Ckt2Vec: Efficient Electrical Encoding for Analog Circuit Representations in Vector Space

Peng Xu, Yapeng Li, Tinghuan Chen, Tsung-Yi Ho, Bei Yu

***Abstract*—Representation learning for analog circuits is challenging due to the continuous electrical characteristics of devices, compared to the discrete states of digital circuits. While graph neural networks (GNNs) show promise in analog circuit tasks, existing methods neglect the intrinsic electrical properties governing device-specific behaviors. Traditional device feature encoding methods present limitations: one-hot encoding is space-consuming and fails to effectively characterize inter-device similarities, while text encoding introduces erroneous estimation. We propose Ckt2Vec, a novel framework that integrates electrical characteristics into analog circuit representation learning. By encoding frequency-domain embeddings of current-voltage (I-V) curves via a spectral extractor, Ckt2Vec compresses nonlinear device-specific behaviors into low-dimensional embeddings while preserving physical fidelity. A graph-based contrastive learning approach further generates hierarchical circuit representations, capturing both block- and system-level interactions. Evaluated on three downstream tasks, including circuit classification, subcircuit detection, and circuit edit distance prediction, Ckt2Vec outperforms traditional one-hot and text-based encoding methods with less space consumption and better capability in capturing analog behavior.**

I. Introduction

ELECTRONIC systems form the foundation of modern technological advancement, powering advancements ranging from healthcare devices to self-driving transportation. Their pervasive presence depends on effective integration between analog and digital elements. Within this electronic design paradigm, **analog circuits** serve as essential bridges connecting physical input with digital computation [1]. Analog design complexity also arises from its inherent sensitivity to physical characteristics. Unlike digital systems confined to binary logic, analog components are continuously affected by parasitic interference, manufacturing inconsistencies, and layout-induced variations. Despite modern engineering practices, analog design workflows, spanning schematic creation to physical layout, remain predominantly manual, requiring extensive expertise and often resulting in a labor-intensive process prone to human error [2].

This work is supported by The National Key Research and Development Program of China (No. 2023YFB4402900), The National Natural Science Foundation of China (No. 92573108 and No. 62304197), The Research Grants Council of Hong Kong SAR (No. CUHK14211824 and No. CUHK14201624), and the MIND project (MINDXZ202404). (Corresponding author: Bei Yu)

Peng Xu, Tsung-Yi Ho, and Bei Yu are with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, NT, Hong Kong SAR.

Yapeng Li and Tinghuan Chen are with the School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, China.

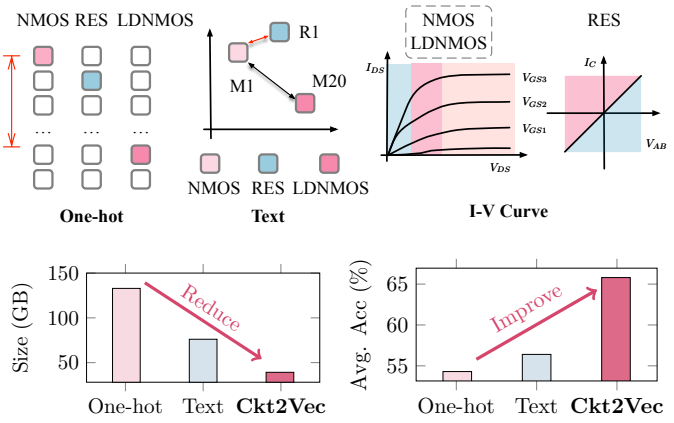


Fig. 1 Previous encoding methods are space-consuming and fail to estimate similarities among devices using one-hot encoding, or lead to wrong estimations using text encoding. For example, LDNMOS is more similar to NMOS than RES, but is wrongly estimated by those two methods, as the identifier names of "M1" and "R1" are closer. In this paper, we utilize the spectral representation of I-V curves to encode devices' electrical information, thereby improving accuracy while reducing space usage.

Recent advancements in machine learning, especially graph neural networks (GNNs) [3], [4], show significant promise in addressing the intricate challenges of analog circuit design. Analog circuits inherently lend themselves to graph-based representations, where components like devices and nets form interconnected nodes and edges. Modern approaches leveraging this graph structure have achieved breakthroughs in key design tasks [5]–[14]. For analog topology classification, graph embeddings are used to identify layout templates for circuits like amplifiers and filters [7]. For subgraph identification, GNN architectures are employed in [5], [6] to minimize symmetry errors in layout synthesis, outperforming traditional manual methods. Node-centric GNNs are adopted to streamline optimization and reduce reliance on iterative simulation [8]–[11]. Furthermore, graph-based circuit representations are applied in diverse design stages, from large-scale subgraph matching during testing [12] to performance-driven routing optimizations [13], [14]. Recent efforts to develop generalized pretraining frameworks for analog circuits (as exemplified in works like [15], [16]) integrate domain knowledge via text embeddings derived from device names and supplementary layout data.

However, while these methods demonstrate potential in layout relationship estimation, representation learning for only the analog circuits level remains an underexplored area in machine learning-driven electronics design.

A critical challenge in analog circuit representation learning lies in the encoding of analog devices, which have traditionally neglected the continuous electrical properties of devices. As illustrated in the upper part of Fig. 1, current methods either adopt one-hot encoding, which is space-consuming and incapable of modeling inter-device similarities, or employ text-based augmentation that might introduce erroneous estimation with the text edit distance. For instance, while laterally-diffused NMOS (LDNMOS) exhibits stronger electrical similarity to NMOS than to resistors (RES), both encoding strategies fail to capture this electrical similarity accurately. In contrast, digital circuit representation learning has successfully embedded foundational encoding with Boolean algebraic operations [17] like in [18]–[23]. Analog circuits, however, lack a comparable mechanism to systematically encode the continuous electrical characteristics of devices into the representations. The intrinsic physical properties of analog devices are still neglected in the representation learning phase.

To address the encoding limitations, we propose **Ckt2Vec**, a framework that incorporates the electrical characteristics of analog devices without requiring circuit-level simulations or bias point extraction. The Ckt2Vec framework first scans input voltages across different devices to obtain I-V curve data under various operating regions defined in the process design kit (PDK), forming initial features for the devices. Then, the framework extracts spectral features using Fast Fourier Transform (FFT) on measured I-V data, effectively reducing dimensionality while preserving the nonlinear physical characteristics of devices. For block-level and circuit-level circuit representation, we apply general graph-based contrastive learning. Evaluated on circuit classification, subcircuit detection, and edit distance prediction, Ckt2Vec surpasses one-hot [7], [24] and text-based encoding [16] methods, demonstrating superior accuracy in capturing analog behavior with much less space usage, as shown in Fig. 1. The main contributions of this paper are listed as follows:

- **Electrical feature extractor**: Employs FFT-based spectral feature extraction to compress I/V curves into low-frequency components, preserving nonlinear device characteristics while reducing space usage significantly.
- **Graph representation learning**: Uses general graph-based contrastive learning to generate hierarchical representations (block- and circuit-level) for analog circuits.
- **Downstream tasks validation**: Demonstrates superior performance over traditional methods (one-hot [7], [24] and text-based encoding [16]) in three downstream tasks-circuit classification and circuit edit distance prediction-proving its effectiveness in analog circuit representation learning with only $1/3$ runtime overhead of original data generation.

To the best of our knowledge, this is the first work to integrate the I-V curves of devices into analog circuit representation learning. Our approach pioneers the integration of device electrical physics in analog characterization. The proposed feature generation method, dataset, and downstream tasks will serve as foundational elements, paving the way for analog circuits representation learning.

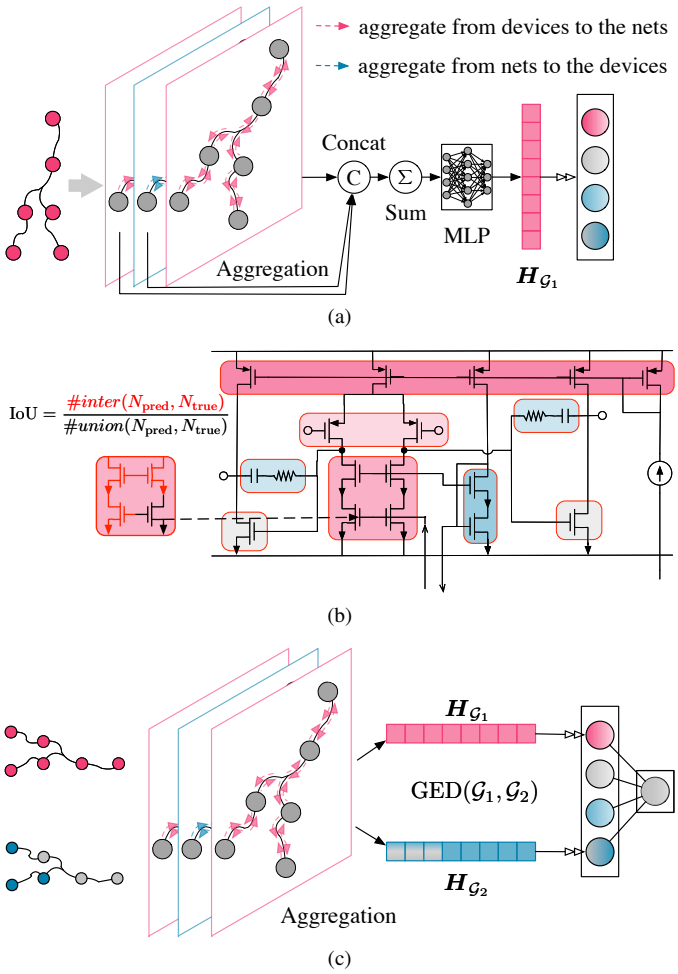


Fig. 2 The illustration of the downstream tasks: (a) *Analog circuit classification*: evaluates the ability of models to classify analog circuits into predefined categories; (b) *Analog subcircuit detection*: tests the precision of identifying and labeling fundamental subcircuits within larger analog circuits; (c) *Analog Graph Edit Distance (GED) prediction*: assesses whether learned embeddings capture the structural similarity between circuits with different functions.

II. Related Work and Preliminaries

A. Graph Neural Networks (GNNs)

Graph Neural Networks (GNNs) are powerful representation learning techniques [25] with many key applications [26]. Concretely, GNNs aggregate node features from neighbors and stack multiple layers to capture long-range dependencies. For instance, GraphSAGE [26] concatenates node features with mean/max/LSTM-pooled neighboring information. GAT [27] aggregates neighbor information using learnable attention

TABLE I Typical analog device types

Device	Symbol	Number of pins	Pin types
NMOS		$n_d+n_g+n_s+n_b$	nd, ng, ns, nb
PMOS		$n_d+n_g+n_s+n_b$	pd, pg, ps, pb
NPN		$n_b+n_c+n_e$	nb, nc, ne
PNP		$n_b+n_c+n_e$	nb, nc, ne
Diode		2	n+, n-
Resistor		2	n+, n-
Capacitor		2	n+, n-
Inductor		2	n+, n-

weights. GIN [25] converts the aggregation as a learnable function based on the Weisfeiler-Lehman test.

GNNs consist of two major components, where the aggregation step aggregates node features of target nodes' neighbors, and the combination step passes the previous aggregated features to networks to generate node embeddings. Mathematically, we can update node v's embedding at the l-th layer by:

$$\begin{aligned} \boldsymbol{e}_v^l &= \text{AGGREGATE}\left(\left\{\boldsymbol{h}_u^{l-1} | \forall u \in \mathcal{N}(v)\right\}\right), \\ \boldsymbol{h}_v^l &= \text{COMBINE}\left(\boldsymbol{h}_v^{l-1}, \boldsymbol{e}_v^l\right), \end{aligned} \tag{1}$$

where $\mathcal{N}(v)$ denotes the neighbours of v. A main difference between different GNNs lies in the design of the aggregation and combine functions, AGGREGATE$(\cdot)$ and COMBINE$(\cdot)$.

B. GNNs for Analog Representation Learning (ARL)

Analog circuits pose unique representation challenges due to their bipartite structure (devices and nets) and heterogeneous device types. Recent work explores GNNs to learn analog circuit representations directly from graph structure [5]–[14].

An analog circuit can be naturally represented as a graph [28]. Traditional approaches convert an analog circuit into a multigraph [5], [29], [30]. In the multigraph model, each device is depicted as a node, and interconnections are modeled as edges, with specific edge types corresponding to different port types. The transformation from circuit to multigraph requires removing all nets while maintaining interconnections. Since this transformation is not one-to-one, it can lead to potential information loss issues. Bipartite graphs [7], [24], [31] address this limitation by explicitly modeling nets as nodes equally.

We give the formal definition of a bipartite analog circuit graph as follows:

Definition 1 (Device)**.** *Each device $v_d \in \mathcal{V}_d$ is associated with attributes such as type (NMOS, CAP, etc.), parameters (e.g., W, L, resistance, capacitance), and connectivity metadata as shown in TABLE I.*

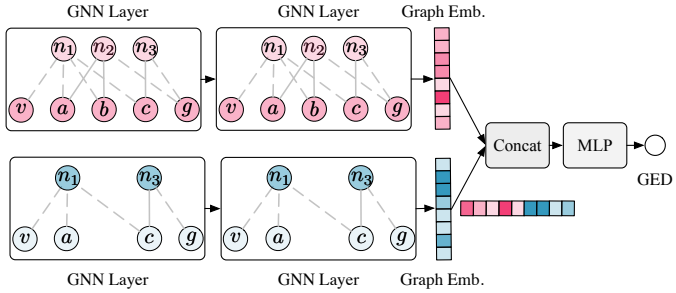


Fig. 3 Architecture of the GED prediction framework.

Definition 2 (Net)**.** *Each net $v_n \in \mathcal{V}_n$ is a junction where multiple devices connect, with topological metrics (degree, betweenness centrality) decided by the connected pin of devices as shown in TABLE I.*

Definition 3 (Analog Circuit Graph)**.** *The analog circuit graph consists of two groups of nodes $\mathcal{V}_d, \mathcal{V}_n$, corresponding to devices and nets. Those two groups of nodes are connected by a set of edges $\mathcal{E}$, presenting a connection Hence, the analog circuit graph has the form of a bipartite graph as $\mathcal{G} = \{\mathcal{V}_d, \mathcal{V}_n, \mathcal{E}\}$.*

C. Downstream Tasks for Analog Representation Learning

In this paper, we introduce three downstream applications as shown in Fig. 2 to verify our pre-trained circuit embedding.

Problem 1 (Analog Circuit Classification)**.** *Assign a class label $y \in \{1, 2, \ldots, C\}$ to a circuit graph $\mathcal{G}$. Let $\mathcal{D}_{cls} = \{(G_i, y_i)\}_{i=1}^N$ be a dataset of labeled circuit graphs. The task is to learn a classifier $f_{cls} : \mathbb{R}^d \to \mathbb{R}^C$ such that:*

$$\hat{y} = softmax(f_{cls}(z)), \quad where\ z = \phi(G).$$

The loss function is typically the cross-entropy function:

$$\mathcal{L}_{cls} = -\sum_{i=1}^{N}\sum_{c=1}^{C} \mathbb{I}(y_i = c) \log \hat{y}_{i,c}.$$

Problem 2 (Analog Subcircuit Detection)**.** *Let $\mathcal{D}_{det} = \{(G_i, S_i, v_j^i, y_{i,j})\}_{i=1}^M$, where $y_{i,j} \in \{0, 1\}$ indicates whether node v_j^i in circuit G_i belongs to subcircuit type S_i. The task is to learn a detector $f_{det} : \mathbb{R}^d \times \mathbb{R}^d \to [0, 1]$ such that:*

$$\hat{y}_{i,j} = \sigma(f_{det}(z_{v_j}, S_i)), \quad where\ z_{v_j} = \phi(v_j),$$

where σ is the sigmoid function. The loss is cross-entropy:

$$\mathcal{L}_{det} = -\sum_{i=1}^{M}\sum_{j=1}^{N_i} \left[y_{i,j} \log \hat{y}_{i,j} + (1 - y_{i,j}) \log(1 - \hat{y}_{i,j})\right],$$

where N_i is the number of nodes in circuit G_i.

Problem 3 (Analog Graph Edit Distance (GED) Prediction)**.** *Predict the minimum number of edit operations (node/edge insertions, deletions, substitutions) required to transform graph*

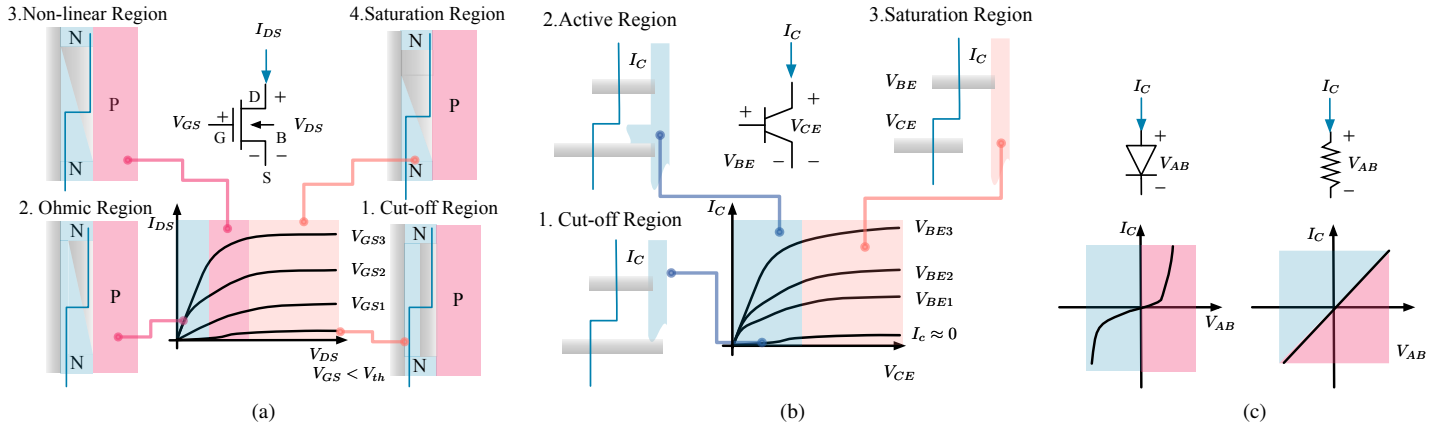


Fig. 4 The typical electrical characteristics of the nonlinear devices and linear devices: (a) The I-V curve of the NMOS devices; (b) The I-V curve of the BJT devices; (c) The I-V curves of the Diode and Resistor devices. I-V curves represent the electrical characteristics of these devices and can effectively reflect the electrical similarity between different devices. We use I-V curves to generate the initial features of different devices.

G_1 into G_2. Let $\mathcal{D}_{ged} = \{(G_1^i, G_2^i, d_i)\}_{i=1}^K$, where d_i is the ground-truth GED between G_1^i and G_2^i. The task is to learn a regressor $f_{ged} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ such that:

$$\hat{d} = f_{ged}(z_1, z_2), \quad \text{where } z_1 = \phi(G_1),\ z_2 = \phi(G_2).$$

The loss is mean squared error (MSE):

$$\mathcal{L}_{ged} = \frac{1}{K}\sum_{i=1}^{K}(\hat{d}_i - d_i)^2.$$

To clarify our GED prediction pipeline, Fig. 3 further illustrates the network architecture. Each input circuit graph is encoded using a shared GNN to produce graph-level embeddings. These embeddings are then fused through concatenation, including pairwise interactions, and passed into an MLP regression head. The MLP outputs the predicted graph edit distance (GED) between the two circuits. This Siamese-style design allows the model to capture both structural and feature-level similarities of circuit graphs.

III. Ckt2Vec: Electrical Physics-Informed Analog Circuits Representation Learning

A. Overview

In this section, we introduce the Ckt2Vec framework, which differs from traditional one-hot or text-based encodings by leveraging device I–V curves to capture physical behaviors. There are several major algorithmic challenges in enabling this methodology: (i) extracting electrical features to represent continuous device characteristics (Section III-B), (ii) reducing the dimensionality of high-resolution I–V curves via spectral analysis (Section III-C), and (iii) learning circuit-level invariances through self-supervised graph representation learning (Section III-D).

As shown in Fig. 5, the Ckt2Vec framework begins with a preprocessing stage, where individual devices from the PDK are simulated under systematic voltage sweeps to generate

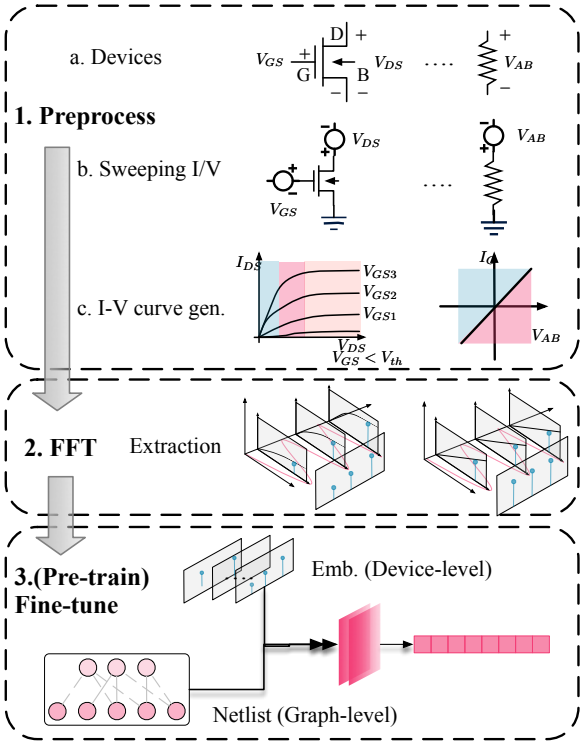


Fig. 5 The overall flow of our Ckt2vec framework.

their corresponding I–V curves. In the next step, an FFT-based feature extraction is applied to the generated curves, yielding compact spectral representations of the electrical characteristics. These device-level embeddings serve as the input to the learning stage, where we perform pre-training and fine-tuning on circuit netlists. By combining device-level fingerprints with graph-level representations, Ckt2Vec provides a unified embedding space that captures both the physical properties of devices and the topological structures of circuits.

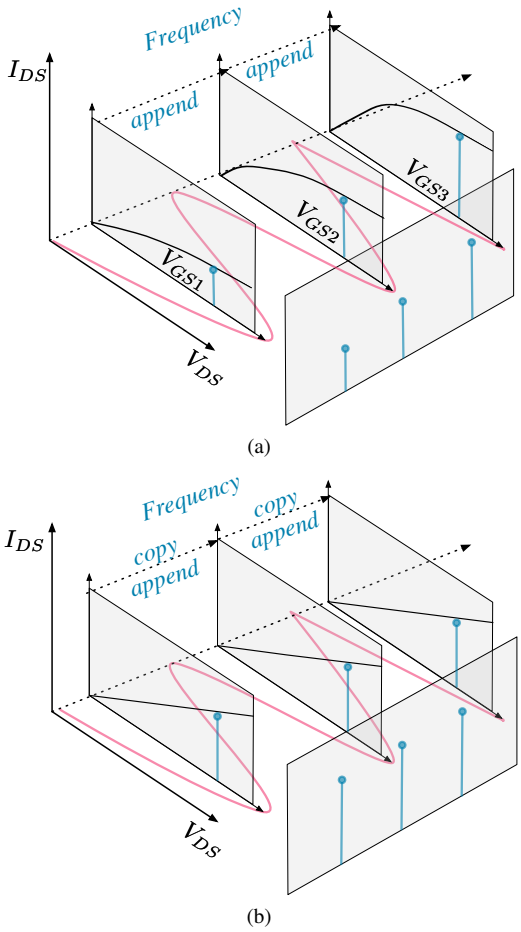


Fig. 6 The visualization of frequency-domain feature extractor for I-V Curves for: (a) Devices necessitating additional excitation voltage (e.g., NMOS/PMOS, PNP/NPN); (b) Devices without excitation voltage requirements (e.g., Diodes, Resistors, Capacitors, Inductors). We use ***append*** to piece together I-V curves from different excitation, and apply ***copy-and-append*** to ensure I-V curves without excitation have the same dimensions. FFT is used to extract the first k components from them.

B. Device Electrical Features Generation

Traditional analog circuit representation learning predominantly employs simplified one-hot encoding for device types [9], [12] and manually engineered features like device name [16], which suffer from a fundamental limitation: discrete encoding fails to capture the continuously varying physical properties of analog devices adequately. Conventional discrete encoding methods inevitably discard higher-order differential features inherent in these continuous physical processes. To overcome these limitations, physics-informed characterization must confront the continuous nature of analog devices. Circuit researchers often employ methods such as **current-voltage (I-V) characteristic curves** to describe the continuous electrical behavior of various devices. The I-V characteristic curves, serving as physical fingerprints of semiconductor devices, comprehensively record dynamic responses in bias space through voltage sweeping [32].

Algorithm 1 Frequency-Domain Feature Extraction for I-V Curves

Input: The raw I-V curves: $\mathcal{C} = \{(V_{GS}^{(i)}, V_{DS}^{(i)}, I_D^{(i)})\}_{i=1}^{N}$ with N sample points;

Output: Device feature vector: $\boldsymbol{f} \in \mathbb{R}^{2k}$;

1: Normalize voltage ranges:

$$V_{GS} \in [V_{th}, V_{DD}], V_{DS} \in [0, V_{DD}];$$

2: Apply FFT to drain the current sequence

$$\boldsymbol{F} = \mathcal{F}([I_D^{(1)}, \ldots, I_D^{(M)}]);$$

3: Preserve k low-frequency components

$$\tilde{\boldsymbol{F}} = [\Re(F_1), \Im(F_1), \ldots, \Re(F_k), \Im(F_k)];$$

4: Concatenate with sizing-related parameters:

$$\boldsymbol{f} = [V_{th}, W, L] \oplus \tilde{\boldsymbol{F}};$$

5: **Return** $\boldsymbol{f}$;

In this work, we adopt the I-V characteristic curves to generate the initial features for devices. Ckt2Vec does not rely on circuit-level bias extraction or testbench simulations. Instead, it uses lightweight device-level voltage sweeps within PDK-defined operating ranges, pre-computed once per device type and reused across different netlists.

Nonlinear and linear behaviors exhibit significant differences in the I-V characteristics of semiconductor devices and passive components. The I-V characteristics of semiconductor devices and passive components exhibit distinct differences between nonlinear (e.g., MOSFETs) and linear behaviors. For metal-oxide-semiconductor field-effect transistors (MOSFETs), operational regions are determined by the gate-source voltage V_{GS} and drain-source voltage V_{DS}. As illustrated in Fig. 4(a), these devices (both NMOS and PMOS) operate in three distinct regimes: cutoff region (active when $V_{GS} < V_{th}$ for NMOS and $V_{GS} > V_{th}$ for PMOS), ohmic and linear region (dominates when $V_{DS} < V_{GS} - V_{th}$ for NMOS and $V_{DS} > V_{GS} - V_{th}$ for PMOS), saturation region (occurs then $V_{DS} \geq V_{GS} - V_{th}$ for NMOS and $V_{DS} \leq V_{GS} - V_{th}$ for PMOS).

As shown in Fig. 4(b), bipolar junction transistors (BJTs, comprising NPN and PNP) switch operational states via the base current I_B: cutoff (near-zero collector current when $I_B = 0$), active amplification ($I_C = \beta I_B$), and saturation (collector current limited by external circuits), with carrier directions differing based on polarity. Diodes exhibit exponential growth under forward bias and minimal leakage current under reverse bias until breakdown, as shown in the left part of Fig. 4(c). Among passive components, resistors adhere to Ohm's law ($V = IR$), as shown in the right part of Fig. 4(c). Capacitors follow $I = C \cdot dV/dt$ (current proportional to voltage change rate), and inductors obey $V = L \cdot dI/dt$ (voltage linked to

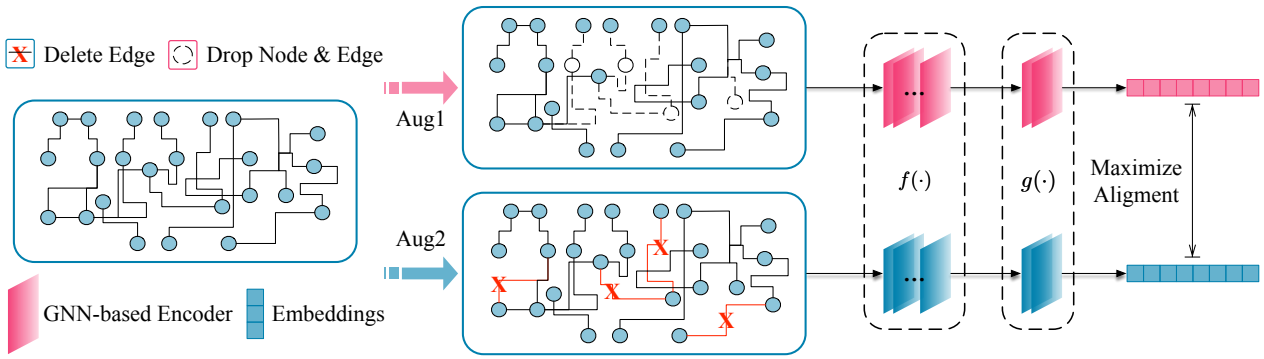


Fig. 7 The general graph contrastive learning framework, which leverages the advancement in general graph representation learning to achieve unsupervised learning for analog circuits.

current change rate).

Considering the AC characteristics of different devices, we also performed AC-related I-V curve scans and obtained the corresponding amplitude and phase information. For the scan results, we adopted a stitching method and frequency-domain dimensionality reduction similar to those used in DC analysis. However, due to space limitations, we omit the detailed description of this part here.

C. Frequency-Domain Based Electrical Feature Extractor

However, the high-dimensional nature of continuous curves (typically $> 10^3$ sampling points) directly leads to the curse of dimensionality in representation learning. More critically, significant nonlinear disparities exist across devices: NMOS exhibits enhancement-mode conduction, PMOS demonstrates depletion-mode behavior, and NPN/PNP involves complex carrier transport mechanisms. These inherent differences pose substantial challenges for constructing unified input feature spaces.

A frequency-domain feature extraction method for physical characterization is proposed to address these issues. The core innovation lies in transforming time-domain I-V curves into the frequency domain via Fast Fourier Transform (FFT), retaining the first k low-frequency components [33]. For devices necessitating additional excitation voltage (e.g., NMOS/PMOS, PNP/NPN), we employ an ***append*** strategy as illustrated in Fig. 6(a). Initially, we define a voltage interval $[0, \Delta V_{DS}]$ to capture the I-V characteristics across all operational regions comprehensively. By systematically varying the external excitation voltage, we generate a series of I-V curves, each corresponding to a distinct excitation level. These curves are sequentially appended to form a periodic signal structured in intervals of $[\Delta V_{DS} \times i, \Delta V_{DS} \times (i+1)]$. Subsequent FFT processing of this composite signal yields a frequency-domain representation, from which we retain the k dominant low-frequency components to construct a compact feature vector.

For devices without excitation voltage requirements (e.g., Diodes, Resistors, Capacitors, Inductors), we implement a ***copy-and-append*** procedure, also depicted in Fig. 6(b), ensuring a standardized format across diverse device types. Here, the interval $[0, \Delta V_{DS}]$ is similarly employed to capture I-V behavior across all operational regions. Through strategic duplication and appending of these curves, we generate a uniform-length periodic signal with intervals of $[\Delta V_{DS} \times i, \Delta V_{DS} \times (i+1)]$, which is subsequently subjected to FFT analysis.

As detailed in Algorithm 1, our technique implements three critical operations: (1) voltage range normalization for operational consistency (line 1); (2) spectral decomposition with adaptive component selection (lines 2–3); (3) hybrid feature vector construction combining spectral coefficients with fundamental device parameters (line 4). The number of retained low-frequency components k in (lines 2–3) balances fidelity and compactness. Empirically, $k = 128$ achieves stable performance, as larger k adds little benefit while increasing runtime cost. The proposed frequency-domain representation effectively captures global shape features of I-V curves and reduces the spatial complexity by keeping the k low-frequency components.

D. Graph Contrastive Learning with Augmentations

Building on recent advances in contrastive learning for graph representation learning, we adopt the Graph Contrastive Learning (GraphCL) framework [34] for self-supervised pre-training of analog circuit graph representations. In graph contrastive learning, pre-training involves maximizing agreement between two augmented views of the same graph by optimizing a contrastive loss in the latent space (as illustrated in Fig. 7). The framework comprises four key components: 1) Graph Data Augmentation: Two related views $\hat{\mathcal{G}}_i$ and $\hat{\mathcal{G}}_j$ of the same graph $\mathcal{G}$ are generated through augmentation techniques to form a positive pair; 2) GNN-Based Encoder: A graph neural network (GNN) encoder $f(\cdot)$ produces graph-level embedding vectors for the augmented graphs. The framework imposes no restrictions on the GNN architecture; 3) Projection Head: A non-linear transformation $g(\cdot)$ maps the embeddings $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$ to an intermediate latent space where the contrastive loss is computed, following recommendations in [35]; 4) Contrastive Loss Function: A contrastive loss $\mathcal{L}(\cdot)$ is designed to maximize similarity between positive pairs ($\boldsymbol{z}_i$, $\boldsymbol{z}_j$) while minimizing agreement with negative pairs.

TABLE II Analog circuit dataset statistics across different circuit categories.

Category (Functionality)	Circuit Examples	Component Counts						
		#NMOS	#PMOS	#RES	#CAP	#Nets	#Nodes	#Edges
Data Converters	DAC_R2R_Type	34	34	103	0	131	307	466
A/D signal conversion	ADC_SAR_Small	27	27	0	0	36	93	196
	ADC_SAR_Large	27	27	0	0	36	93	196
Digital Logic	Decoder_3to7	38	38	0	0	44	123	274
Binary operations	Digital_Mux	35	35	0	0	43	115	223
	ResNetwork_PNP	0	0	80	0	80	163	161
	SignBit_Detector	27	27	4	0	32	93	208
Signal Circuits	LDO_Regular	18	18	33	0	54	126	199
Current Reference	CurrentRef_5uA	24	29	0	0	35	91	183
Op Amp	OpAmp_FirstStage	17	25	0	0	28	73	142
Resistor Divider	ResistorDivider	63	63	60	0	72	261	578
Bandgap Reference	BandgapRef_Mux	20	20	0	0	26	69	147
Timing & I/O	PLL_PhaseDetector	34	34	4	0	44	119	251
Clock Circuits	PLL_PostProcessor	76	76	0	0	76	230	499
I/O Circuits	IO_Buffer_Pad	19	25	5	0	31	83	159
Buffers	Buffer_Calibrated	50	76	1	1	77	208	441
Other Components	SpareCell_Standard	90	90	2	0	90	276	582

Data Augmentation for Analog Circuit Graphs. Given a graph $\mathcal{G} \in \{\mathcal{G}_m : m \in M\}$ in the dataset of M graphs, we formulate the augmented graph $\hat{\mathcal{G}}$ satisfying: $\hat{\mathcal{G}} \sim q(\hat{\mathcal{G}} \mid \mathcal{G})$, where $q(\cdot \mid \mathcal{G})$ is the augmentation distribution conditioned on the original graph. We apply four general data augmentations for analog circuit graph data:

- Node dropping. Given the graph $\mathcal{G}$, node dropping will randomly discard a certain portion of the nodes along with their connections.
- Edge perturbation. It will perturb the connectivity in $\mathcal{G}$ by randomly dropping a certain ratio of edges.
- Attribute masking. Attribute masking prompts models to recover masked vertex attributes using their context information.
- SubCircuit Dropping. This augmentation samples a subcircuit to be dropped off from $\mathcal{G}$ using the information from the netlist. It assumes that the semantics of $\mathcal{G}$ can be much preserved in its (partial) local structure.

Implementation for Graph Contrastive Learning. We employ the normalized temperature-scaled cross-entropy loss (NT-Xent) [36] for contrastive learning. During pre-training, a minibatch containing N graphs is randomly selected and processed through the contrastive framework, generating $2N$ augmented graphs. For the n-th graph in the minibatch, we denote its augmented representations as $\boldsymbol{z}_{n,i}$ and $\boldsymbol{z}_{n,j}$, with corresponding contrastive loss optimization. Negative pairs are implicitly derived from the remaining $N-1$ augmented graphs in the same minibatch, following established practices [37]. Let the cosine similarity function be defined as:

$$\mathrm{sim}(\boldsymbol{z}_{n,i}, \boldsymbol{z}_{n,j}) = \frac{\boldsymbol{z}_{n,i}^{\top} \boldsymbol{z}_{n,j}}{\|\boldsymbol{z}_{n,i}\| \|\boldsymbol{z}_{n,j}\|}. \tag{2}$$

The NT-Xent loss for the n-th graph is then formulated as:

$$\ell_n = -\log \frac{\exp\left(\mathrm{sim}(\boldsymbol{z}_{n,i}, \boldsymbol{z}_{n,j})/\tau\right)}{\sum_{\substack{n'=1 \\ n' \neq n}}^{N} \exp\left(\mathrm{sim}(\boldsymbol{z}_{n,i}, \boldsymbol{z}_{n',j})/\tau\right)}, \tag{3}$$

where τ denotes the temperature parameter. The final loss is computed across all positive pairs in the minibatch. Although pretraining enhances global feature quality, its gains are degraded for some networks on tasks where fine-grained structural details dominate performance. Future work will investigate task-specific pretraining and adaptive fine-tuning to address these limitations.

IV. Experiments

A. Experimental Setup

Dataset Construction. We generate the dataset using the analog circuit topology synthesis framework [38], named as *ANALOG*, containing fundamental topologies ranging from basic building blocks to complex industrial-scale systems. TABLE II provides the details for circuit topologies: transistor counts (PMOS/NMOS), passive components (#C capacitors, #R resistors), and node counts after graph conversion. Our hierarchical benchmark structure enables comprehensive evaluation across different complexity tiers: (1) small-scale: matches core building blocks (inverters, push-pull stages) with complex implementations; (2) medium-scale: tests intermediate structures (operational amplifiers, output stages); (3) large-scale: evaluates system-level designs including ADC cores, and Industrial SerDes circuit ($> 10,000$ nodes). This framework spans circuit scales from 10^1 to 10^3 nodes, covering diverse analog circuit topologies.

Downstream Tasks for Analog Representation Learning. For the *analog circuit classification* task, we expanded the dataset scale by adjusting sizing parameters and further processing of the *ANALOG* dataset classified into 12 general

TABLE III The averaged performance comparison on analog circuits classification task (CLS).

Merthods	Models	Acc@1↑	Acc@2↑	Acc@5↑	Recall↑	F1 Score↑	AUC↑
DATE'20 & TCAD'23 (One-hot) [7], [24]	GCN	0.519	0.736	0.849	0.476	0.419	0.912
	GAT	0.384	0.512	0.790	0.182	0.128	0.840
	GATv2	0.232	0.431	0.624	0.150	0.079	0.794
	GIN	0.783	0.905	0.959	0.767	0.744	0.967
	SAGE	0.795	0.936	1.000	0.790	0.765	0.985
	Average	0.543	0.704	0.844	0.473	0.427	0.900
	Ratio	0.751	0.805	0.885	0.698	0.668	0.943
ICCAD'22 (Text) [16]	GCN	0.556	0.740	0.825	0.537	0.511	0.894
	GAT	0.441	0.628	0.729	0.429	0.385	0.861
	GATv2	0.117	0.298	0.484	0.105	0.024	0.680
	GIN	0.772	0.886	0.959	0.730	0.716	0.965
	SAGE	0.822	0.952	0.998	0.793	0.787	0.987
	Average	0.564	0.690	0.791	0.545	0.514	0.875
	Ratio	0.781	0.788	0.829	0.804	0.805	0.917
Ckt2Vec (ours)	GCN	0.596	0.828	0.955	0.509	0.482	0.939
	GAT	0.496	0.756	0.868	0.356	0.306	0.867
	GATv2	0.533	0.736	0.844	0.360	0.313	0.870
	GIN	0.825	0.988	1.000	0.791	0.767	0.992
	SAGE	0.838	0.935	0.981	0.817	0.799	0.986
	Average	0.658	0.849	0.930	0.567	0.533	0.931
	Ratio	0.910	0.970	0.974	0.836	0.835	0.976
Ckt2Vec (ours) with. *Pretrain*	$\text{GraphCL}_{\text{GCN}}$	0.725	0.842	0.958	0.713	0.682	0.958
	$\text{GraphCL}_{\text{GAT}}$	0.570	0.840	0.968	0.455	0.406	0.933
	$\text{GraphCL}_{\text{GATv2}}$	0.522	0.688	0.852	0.483	0.415	0.900
	$\text{GraphCL}_{\text{GIN}}$	0.887	0.996	1.000	0.885	0.865	0.991
	$\text{GraphCL}_{\text{SAGE}}$	0.910	0.976	0.998	0.887	0.869	0.991
	Average	**0.722**	**0.875**	**0.955**	**0.678**	**0.639**	**0.954**
	Ratio	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**

classes. The dataset is partitioned into training, validation, and testing sets of $360,000$, $28,800$, and $39,600$ analog circuit graphs, referred to as `ANALOG-CLS-428k`.

For the *analog subcircuit detection* task, we followed the experimental framework of Kunal et al. [7], classifying fundamental circuit elements into 12 distinct base categories through manual annotation. We adhere to the electrical characteristics of these components to integrate different circuit connections into a larger circuit. The *voltage* and *ground* nodes are explicitly omitted from detection targets. The device nodes and net nodes retain their original classification labels. The resulting dataset `ANALOG-DET-242k` contains $242,320$ samples distributed as training: 76.9%, validation: 7.7%, testing: 15.4%.

For the *analog graph edit distance (GED) prediction* task, we generated labeled training (70%), validation (10%), and test (20%) datasets by performing mutating operations on the existing data with 194,400 samples. Based on their embeddings, we aim to predict the graph edit distance (GED) between two circuit graphs. This task's purpose is to demonstrate whether the learned embeddings of analog circuits can preserve the circuits' structural similarity. We named this dataset `ANALOG-GED-194k`. We ensure a balanced graph edit distance distribution across all phases.

Baseline Methods. The Baseline methods we use involve common one-hot encoding used in DATE'20 and TCAD'23 [7], [24] and encoding enhancement via text embedding in ICCAD'22 [16]. The one-hot encoding [7], [24] is set to the number of all used device categories, with the default input dimension being 550. For text encoding, we used fastText as in [16], with the default input dimension set to 300. For the encoding of Ckt2Vec, we took the amplitudes and phases of the first $k = 64$ components, resulting in a final input encoding dimension of 142.

Experimental Implementation Details. All experiments were conducted on an Amazon EC2 cluster equipped with an Intel Xeon E5-2686 v4 CPU, 61GB DRAM, and an NVIDIA Tesla V100 GPU. We employ a 3-layer GNN backbone in this paper.

- *Graph Augmentation*: Default augmentation strength is set to 0.2. The optimization trade-off hyperparameter γ is 0.1.
- *Optimizer*: Model parameters (w) are updated using SGD with a learning rate of 0.025 (cosine decay over 200 epochs) and momentum of 0. Weight decay (ρ_1) is fixed at 0.00125.
- *Batch Size*: All pretraining methods use an in-memory batch size of 32.

Downstream Task Training and Finetuning Configuration. Downstream tasks utilize the SGD optimizer with cosine learning rate decay. All models train for 20 epochs. Task-specific settings are as follows:

- `Analog circuit classification`: Learning rate 0.01; weight decay 0.005; GNN: $hidden_{size} = 64$, $dropout = 0.6$, $ReLU$ activation.
- `Analog subcircuit detection`: Learning rate

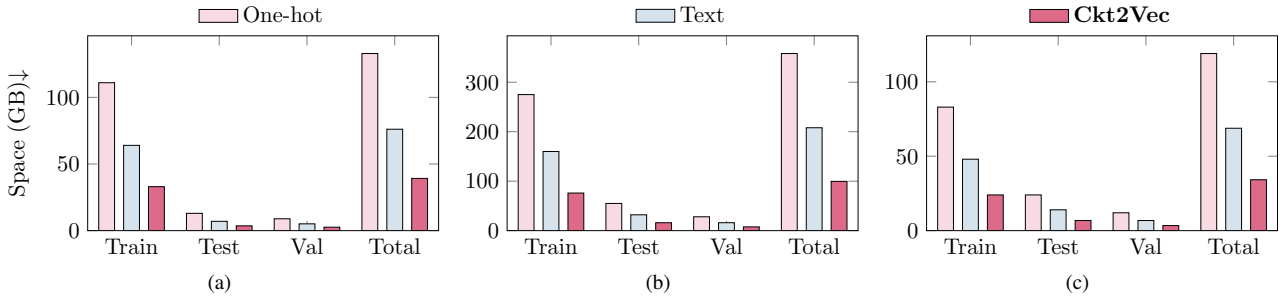


Fig. 8 The comparison of the space usage of encoding methods across different downstream tasks: (a) analog circuit classification; (b) analog subcircuit detection; (c) analog graph edit distance prediction. We measured the disk space taken up by datasets for different encoding methods. Ckt2Vec uses the space about 29% of the One-hot [7], [24] and 51% of the Text encoding [16] method. However, the key information of devices is not lost, which is shown by the better performance.

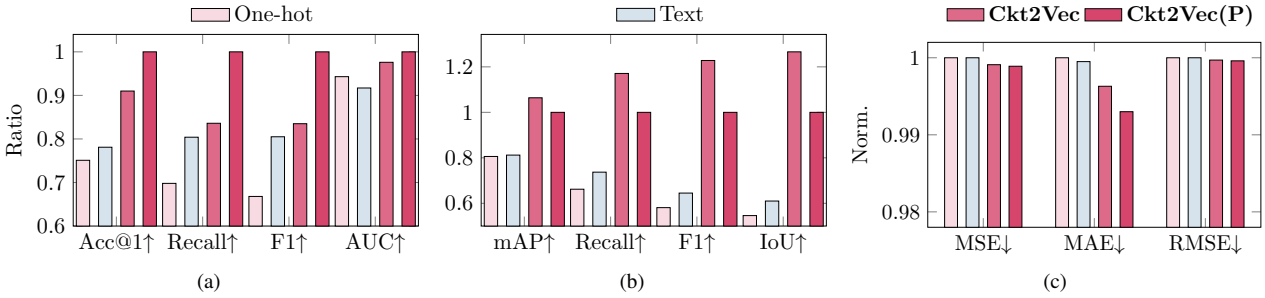


Fig. 9 The visualization of the averaged performance across downstream tasks and runtime breakdown: (a) classification; (b) subcircuit detection; (c) GED prediction. Here, **Ckt2Vec(P)** represents the pretrained variant. Ckt2Vec delivers an average performance boost across all three tasks, with classification and detection tasks seeing more significant improvements.

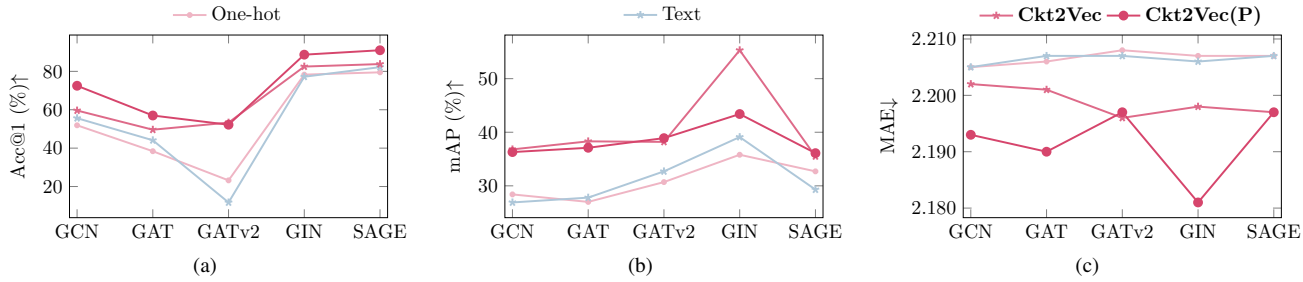


Fig. 10 Visualization of different graph neural networks' performance across downstream tasks: (a) classification; (b) subcircuit detection; (c) GED prediction. Our Ckt2Vec shows general improvements across different graph neural network structures.

0.03; weight decay 0.005; GNN: Identical architecture as circuit classification.

- `Graph edit distance prediction`: Learning rate 0.01; weight decay 0.0025; GNN: $hidden_{size} = 64$, $dropout = 0.6$, $ReLU$ activation.

Evaluation Metrics. To fully explore the effectiveness of our encoding method, we test different mainstream GNN encoders. For the mainstream GNN encoders, we adopted the following commonly used GNN frameworks as our analog circuits encoder models: GCN [39], GAT [27], GIN [25], Graphsage [26], GAT_v2 [40]. For the three downstream tasks, we set the training time for all methods at 20 epochs, 5 epochs, and 20 epochs, respectively. The pre-training is performed with 200 epochs and a batch size of 32. For the analog circuit classification task, we adopt standard classification metrics for evaluation: top-k accuracy (Acc@1, Acc@2, Acc@5), True Positive Rate (Recall), F1 Score, and Area Under the ROC Curve (AUC). For the analog subcircuit detection task, we use common evaluation metrics in detection tasks, including mAP (mean Average Precision), Recall, F1 Score, and IoU (Intersection over Union). For the graph edit distance prediction task, we use common evaluation metrics in regression tasks, including Mean Squared Error (MSE), Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE). The reported results are the averaged results in 4 runs from different random seeds.

TABLE IV Averaged performance comparison on subcircuits detection task (DET).

Models	mAP↑	Recall↑	F1 Score↑	AUC↑	IoU↑
DATE'20 & TCAD'23 (One-hot) [7], [24]					
GCN	0.284	0.180	0.123	0.692	0.077
GAT	0.270	0.097	0.059	0.641	0.035
GATv2	0.307	0.175	0.125	0.701	0.078
GIN	0.358	0.337	0.315	0.827	0.220
SAGE	0.327	0.215	0.169	0.739	0.105
Average	0.309	0.201	0.158	0.720	0.103
ICCAD'22 (Text) [16]					
GCN	0.269	0.178	0.117	0.685	0.073
GAT	0.278	0.171	0.115	0.688	0.071
SAGE	0.327	0.215	0.169	0.739	0.105
GIN	0.391	0.366	0.341	0.834	0.243
GATV2	0.293	0.187	0.136	0.696	0.084
Average	0.312	0.223	0.176	0.728	0.115
Ckt2Vec (ours)					
GCN	0.368	0.260	0.225	0.796	0.144
GAT	0.383	0.290	0.262	0.808	0.170
GATv2	0.382	0.303	0.279	0.807	0.183
GIN	0.553	0.667	0.695	0.932	0.565
SAGE	0.355	0.255	0.211	0.785	0.133
Average	**0.408**	**0.355**	**0.334**	**0.826**	**0.239**
Ckt2Vec (ours) with. *Pretrain*					
$\text{GraphCL}_{\text{GCN}}$	0.363	0.222	0.181	0.762	0.115
$\text{GraphCL}_{\text{GAT}}$	0.371	0.231	0.193	0.757	0.121
$\text{GraphCL}_{\text{GATv2}}$	0.389	0.271	0.236	0.779	0.154
$\text{GraphCL}_{\text{GIN}}$	0.434	0.539	0.539	0.898	0.420
$\text{GraphCL}_{\text{SAGE}}$	0.361	0.253	0.213	0.776	0.134
Average	0.384	0.303	0.272	0.794	0.189

TABLE V Averaged performance comparison on analog circuits graph edit distance prediction task (GED).

Models	MSE↓	MAE↓	RMSE↓
DATE'20 & TCAD'23 (One-hot) [7], [24]			
GCN	7.161	2.205	2.676
GAT	7.161	2.206	2.676
GATv2	7.164	2.208	2.677
GIN	7.164	2.207	2.677
SAGE	7.163	2.207	2.676
Average	7.163	2.207	2.676
ICCAD'22 (FastText) [16]			
GCN	7.161	2.205	2.676
GAT	7.163	2.207	2.676
GATv2	7.163	2.207	2.676
GIN	7.163	2.206	2.676
SAGE	7.164	2.207	2.677
Average	7.163	2.206	2.676
Ckt2Vec (ours)			
GCN	7.158	2.202	2.675
GAT	7.157	2.201	2.675
GATv2	7.157	2.196	2.675
GIN	7.155	2.198	2.675
SAGE	7.155	2.197	2.675
Average	7.156	2.199	2.675
Ckt2Vec (ours) with. *Pretrain*			
$\text{GraphCL}_{\text{GCN}}$	7.154	2.193	2.675
$\text{GraphCL}_{\text{GAT}}$	7.154	2.190	2.675
$\text{GraphCL}_{\text{GATv2}}$	7.155	2.197	2.675
$\text{GraphCL}_{\text{GIN}}$	7.159	2.181	2.676
$\text{GraphCL}_{\text{SAGE}}$	7.155	2.197	2.675
Average	**7.155**	**2.192**	**2.675**

B. Experimental Results

Analog Circuits Graph Classification. As shown in TABLE III, Fig. 8(a), Fig. 9(a), and Fig. 10(a), our proposed Ckt2Vec framework achieves the best results across all evaluation metrics with significant performance improvements over prior encoding methods and space usage reduction. Unlike baseline methods relying solely on one-hot/textual info, I/V curve embeddings capture dynamic circuit operational characteristics, enabling the model to distinguish info differences among circuit categories. From experimental data, compared to the one-hot and text-based encoding methods, our approach achieves up to a 15.9% increase in Acc@1 and a 16.5% increase in Recall with only 29% and 51% of the space usage, respectively. Applying graph contrastive learning in the pre-training phase further boosts the model's ability. It consistently demonstrates better generalization across most GNN architectures, resulting in an average 9% increase in Acc@1 and a 3% increase in Recall. This indicates that generating diverse augmented views of circuit graphs and maximizing their mutual information allows the model to learn a more robust and discriminative feature space.

Analog Subcircuits Detection. As shown in TABLE IV, Fig. 8(b), Fig. 9(b), and Fig. 10(b), our proposed Ckt2Vec framework also achieves superior performance across all evaluation metrics for the detection task.

When compared against established encoding baselines, namely one-hot and text-based approaches, the experimental results reveal that our method delivers substantial performance enhancements, with gains of up to 25.8% in mAP and a remarkable 50.9% increase in Recall metrics. While general pretraining offers certain gains in this task, we've observed noticeable degradation in some architectures, like GIN. This indicates that conventional graph pretraining paradigms, often developed for general graph domains, are suboptimal for capturing the nuanced physical relationships inherent in analog circuits. Consequently, these results highlight a clear opportunity for future work on developing specialized pretraining schemes explicitly tailored to analog circuit representation learning.

Analg Circuits Graph-Edit-Distance Prediction. Our proposed Ckt2Vec framework, implemented initially without pretraining, demonstrates a slight but consistent improvement over the baseline methods. Specifically, it reduces the Mean Squared Error (MSE) to 7.156 and the Mean Absolute Error (MAE) to 2.199, while maintaining a Root Mean Squared Error (RMSE) comparable to the baselines with less space usage. These results are detailed in TABLE V and visualized in Fig. 8(c), Fig. 9(c) and Fig. 10(c). After incorporating graph contrastive learning for pretraining, the enhanced Ckt2Vec models achieve the lowest average errors across all metrics: an MSE of 7.155, an MAE of 2.192, and an RMSE of 2.675.

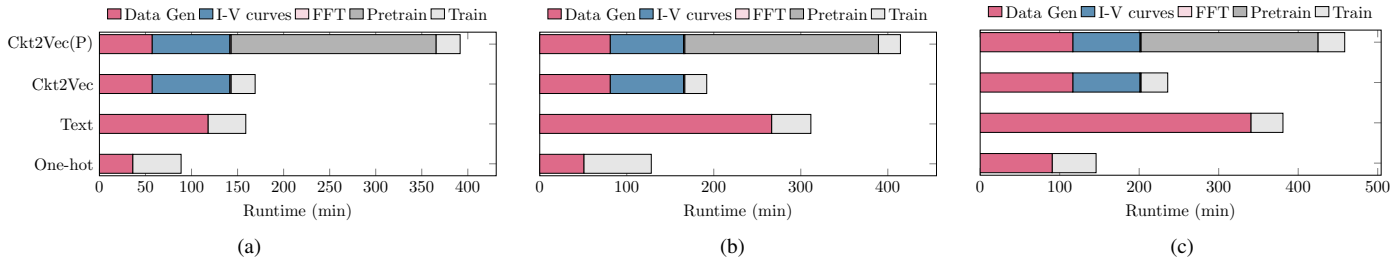


Fig. 11 The runtime comparison of different methods on: (a) CLS task; (b) DET task; (c) GED task.

Fig. 12 The visualization of detailed analysis on MAE comparison for GED prediction error across representative circuit categories.

TABLE VI Runtime breakdown (in minutes) for CLS, DET, and GED tasks across different data representations.

Methods	One-hot	Text	Ckt2Vec	Ckt2Vec(P)
CLS				
Data Gen	36.14	117.87	57.16	57.16
I-V curves	0.00	0.00	84.32	84.32
FFT	0.00	0.00	1.34	1.34
Pretrain	0.00	0.00	0.00	222.60
Train	52.54	41.05	26.25	26.25
Total	88.68	158.92	132.01	354.61
DET				
Data Gen	50.66	266.47	80.94	80.94
I-V curves	0.00	0.00	84.32	84.32
FFT	0.00	0.00	1.34	1.34
Pretrain	0.00	0.00	0.00	222.60
Train	77.54	45.11	25.31	25.31
Total	128.20	311.58	139.42	362.02
GED				
Data Gen	90.55	340.45	116.51	116.51
I-V curves	0.00	0.00	84.32	84.32
FFT	0.00	0.00	1.34	1.34
Pretrain	0.00	0.00	0.00	222.60
Train	55.27	40.39	33.73	33.73
Total	145.82	380.84	160.34	382.94

C. Runtime Analysis

To assess computational efficiency, we compare the runtime of different methods for circuit classification, subcircuit detection, and graph edit distance prediction. As shown in TABLE VI and Fig. 11, the overall runtime is divided into data generation, I–V curve extraction, FFT-based feature encoding, pretraining, and task-specific training. The results indicate that I–V curve extraction adds moderate overhead while FFT

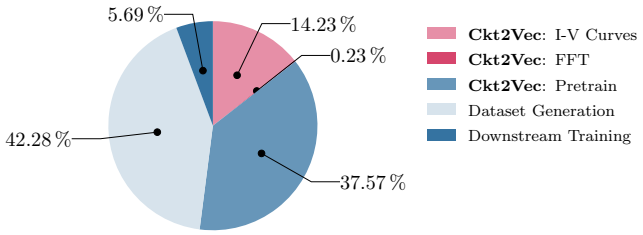


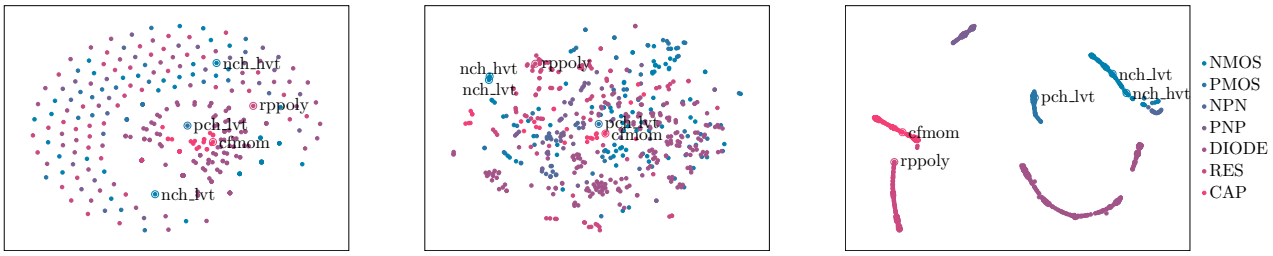
Fig. 13 The device-specific I-V curves generation of Ckt2Vec is efficient, accounting for only $1/3$ overhead of data generation.

TABLE VII The comparison of clustering evaluation metrics for different encoding methods.

Methods	SC↑	CH↑	DB↓
One-hot	0.013	1.990	9.251
Text	-0.108	3.107	6.105
Ckt2Vec	**0.457**	**318.345**	**1.185**

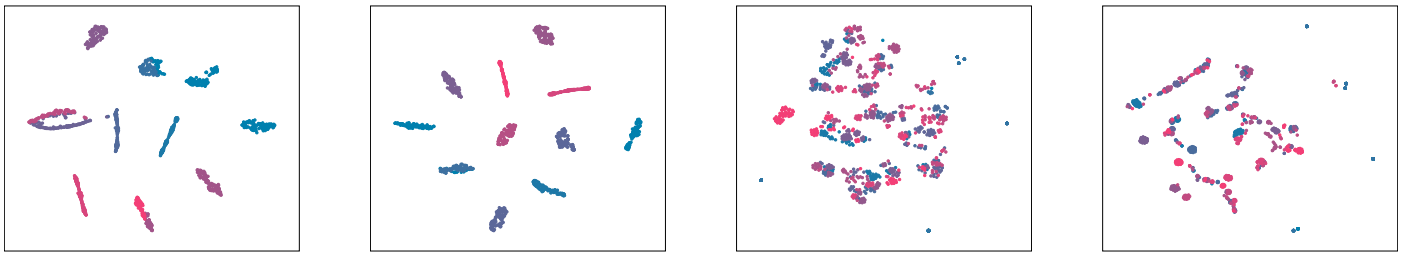
encoding is negligible. The main additional cost arises from the optional pretraining stage. Without pretraining, Ckt2Vec remains comparable to one-hot encoding and more efficient than text-based encoding, while providing improved accuracy and reduced storage requirements.

Additionally, we conducted a detailed runtime breakdown analysis for the entire encoding and training pipeline across the three tasks, as visualized in Fig. 13. Notably, the generation of device-specific current-voltage (I-V) curves proved highly efficient, contributing only one-third of the total additional overhead associated with the overall data generation process. This efficiency profile highlights the practicality of the



(a) (b) (c)

Fig. 14 The t-SNE visualization of device embeddings from different encoding methods: (a) One-hot encoding; (b) Text encoding; (c) Ckt2Vec (**Ours**). Labeled dots represent key device types, including nch_lvt (LDNMOS), nch_hvt (LVTNMOS), pch_lvt (LDPMOS), rppoly (resistor), and cfmom (capacitor). Compared with the scattered distributions produced by one-hot and text-based encodings, Ckt2Vec generates compact and physically meaningful clusters.



(a) (b) (c) (d)

Fig. 15 The t-SNE visualization of learned embeddings for different downstream tasks of **Ckt2Vec** with and without pre-training: (a) Analog circuit classification without pre-training; (b) Analog circuit classification with pre-training; (c) Subcircuit detection without pre-training; (d) Subcircuit detection with pre-training.

Ckt2Vec framework in graph learning for analog circuits.

D. Visualization and Discussion

Detailed Analysis and Discussion on GED task. Fig. 12 presents the normalized MAE comparison of GED prediction across circuit categories. While the dataset-level gains of Ckt2Vec over one-hot and text encodings appear modest, the per-circuit breakdown reveals consistent improvements across many categories. This indicates that GED, being predominantly a structural metric, is naturally less sensitive to device-level electrical fingerprints. However, Ckt2Vec still provides steady robustness by embedding device physics. Furthermore, the pretrained variant Ckt2Vec(P) achieves the lowest—or equal—error across categories. This demonstrates that incorporating electrical features does not compromise topology-based similarity prediction, while also yielding significant space savings. It is worth noting that GED prediction is inherently difficult, since it evaluates discrete structural edits (node/edge insertions, deletions, substitutions) rather than continuous device behaviors [41], [42]. As GED is NP-hard and largely insensitive to electrical similarity, the potential benefits of physics-informed features are necessarily bounded.

Visualization of Different Encoding Methods. As discussed earlier, conventional encoding schemes face limitations: one-hot encoding is space-intensive and cannot capture device similarity, while text-based methods often introduce estimation errors. To address these issues, we propose a spectral representation of I–V curves that encodes device electrical characteristics with higher accuracy and lower storage cost. We validate this approach experimentally by comparing embeddings generated with one-hot, fastText, and Ckt2Vec encodings. Embeddings from the same general device family naturally cluster by category, motivating the use of clustering-oriented metrics for evaluation. Specifically, clustering quality is assessed using the Silhouette Coefficient (SC), Calinski–Harabasz Index (CH), and Davies–Bouldin Index (DB). SC measures intra-cluster cohesion and inter-cluster separation (higher is better), CH evaluates cluster density and separation (higher is better), and DB reflects inter-cluster similarity (lower is better).

Results in Fig. 14 and TABLE VII show that Ckt2Vec embeddings consistently achieve superior cluster quality. FastText provides moderate improvements over one-hot, while Ckt2Vec further enhances both separability and interpretability by leveraging I–V curves as device fingerprints. Visualizations corroborate the metric-based analysis: one-hot produces weak discrimination, fastText offers partial separability but irregular spacing, whereas Ckt2Vec yields clear, well-structured clusters aligned with device categories.

Visualization of Pretraining Effect on Tasks. We further analyze the effect of pre-training on different downstream tasks. As shown in Fig. 15, pre-training leads to more compact and well-separated clusters for the classification (CLS) task (b), while embeddings without pre-training (a) are less

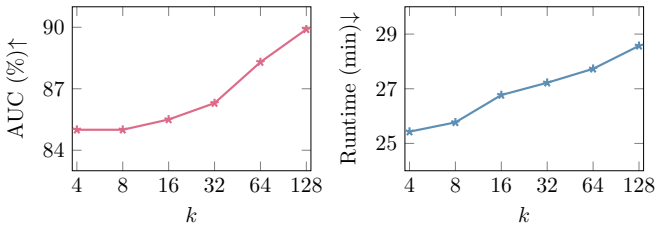


Fig. 16 The comparison of different hyperparameter k of FFT in our Ckt2Vec framework.

discriminative. However, for the device-type detection (DET) task, pre-training (d) does not consistently improve clustering compared to training from scratch (c). We attribute this to the fact that our current pre-training strategy emphasizes global circuit-level features, which benefits classification but lacks the granularity needed to capture local device-specific features required for detection. This suggests that extending pre-training to incorporate both global and local representations could further enhance performance across tasks.

E. Ablation Study

To investigate the effect of varying FFT parameters in our proposed Ckt2Vec encoding method, we conducted more experiments on the design space of the CLS task. To investigate the effect of different FFT parameters in our proposed Ckt2Vec encoding method, we conducted additional experiments on the design space of the CLS task. As shown in Fig. 16, we compared the classification performance (AUC) under different parameter settings from $k = 4$ to $k = 128$, where the parameter k controls the fidelity of the FFT to the original information of the IV curves. With the increase of k, the AUC results clearly improve, indicating that incorporating more frequency-domain components contributes to better classification performance. However, as k increases, the training time also grows linearly, and the storage cost is affected. Therefore, in our experiments, we set the threshold p to 128.

V. Conclusion

This paper introduces Ckt2Vec, a novel analog circuit representation learning framework that advances beyond traditional one-hot and text encoding. It extracts features from I-V curves, employs frequency-domain feature extraction for high-dimensional data, and uses graph contrastive learning for self-supervised pre-training. Ckt2Vec effectively captures device physical characteristics, thereby enhancing accuracy and reducing feature dimensionality. Experimental results show superior performance in multiple analog circuit representation learning tasks. While pretraining enhances feature quality, its benefits are degraded for some networks on subcircuit detection and GED prediction, suggesting the need for task-specific pretraining strategies. This work lays the foundation for incorporating electrical features into analog representation learning. Future research can integrate analog electrical characteristics into the representation learning.

References

[1] P. R. Gray, P. J. Hurst, S. H. Lewis, and R. G. Meyer, *Analysis and Design of Analog Integrated Circuits*. John Wiley & Sons, 2009.

[2] P. Xu, J. Li, T.-Y. Ho, B. Yu, and K. Zhu, "Performance-driven Analog Layout Automation: Current Status and Future Directions," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2024, pp. 679–685.

[3] W. L. Hamilton, "Graph Representation Learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 14, no. 3, pp. 1–159, 2020.

[4] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A Comprehensive Survey on Graph Neural Networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24, 2021.

[5] H. Chen, K. Zhu, M. Liu, X. Tang, N. Sun, and D. Z. Pan, "Universal Symmetry Constraint Extraction for Analog and Mixed-signal Circuits with Graph Neural Networks," in *ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 1243–1248.

[6] X. Gao, C. Deng, M. Liu, Z. Zhang, D. Z. Pan, and Y. Lin, "Layout Symmetry Annotation for Analog Circuits with Graph Neural Networks," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2021, pp. 152–157.

[7] K. Kunal, T. Dhar, M. Madhusudan, J. Poojary, A. Sharma, W. Xu, S. M. Burns, J. Hu, R. Harjani, and S. S. Sapatnekar, "GANA: Graph Convolutional Network Based Automated Netlist Annotation for Analog Circuits," in *IEEE/ACM Proceedings Design, Automation and Test in Eurpoe (DATE)*, 2020, pp. 55–60.

[8] K. Settaluri, A. Haj-Ali, Q. Huang, K. Hakhamaneshi, and B. Nikolic, "AutoCkt: Deep Reinforcement Learning of Analog Circuit Designs," in *IEEE/ACM Proceedings Design, Automation and Test in Eurpoe (DATE)*, 2020, pp. 490–495.

[9] H. Wang, K. Wang, J. Yang, L. Shen, N. Sun, H.-S. Lee, and S. Han, "GCN-RL Circuit Designer: Transferable Transistor Sizing with Graph Neural Networks and Reinforcement Learning," in *ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.

[10] Z. Dong, W. Cao, M. Zhang, D. Tao, Y. Chen, and X. Zhang, "CktGNN: Circuit Graph Neural Network for Electronic Design Automation," in *International Conference on Learning Representations (ICLR)*, 2023.

[11] Y. Hou, J. Zhang, H. Chen, M. Zhou, F. Yu, H. Fan, and Y. Yang, "CktGen: Specification-Conditioned Analog Circuit Generation," *arXiv preprint arXiv:2410.00995*, 2024.

[12] J. Tu, Y. Li, P. Li, P. Xu, Q. Zhang, S. Wan, Y. Sun, B. Yu, and T. Chen, "SMART: Graph Learning-Boosted Subcircuit Matching for Large-Scale Analog Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 44, no. 10, pp. 4018–4031, 2025.

[13] P. Xu, G. Chen, K. Zhu, T. Chen, T.-Y. Ho, and B. Yu, "Performance-driven Analog Routing via Heterogeneous 3DGNN and Potential Relaxation," in *ACM/IEEE Design Automation Conference (DAC)*, 2024, pp. 1–6.

[14] P. Xu, J. Tu, G. Chen, K. Zhu, T. Chen, T.-Y. Ho, and B. Yu, "PARoute2: Enhanced Analog Routing via Performance-Drive Guidance Generation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 44, no. 10, pp. 3654–3667, 2025.

[15] H. Ren, G. F. Kokai, W. J. Turner, and T.-S. Ku, "ParaGraph: Layout Parasitics and Device Parameter Prediction using Graph Neural Networks," in *ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.

[16] K. Zhu, H. Chen, W. J. Turner, G. F. Kokai, P.-H. Wei, D. Z. Pan, and H. Ren, "TAG: Learning Circuit Spatial Embedding From Layouts," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2022, pp. 1–9.

[17] B. Jónsson, "Relation Algebras and Schröder Categories," *Discrete Mathematics*, vol. 70, no. 1, pp. 27–45, 1988.

[18] M. Li, S. Khan, Z. Shi, N. Wang, H. Yu, and Q. Xu, "DeepGate: Learning Neural Representations of Logic Gates," in *ACM/IEEE Design Automation Conference (DAC)*, 2022, pp. 667–672.

[19] Z. Shi, H. Pan, S. Khan, M. Li, Y. Liu, J. Huang, H.-L. Zhen, M. Yuan, Z. Chu, and Q. Xu, "DeepGate2: Functionality-aware Circuit Representation Learning," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2023, pp. 1–9.

[20] Z. Shi, Z. Zheng, S. Khan, J. Zhong, M. Li, and Q. Xu, "DeepGate3: Towards Scalable Circuit Representation Learning," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2024, pp. 1–9.

[21] Z. Zheng, S. Huang, J. Zhong, Z. Shi, G. Dai, N. Xu, and Q. Xu, "DeepGate4: Efficient and Effective Representation Learning for Circuit Design at Scale," in *International Conference on Learning Representations (ICLR)*, 2025.
[22] Z. Wang, C. Bai, Z. He, G. Zhang, Q. Xu, T.-Y. Ho, Y. Huang, and B. Yu, "FGNN2: A Powerful Pretraining Framework for Learning the Logic Functionality of Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 44, no. 1, pp. 227–240, 2025.
[23] H. Wu, H. Zheng, Y. Pu, and B. Yu, "Circuit Representation Learning with Masked Gate Modeling and Verilog-AIG Alignment," *International Conference on Learning Representations (ICLR)*, 2025.
[24] K. Kunal, T. Dhar, M. Madhusudan, J. Poojary, A. K. Sharma, W. Xu, S. M. Burns, J. Hu, R. Harjani, and S. S. Sapatnekar, "GNN-based Hierarchical Annotation for Analog Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 42, no. 9, pp. 2801–2814, 2023.
[25] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How Powerful are Graph Neural Networks?" in *International Conference on Machine Learning (ICML)*, 2019.
[26] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," in *Annual Conference on Neural Information Processing Systems (NIPS)*, 2017.
[27] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," in *International Conference on Learning Representations (ICLR)*, 2018.
[28] H. Ren, S. Nath, Y. Zhang, H. Chen, and M. Liu, "Why are Graph Neural Networks Effective for EDA Problems?" in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2022, pp. 1–8.
[29] T. Chen, Q. Sun, C. Zhan, C. Liu, H. Yu, and B. Yu, "Analog IC Aging-induced Degradation Estimation via Heterogeneous Graph Convolutional Networks," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2021, pp. 898–903.
[30] Z. Wu, I. Song, and I. Savidis, "Hybrid Utilization of Subgraph Isomorphism and Relational Graph Convolutional Networks for Analog Functional Grouping Annotation," in *ACM/IEEE Workshop on Machine Learning CAD (MLCAD)*, 2023, pp. 1–6.
[31] H. Graeb and M. Leibl, "Learning from the Implicit Functional Hierarchy in an Analog Netlist," in *ACM International Symposium on Physical Design (ISPD)*, 2023, pp. 93–100.
[32] A.-J. Annema, "Analog Circuit Performance and Process Scaling," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 46, no. 6, pp. 711–725, 1999.
[33] P. Heckbert, "Fourier Transforms and the Fast Fourier Transform (FFT) Algorithm," *Computer Graphics*, vol. 2, no. 1995, pp. 15–463, 1995.
[34] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, "Graph Contrastive Learning with Augmentations," in *Annual Conference on Neural Information Processing Systems (NIPS)*, vol. 33, 2020, pp. 5812–5823.
[35] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A Simple Framework for Contrastive Learning of Visual Representations," in *International Conference on Machine Learning (ICML)*, 2020, pp. 1597–1607.
[36] K. Sohn, "Improved Deep Metric Learning with Multi-class N-pair Loss Objective," in *Annual Conference on Neural Information Processing Systems (NIPS)*, 2016.
[37] T. Chen, Y. Sun, Y. Shi, and L. Hong, "On Sampling Strategies for Neural Network-based Collaborative Filtering," in *ACM International Conference on Knowledge Discovery and Data Mining (KDD)*, 2017, pp. 767–776.
[38] Z. Zhao and L. Zhang, "Analog Integrated Circuit Topology Synthesis with Deep Reinforcement Learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 41, no. 12, pp. 5138–5151, 2022.
[39] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," in *International Conference on Learning Representations (ICLR)*, 2017.
[40] S. Brody, U. Alon, and E. Yahav, "How Attentive are Graph Attention Networks?" in *International Conference on Learning Representations (ICLR)*, 2022.
[41] R. Myers, R. Wison, and E. R. Hancock, "Bayesian Graph Edit Distance," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 22, no. 6, pp. 628–635, 2000.
[42] K. Riesen, "Structural Pattern Recognition with Graph Edit Distance," *Advances in computer vision and pattern recognition*, pp. 1–164, 2015.

Peng Xu is currently a Ph.D. student at the Department of Computer Science and Engineering, The Chinese University of Hong Kong (CUHK), under the supervision of Prof. Bei Yu from Fall 2022. Previously, he received his B.S. from Central South University and his M.S. from Harbin Institute of Technology (Shenzhen). His research interests include machine learning for analog physical design and optimization in EDA problems.

Yapeng Li is currently a Ph.D. student at the School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, from Fall 2023. Previously, he received his B.Eng. degree in microelectronics science and engineering from South China University of Technology in 2023. His research interests include machine learning and algorithm optimization for analog EDA.

Tinghuan Chen (M'21) received his B.Eng. and M.Eng. degrees in electronics engineering from Southeast University in 2014 and 2017, and Ph.D. degree in Computer Science and Engineering from The Chinese University of Hong Kong in 2021. He is currently an Assistant Professor in the School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen. His research interests include machine learning for EDA and deep learning accelerators.

Tsung-Yi Ho (F'24) received the Ph.D. degree in Electrical Engineering from National Taiwan University in 2005. He is currently a Professor in the Department of Computer Science and Engineering, at the Chinese University of Hong Kong (CUHK). He was a recipient of the Best Paper Award at IEEE TCAD in 2015. Currently, he serves as the VP Conference of IEEE CEDA, and the Executive Committee of ASP-DAC and ICCAD. He is a Distinguished Member of ACM and a Fellow of IEEE.

Bei Yu (M'15-SM'22) received the Ph.D. degree from The University of Texas at Austin in 2014. He is currently a Professor in the Department of Computer Science and Engineering, The Chinese University of Hong Kong. He has served as TPC Chair of ACM/IEEE Workshop on Machine Learning for CAD, and in many journal editorial boards and conference committees. He received eleven Best Paper Awards from ICCAD 2024 & 2021 & 2013, IEEE TSM 2022, DATE 2022, ASPDAC 2021 & 2012, ICTAI 2019, Integration, the VLSI Journal in 2018, ISPD 2017, SPIE Advanced Lithography Conference 2016, six ICCAD/ISPD contest awards, IEEE CEDA Ernest S. Kuh Early Career Award in 2021, DAC Under-40 Innovator Award in 2024, and Hong Kong RGC Research Fellowship Scheme (RFS) Award in 2024.