2025 Design, Automation & Test in Europe Conference (DATE 2025)

LLM-SRAF: Sub-Resolution Assist Feature Generation Using Large Language Model

Tianyi Li[1†], Zhexin Tang[1†], Tao Wu[1], Bei Yu[2], Jingyi Yu[1§], Hao Geng[1§]
[1]ShanghaiTech University
[2]Chinese University of Hong Kong

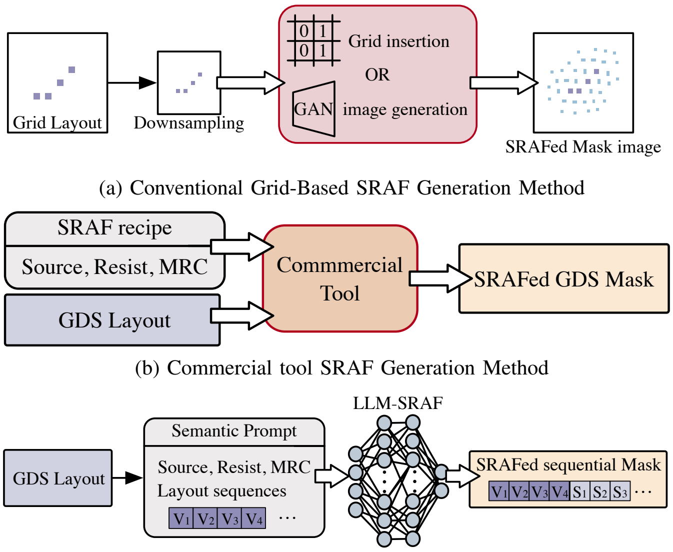
***Abstract*—As integrated circuit (IC) feature sizes continue to shrink, using sub-resolution assist features (SRAF) becomes increasingly crucial for improving wafer pattern resolution and fidelity. However, model-based SRAF insertion techniques, while accurate, require substantial computational resources and are often impractical for industrial scenarios. This demands more efficient and industry-compatible methods that maintain high performance. In this work, we introduce LLM-SRAF, a novel framework for SRAF generation driven by a large language model fine-tuned on an SRAF dataset. LLM-SRAF accepts semantic prompt inputs, including SRAF generation task descriptions, OPC recipe, lithography conditions, mask rules, and sequential layout descriptions, to directly generate SRAFs. Both supervised fine-tuning and reinforcement learning with human feedback (RLHF) are employed to enable the model to acquire domain-specific knowledge and specialize in SRAF generation. Experimental results show that LLM-SRAF outperforms existing state-of-the-art methods in metrics of mask quality, including edge placement error (EPE) and process variation band (PVB) area. Moreover, the runtime of LLM-SRAF is also 3x faster compared to the Calibre commercial tool.**

I. Introduction

As integrated circuit process nodes advance, various resolution enhancement techniques (RETs) have been proposed to improve fidelity and printability during mask optimization. One widely used RET is the Subresolution Assist Feature (SRAF) [1] [2]. SRAF involves adding small, non-printing features to the mask that aid in transferring light to the target pattern positions with the correct phase. This technique significantly enhances the printability and lithographic process window of isolated and semi-isolated features, improving robustness against process variations.

Recent SRAF generation methods can be categorized into three types: rule-based [3] [4] [5], model-based [3] [6] [7], and machine learning-based [1] [8] [9] [10]. Rule-based methods, while capable of achieving acceptable accuracy quickly and simply based on engineers' experience, require significant preprocessing efforts for increasingly complex layouts. Model-based methods eliminate the need for human expertise and achieve high accuracy even with complex patterns. However, many of these methods require extensive post-processing to generate manufacturing-friendly Manhattan shapes, and they always come at the cost of significant computing resources.

With the rapid advancement of machine learning algorithms, these technologies offer promising solutions to the SRAF insertion problem by reducing the computational costs of traditional methods. Geng et al. [10] and Xu et al. [1] used probability maps to predict SRAF placement, but required extensive post-processing. Liu et al. [9] developed a framework combining machine learning and reinforcement learning to generate SRAF layouts without post-processing. However, for different lithographic conditions, the model parameters still need to be adjusted by transfer learning. GAN-SRAF [8], which employs a generative adversarial network with multichannel heatmap encoding for end-to-end SRAF generation, shows potential for superior results with high-quality datasets, but still requires post-processing and struggles with varying layout sizes and lithography conditions.



(a) Conventional Grid-Based SRAF Generation Method

(b) Commercial tool SRAF Generation Method

(c) LLM-SRAF SRAF Generation Method

Fig. 1 Comparison of our LLM-SRAF generation method with conventional grid-based and commercial SRAF generation methods. (a) Previous SOTA SRAF generation methods use grid-based or image-based manner, necessitating layout downsampling for acceleration. It causes lithographic performance degradation. Besides, they have limited adaptability to different lithography conditions. (b) Commercial tools are time-intensive while supporting user-defined lithography conditions through verification scripts. (c) Our LLM-SRAF method, like commercial tools, incorporates lithography conditions and a losslessly processed sequential GDS layout into a semantic prompt. The resulting SRAF mask achieves commercial-level accuracy with faster generation.

[§] Corresponding author
[†] Author One and Author Two contributed equally to this work.

978-3-9826741-0-0/DATE25/© 2025 EDAA

In the integrated circuits industry, layout patterns are often stored in binary formats such as GDSII or OASIS. Traditional pre-processing methods frequently convert these GDS files into images for feature extraction and analysis. Further, grid insertion techniques or image generation neural networks are employed, as illustrated in Fig. 1(a). Despite its widespread use, the image-based approach has notable limitations, especially for sparse, large-size via/contact layouts. The image-based approach introduces unnecessary redundancy, distorts layouts during downsampling, and struggles with handling discrete images due to its low efficiency [11].

Recent advancements in Large Language Models (LLMs), have shown remarkable abilities in processing and generating coherent text, excelling in tasks requiring contextual understanding and accurate information retrieval [12] [13]. These models have recently been successfully applied to layout hotspot detection [11]. Inspired by these progress, we recognize LLMs' potential to enhance lithographic performance and computational efficiency in SRAF generation. Unlike commercial tools that require strict SRAF recipe writing, LLMs can simplify the process through a simpler question-and-answer format, thereby reducing tool usage complexity.

In this work, we propose an LLM-driven SRAF generation method using semantic prompts. To be specific, the workflow, as illustrated in Fig. 1(c), consists of three main stages: 1) Prompt preparation: GDS polygon are converted into layout sequences, combined with lithography conditions and mask rules, to form a semantic prompt. 2) LLM generation procedure: the prompt is input into the fine-tuned LLM, which processes it through tokenization, encoding, decoding, and detokenization to generate a semantic output containing size and position information of the mask layout. 3) semantic output to mask GDS: the SRAFed mask sequences are extracted from the semantic output and converted into a mask GDS with SRAFs inserted.

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

- To the best of our knowledge, this is the first instance of an LLM being specifically applied to SRAF generation. It closely resembles certain functionalities of commercial tools and establishes a framework for future applications in computational lithography.
- The SRAF generation problem is transformed into a sequence-to-sequence generation task. To cater to this transformation, a Manhattan encoding method is specifically designed for layout description, enhancing memory efficiency while ensuring the model's awareness of pattern positional, shape, and size attributes.
- Based on a designed semantic representation of the layout description dataset, a large language model is fine-tuned using supervised learning. This fine-tuning process enable the LLM to acquire the capability to generate sub-resolution assist features.
- A reward model based on commercial tool's lithography compliance Check (LCC) results is proposed to fine-tune the LLM using Reinforcement Learning from Human Feedback (RLHF), providing the LLM with insights into the physical aspects of the lithography model.

The rest of the paper is organized as follows. Section II describes the problems. Section III explains the algorithms in detail while Section IV presents the experimental results. Followed is the conclusion and potential future work in Section V.

II. Preliminaries

A. SRAF Generation Metrics

In line with previous research approaches, this paper presents the SRAF shapes as Manhattanized rectangles and utilizes process variation band (PVB) area and edge placement error (EPE) as evaluation metrics to assess the LCC performance of the LLM-SRAF mask optimization results.

In the LLM prompt, it is essential to include lithography conditions, mask rules, and layout descriptions. However, traditional GDS layout representations that use four points to define a polygon may lead to shape distortion, as the LLM could adapt these points into non-rectangular quadrilaterals to optimize lithography performance, potentially violating mask rules. Additionally, using four points to represent predefined rectangles is inefficient and increases token usage.

B. Problem Formulation

Problem 1 (Semantic prompt generation)**.** *Given a set of information that includes lithography conditions, mask rules, and layout files, all details must be losslessly transformed into a semantic prompt. Particularly for layout descriptions, a more robust and efficient sequence layout representation method should be designed.*

SRAF generation aims to optimize mask quality, which is evaluated by metrics such as PVB and EPE. Therefore, the LLM must be well-trained to act as an expert, understanding the relationship between the main pattern and SRAF, and accurately predicting the optimal SRAF positions and sizes for a given main pattern.

Problem 2 (SRAF generation via LLM)**.** *Given a set of semantic prompts, the objective of SRAF generation is to fine-tune an LLM that can place SRAFs in layouts to minimize the corresponding PVB area and EPE within an acceptable time.*

III. SRAF Generation using fine-tuned LLM

We embrace a three-step training process for the generation of LLM-SRAF. In the first step, we convert the layout dataset into semantic prompts with lithography condition, mask rules, and layout description inside. Second, the semantic SRAF dataset is exploited to supervise the fine-tuning of the LLM. Finally, we align the LLM's generated results to the Calibre tool and design a reward model based on MRC and lithography results for Reinforcement Learning from Human Feedback to fine-tune the LLM further. An overview of our proposed SRAF generation training process is shown in Fig. 2.

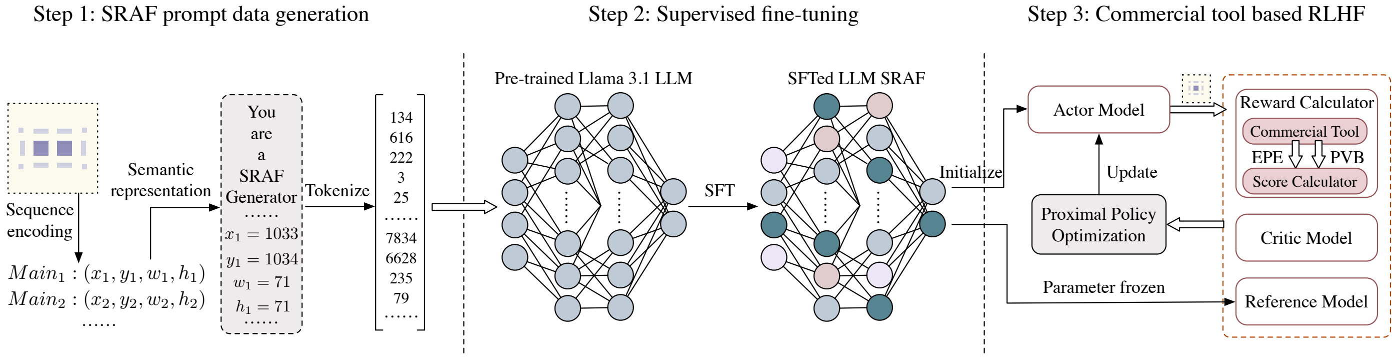


Fig. 2 The training flow of LLM-SRAF.

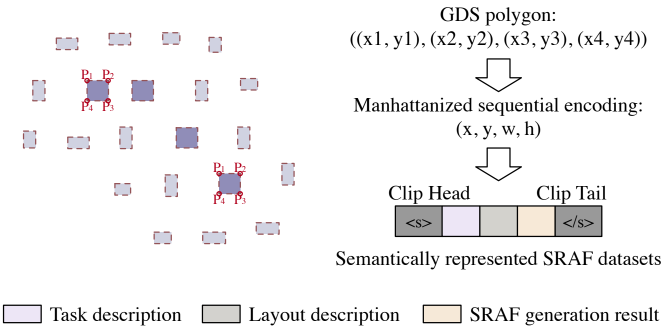


Fig. 3 SRAFs Manhattanized sequence encoding.

A. *SRAF Generation Prompt Design*

Since the LLM is pre-trained on natural language data, providing clear prompts is essential to reduce misunderstandings, ensure output quality, and maintain task relevance. To fully leverage the strengths of LLMs, we must design a semantic encoding framework that enhances the accuracy and quality of SRAF generation. As illustrated in Fig. 3, the SRAF semantic prompt is structured into three key components:

1) **Task description**: Provides semantic instructions, including background information, as well as details about the light source, photoresist, and mask rules.
2) **Layout description**: Contains details about the layout, focusing on the number, position, and dimensions of the main patterns. It serves as a semantic representation of the newly introduced sequence encoding method described in the following section of this paper.
3) **SRAF output description**: The LLM is expected to output the desired positions and dimensions of the generated SRAF based on the given layout description and the lithography conditions described in the task description.

To be specific about the prompt, the task description begins with a background introduction that outlines the LLM's role and the task at hand. This is followed by a description of the mask rules, the source information and the resist information, which provide the LLM with fundamental setup information. This information will later be integrated into the model through reinforcement learning during the RLHF phase.

For layout description, a new sequence encoding method is required to improve accuracy, shorten token length for efficiency, and simplify interpretation by the LLM. To address the inefficiencies and potential errors introduced by downsampling in image-based SRAF generation methods, this encoding method involves only minor modifications to GDS files and focuses solely on recording pattern information, thereby avoiding loss of shape details while enhancing efficiency simultaneously. As shown in Fig. 3, traditional GDS files use 2x4 tuples to describe rectangular polygons by recording the four vertex coordinates. However, this approach does not guarantee that shapes remain rectangles after LLM processing. Our Manhattanized sequence encoding method resolves this issue by representing SRAF with its center coordinates $(\boldsymbol{x}, \boldsymbol{y})$ and dimensions $\boldsymbol{w}$ (width) and $\boldsymbol{h}$ (height). This SRAF sequence encoding method ensures accurate shape representation with just four numbers, offering a more efficient storage solution while preserving the integrity of the rectangles.

To avoid printing issues or mask unmanufacturability, SRAFs in the dataset must meet specific size constraints. Therefore, we have further optimized this method to ensure compliance with these constraints. To keep SRAFs within the desired range, a linear transformation using the sigmoid function is applied to the original width $\boldsymbol{w_{ori}}$ and height $\boldsymbol{h_{ori}}$. This transformation converts these dimensions into a pair of width and height parameters, $\boldsymbol{w}$ and $\boldsymbol{h}$, which are then constrained within the range $[\alpha, \beta]$. The linear transformation is defined by Equation (1) and Equation (2).

$$\frac{\boldsymbol{w_{ori}} - \alpha}{\beta - \alpha} = \text{Sigmoid}(\boldsymbol{w}), \frac{\boldsymbol{h_{ori}} - \alpha}{\beta - \alpha} = \text{Sigmoid}(\boldsymbol{h}), \qquad (1)$$

$$\boldsymbol{w} = \ln \frac{\boldsymbol{w_{ori}} - \alpha}{\beta - \boldsymbol{w_{ori}}}, \boldsymbol{h} = \ln \frac{\boldsymbol{h_{ori}} - \alpha}{\beta - \boldsymbol{h_{ori}}}. \qquad (2)$$

With the center coordinates and the transformed width parameter $\boldsymbol{w}$ and height parameter $\boldsymbol{h}$, each via pattern or SRAF in the layout file can be represented by a tuple $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w}, \boldsymbol{h})$. The entire dataset employs a series of these $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w}, \boldsymbol{h})$ tuples as layout descriptions.

In the dataset, the SRAF output description section must include the generated SRAF information, representing the expected results of SRAF generation during LLM inference. During supervised learning, this section also serves as the reference for the output. To help the LLM better understand the principles of SRAF generation and enhance its spatial awareness, this section clearly establishes the correspondence between each main pattern and its most relevant SRAF. A Nearest-Neighbor-Search algorithm is employed to link each SRAF to its corresponding main pattern by analyzing their geometric and positional relationships.

With all three sections well introduced, here is an example of the designed semantic prompt:

Prompt design for LLM-SRAF

# Task description
You are an assistant tasked with addressing SRAF generation issues on a two-dimensional plane. The lithography setup is:
Illumination:
SoftAnnular, 193nm, 1.35NA, o0.X0/i0.X0
Defocus -60 -30 0 30 60
Resist: thickness$_1$, n$_1$, k$_1$
BARC: thickness$_2$, n$_2$, k$_2$
$\cdots\cdots$
Mask Rule: d1 metric: 0.040 SQUARE

# Layout description
There are **N** main patterns in total, $\boldsymbol{x}$ and $\boldsymbol{y}$ are the coordinates and $\boldsymbol{w}$ and $\boldsymbol{h}$ are the dimensions.
main$_0$: $\boldsymbol{x} = x_0$, $\boldsymbol{y} = y_0$, $\boldsymbol{w} = w_0$, $\boldsymbol{h} = h_0$.
main$_1$: $\boldsymbol{x} = x_1$, $\boldsymbol{y} = y_1$, $\boldsymbol{w} = w_1$, $\boldsymbol{h} = h_1$.
$\cdots\cdots$

# SRAF output description
Generated by main$_0$: sraf: $\boldsymbol{x} = x_0$, $\boldsymbol{y} = y_0$, $\boldsymbol{w} = w_0$, $\boldsymbol{h} = h_0$
$\cdots\cdots$
Generated by main$_1$: sraf: $\boldsymbol{x} = x_1$, $\boldsymbol{y} = y_1$, $\boldsymbol{w} = w_1$, $\boldsymbol{h} = h_1$
$\cdots\cdots$

For ease of use, a fixed prompt template can be employed for the task description. When using this template, it is only necessary to adjust parameters such as the inner and outer diameters of the light source, as well as data like the thickness, n and k of the resist, according to specific requirements.

B. LLM Supervised Fine-tuning

The fine-tuning process of Large Language Models involves adapting pre-trained models to perform effectively on domain-specific tasks by adjusting their parameters. In this study, fine-tuning is employed to specialize models in generating SRAFs for specific layouts under various lithography conditions, utilizing the SRAF dataset. The foundation model chosen for this task is Meta-Llama-3.1-8B [14], which has 8 billion parameters and a deep Transformer architecture, making it well-suited for generating SRAFs from semantic inputs.

Fig. 2 illustrates the complete fine-tuning process for LLM-SRAF. Initially, mask rules, simulation conditions, and layout descriptions are converted into semantic prompts, which are then mapped to tokenized numerical representations before being fed into the LLM. The LLM is then trained to generate the corresponding SRAF descriptions for each layout based on the specific conditions provided by the prompt.

Instead of full fine-tuning, which modifies all model parameters, parameter-efficient fine-tuning methods are preferred. Specifically, Low-Rank Adaptation (LoRA) is employed. LoRA refines only a subset of model parameters by incorporating low-rank matrices into the existing weight matrices, thereby preserving the original weights. This approach updates fewer parameters while still achieving significant performance improvements with minimal computational cost, thus conserving both resources and time [15].

C. RLHF Based on Commercial Tool Results

Reinforcement Learning with Human Feedback (RLHF) follows supervised fine-tuning to further improve model performance by integrating reinforcement learning with human insights. It uses rewards to guide the model towards better outputs and addresses complex goals, thereby improving the quality and relevance of results [16].

The RLHF module consists of four key components: the Actor model, the Critic model, the Reward calculator, and the Reference model. The Actor model, which represents the policy being optimized, is initialized using the SFTed LLM. The final optimized policy results in an LLM that has been successfully trained with RLHF. The Reference model, a frozen version of the SFTed LLM, serves as a benchmark in the RLHF process. It computes a KL divergence penalty to measure the difference in token logarithmic probabilities between the Actor and Reference models, ensuring stability during RL training.

The Reward calculator, which is typically referred to as the reward model in RLHF, computes the immediate reward for generated tokens. Unlike traditional RLHF methods that rely on human annotators for evaluation, our approach employs a scoring model based on commercial tool simulations, achieving more equitable and convincing results compared to other state-of-the-art methods. Additionally, the development of a faster lithography simulator could further accelerate model training. Using an input prompt detailing mask rules, light sources, and photoresists, a script is run in the commercial tool with specified settings to perform lithography simulations and generate metrics such as EPE, PVB area, and MRC violations for scoring. The scoring mechanism of the Reward calculator is defined as follows:

$$\text{Score} = \frac{\alpha}{\text{EPE}} + \frac{\beta}{\text{PVB}} - \gamma \times \text{MRC violation}, \tag{3}$$

where α, β, and γ are the weighting coefficients that balance the contributions of EPE, PVB, and MRC violations to the overall reward. As illustrated in Fig. 4, the reward calculator ranks the outputs from the actor model based on these scores. Consequently, the actor model will increase the output probabilities of higher-ranked outputs while decreasing the output probabilities of lower-ranked ones.

The Critic model predicts the expected total reward and is trained using the reward signals provided by the Reward Calculator during the reinforcement learning process. It is specifically designed to assess the state value and the action value of sizing and positioning actions. As illustrated in Fig. 5,

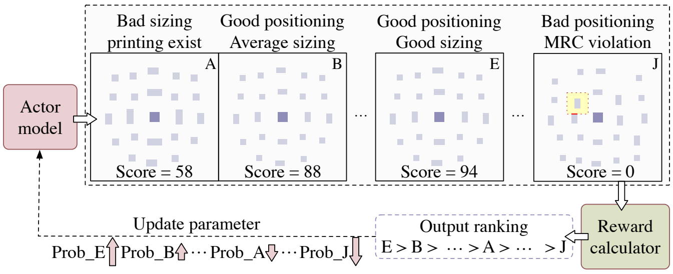


Fig. 4 The operating mechanism of Reward calculator.

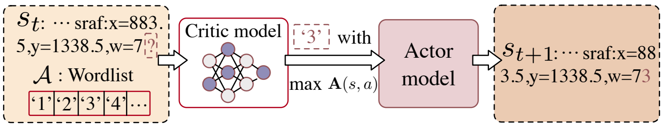


Fig. 5 The operating mechanism of Critic model.

the Critic model processes the current state and action space to compute the advantage function, guiding the policy toward actions that are expected to yield higher rewards compared to the average reward.

To optimize the LLM actor, Proximal Policy Optimization (PPO) was employed [17]. The objective function of PPO is shown below:

$$\mathcal{L}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(r_t(\theta) \hat{A}_t, \text{clip} \left(r_t(\theta), 1 - \epsilon, 1 + \epsilon \right) \hat{A}_t \right) \right] . \tag{4}$$

- $\hat{\mathbb{E}}_t$ represents the expected cumulative reward from time step t onward based on the current policy.
- $r_t(\theta)$ is the ratio of token probability of RLHFed LLM SRAF $\pi_\theta(a_t|s_t)$ and the token probability of reference only SFTed LLM SRAF $\pi_{\theta,ref}(a_t|s_t)$.
- $\hat{A}_t$ is the advantage estimate derived from the critic model.
- ϵ is a positive number that controls the extent of clipping to prevent large policy updates.
- clip($r_t(\theta)$, 1 - ϵ, 1 + ϵ) is a clip function that restricts $r_t(\theta)$ within a defined range (1 - ϵ, 1 + ϵ).

PPO updates the policy $\pi_\theta(a_t|s_t)$ by iteratively improving it through small, stable updates. This process improves the final score by reducing EPE, PVB, and MRC violations, resulting in an RLHFed LLM that performs better on these metrics.

D. Optimized SRAF Insertion for Large Layouts Using Small-Scale Segments

In large-scale layouts, utilizing the LLM for output requires an excessive number of tokens, resulting in slow processing speeds and impractical time consumption. To address this issue, we propose an algorithm for rapid SRAF insertion based on small-scale layout results. This method involves segmenting a large layout into smaller sections within an optical diameter range and generating SRAF for the entire large layout based on these smaller sections.

The proposed algorithm uses distance-based clustering to split SRAF into inner and outer loops. The inner loop's distance from the main pattern is optimized to minimize violations with adjacent layouts. A simple merging algorithm then resolves any mask rule issues introduced by the inner loop, ensuring it is optimally sized and positioned before addressing the outer loop. Finally, the outer loop is inserted without new violations, followed by a final mask rule check to ensure manufacturability.

IV. Experimental Results

A. Experimental Setups

Our SFT process utilizes an augmented version of the ICCAD2020 via layer dataset, consisting of GDS files ranging in size from $2\mu m \times 2\mu m$ to $10\mu m \times 10\mu m$ with varying via pattern densities. The augmentation involves adjusting the illumination source's inner and outer radii to three specific ratios: 0.6/0.8, 0.6/0.9, and 0.7/0.9. Additionally, SRAF sizes are randomly scaled by factors ranging from 0.96 to 1.08. This results in an augmented dataset with several hundred thousand samples. We train the Meta-Llama-3.1-8B model using Distributed Data Parallel across four Nvidia H100 GPUs, with inference performed on a single Nvidia H100 GPU. The AdamW optimizer, with an initial learning rate of 5×10^{-5}, is adjusted using a linear warmup and cosine annealing schedule to ensure stable and efficient convergence.

B. Lithography Performance Against SOTA Works

After generating the SRAF based on the main pattern, model-based OPC and LCC are performed using a commercial tool [19]. We compare model-based [?], SODL + NewILP [10], RL-SRAF [9], CTM-SRAF [18], and LLM-SRAF in terms of PVB and EPE in the same benchmark suite with 18 testing layout clips as utilized in [9], [10]. TABLE I presents the PVB area and EPE values for each method. LLM-SRAF achieves the best average PVB and EPE in all test clips, outperforming the best baseline [9] by reducing PVB by 2% and EPE by 4%. In terms of the result PVB area, although LLM-SRAF slightly underperforms in 3 out of 8 dense test cases compared to other SOTA methods, this is attributed to the small number of main patterns in these cases, where minor changes can have a significant impact. However, LLM-SRAF outperforms all other methods on large-scale layouts, demonstrating its superior global optimization capabilities. This advantage is especially valuable in industrial scenarios such as memory device manufacturing, where large-scale vias patterning is essential.

The EPE result is determined by the custom reward calculator, which sets a baseline based on the best SOTA performance. If the EPE performance in a case surpasses the baseline, full points are awarded, with no additional reward for further EPE improvement. This approach encourages the model to shift its focus toward achieving a higher PVB score. such a score evaluation method allows users to customize the score calculator based on their specific requirements for EPE and PVB, providing flexibility in balancing these metrics.

TABLE I Comparison of different SRAF generation methods on dense and sparse testing layout clips.

Testbench	SODL+NewILP [10]		RL-SRAF [9]		CTM-SRAF [18]		Model-based [19]		LLM-SRAF	
	PVB $.001\mu m^2$	EPE nm	PVB $.001\mu m^2$	EPE nm	PVB $.001\mu m^2$	EPE nm	PVB $.001\mu m^2$	EPE nm	PVB $.001\mu m^2$	EPE nm
Dense1	1.850	0.667	1.876	0.708	1.857	0.458	**1.801**	**0.375**	1.844	0.458
Dense2	1.987	0.438	**1.975**	0.250	2.538	**0.000**	2.828	0.750	1.999	0.062
Dense3	2.545	1.750	2.442	1.125	2.617	1.375	2.449	1.000	**2.391**	**0.875**
Dense4	2.363	1.125	2.392	1.500	2.445	1.000	2.360	1.250	**2.215**	**0.937**
Dense5	2.413	0.938	2.265	1.500	2.515	1.375	2.356	1.437	**2.255**	**0.937**
Dense6	**2.538**	**0.000**	2.828	0.750	3.008	1.000	2.821	0.250	2.774	**0.000**
Dense7	2.277	1.803	2.372	1.167	2.484	1.667	2.336	**1.083**	**2.254**	**1.083**
Dense8	2.445	1.000	2.360	1.250	2.490	1.083	2.364	1.250	**2.278**	**0.916**
Sparse1	2.813	0.500	2.774	0.500	2.931	0.500	2.778	0.438	**2.749**	**0.375**
Sparse2	2.803	0.625	2.727	0.516	2.982	**0.469**	2.753	0.515	**2.720**	1.000
Sparse3	2.764	0.563	2.749	0.507	2.969	**0.368**	2.765	0.453	**2.719**	0.464
Sparse4	2.785	0.547	2.753	0.559	2.958	**0.476**	2.735	0.687	**2.621**	0.775
Sparse5	2.799	0.633	2.766	0.559	2.978	**0.475**	2.738	1.068	**2.717**	0.961
Sparse6	2.789	0.552	2.768	0.514	2.998	**0.502**	2.742	1.150	**2.716**	0.818
Sparse7	2.786	0.536	2.767	0.531	2.987	**0.475**	2.734	0.704	**2.709**	0.553
Sparse8	2.780	0.610	2.751	0.526	2.986	0.497	2.740	1.125	**2.719**	**0.464**
Sparse9	2.801	0.573	2.753	0.535	2.986	0.491	2.746	1.885	**2.731**	**0.411**
Sparse10	2.790	0.555	2.763	0.525	2.974	**0.497**	2.748	0.593	**2.668**	0.661
Average	2.574	0.705	2.546	0.675	2.733	0.737	2.548	0.865	**2.505**	**0.652**
Ratio	1.027	1.081	1.016	1.035	1.091	1.130	1.017	1.326	**1.000**	**1.000**

*Results of SODL+NewILP and RL-SRAF are directly quoted from [9] and [10]. The CTM-SRAF is re-implemented.

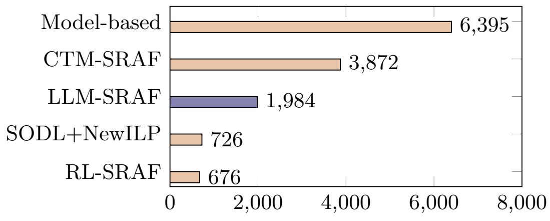


Fig. 6 Runtime comparison in seconds.

C. Runtime Analysis

Fig. 6 compares the runtime of five methods. Runtime for RL SRAF and SODL+NewILP are sourced from [9] and [10], respectively. To ensure fairness, we used 8 CPU cores for the commercial tool's model-based SRAF generation. Although the LLM-based SRAF is slower than RL SRAF and SODL+NewILP due to large model complexity, it still significantly outperforms traditional model-based methods, running 3x faster than the Calibre commercial tool. Additionally, RLHF's mask rule violation penalties reduce the need for post-processing, further decreasing the overall runtime.

D. Ablation Study of RLHF's Impact on SRAF Generation Performance

An ablation study evaluated RLHF's impact on SRAF generation. We compared an LLM-SRAF model trained only with supervised fine-tuning to one that also includes RLHF. Both models generated 10 SRAF sets, which were evaluated through lithography simulation. Results, as shown in TABLE II, show that the RLHF model, using a reward model based on EPE and PVB metrics, significantly outperforms the model that only underwent supervised fine-tuning in lithography quality.

TABLE II Impact of RLHF on SRAF generation performance.

LLM Without RLHF		LLM With RLHF	
PVB 2.571	EPE 0.719	PVB 2.505	EPE 0.652

Although the LLM typically avoids MRC violations after supervised fine-tuning, occasional content redundancy can lead to excessive SRAF, resulting in MRC violations. Out of 80 generated tests, 7 outputs experience MRC violations before RLHF. However, after RLHF, the probability of MRC violations drops to zero.

V. Conclusion and Future works

This paper presents an LLM-based SRAF generation model that replicates key functionalities of traditional commercial tools, paving the way for future LLM applications in computational lithography. We developed a specialized semantic data processing method and combined supervised learning with RLHF to integrate lithography simulation data into the LLM. Experimental results demonstrate that our approach significantly improves PVB area and EPE, outperforming state-of-the-art methods and being at least three times faster than model-based approaches despite increased complexity. These findings highlight the great potential of LLMs to advance computational lithography.

While this paper focuses on generating SRAFs for via layers, the LLM-based approach can also extend to metal layers. Via layers require dedicated sequence encoding, whereas metal layers use GDS's path-based representation, preserving the Manhattan structure without additional encoding. Traditional grid-based or image-based methods are limited by maximum layout sizes per pass, necessitating partitioning that can cause conflicts when merging segmented metal lines with SRAFs. By leveraging LLMs' efficient data handling, partitioning time is reduced, enabling more global SRAF generation for large metal layouts.

References

[1] 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 *Proc. ISPD*, 2016, p. 161–168.

[2] C. H. Wallace, S. S. Sivakumar, and P. A. Nyhus, “Sub-resolution assist features,” 2006.

[3] R. Viswanathan, J. T. Azpiroz, and P. Selvam, “Process optimization through model based SRAF printing prediction,” in *Optical Microlithography XXV*, vol. 8326, 2012, p. 83261A.

[4] Y. Ping, S. McGowan, Y. Gong, Y. Foong, J. Liu, J. Qiu, V. Shu, B. Yan, J. Ye, P. Li, H. Zhou, T. Pandey, J. Liang, C. Aquino, S. Baron, and S. Kapasi, “Process window enhancement using advanced ret techniques for 20nm contact layer,” *Proc. SPIE*, 02 2014.

[5] A. Chen, S. Hansen, M. Moers, J. Shieh, A. Engelen, K. van Ingen Schenau, and S.-E. Tseng, “The contact hole solutions for future logic technology nodes,” *Proc. SPIE*, 03 2008.

[6] L. Pang, Y. Liu, T. Dam, K. Mihic, T. Cecil, and D. Abrams, “Inverse lithography technology (ILT): keep the balance between SRAF and MRC at 45 and 32 nm,” in *Photomask Technology 2007*, ser. Proc. SPIE, R. J. Naber and H. Kawahira, Eds., vol. 6730, Oct. 2007, p. 673052.

[7] S. D. Shang, S. Lisa, and G. Yuri, “Model-based SRAF insertion,” 2012.

[8] M. B. Alawieh, Y. Lin, Z. Zhang, M. Li, Q. Huang, and D. Z. Pan, “GAN-SRAF: Sub-Resolution Assist Feature Generation Using Generative Adversarial Networks,” *IEEE TCAD*, vol. 40, no. 2, pp. 373–385, 2021.

[9] G.-T. Liu, W.-C. Tai, Y.-T. Lin, I. H.-R. Jiang, J. P. Shiely, and P.-J. Cheng, “Sub-Resolution Assist Feature Generation with Reinforcement Learning and Transfer Learning,” in *Proc. ICCAD*, 2022.

[10] H. Geng, W. Zhong, H. Yang, Y. Ma, J. Mitra, and B. Yu, “SRAF Insertion via Supervised Dictionary Learning,” *IEEE TCAD*, vol. 39, no. 10, pp. 2849–2859, 2020.

[11] Y. Chen, Y. Wu, J. Wang, T. Wu, X. He, J. Yu, and H. Geng, “LLM-HD: Layout Language Model for Hotspot Detection with GDS Semantic Encoding,” in *Proc. DAC*, 2024.

[12] OpenAI, “Gpt-4 technical report,” OpenAI, Tech. Rep., 2023.

[13] H. Touvron, L. Dickersin, F. R. H. G. de Melo, A. Sablayrolles *et al.*, “LLaMA: Open and Efficient Foundation Language Models,” *arXiv preprint arXiv:2302.13971*, 2023.

[14] AI@Meta, “Llama 3 model card,” 2024. [Online]. Available: https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md

[15] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, “LoRA: Low-Rank Adaptation of Large Language Models,” in *Proc. ICLR*, 2022.

[16] P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, “Deep Reinforcement Learning from Human Preferences,” in *Proc. NIPS*, 2017, p. 4302–4310.

[17] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal Policy Optimization Algorithms,” 2017.

[18] Z. Yu, P. Liao, Y. Ma, B. Yu, and M. D. F. Wong, “CTM-SRAF: Continuous Transmission Mask-Based Constraint-Aware Sub-Resolution Assist Feature Generation,” *IEEE TCAD*, vol. 42, no. 10, p. 3402–3411, oct 2023.

[19] “Calibre Computational Lithography,” [Online]. Available: ”https://eda.sw.siemens.com/en-US/ic/calibre-manufacturing/computational-lithography/”, accessed: Sept. 12, 2024.