Novel Transformer Model Based Clustering Method for Standard Cell Design Automation

Chia-Tung Ho*
Nvidia Research
Santa Clara, CA, USA
chiatungh@nvidia.com

Ajay Chandna
Nvidia
Santa Clara, CA, USA
AChandna@nvidia.com

David Guan
Nvidia
Santa Clara, CA, USA
dguan@nvidia.com

Alvin Ho
Nvidia
Santa Clara, CA, USA
alho@nvidia.com

Minsoo Kim
Nvidia
Austin, TX, USA
minsook@nvidia.com

Yaguang Li
Nvidia
Austin, TX, USA
yaguangl@nvidia.com

Haoxing Ren*
Nvidia Research
Austin, TX, USA
haoxingr@nvidia.com

ABSTRACT

Standard cells are essential components of modern digital circuit designs. With process technologies advancing beyond 5nm, more routability issues have arisen due to the decreasing number of routing tracks (RTs), increasing number and complexity of design rules, and strict patterning rules. The standard cell design automation framework is able to automatically design standard cell layouts, but it is struggling to resolve the severe routability issues in advanced nodes. As a result, a better and more efficient standard cell design automation method that can not only resolve the routability issue but also scale to hundreds of transistors to shorten the development time of standard cell libraries is highly needed and essential.

High quality device clustering with the considerations of routability in the layouts of different technology nodes can reduce the complexity and assist finding the routable layouts faster. In this paper, we develop a novel transformer model-based clustering methodology - training the model using LVS/DRC clean cell layouts and leveraging the personalized page rank vectors to cluster the devices with the attentions to netlist graph and learned embeddings from the actual LVS/DRC clean layouts. On a benchmark of 94 complex and hard-to-route standard cells, the proposed method not only generates 15% more LVS/DRC clean layouts, but also achieves average 12.7× faster than previous work. The proposed method can generate 100% LVS/DRC clean cell layouts over 1000 standard cells and achieve 14.5% smaller cell width than an industrial standard cell library.

CCS CONCEPTS

• **Hardware → Standard cell libraries**; • **Computing methodologies → Machine learning**.

ISPD '24, March 12–15, 2024, Taipei, Taiwan
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0417-8/24/03.
https://doi.org/10.1145/3626184.3633314

KEYWORDS

Standard cell design automation, Electronic design automation, Machine learning

ACM Reference Format:
Chia-Tung Ho, Ajay Chandna, David Guan, Alvin Ho, Minsoo Kim, Yaguang Li, and Haoxing Ren. 2024. Novel Transformer Model Based Clustering Method for Standard Cell Design Automation. In *Proceedings of the 2024 International Symposium on Physical Design (ISPD '24), March 12–15, 2024, Taipei, Taiwan.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3626184.3633314

1 INTRODUCTION

Standard cells are the essential components of digital Very Large-Scale Integration (VLSI) designs. As process technologies relentlessly advance beyond 5nm, the decreasing number of routing tracks, increasing number of design rules, and strict patterning rules are leading to severe routability issues in standard cell layouts. Experienced standard cell designers are facing enormous challenges to design the high-quality standard cell layouts and deliver cell libraries, which have thousands of standard cells for each technology node, in time for VLSI circuit design and physical implementation, because of the limited in-cell routing resources, increasing number and complexity of design rules, and strict patterning rules. Therefore, a fast and efficient automatic standard cell design automation method that can not only resolve the routability issue but also scale to hundreds of transistors is of great importance in advanced technology nodes.

Recently, some automated standard cell synthesis tools such as NVCell [1] and BonnCell [2], have been shown to generate high quality cell layouts on advanced technology nodes. However, one of the key challenges of these automated standard cell synthesis tools is that the generated placement for any given cell could be unroutable or unable to be routed without DRC errors. NVCell2 [3] develops a lattice graph routability model and successfully improves the routability in the advanced technology nodes. However, its performance is not scale to hundreds of transistors because the model inference needs to be performed for every action in the simulated annealing-based placement algorithm [1] and the cell-level metrics (i.e., cell width (CW) and total wirelength (TWL)) are compromised for routability.

High quality device clustering with the considerations of diffusion sharing/break, routability, and DRCs of routing metals in the layout of different technology nodes can reduce the complexity, narrow down the searching spaces, and assist finding the routable layouts faster in the placement stage as shown in Figure 1. In this paper, we developed a novel transformer model based clustering methodology - training the model using LVS/DRC clean cell layouts and leveraging the personalized page rank vectors [4, 5] to draw the local attentions on netlist graph and consider the learned embeddings preference from the actual LVS/DRC clean layouts of each device for clustering. We demonstrate that the proposed novel transformer model based clustering method can successfully generate 100% LVS/DRC clean cell layouts over 1000 standard cells in an industrial standard cell library, and achieves 14.5% smaller cell area than the industrial standard cell library.

Our main contributions are as follows.

- We propose a novel transformer model based clustering methodology for standard cell layout automation. We train the model using LVS/DRC clean cell layouts and leverage the personalized page rank vectors [4, 5] to cluster the devices with the attentions to netlist graph and learned embeddings from the actual LVS/DRC clean layouts. The proposed novel transformer model based clustering methodology can generate the device clusters with the considerations of the routability, design rules, and netlist structure, simultaneously.
- On a set of 94 hard-to-route cell benchmark on the advanced nodes, the proposed method not only generates 15% more LVS/DRC clean layouts than one of the state-of-the-arts routability-driven standard cell design automation method, NVCell2 [3], but also improves avg. Cell Width (CW) and avg. Total Wirelength (TWL) by 3.9%, and 3.3%, respectively. Moreover, the proposed method achieves 12.7× faster than NVCell2 [3] on average for the complex cells.

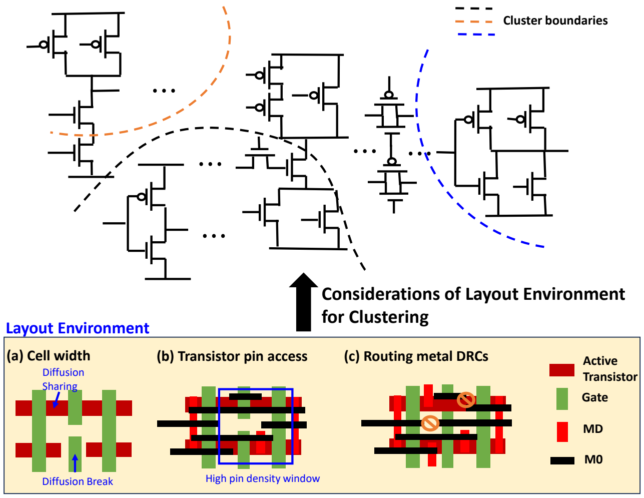

Figure 1: An illustration of high quality clustering with the considerations of diffusion break/sharing, routability, and DRCs of routing metals in the layout can reduce the complexity, narrow down the solution space, and find the routable layouts faster in the placement stage.

- We use multi-objective BOHB [6, 7] to explore the weights of cell width, routability, and cluster design constraints to generate competitive cell layouts in terms of cell width, routability, number of DRC errors, and total wirelength.
- The proposed novel transformer model based clustering methodology for standard cell layout automation achieves cell layouts with smaller cell area than the existing industrial standard cell library for 14.5% of over 1000 cells.

The remaining sections are organized as follows. Section 2 reviews prior works in standard cell layout automation and gives a brief overview of the original NVCell [1] and NVCell2 [3] that this work is built upon. Section 3 describes our novel transformer model based clustering methodology. Section 4 presents our main experiments. Section 5 concludes the paper.

2 BACKGROUND

Standard cell layout automation includes placement and routing steps. The placement step places devices; the routing step connects device terminals and pins based on net connectivity.

Sequential standard cell synthesis approach: It performs the placement step first and then the routing step such as [8], [2], [9], [1], and [3]. Placement techniques include heuristic based methods, exhaustive search based methods, and mathematical programming based methods. Routing techniques include channel routing, SAT, and Mixed-Integer Linear Programming (MILP) based routing methods. [8] leveraged MILP algorithms to find optimal device placement. [2] and [9] used branch and bound or dynamic programming techniques to explore optimal transistor placement exhaustively and then formulate the MILP for in-cell routing. Recently, [1] and [3] used the simulated annealing technique to generate optimal transistor placement, leveraged genetic algorithms for routing, and applied reinforcement learning to fix the design rule violation. Ho *et al.* [3] proposed lattice graph routability model to improve the routability of generated placements, but its performance is not scale to hundreds of transistors because the model inference needs to be performed for every action in the simulated annealing-based placement algorithm and the cell-level metrics (i.e., CW and TWL) are compromised for routability.

Simultaneously standard cell synthesis approach: There are also several works proposing to solve the placement and routing problems simultaneously using Satisfiability-modulo-theory (SMT) in [10], [11], [12], and [13]. By encoding the design rules in the engine, simultaneously standard cell synthesis approach can generate routable standard cell layouts. However, their scalability is worse than sequential standard cell synthesis approaches on large and complex standard cell designs (i.e., more than 50 devices).

2.1 NVCell Framework

The NVCell framework [1] [3] is a sequential standard cell automation approach, which consists of placement and routing stages. In the placement stage, given a set of PFET and NFET devices, the goal of placement is to place them on the PFET row and NFET row of the cell layout while satisfying technology constraints. Here, the conventional simulated annealing algorithm is selected because its adaptability to custom layout constraints and ease of implementation. The designed simulated annealing based algorithm does both pairing and ordering simultaneously. Simulated annealing

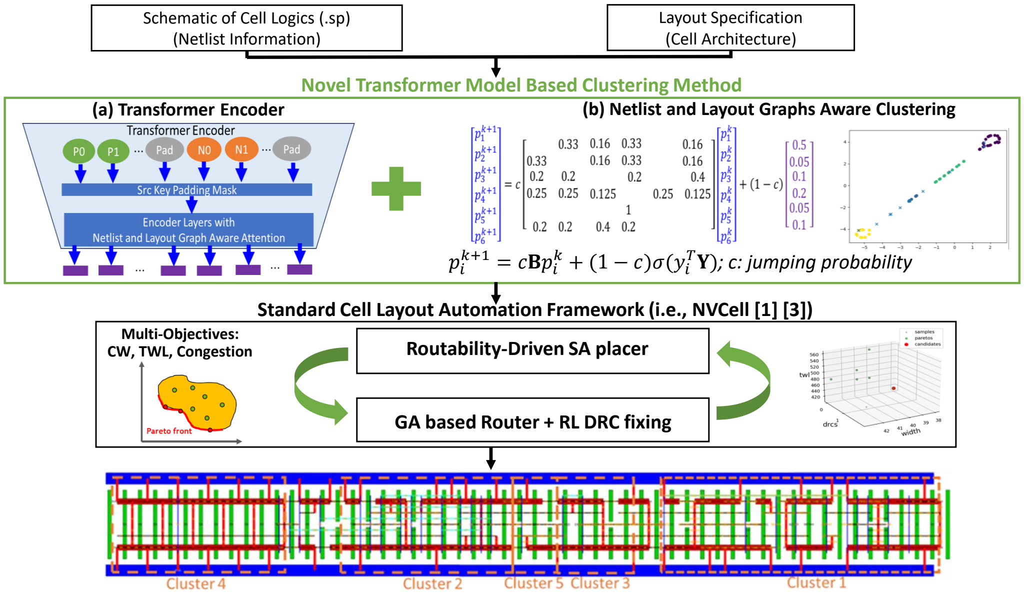
Schematic of Cell Logics (.sp)
(Netlist Information)
Layout Specification
(Cell Architecture)
Novel Transformer Model Based Clustering Method
(a) Transformer Encoder
Transformer Encoder
P0
P1
Pad
N0
N1
Pad
Src Key Padding Mask
Encoder Layers with
Netlist and Layout Graph Aware Attention
(b) Netlist and Layout Graphs Aware Clustering
$p_i^{k+1} = c\mathbf{B}p_i^k + (1-c)\sigma(y_i^T\mathbf{Y})$; c: jumping probability
Standard Cell Layout Automation Framework (i.e., NVCell [1] [3])
Multi-Objectives:
CW, TWL, Congestion
Pareto front
Routability-Driven SA placer
GA based Router + RL DRC fixing
twl
drcs
width
Cluster 4
Cluster 2
Cluster 5
Cluster 3
Cluster 1

Figure 2: Overview of the proposed transformer model based clustering methodology and standard cell layout automation framework flow. The generated device clusters are fed into the standard cell layout automation framework to reduce the complexity and assist finding routable solutions faster.

makes moves on a placement representation which specifies the placement order of pins, ordering of NFET and PFET devices, and whether to flip a device orientation (switching the source and drain positions). It optimizes a scoring function which is a weighted sum of cell width, routability estimation, and estimated wirelength. These moves can be categorized either by the types of moves or by the targeted devices of the moves. The Flip changes all targeted devices flip flag. The Swap swaps targeted devices. The Move moves targeted devices to a specific location. The target devices can be either consecutive PFET devices, consecutive NFET devices, or consecutive PFET/NFET device pairs. The simulated annealing algorithm is implemented based on the modified Lam annealing schedule [14] that requires no hyperparameter tuning.

There are two routabiltiy-driven placement methods in the NVCell framework: a pin density aware congestion (PDA) metric and a lattice graph routability model. The PDA metric is used to capture the number of required contacts to lowest metal layer (i.e., M0) in a local area and number of crossing nets, which exclude the nets within the local area. The lattice graph routability modelling approach captures the routability of local areas, routability impacts between local areas, and global net connections in the standard cell. These routability-driven placement methods help improve the success rate of routable placement generation, but they are not efficient enough to scale to hundreds of devices [3] and the cell-level metrics (i.e., CW and TWL) are compromised for routability.

In the routing phase, there are two steps: a genetic algorithm-based routing step and a Reinforcement Learning (RL)-based DRC fixing step [15]. The genetic algorithm drives a maze router to create many routing candidates, and the DRC RL agent reduces the number of DRCs of a given routing candidate [1]. The genetic algorithm based routing algorithm uses routing segments as the genetic representation in that it ensures that good routing islands in the routing structure are preserved during genetic operations such as crossover and mutation. The fitness of each individual chromosome in a generation is evaluated based on two metrics: the number of unrouted terminal pairs and the number of DRCs. The standard cell external pins are dynamically determined with dynamic external pin allocation methods [3] in the routing phase.

3 NOVEL TRANSFORMER MODEL BASED CLUSTERING METHODOLOGY

We introduce the novel transformer model based clustering methodology for standard cell layout automation frameworks as shown in Figure 2. Given a cell netlist and layout specification, the proposed novel transformer model based clustering methodology generates the device clusters. It is used to assist the placer to find optimal solutions faster in the standard cell layout automation framework. Here, we integrate the novel transformer model based clustering method with NVCell framework [1, 3], which uses a simulated annealing based placer [1] with PDA metric and device clustering constraint to generate placements. Then, the genetic algorithm based router with dynamic external pin allocation [3] are leveraged to generate optimized standard cell layouts. We firstly show transformer encoder modeling approach in section 3.1. Then, we introduce the novel netlist and layout graphs aware clustering approach in section 3.2. Lastly, we introduce the methodology of using the generated clustering result for standard cell design automation in section 3.3. The notations are in Table 1.

Table 1: Notation Table

Term	Description
G	Netlist logic graph.
G_l	Layout graph.
v	Represent a device node in G or G_l.
D	Set of devices in the netlist. $\|D\|$ is the number of devices.
$N(v)/N_l(v)$	Set of neighbor nodes of v in G/G_l.
d_v	Extracted graph embedding of device v.
h_v	Hidden representation of device v in transformer encoder model.
y_v	Representative embedding of device v from transformer encoder model.
dim	The dimension of d_v and y_v. Here, dim=128.
$\mathbf{Y}$	Representative embedding matrix with $\|D\| \times dim$ from transformer encoder model.
$\mathbf{B}$	Device placement-aware transition matrix of G.
C_i	Clustering cost of i^{th} cluster in a given placement.

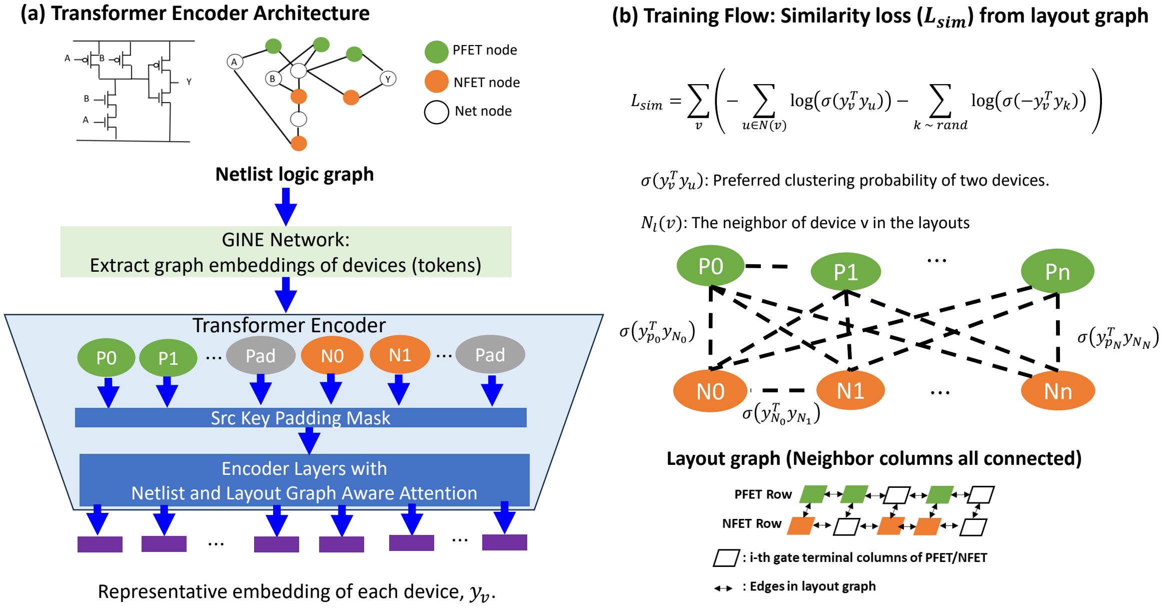
(a) Transformer Encoder Architecture
A
B
B
A
Y
A
B
Y
PFET node
NFET node
Net node
Netlist logic graph
GINE Network:
Extract graph embeddings of devices (tokens)
Transformer Encoder
P0
P1
Pad
N0
N1
Pad
Src Key Padding Mask
Encoder Layers with
Netlist and Layout Graph Aware Attention
Representative embedding of each device, y_v.
(b) Training Flow: Similarity loss (L_{sim}) from layout graph
$L_{sim} = \sum_v \left(-\sum_{u \in N(v)} \log(\sigma(y_v^T y_u)) - \sum_{k \sim rand} \log(\sigma(-y_v^T y_k)) \right)$
$\sigma(y_v^T y_u)$: Preferred clustering probability of two devices.
$N_l(v)$: The neighbor of device v in the layouts
P0
P1
Pn
N0
N1
Nn
$\sigma(y_{P_0}^T y_{N_0})$
$\sigma(y_{P_N}^T y_{N_N})$
$\sigma(y_{N_0}^T y_{N_1})$
Layout graph (Neighbor columns all connected)
PFET Row
NFET Row
: i-th gate terminal columns of PFET/NFET
: Edges in layout graph

Figure 3: (a) The transformer encoder architecture. (b) Training flow: the transformer encoder model is trained with similarity loss (L_{sim}) based on the layout graph.

3.1 Transformer Encoder Modeling

We introduce the transformer encoder modeling approach, which generates the representative embeddings of devices in the layout graph from the input netlist logic graph. Figure 3 shows the transformer encoder modeling approach including the transformer encoder architecture and training flow. We extract the netlist logic graph from the circuit netlist as described in [1]. The layout graph consists of PFET and NFET gate terminal columns from left to right, and PFETs and NFETs including dummy devices are placed on the intersecting grid points. Each grid point is connected to the adjacent grid points in the layout graph. There are multiple possible layout graphs for a netlist logic graph. Given the netlist logic graph, we use the transformer encoder model to learn the device clusters to optimize the cell-level metrics and routability on the layout graph. Firstly, we introduce the input features and device token extraction. Secondly, we show the proposed transformer model architecture. Lastly, we introduce the training flow.

Input Features and Device Token Generation: Given a cell netlist, we construct a netlist logic graph, $G(V, E)$, where the pins, nets, and devices are represented as nodes and the connectivity between the nodes are derived from the types of connections, such as source-to-net, net-to-gate, pin-to-net, etc. Then, we use GINE [16] network, a modified Graph Isomorphism Network (GIN) [17] to incorporate edge features into the aggregation procedure, to extract the netlist logic graph embeddings of devices as the input tokens to the transformer model. The GINE updates node representations as Equation (1).

$$d_v^{(k)} = \Theta((1+\epsilon)d_v^{k-1} + \sum_{u \in N(v)} ReLU(d_u^{k-1} + m_\theta(e_{u,v}))) \tag{1}$$

where d_v^k and d_v^{k-1} are the new and previous embeddings of node v, respectively; d_u^{k-1} is the previous embedding of node v's neighbor node u; $e_{u,v}$ is the edge features of edge between u and v; m_θ is a linear network that maps edge feature dimensions to node embedding dimensions; and Θ is another linear network that maps input node embedding dimensions to output node embedding dimensions.

Transformer Model Architecture: The transformer model allows each input token attends to the information at any position globally and then process its representative embeddings. The locality of a sequential data are encoded as the relative distance of any two positions [18] in the transformer layer. However, the nodes are not arranged as a sequence for graphs. They can be in a multi-dimensional spatial space and connected by edges. To encode the structural information in the netlist logic and layout graphs, we use the netlist and layout graph aware multi-head attention layer to capture the spatial relation of devices in the netlist logic graph, and the device placement relation (i.e., diffusion sharing, vertical PFET-NFET gate/diffusion connection, etc.) in the layout graph.

Figure 4 shows the netlist and layout graph aware multi-head attention mechanism. For spatial relation, given any netlist logic graph $G(V, E)$, we can use a function $\phi(v, u)$ to measure the spatial relation between v and u in the graph G. The function ϕ can be defined by the connectivity between nodes. In this paper, we define $\phi(v, u)$ as the shortest path distance of v and u in the graph G.

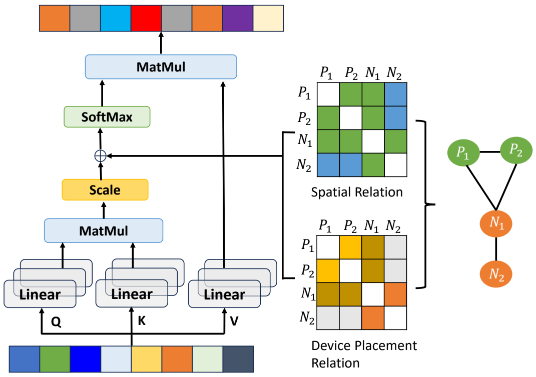

Figure 4: Netlist and layout graph aware multi-head attention mechanism.

For device placement relation, we can extract the potential diffusion sharing and vertical PFET-NFET gate/diffusion connection relations ($\kappa(v,u)$) of device pairs in the layout graph from the connectivity of devices in the spice netlist. In the netlist and layout graph aware multi-head attention layer, we assign learnable scalars for spatial relation and device placement relation and use them as bias term in the self-attention module. The Query-Key product ($a_{v,u}$) of v and u can be written as Equation (2) below.

$$a_{v,u} = \frac{(h_v \mathbf{W_Q})(h_u \mathbf{W_K})^T}{\sqrt{dim}} + b_{\phi(v,u)} + b_{\kappa(v,u)} \tag{2}$$

where $h_v, h_u \in R^{1\times dim}$ are hidden representation of v and u, respectively. The $\mathbf{W_Q}$ and $\mathbf{W_K}$ are the projection matrix for Query and Key, respectively. Here, we consider the single-head self-attention for simplicity of illustration.

The transformer encoder layer is composed of 3 stacked identical layers in this paper. Each layer has two sub-layers. The first is a multi-head self-attention mechanism [19] with the relative graph distance and layout aware diffusion attention bias, and the second is a simple, position wise fully connected feed-forward network. The residual connection [20] is employed around each of the two sub-layers, followed by layer normalization [21]. The residual connections, all sub-layers in the model, as well as the embedding layers, produce outputs of dimension dim=128. The number of heads is 8 in the multi-head self-attention layer.

Training Flow: We adopt unsupervised learning to learn the representative embeddings of devices in layout graph for device clustering. We use the "similarity loss" (L_{sim}) as the objective for training flow and it is directly calculated from the learned representative embeddings of devices as shown in Figure 3 (b). The key idea behind is to learn the relations of devices being placed adjacently in the LVS/DRC clean layout graph and encourage the devices which share similar relations to have higher probabilities of being assigned into the same cluster, while making devices that don't have the similar relations in layout graph to have lower probabilities. As a result, the trained transformer model can generate device clusters that considering the device accessibility, routability, and DRC in layout graph. The loss function L_{sim} is designed as:

$$L_{sim} = \sum_{v}(-\sum_{u \in N_l(v)} log(\sigma(y_v^T y_u)) - \sum_{k \sim rand} log(\sigma(-y_v^T y_k))) \tag{3}$$

where y_v denotes the learned representative embeddings of node v, σ denotes the sigmoid function, and $k \sim rand$ denotes the random sampling operation over the full LVS/DRC clean layout graph. $N_l(v)$ represents the neighbor nodes of v in the LVS/DRC clean layout graph. By minimizing Equation (3), the neighboring nodes in the layout graph will be encouraged to have similar representative embeddings y, which increases the probability of them being assigned to the same cluster and hence reduces the complexity and assist finding the routable layout more efficient and faster.

3.2 Netlist and Layout Graphs Aware Clustering

We leverage the personalized page rank vectors [4, 5] of devices to draw the local attentions on netlist graph and consider the learned embeddings from the transformer encoder model to generate robust clustering results across different cell designs, Here, the personalized page rank vector of a device is a stationary distribution to other devices for the random walk on the netlist logic graph with the predicted preference from the trained model. Given the representative embeddings of i^{th} device, y_i, the predicted cluster preference probability vector of i^{th} device is $\sigma(y_i^T \mathbf{Y})$. $\mathbf{Y}$ is the representative embedding matrix with $|D| \times dim$ from the trained model. The personalized page rank vector of i^{th} device (p_i) can be obtained by using power iterations of Equation (4). Then, we apply DBScan algorithm [22] to cluster devices on the obtained personalized page rank vectors of devices.

$$p_i^k = c\mathbf{B}p_i^{k-1} + (1-c)\sigma(y_i^T \mathbf{Y}) \tag{4}$$

where $p_i^k \in R^{|D|\times 1}$ is the personalized page rank vector of i^{th} device in k^{th} iteration. The p_i^0 is set to equal to the $\sigma(y_i^T \mathbf{Y})$ for power iterations. c and $(1-c)$ represent probabilities jumping between the netlist logic graph the predicted preference probability vector of i^{th} device. $\mathbf{B} \in R^{|D|\times|D|}$ is the device placement-aware transition matrix of netlist logic graph, G, and $|D|$ is the number of devices in the netlist. Different types of connection edge of device terminals have different edge weights based on the diffusion sharing and vertical gate/diffusion connection property in $\mathbf{B}$. For example, the edge weights of diffusion sharing edges, and vertical gate/diffusion connection edges are set to larger weights than other edges (i.e., drain-to-gate, pin-to-gate, etc.). The element, $b_{i,j}$, in $\mathbf{B}$ is written in the Equation (5).

$$b_{i,j} = \frac{we_{j,i}}{\sum_{k \in out(j), k \neq j} we_{j,k}} \tag{5}$$

where $we_{j,i}$ is the weight of edge from j^{th} device to i^{th} device. $out(j)$ represents a set of devices that connected to j^{th} device. In summary, the proposed novel transformer model-based clustering methodology can leverage netlist logic connections, and the predicted routability-aware preference probability vector to generate high quality device clusters for reducing complexity, narrowing down the searching spaces, and assist finding routable placements.

3.3 Clustering Constraints for Standard Cell Design Automation

We introduce the methodology of using generated clustering constraints for standard cell design automation. We use a device clustering cost (i.e., C_i) which measures the number of transistors, that are not in i^{th} cluster, being placed inside the i^{th} cluster bounding box region in the current placement. Then, we minimize the sum of device clustering costs of clusters to generate the standard cell layouts. Figure 5 shows an illustration of the clustering cost calculation. In this example, C_1 is 3 since there are 3 transistors (i.e., 2 transistors are C2 and 1 transistor is C3) being placed in the bounding box region of cluster 1. C_2 and C_3 can be calculated accordingly.

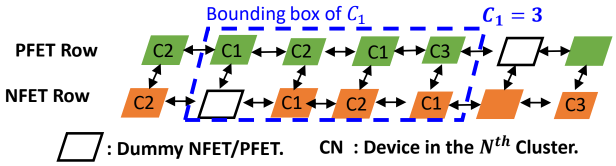

Figure 5: An example of clustering cost (C_i) calculation.

The device clustering cost can be integrated into the objective functions of standard cell synthesis framework for multi-objective optimization [3] in Equation (6).

$$\textbf{Minimize } w_a \times CW + w_m \times TWL + w_c \times \sum_{i \in |C|} C_i \tag{6}$$

where w_a, w_m,and w_c are the weights of cell area, total wire length, and the summation of device clustering costs, respectively. Here, we use multi-objective BOHB [6, 7] method for multi-objective optimization for Equation (6), which includes multi-objective tree-structured parzen Estimator (MOTPE) [7], and HyperBand [23]. MOTPE is a multi-objective Bayesian optimization algorithm for the hyperparameter optimization with categorical hyper-parameters in a tree-structured [7]. Hyperband searches the best time allocation for each of the hyperparameter configurations [23].

4 EXPERIMENTAL RESULTS

Our work is implemented with Python and runs on a server with multiple Intel Xeon CPUs. It generates a simplified grid-based cell layout, which is given to a separate Perl program called Sticks to handle DRC checking and conversion to tapeout quality Cadence Virtuoso layout.

We firstly study the clustering quality of various models and representative node vectors for DBScan clustering method [22] on training set, which has 250 LVS/DRC clean cell layouts, using silhouette coefficient[1]. To demonstrate that the high quality device clustering with the considerations of routability in the layouts can reduce the complexity and assist finding the routable layouts faster, we conduct extensive studies on the routability and performance using the selected 94 complex and hard to route standard cells in an advanced node from an industrial technology [3]. Lastly, we apply the proposed transformer model based clustering method to generate optimized cell layouts on the single row cells (i.e., over 1000 cells) in an industrial standard cell library.

Figure 6 shows the number of devices and nets statistics of 250 cells training set and 94 complex cell benchmark. The maximum number of devices and nets in 250 cells training set are 84 and 35, respectively. For the 94 complex cell benchmark, there are 5 cells with number of devices larger than 90, and the maximum number of nets is around 50.

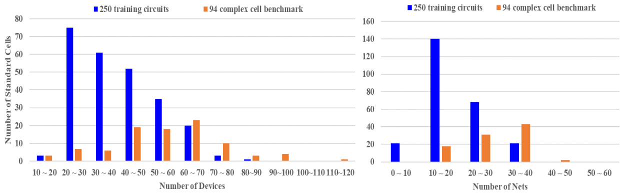

Figure 6: Cell statistics of 250 cells training set and 94 complex cell benchmark.

4.1 Clustering Quality Study

We study the clustering quality of various models and representative node vectors for DBScan clustering method [22] on 250 cells training set using silhouette coefficient. The silhouette coefficient is calculated based on the clustering result and the actual LVS/DRC clean layout placement. For model comparison, we train GINE network and the proposed transformer model based architecture using the similarity loss L_{sim} to learn the representative embeddings of each devices in layout graph. We use the following representative vectors with DBScan clustering method for comparison.

Table 2: Clustering Quality Table. Silhouette coefficients of various models and node representative vectors for DBScan clustering method on 250 cells for training. Pred. preference=Predicted preference. PPR without Pred. Preference=personalized page rank vector without predicted preference. PPR with Pred. Preference=personalized page rank vector with predicted preference.

Rep. Node Vectors	Model	Avg. Silhouette	Impr. (%)
Pred. preference	GINE	0.21	200%
	Transformer based	0.42	50%
PPR without Pred. Preference	-	0.22	186%
PPR with Pred. Preference	GINE	0.37	70%
	Transformer based (proposed)	0.63	-

- **Predicted preference:** We apply the predicted preference probability vector of each device (i.e., $\sigma(y_i^T \mathbf{Y})$) from the trained models directly for DBScan clustering.
- **Personalized page rank vector without predicted preference:** We calculate the personalized page rank vector of each device on netlist logic graph and the preference probability vector distribution which is 1 for the corresponding device node and 0 elsewhere [24].
- **Personalized page rank vector with predicted preference:** We calculate the personalized page rank vector of each device on netlist logic graph and the predicted preference probability vector from trained models (i.e., Equation (4)).

Table 2 shows the silhouette coefficients of various models and node representative vectors for DBScan clustering method on 250 cells training set. For model comparison, we observed that transformer based model architecture achieves 200%, and 70% improvements on avg. silhouette coefficients compared to GINE network using predicted preference vector only, and personalized page rank vector with predicted preference for clustering, respectively. The advantages of proposed transformer based model architecture over GINE are from the globally receptive field, structural information of netlist graph, and device placement relations in layout graph. Figure 7 shows the training loss curves of the proposed transformer based model architecture and GINE. We can observed that the proposed transformer based model architecture can learn and capture the relations of devices in layout graph given the netlist logic graph more efficiently than GINE model. For representative node vectors comparison, the proposed method achieves 50%, and 186% larger silhouette coefficients than predicted preference vector only, and personalized page rank vector without predicted preference, respectively. The avg. probability c is 0.52 in the proposed method, which indicates that the proposed clustering method leverages the

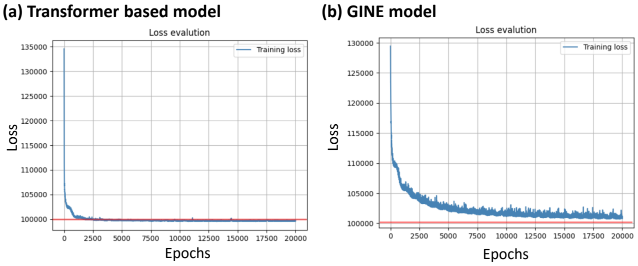

Figure 7: The training loss, L_{sim}, curves of using (a) transformer based model architecture, and (b) GINE model. The red line indicate the L_{sim} = 100000.

[1] The silhouette Coefficient is widely used to measure the quality of clusters by calculating the mean intra-cluster distance and the mean nearest-cluster distance for each sample.

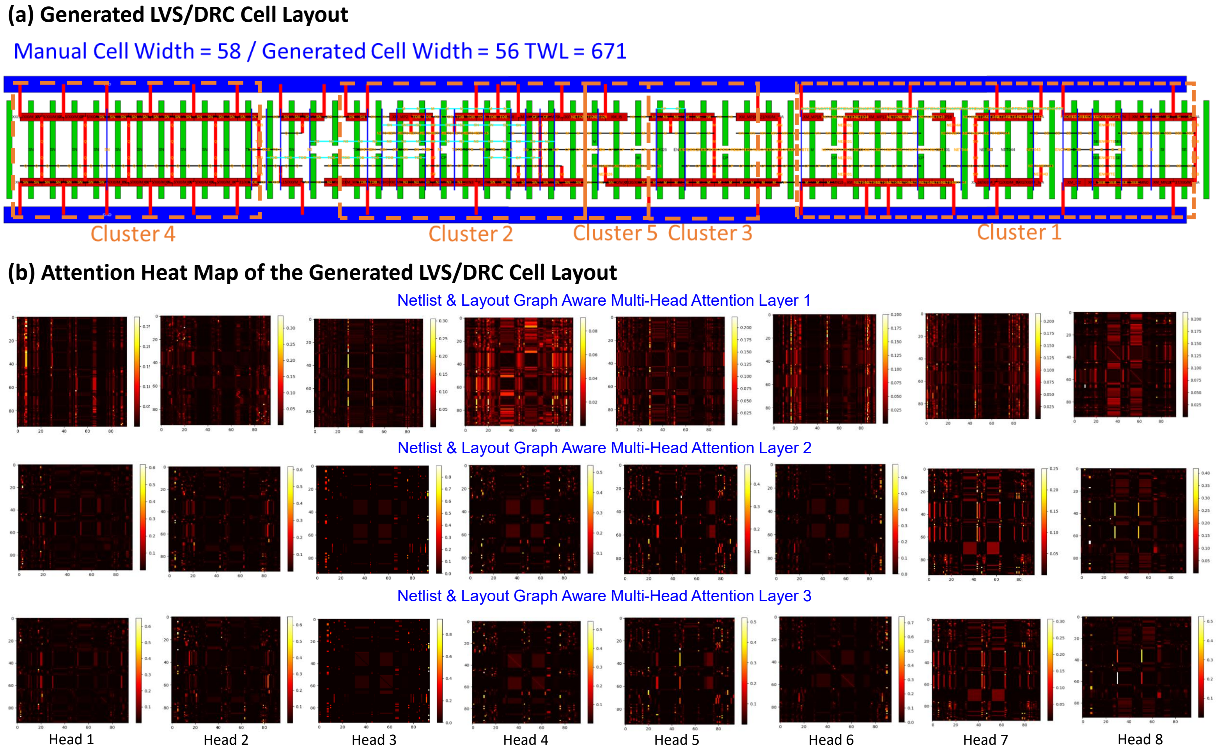
(a) Generated LVS/DRC Cell Layout
Manual Cell Width = 58 / Generated Cell Width = 56 TWL = 671
Cluster 4
Cluster 2
Cluster 5
Cluster 3
Cluster 1
(b) Attention Heat Map of the Generated LVS/DRC Cell Layout
Netlist & Layout Graph Aware Multi-Head Attention Layer 1
Netlist & Layout Graph Aware Multi-Head Attention Layer 2
Netlist & Layout Graph Aware Multi-Head Attention Layer 3
Head 1
Head 2
Head 3
Head 4
Head 5
Head 6
Head 7
Head 8

Figure 8: (a) Generated LVS/DRC clean layout of a latch cell with 56 CPPs. Manual cell width is 58 CPPs. (b) Attention heat map of the latch cell.

device placement-aware connections in netlist logic graph and the predicted cluster preference probability considering the transistor terminal accessibility, and routability in the layout graph together.

In summary, the proposed clustering method using transformer based model architecture and the personalized page rank vector with predicted preference can generate more robust and high quality cluster results.

4.2 Routability and Performance Experiment

It is very challenging and critical for standard cell design automation framework to generate the LVS/DRC clean cell layout for complex and hard to route cells with industrial cell standard and be able to scale up to hundreds of transistors. To demonstrate that the high quality device clustering with the considerations of routability in the layouts can reduce the complexity and assist finding the routable layouts faster, we use the 94 hard to route standard cell benchmark [3] for the routability and performance study. The number of transistors is from 14 to 114. There are 57 Flip-flops, 21 Combinational cells, and 15 Latchs. There are 35 unseen cell designs for the model in the 94 cell benchmark.

Table 3 shows the cell-level metrics (i.e., CW, TWL, and routability) of NVCell [1], NVCell2 [3], and the proposed method using the 94 cell benchmark. We don't measure the improvement of the proposed method over NVCell [1] since NVCell [1] failed to generate any routable cell layouts of the 94 cell benchmark. The proposed method not only generates 15% more LVS/DRC clean layouts than NVCell2 [3], but also improves avg. CW and avg. TWL by 3.9%, and 3.3%, respectively. Moreover, the avg. CW of the proposed method is 0.8% smaller than the manual cells from the industrial library. Compared to NVCell2 [3], the proposed method increases 33.3% and 57.1% of the number of smaller CW and same CW layouts, respectively. In the meanwhile, the number of larger CW layouts are decreased by 57.7%. Figure 8 shows the generated LVS/DRC clean cell layouts of a latch cell (~100 devices) using the proposed method in stick format and its attention heat maps. For the performance, Figure 9 shows that the proposed method achieves 12.7× faster than NVCell2 [3] on average for the cells with #devices larger than 80 in the 94 cell benchmark.

4.3 Entire Cell Library Experiment

We apply the proposed transformer model based clustering method to generate optimized cell layouts on the single row cells (i.e., over 1000 cells) in an industrial standard cell library. Given the device

Table 3: Cell statistics of the 94 hard-to-route cell benchmark of proposed methodology and NVCell2 [3]. NVCell [1] is not able to generate any routable layouts for the 94 cells. The average cell width (CW) of the 94 cells from industrial library is 33.90.

	Avg. CW	Avg. TWL	#Routable Cells	#LVS/DRC Clean Cells	Compared with Manual CW		
					Smaller	Same	Larger
NVCell [1]	N/A	N/A	0	0	N/A	N/A	N/A
NVCell2 [3]	34.97	436.67	93	82	21	35	26
Proposed	33.60	422.37	94	94	28	55	11
Impr. [3]	3.9%	3.3%	1%	15%	33.3%	57.1%	-57.7%

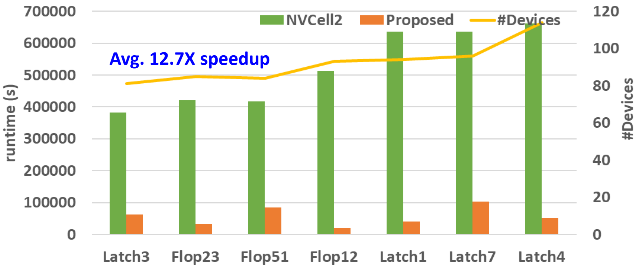

Figure 9: Performance evaluation with cells has more than 80 devices.

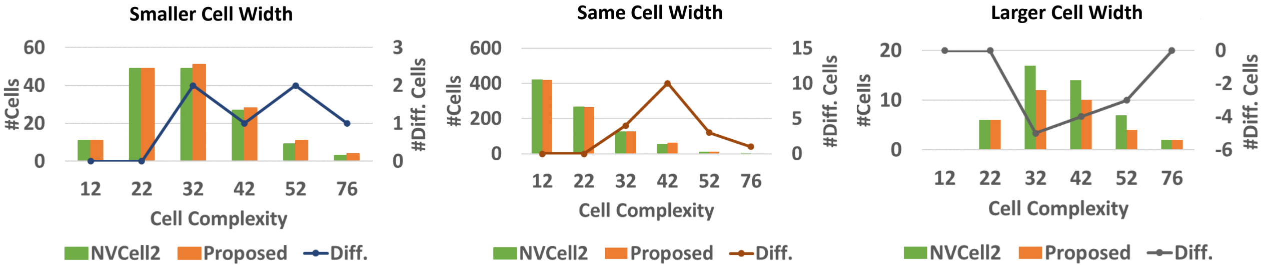

Figure 10: Cell width comparison of NVCell2 [3] and the proposed method. The baseline is the manual designed cell width in the existing industrial library. The proposed method successfully achieves more number of smaller cell width and number of same cell width cells which has devices more than 32.

clusters from the proposed method, we perform multi-objective BOHB [6, 7] method to explore the optimal CW, routability, and TWL in standard cell design automation framework [3].

Table 4 shows the summary of the number of LVS/DRC clean cells, and cell width comparison to the manual designed cells in the existing industrial library. Compared to NVCell2 [3], the proposed transformer model-based clustering method can not only generate LVS/DRC clean layouts of all single row cells in the industrial library but also reduce the number of cells with wider CW by 47.8%. Figure 10 shows the summary of the number of smaller, same, and larger cell with statistics of NVCell2 [3] and the proposed method. The proposed transformer model based clustering method achieves generating more number of smaller and same cell width layouts with #devices from 32 to more than 76, which are more complex and challenging for experienced standard cell designers. Moreover, we select a set of flip-flop cell netlists (i.e., 8 flip-flop cells) and compare the power, performance, and area (PPA) of generated cell layout from the proposed approach using characterization tool with the PPA of manual designed layout. The performance, power, and area are improved up to 7%, 8%, and 4%, respectively. From the studies, we demonstrate the proposed method successfully generates high quality device clusters for reducing the complexity, and assisting finding optimum cell layouts robustly. Overall, with the proposed transformer model-based clustering method, the standard cell design automation framework [3] can automatically generate 100% of over 1000 single row cells and achieve 14.5% smaller and 83.3% same CW than existing industrial library.

Table 4: Cell statistics of NVCell [1], NVCell2 [3], and the proposed transformer model based clustering method on the entire library set (i.e., over 1000 cells). The baseline cell width for cell width comparison is the manual designed cell width in the existing industrial library.

	#LVS/DRC	Cell Width Comparison		
	Clean Cells	Smaller	Same	Larger
NVCell [1]	91.2%	11.8%	77.6%	1.8%
NVCell2 [3]	98.8%	13.7%	80.1%	4.3%
Proposed	100.0%	14.5%	83.3%	2.2%

5 CONCLUSION

We propose a novel transformer model based clustering method for standard cell design automation - training the model using LVS/DRC clean cell layouts and leveraging the personalized page rank vector [4, 5] to cluster the devices with the attentions to netlist graph and learned embeddings from the actual LVS/DRC clean layouts in the layout graph. We firstly demonstrate that the proposed method not only improves the number of LVS/DRC clean cell layouts by 15%, but also improves avg CW and avg TWL by 3.9% and 3.3%, respectively, compared to [1] using the 94 complex and hard to route cell benchmark. In total, the proposed framework can generate 100% LVS/DRC clean cell layouts over 1000 standard cells in an industrial standard cell library. Then, we demonstrate that the proposed method can generate smaller cell layouts for 14.5% of cells compared to an existing industrial standard cell library of over 1000 cells. Lastly, we show the performance, power, and area of a set of generated flip-flop cell layouts are improved up to 7%, 8%, and 4%, respectively, compared to the manual designs.

REFERENCES

[1] Haoxing Ren and Matthew Fojtik. Nvcell: Standard cell layout in advanced technology nodes with reinforcement learning. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 1291–1294. IEEE, 2021.

[2] Pascal Van Cleeff, Stefan Hougardy, Jannik Silvanus, and Tobias Werner. Bonncell: Automatic cell layout in the 7-nm era. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 39(10):2872–2885, 2019.

[3] Chia-Tung Ho, Alvin Ho, Matthew Fojtik, Minsoo Kim, Shang Wei, Yaguang Li, Brucek Khailany, and Haoxing Ren. Nvcell 2: Routability-driven standard cell layout in advanced nodes with lattice graph routability model. In *Proceedings of the 2023 International Symposium on Physical Design*, pages 44–52, 2023.

[4] Lawrence Page. The pagerank citation ranking: Bringing order to the web. technical report. *Stanford Digital Library Technologies Project, 1998*, 1998.

[5] Taher Haveliwala, Sepandar Kamvar, Glen Jeh, et al. An analytical comparison of approaches to personalizing pagerank. Technical report, Technical report, Stanford University, 2003.

[6] Stefan Falkner, Aaron Klein, and Frank Hutter. Bohb: Robust and efficient hyperparameter optimization at scale. In *International Conference on Machine Learning*, pages 1437–1446. PMLR, 2018.

[7] Yoshihiko Ozaki, Yuki Tanigaki, Shuhei Watanabe, and Masaki Onishi. Multiobjective tree-structured parzen estimator for computationally expensive optimization problems. In *Proceedings of the 2020 genetic and evolutionary computation conference*, pages 533–541, 2020.

[8] Ang Lu, Hsueh-Ju Lu, En-Jang Jang, Yu-Po Lin, Chun-Hsiang Hung, Chun-Chih Chuang, and Rung-Bin Lin. Simultaneous transistor pairing and placement for cmos standard cells. In *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1647–1652. IEEE, 2015.

[9] Yih-Lang Li, Shih-Ting Lin, Shinichi Nishizawa, Hong-Yan Su, Ming-Jie Fong, Oscar Chen, and Hidetoshi Onodera. Nctucell: A dda-aware cell library generator for finfet structure with implicitly adjustable grid map. In *Proceedings of the 56th Annual Design Automation Conference 2019*, pages 1–6, 2019.

[10] Daeyeal Lee, Dongwon Park, Chia-Tung Ho, Ilgweon Kang, Hayoung Kim, Sicun Gao, Bill Lin, and Chung-Kuan Cheng. Sp&r: Smt-based simultaneous place-and-route for standard cell synthesis of advanced nodes. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 40(10):2142–2155, 2020.

[11] Chung-Kuan Cheng, Chia-Tung Ho, Daeyeal Lee, and Dongwon Park. A routability-driven complimentary-fet (cfet) standard cell synthesis framework using smt. In *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pages 1–8. IEEE, 2020.

[12] Chung-Kuan Cheng, Chia-Tung Ho, Daeyeal Lee, Bill Lin, and Dongwon Park. Complementary-fet (cfet) standard cell synthesis framework for design and system technology co-optimization using smt. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 29(6):1178–1191, 2021.

[13] Chung-Kuan Cheng, Chia-Tung Ho, Daeyeal Lee, and Bill Lin. Multirow complementary-fet (cfet) standard cell synthesis framework using satisfiability modulo theories (smts). *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, 7(1):43–51, 2021.

[14] Vincent A Cicirello. On the design of an adaptive simulated annealing algorithm. In *Proceedings of the international conference on principles and practice of constraint*

programming first workshop on autonomous search, 2007.

[15] Haoxing Ren and Matthew Fojtik. Standard cell routing with reinforcement learning and genetic algorithm in advanced technology nodes. In *Proceedings of the 26th Asia and South Pacific Design Automation Conference*, pages 684–689, 2021.

[16] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019.

[17] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[18] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018.

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[21] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[22] Nadia Rahmah and Imas Sukaesih Sitanggang. Determination of optimal epsilon (eps) value on dbscan algorithm to clustering data on peatland hotspots in sumatra. In *IOP conference series: earth and environmental science*, volume 31, page 012012. IoP Publishing, 2016.

[23] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research*, 18(1):6765–6816, 2017.

[24] Fan Chung Graham and Alexander Tsiatas. Finding and visualizing graph clusters using pagerank optimization. In *International Workshop on Algorithms and Models for the Web-Graph*, pages 86–97. Springer, 2010.