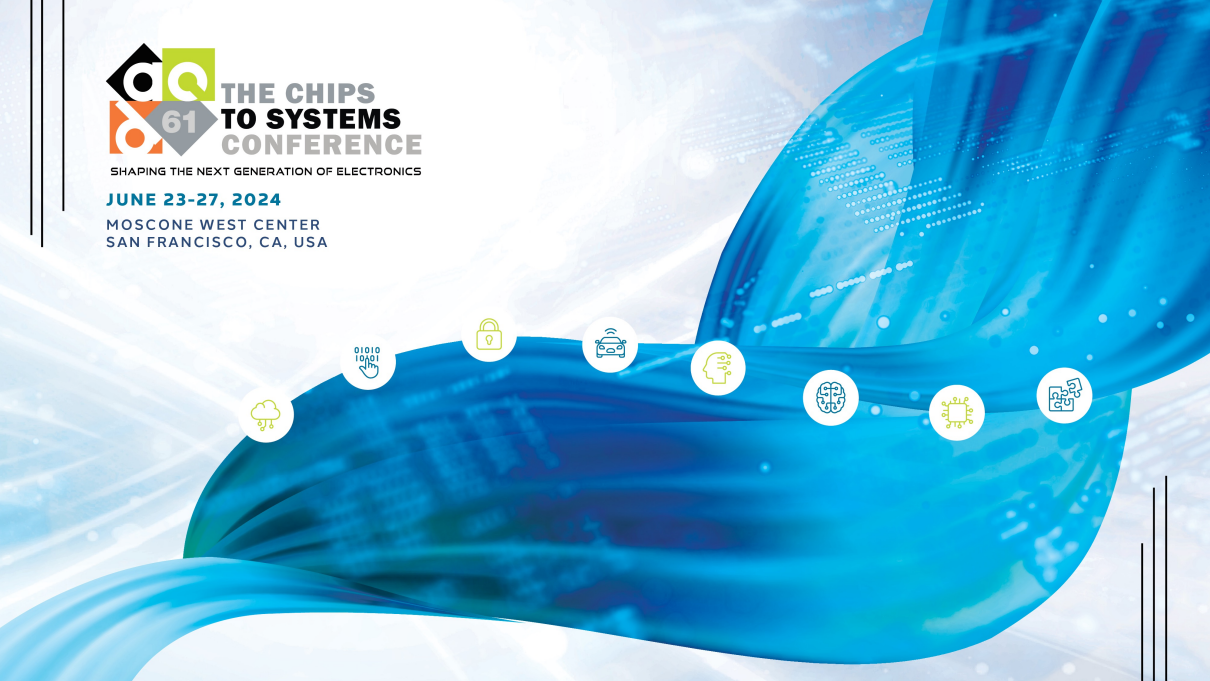

61
THE CHIPS
TO SYSTEMS
CONFERENCE
SHAPING THE NEXT GENERATION OF ELECTRONICS
JUNE 23-27, 2024
MOSCONE WEST CENTER
SAN FRANCISCO, CA, USA

61 THE CHIPS TO SYSTEMS CONFERENCE
SHAPING THE NEXT GENERATION OF ELECTRONICS
JUNE 23-27, 2024
MOSCONE WEST CENTER
SAN FRANCISCO, CA, USA

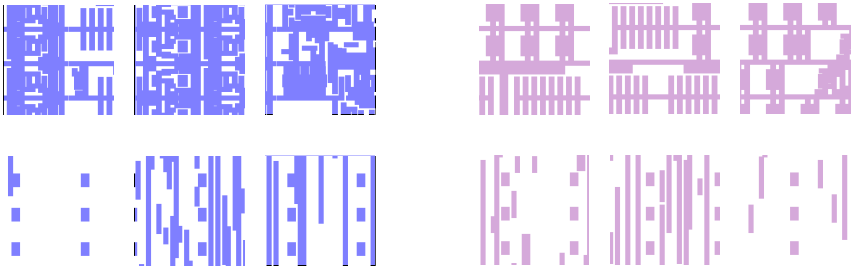
# ChatPattern: Layout Pattern Customization via Natural Language

**Zixiao Wang**[1], Yunheng Shen[2], Xufeng Yao[1], Wenqian Zhao[1], Yang Bai[1], Farzan Farnia[1], Bei Yu[1]

[1]Chinese University of Hong Kong
[2]Tsinghua University

# Background

# Layout Pattern Generation



Existing Patterns

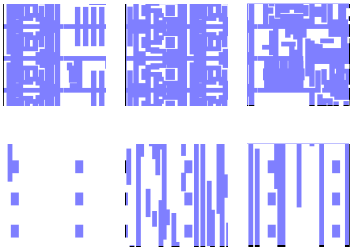
Generated Patterns

VLSI layout patterns provide critical resources in various designs for manufacturability research. Pattern Generation task aims to mimic the distribution of existing patterns.

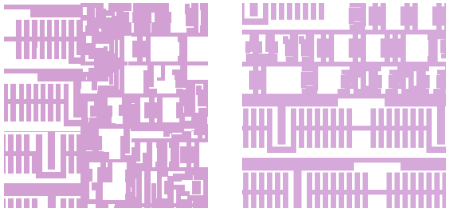
# From Generation to Customization

User Requirement

I need Patterns in upper row style with 4 times larger in size.
The design rule should be changed to XXX



Existing Patterns

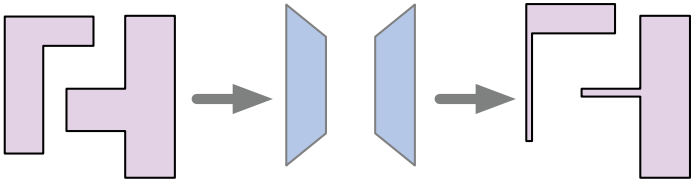


Customized Patterns

The requirements on layout pattern distributions can vary in real cases. Pattern Customization task aims to generate patterns to meet specialized requirements.

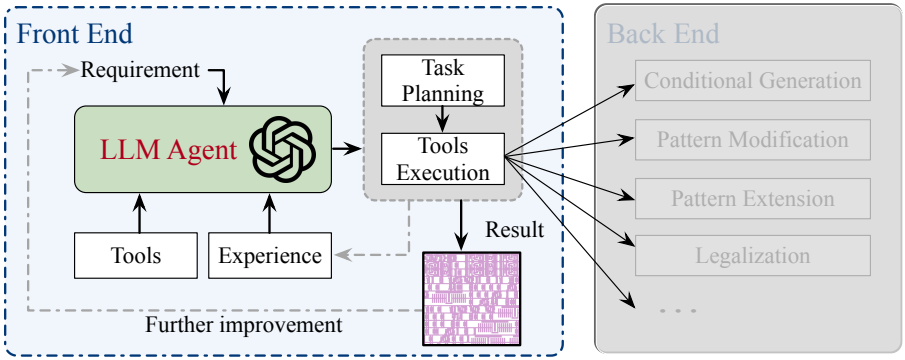
# Let's Employ a LLM

- Training a LLM from scratch? **NO**, Too expensive.
- Utilizing Pre-trained LLM? **Yes**, **but, how can LLM get access to the Layout Patterns?**
  - Encoding a pattern as a sequence of direction and distance?
  - Embedding a pattern as a pattern token?
  - Manipulating pattern-generation tools?



# ChatPattern

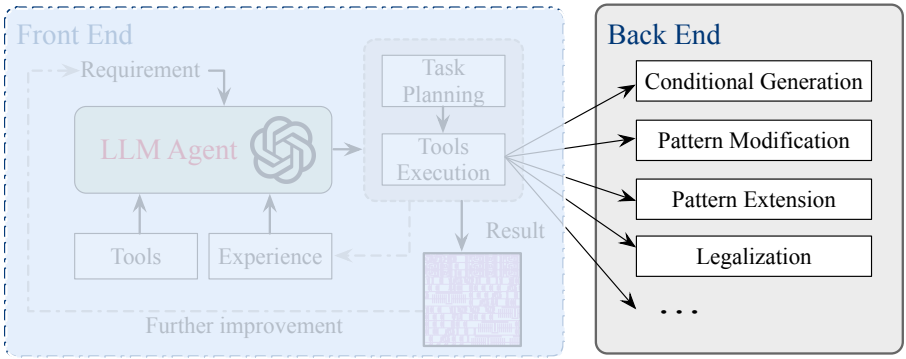
# ChatPattern




An illustration of ChatPattern

ChatPattern seamlessly integrates a front-end powered by a Large Language Model with a back-end that employs a conditional discrete diffusion model for layout pattern generation.

# ChatPattern




An illustration of ChatPattern

ChatPattern seamlessly integrates a front-end powered by a Large Language Model with a back-end that employs a conditional discrete diffusion model for layout pattern generation.

# The LLM agent

The LLM agent is designed to communicate with users via natural language, and is able to:

- Auto-Format Requirement
- Plan and Execute Task
- Learn and Apply Tool Functions
- Learn from Documents and Experience

# Flexible Layout Pattern Generative Model

To construct a pattern library, certain fundamental tools or APIs are indispensable:

- Random Topology Generation
- Topology Legalization[1]

[1] Zixiao Wang et al. (2023). "Diffpattern: Layout pattern generation via discrete diffusion". In: *2023 60th ACM/IEEE Design Automation Conference (DAC)*. IEEE, pp. 1–6.

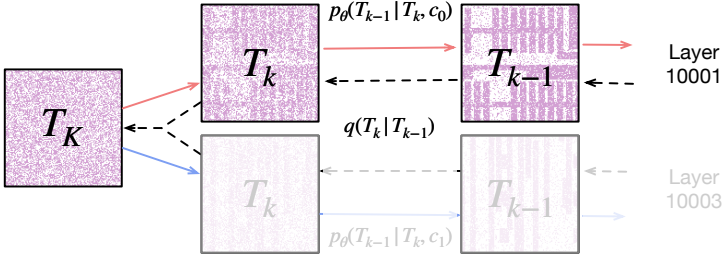
# Flexible Layout Pattern Generative Model

To construct a pattern library, certain fundamental tools or APIs are indispensable:

- **Conditional Topology Generation**
- Topology Legalization[1]
- **Topology Modification**
- **Topology Extension**

[1] Zixiao Wang et al. (2023). "Diffpattern: Layout pattern generation via discrete diffusion". In: *2023 60th ACM/IEEE Design Automation Conference (DAC)*. IEEE, pp. 1–6.

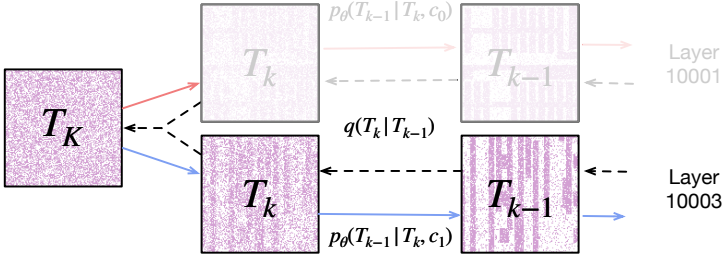
# Property-Conditional Topology Generation




$$p_\theta(\boldsymbol{T}_0|\boldsymbol{T}_K, \boldsymbol{c}) = p_\theta(\boldsymbol{T}_0|\boldsymbol{T}_1, \boldsymbol{c}) \prod_{k=2}^{K} p_\theta(\boldsymbol{T}_{k-1}|\boldsymbol{T}_k, \boldsymbol{c}), \tag{1}$$

$$L = D_{\mathrm{KL}}\left(\boldsymbol{q}\left(\boldsymbol{x}_{k-1}|\boldsymbol{x}_k, \boldsymbol{x}_0\right) \parallel \boldsymbol{p}_\theta\left(\boldsymbol{x}_{k-1}|\boldsymbol{x}_k, \boldsymbol{c}\right)\right) - \lambda \log \boldsymbol{p}_{\boldsymbol{\theta}}\left(\boldsymbol{x}_0|\boldsymbol{x}_k, \boldsymbol{c}\right). \tag{2}$$

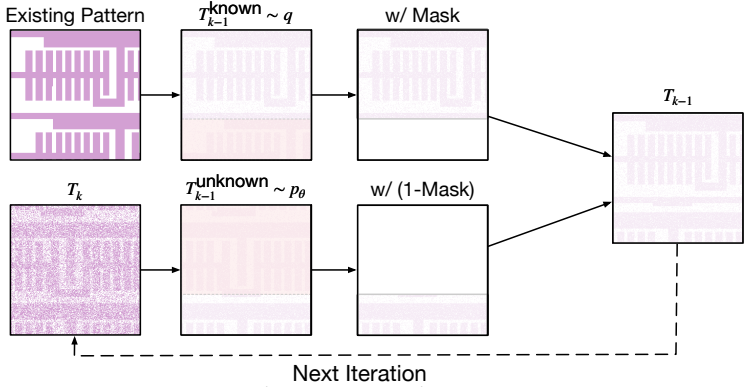
# Property-Conditional Topology Generation




$$p_\theta(\boldsymbol{T}_0|\boldsymbol{T}_K, \boldsymbol{c}) = p_\theta(\boldsymbol{T}_0|\boldsymbol{T}_1, \boldsymbol{c}) \prod_{k=2}^{K} p_\theta(\boldsymbol{T}_{k-1}|\boldsymbol{T}_k, \boldsymbol{c}), \tag{1}$$

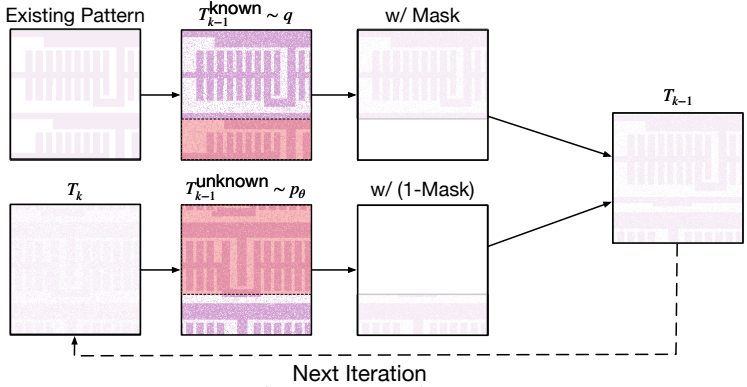
$$L = D_{\mathrm{KL}}\left(\boldsymbol{q}\left(\boldsymbol{x}_{k-1}|\boldsymbol{x}_k, \boldsymbol{x}_0\right) \parallel \boldsymbol{p}_\theta\left(\boldsymbol{x}_{k-1}|\boldsymbol{x}_k, \boldsymbol{c}\right)\right) - \lambda \log \boldsymbol{p}_{\boldsymbol{\theta}}\left(\boldsymbol{x}_0|\boldsymbol{x}_k, \boldsymbol{c}\right). \tag{2}$$

# Pattern Modification



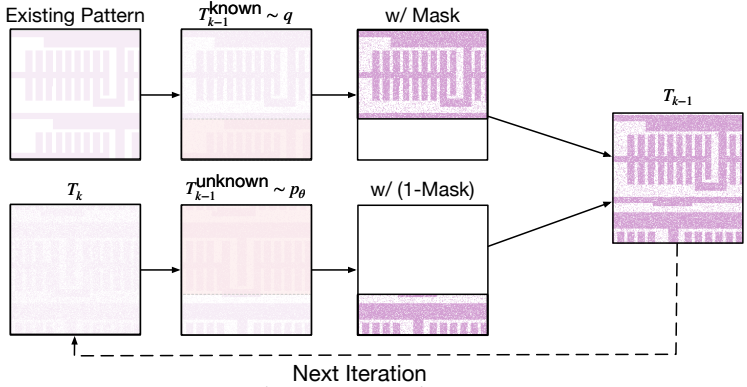

$$
\begin{aligned}
\boldsymbol{T}_{k-1}^{\text{known}} &\sim \boldsymbol{q}\left(\boldsymbol{T}_{k-1}|\boldsymbol{T}_0^{\text{known}}\right), \\
\boldsymbol{T}_{k-1}^{\text{unknown}} &\sim \boldsymbol{p_\theta}\left(\boldsymbol{T}_{k-1}|\boldsymbol{T}_k, \boldsymbol{c}\right), \\
\boldsymbol{T}_{k-1} &= \boldsymbol{M} \odot \boldsymbol{T}_{k-1}^{\text{known}} + (1-\boldsymbol{M}) \odot \boldsymbol{T}_{k-1}^{\text{unknown}},
\end{aligned}
\tag{3}
$$

# Pattern Modification



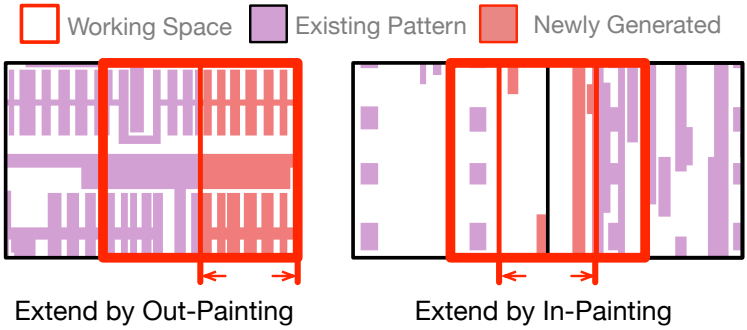

$$
\begin{aligned}
\boldsymbol{T}_{k-1}^{\text{known}} &\sim \boldsymbol{q}\left(\boldsymbol{T}_{k-1}|\boldsymbol{T}_0^{\text{known}}\right), \\
\boldsymbol{T}_{k-1}^{\text{unknown}} &\sim \boldsymbol{p_\theta}\left(\boldsymbol{T}_{k-1}|\boldsymbol{T}_k, \boldsymbol{c}\right), \\
\boldsymbol{T}_{k-1} &= \boldsymbol{M} \odot \boldsymbol{T}_{k-1}^{\text{known}} + (1-\boldsymbol{M}) \odot \boldsymbol{T}_{k-1}^{\text{unknown}},
\end{aligned}
\tag{3}
$$

# Pattern Modification

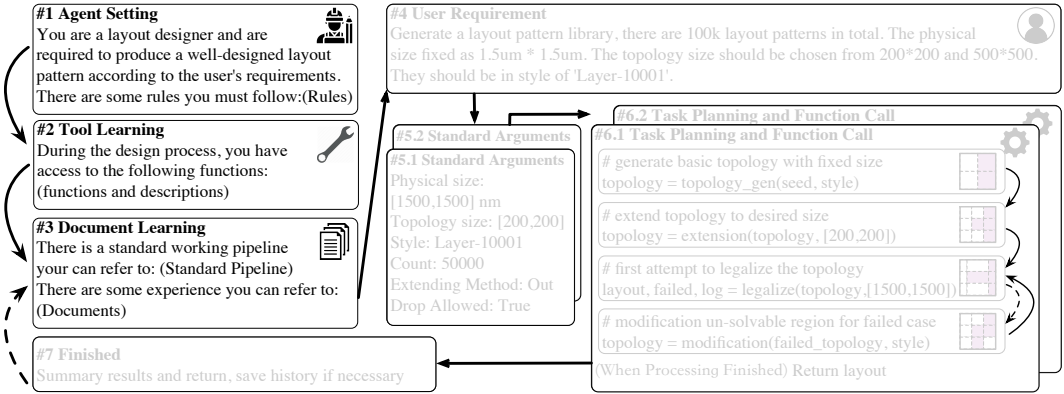



$$
\begin{aligned}
\boldsymbol{T}_{k-1}^{\text{known}} &\sim \boldsymbol{q}\left(\boldsymbol{T}_{k-1}|\boldsymbol{T}_0^{\text{known}}\right), \\
\boldsymbol{T}_{k-1}^{\text{unknown}} &\sim \boldsymbol{p_\theta}\left(\boldsymbol{T}_{k-1}|\boldsymbol{T}_k, \boldsymbol{c}\right), \\
\boldsymbol{T}_{k-1} &= \boldsymbol{M} \odot \boldsymbol{T}_{k-1}^{\text{known}} + (1-\boldsymbol{M}) \odot \boldsymbol{T}_{k-1}^{\text{unknown}},
\end{aligned}
\tag{3}
$$

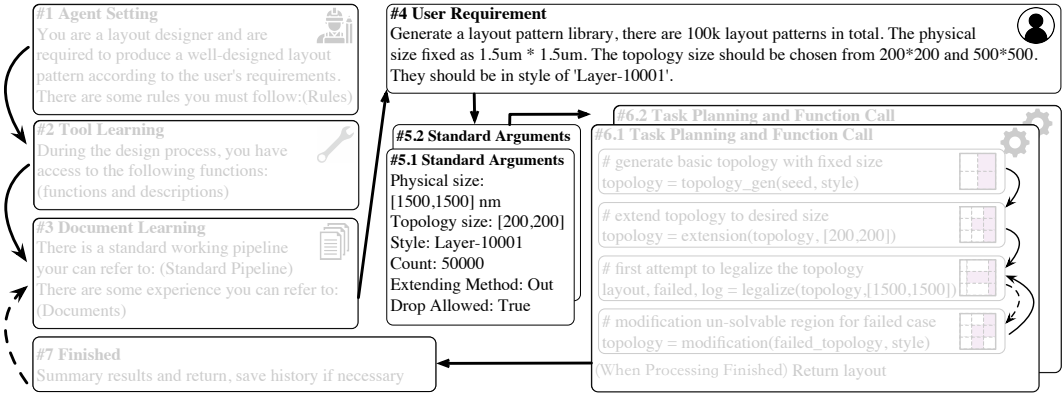
# Pattern Extension



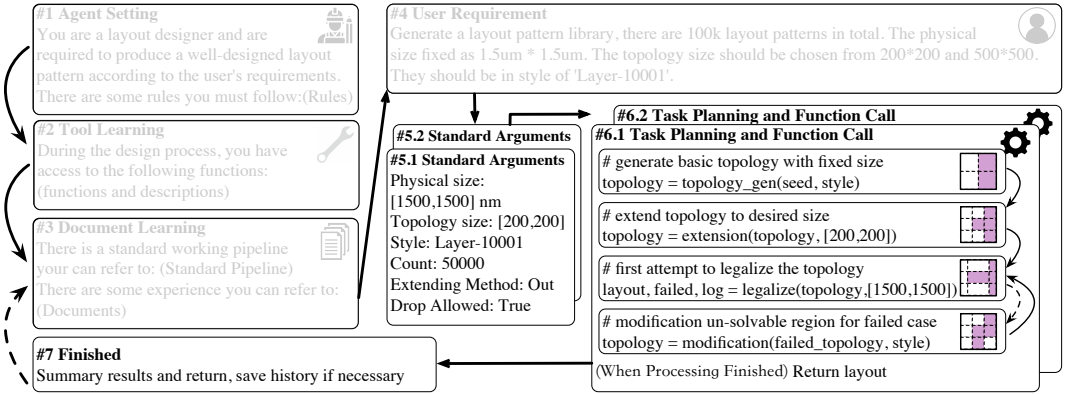
# Example-Pipeline



# Example-Pipeline



# Example-Pipeline



# Experiments

# Evaluation

- Pattern Diversity. Shannon entropy of the pattern complexity.

$$H = -\sum_{i}\sum_{j} P(c_{xi}, c_{yj}) \log P(c_{xi}, c_{yj}), \tag{4}$$

- Pattern Legality.

$$L = \frac{\#\text{ Legal Patterns}}{\#\text{ All Patterns}}. \tag{5}$$

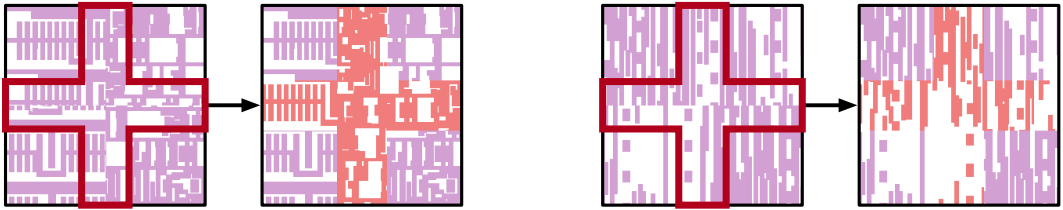
# Free-size Pattern Generation

| Task | Set/Method | Training Set* | Size | Layer-10001 | | Layer-10003 | | Total† | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Legality (↑) | Diversity (↑) | Legality (↑) | Diversity (↑) | Legality (↑) | Diversity (↑) |
| Fixed-size | Real Patterns | / | $128^2$ | / | 10.731 | / | 8.769 | / | 10.625 |
| | CAE+LegalGAN [ICCAD'20] | Layer-10001 | | 3.74% | 5.814 | / | / | / | / |
| | VCAE+LegalGAN [ICCAD'20] | Layer-10001 | | 84.51% | 9.867 | / | / | / | / |
| | LayouTransformer [ICCAD'22] | Layer-10001 | | 89.73% | 10.527 | / | / | / | / |
| | DiffPattern [DAC'23] | Layer-10001/10003 | | **99.97%** | 10.711 | 99.98% | 8.578 | **99.98%** | 10.633 |
| | ChatPattern | Layer-10001/10003 | | **99.97%** | **10.796** | **99.99%** | **8.625** | **99.98%** | **10.650** |
| Free-size | Real Patterns | / | $256^2$ | / | 12.702 | / | 10.696 | / | 12.695 |
| | [DAC'23] w/ Concatenation | Layer-10001/10003 | | 57.78% | 10.719 | 93.69% | 10.511 | 75.74% | 11.706 |
| | ChatPattern | Layer-10001/10003 | | **87.36%** | **11.154** | **99.78%** | **10.556** | **93.57%** | **11.830** |
| | Real Patterns | / | $512^2$ | / | 13.435 | / | 12.139 | / | 13.787 |
| | [DAC'23] w/ Concatenation | Layer-10001/10003 | | 0.29% | 5.714 | 40.83% | 11.555 | 20.56% | 11.359 |
| | ChatPattern | Layer-10001/10003 | | **36.42%** | **10.401** | **98.86%** | **11.620** | **67.64%** | **12.133** |
| | Real Patterns | / | $1024^2$ | / | 13.573 | / | 12.644 | / | 14.109 |
| | [DAC'23] w/ Concatenation | Layer-10001/10003 | | 0.00% | 0.000 | 0.64% | 6.926 | 0.32% | 6.926 |
| | ChatPattern | Layer-10001/10003 | | **1.19%** | **6.438** | **94.96%** | **11.981** | **47.80%** | **11.992** |

## Insights

When pattern size is larger than $256^2$, the legality of ChatPattern sometimes can reach more than **100**× higher than DiffPattern[DAC'23].

# In-Painting

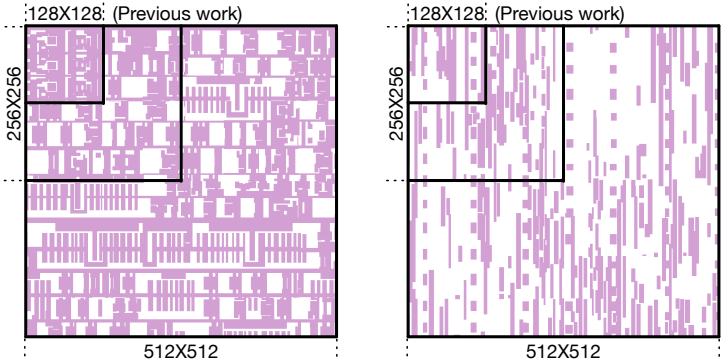


(a) Layer-10001 style

(b) Layer-10003 style

Example of 256×256 topology matrix generated by In-Painting.

# Out-Painting




(a) Layer-10001 style (b) Layer-10003 style

512×512 topology matrix generated by Out-Painting.

# Requirement Auto-formatting

## User Input

Generate a layout pattern library, there are 100k layout patterns in total. The physical size fixed as 1.5um * 1.5um. The topology size should be chosen from 200*200 and 500*500. They should be in style of 'Layer-10001'.

# Requirement - subtask 1
## Basic Part: Topology Size: [200, 200], Physical Size: [1500, 1500] nm, Style: Layer-10001, Count: 50000,
## Advanced Part: Extension Method: Out (Default: Out), Drop Allowed: True (Default: True), Time Limitation: None (Default: None).

# Unseen Mistake-processing

We present a case where the LLM agent adeptly navigates unanticipated mistakes during legalization. It is noteworthy that such error-handling expertise is not pre-coded within the standard operational procedures or the experiential documentation.

% In previous attempts, one pattern can't pass the legalization.
**Thought**: Since legalization has failed twice in the same region, I will try to in-paint that specific area with the same style and then attempt legalization again.
**Action**: Topology_Modification
**Action Input**: "topology_path":${path}, "upper": 12, "left": 56, "bottom": 33, "right": 73, "style": "Layer-10001", "seed": 42
% Continue Processing

THE CHIPS TO SYSTEMS CONFERENCE

SHAPING THE NEXT GENERATION OF ELECTRONICS

JUNE 23-27, 2024

MOSCONE WEST CENTER
SAN FRANCISCO, CA, USA

# Thanks!