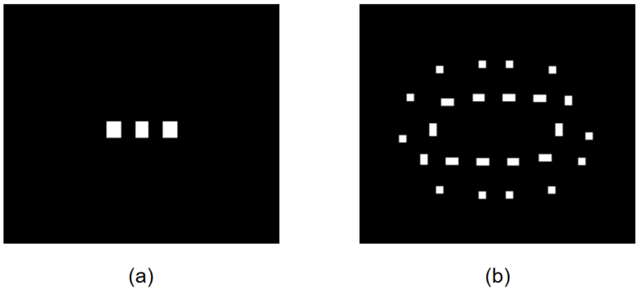

Fig. 3 SRAF generation task can be cast as an image translation problem where layout with target contacts (a) is translated to SRAF (b)

Algorithm 1 Training

```
Input: Total diffusion steps $T$, target and SRAF image
    datasets $D_{tar}$, $D_{SRAF}$
1: repeat
2:     $\boldsymbol{x}_k \sim D_{SRAF}$          ▷ Sample an image with SRAF pattern
3:     $\boldsymbol{C}_k \sim D_{Tar}$          ▷ Corresponding target pattern
4:     $t \sim \text{Uniform}(\{1, \dots, T\})$
5:     $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{n \times n})$
6:     $\bar{\alpha}_t = \prod_{i=1}^{t}(1 - \beta_i)$
7:     $\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_k + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}$          ▷ Add noise
8:     Take gradient descent step to $\mathcal{L}_{simple}$
9: until converged
```

Algorithm 2 Sampling

```
Input: Total diffusion steps $T$, target image $\boldsymbol{C}$
1: $\boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{n \times n})$
2: for $t = T, \dots, 1$ do
3:     Use Equation (8) to calculate $\boldsymbol{x}_{t-1}$
4: end for
5: return $\boldsymbol{x}_0$          ▷ SRAF prediction
```

is added in step 8.

Mathematically, the loss function used for training the DDPM can be given as [16]:

$$\mathcal{L}_{simple} = \mathbb{E}_{t, \boldsymbol{x}_k, \boldsymbol{C}_k, \boldsymbol{\epsilon}} ||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, \boldsymbol{C}_k, t)||^2 \tag{5}$$

where $\boldsymbol{\epsilon}$ is the added Gaussian noise, $\boldsymbol{x}_k$ and $\boldsymbol{C}_k$ denote the paired conditional inputs and target image, and $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, \boldsymbol{C}_k, t)$ is a neural network that predicts the noise added from time 0 to time t.

In DDPM, a U-Net architecture is employed as the core model for noise prediction. The U-Net is characterized by its distinctive U-shaped structure, composed of an encoder, a decoder, and residual connections that link the two. The encoder's role is to downsample the input image into feature representations, while the decoder upsamples these features to predict the target noise. The skip connections serve to concatenate features from the encoder to the corresponding layers in the decoder, thereby preserving and utilizing contextual information effectively. In our work, $\boldsymbol{\epsilon}_\theta$ can be expressed in the following form:

$$\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, \boldsymbol{C}_k, t) = D(E(F(\boldsymbol{x}_t) + G(\boldsymbol{C}_k), t), t) \tag{6}$$

In this architecture, the U-Net's decoder D is conventional and its encoder is broken down into three networks: E, F, and G. The last encodes the input image, while F encodes the segmentation map of the current step $\boldsymbol{x}_t$. The two processed inputs have the same spatial dimensionality and number of channels. We sum these signals $F(\boldsymbol{x}_t) + G(\boldsymbol{C}_k)$. This sum then passes to the rest of the U-Net encoder E. Within the convolutional modules of the U-Net, we utilize the wide residual network (WRN) [22] as its foundational structure. Compared to standard residual networks, WRNs have fewer layers but more channels, achieving a favorable balance between computational efficiency and representational capacity. For the downsampling and upsampling operations within the U-Net, we employ strided convolutions and transposed convolutions, respectively, both with a stride of 2.

C. Sampling

The DDPM sampling process starts with pure noise $\boldsymbol{x}_t$ and iteratively denoises it using the reverse process. For each time step $t = T, T-1, \dots, 1$, DDPM predict the previous step $\boldsymbol{x}_{t-1}$ by sampling:

$$\boldsymbol{x}_{t-1} = \mu_\theta(\boldsymbol{x}_t, t) + \sigma_\theta(\boldsymbol{x}_t, t)\boldsymbol{\epsilon} \tag{7}$$

This process involves many steps (typically more than 1000) to produce high-quality images. Therefore, we modify the reverse diffusion process to speed up sampling. The improved reverse process is the same as [23]. It uses a deterministic trajectory that removes the randomness and speeds up the generation. This means that it does not need to sample from the Gaussian distribution at each step as DDPM does, but directly calculates the result of denoising with condition information:

$$\begin{aligned} \boldsymbol{x}_{t-1} = \sqrt{\alpha_{t-1}} & \left(\frac{\boldsymbol{x}_t - \sqrt{1 - \alpha_t}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, \boldsymbol{C}, t)}{\sqrt{\alpha_t}} \right) \\ & + \sqrt{1 - \alpha_{t-1}} \cdot \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, \boldsymbol{C}, t) \end{aligned} \tag{8}$$

This sampling procedure is detailed in Algorithm 2. The input to the trained model is the total diffusion steps T and target image $\boldsymbol{C}$. Step 1 randomly samples an input noise from the standard normal distribution. In Step 2 $\sim$ 3, from time T to 1 we use the trained DDPM model to calculate $\boldsymbol{x}_{t-1}$. Step 5 outputs the SRAF prediction for the input target image.

D. Post Processing

Ideally, after training, the DDPM model can be able to output specific SRAF generation for each input target pattern, which means the output is still in the form of a binary image, we need to convert it back to GDSII format. However, this layout is not guaranteed to follow all SRAF manufacturing rules such as minimum spacing. Since we did not specify the shape of the SRAF in the model, the SRAF may not be a perfect rectangle in the resulting image. The pixel-level noise in the generated image also affects the converted GDSII layout. In order to solve these problems, we keep only the SRAFs of sufficient size during the format conversion process and rearrange these SRAFs to ensure that they are rectangular.

IV. Experimental Results

A. Implementation Details

We implement our framework with PyTorch. Both mask optimization and evaluation are conducted under a Linux system equipped with a 2.1GHz Intel Xeon Platinum 8352V 32-core processor and 1 Nvidia 4090 GPU. We selected the via part from the Lithobench dataset [21] as the target pattern. It is a dataset of more than 120,000 via layouts, from which we extracted 10,199 as examples. After using the commercial tool to generate the SRAF for via, we separated the target pattern and the generated SRAF for each result in images with the same size of a high-resolution scale 2048 × 2048.

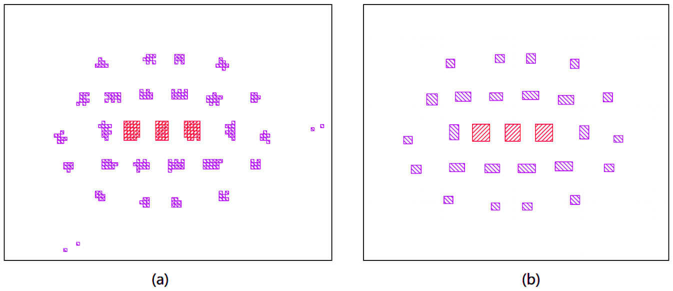

Fig. 4 (a) shows the result without post-processing and (b) shows the results after post-processing

We set the total diffusion step $T = 50$ for all experiments so that the number of neural network evaluations needed during sampling matches other image generation tasks. We set the forward process variances to constants increasing linearly from $\beta_1 = 10^{-4}$ to $\beta_T = 0.02$. These settings are the same as [16] because these constants were chosen to ensure that the reverse and forward processes have approximately the same functional form while keeping the signal-to-noise ratio at $\boldsymbol{x}_T$ as small as possible. The optimization process consists of 20 iterations, adopted with Adam optimizer [24]. Post-processing was required after the training was completed, which was the same as the prediction mentioned above. We first replace all irregular SRAFs with the smallest rectangles that can accommodate them, then eliminate all rectangles whose length and width are both less than $10nm$ (which are regarded as noise), while retaining the target pattern and the rectangular SRAFs that meet the conditions. Fig. 4 shows the effect of the post-processing procedure. Since the final output picture size is set to 256×256, we need to adjust the resolution of the two images to be consistent before combining. In MRC, all the results are successfully verified.

B. Comparison with Commercial Tool

To demonstrate the efficacy of our proposed approach, we compare the layouts generated from the DDPM approach (denoted `Ours`), the proposed efficient approach with improved sampling (denoted `Ours-E`), as well as those obtained from the commercial tool SRAF generation (denoted `CT`). The performance of the two different methods, under the same setup, can be visualized in Fig. 5 which shows the result for SRAF generation using our model and `CT` for two layout clips in the testing dataset.

By examining the results in Fig. 5, we can conclude that although the diffusion model's prediction of SRAF generation does not exactly match the commercial tool results, it has learned the rule of how the commercial tool methods generate SRAF for the target pattern.

To quantify the quality of our proposed approach, we integrate the SRAF generation with a complete mask optimization flow using the commercial tool. We collect the PV band, EPE, and runtime values for each contact and plot the data in histograms as shown in Figs. 6 to 8. Although our generative model is not as good as the commercial tool in EPE performance, it has significantly improved PV band performance. In view of this situation, we use the score function provided in [25] to make a comprehensive evaluation of our model's effect:

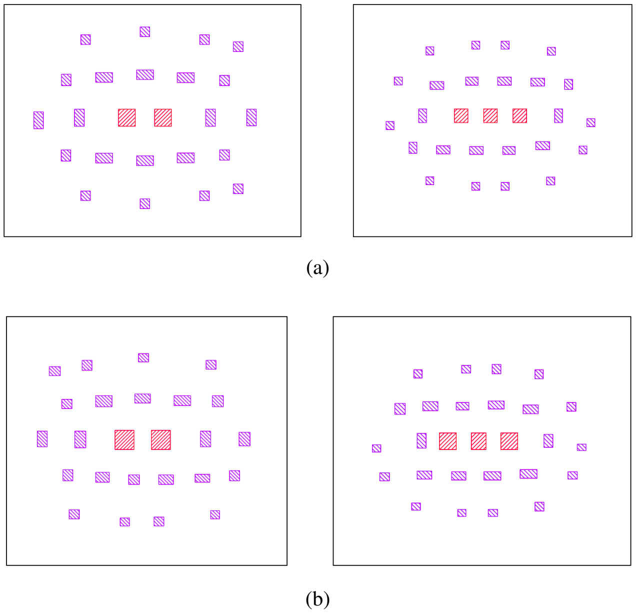

Fig. 5 SRAF generation for two clips in the test data. (a) Commercial tool. (b) Ours.

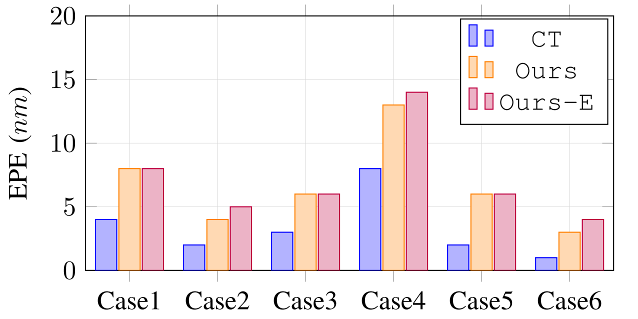

Fig. 6 The comparison of EPE distribution across different cases.

$$Score = Runtime + 4 \times PV\ band + 5000 \times EPE\ violation \tag{9}$$

With the given scoring function, our approach successfully achieves a better result (lower score), which is shown in Fig. 9. In the dataset, the via size is all $70nm \times 70nm$, so it is reasonable to use this dataset for runtime comparison. The average runtime for our model and the commercial tool are 3.2s and 17.83s, respectively. Although we are using a different programming language from the commercial tool, we still obtain over 5× speed-up from machine learning-based SRAF generation compared to the commercial tool. We also check the runtime of the OPC from different SRAF generation approaches and ensure that they are approximately the same.

C. Ablation Study

As mentioned in Section III, we use the improved sampling method to replace the original method in DDPM. Here we treat the variance as a hyperparameter that can be manually

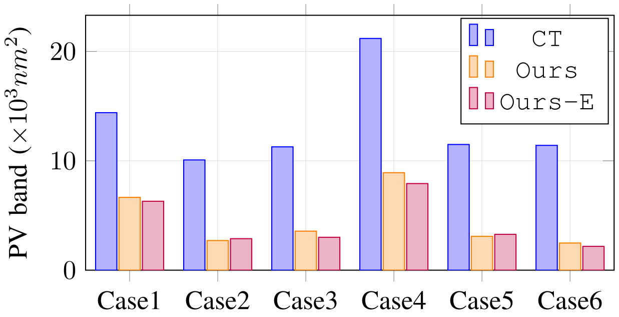

Fig. 7 The comparison of PV band distribution across different SRAF generation cases

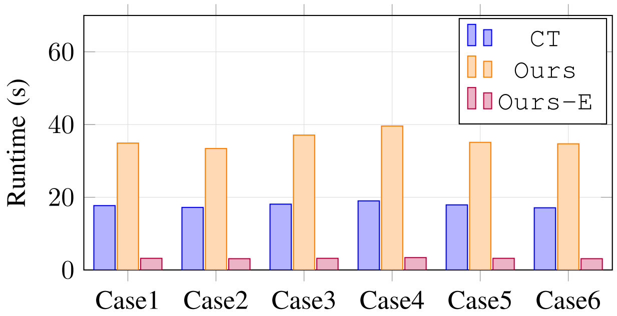

Fig. 8 The comparison of runtime distribution across different SRAF generation cases

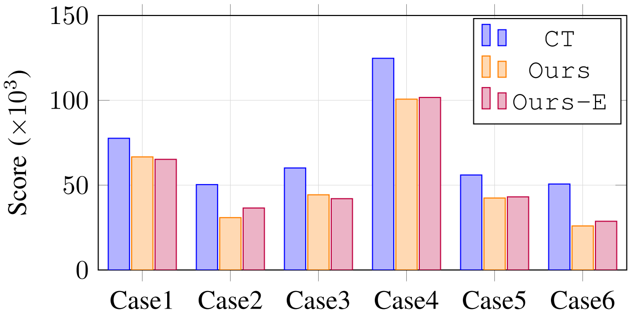

Fig. 9 The comparison of score distribution across different SRAF generation cases

adjusted and set to 0. We hope that the improved method can generate high-quality images in fewer time steps, making the sampling process faster. We also compare the performance in Figs. 6 to 9. The experimental results show that `Ours-E` significantly reduces the runtime by more than 10× while achieving comparable quality.

V. Conclusion

In this paper, a conditional generative diffusion mode 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 the SRAF domain. Experimental results show that our proposed framework achieves more than 5× speed-up over the commercial tool while achieving comparable results.

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*, 2013.

[2] X. Liang, Y. Ouyang, H. Yang, B. Yu, and Y. Ma, "RL-OPC: Mask optimization with deep reinforcement learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2023.

[3] X. Liang, H. Yang, K. Liu, B. Yu, and Y. Ma, "CAMO: Correlation-aware mask optimization with modulated reinforcement learning," in *Proceedings of the 61st ACM/IEEE Design Automation Conference*, 2024.

[4] J. Jun, M. Park, C. ha Park, H. jo Yang, D. gyu Yim, M. Do, D. Lee, T. Kim, J. Choi, G. Luk-Pat, and A. Miloslavsky, "Layout optimization with assist features placement by model based rule tables for 2x node random contact," in *Advanced Lithography*, 2015.

[5] S. Jayaram, P. LaCour, J. Word, and A. Tritchkov, "Effective model-based sraf placement for full chip 2d layouts," in *Optical Microlithography XXVI*, vol. 8683, 2013.

[6] R. Viswanathan, J. T. Azpiroz, and P. Selvam, "Process optimization through model based sraf printing prediction," in *Optical Microlithography XXV*, vol. 8326, 2012.

[7] K. Sakajiri, A. Tritchkov, and Y. Granik, "Model-based sraf insertion through pixel-based mask optimization at 32nm and beyond," in *Photomask and Next-Generation Lithography Mask Technology XV*, vol. 7028, 2008.

[8] 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 *Photomask Technology 2008*, vol. 7122, 2008.

[9] X. Xu, Y. Lin, M. Li, T. Matsunawa, S. Nojima, C. Kodama, T. Kotani, and D. Z. Pan, "Subresolution assist feature generation with supervised data learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2017.

[10] Y. Lin, M. B. Alawieh, W. Ye, and D. Z. Pan, "Machine learning for yield learning and optimization," in *2018 IEEE International Test Conference (ITC)*, 2018.

[11] M. B. Alawieh, Y. Lin, Z. Zhang, M. Li, Q. Huang, and D. Z. Pan, "Gan-sraf: Sub-resolution assist feature generation using conditional generative adversarial networks," in *Proceedings of the 56th Annual Design Automation Conference 2019*, 2019.

[12] X. Xu, T. Matsunawa, S. Nojima, C. Kodama, T. Kotani, and D. Z. Pan, "A machine learning based framework for sub-resolution assist feature generation," in *Proceedings of the 2016 on International Symposium on Physical Design*, 2016.

[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, 2020.

[14] H. Geng, H. Yang, Y. Ma, J. Mitra, and B. Yu, "Sraf insertion via supervised dictionary learning," in *Proceedings of ASPDAC*, 2019.

[15] T. Matsunawa, B. Yu, and D. Z. Pan, "Optical proximity correction with hierarchical bayes model," in *Optical Microlithography XXVIII*, vol. 9426, 2015.

[16] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, 2020.

[17] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," *Advances in neural information processing systems*, 2021.

[18] D. Kingma, T. Salimans, B. Poole, and J. Ho, "Variational diffusion models," *Advances in neural information processing systems*, 2021.

[19] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *International conference on machine learning*, 2021.

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

[21] S. Zheng, H. Yang, B. Zhu, B. Yu, and M. Wong, "Lithobench: Benchmarking ai computational lithography for semiconductor manufacturing," in *Advances in Neural Information Processing Systems*, vol. 36, 2023.

[22] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *British Machine Vision Conference 2016*, 2016.

[23] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," *arXiv preprint arXiv:2010.02502*, 2020.

[24] D. P. Kingma, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[25] S. Banerjee, Z. Li, and S. R. Nassif, "Iccad-2013 cad contest in mask optimization and benchmark suite," in *2013 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2013.