KCLNet: Electrically Equivalence-Oriented Graph Representation Learning for Analog Circuits

Peng Xu[*1], **Yapeng Li**[*2], **Tinghuan Chen**[2], **Tsung-Yi Ho**[1], **Bei Yu**[1]

[1] Department of Computer Science & Engineering, The Chinese University of Hong Kong
[2] School of Science and Engineering, The Chinese University of Hong Kong (Shenzhen)

Abstract

Digital circuit representation learning has made remarkable progress in the electronic design automation domain, effectively supporting critical tasks such as testability analysis and logic reasoning. However, representation learning for analog circuits remains challenging due to their continuous electrical characteristics compared to the discrete states of digital circuits. This paper presents a direct current (DC) electrically equivalent-oriented analog representation learning framework, named **KCLNet**. It comprises an asynchronous graph neural network structure with electrically-simulated message passing and a representation learning method inspired by Kirchhoff's Current Law (KCL). This method maintains the orderliness of the circuit embedding space by enforcing the equality of the sum of outgoing and incoming current embeddings at each depth, which significantly enhances the generalization ability of circuit embeddings. KCLNet offers a novel and effective solution for analog circuit representation learning with electrical constraints preserved. Experimental results demonstrate that our method achieves significant performance in a variety of downstream tasks, e.g., analog circuit classification, subcircuit detection, and circuit edit distance prediction. Our codes are available at https://github.com/shipxu123/KCLNet.

Introduction

Analog and mixed-signal electronic systems have become the backbone of modern technological advancement, driving innovations from medical instrumentation to autonomous vehicles. Their pervasive presence relies fundamentally on the seamless integration of analog and digital circuits. Within this electronic design paradigm, analog circuits emerge as the critical interface bridging the physical and digital worlds (Gray et al. 2009). As shown in Figure 1, analog circuit components such as operational amplifiers and data converters translate real-world signals—temperature variations, sound waves, or wireless transmissions—into digitally processable information while maintaining signal fidelity. Despite occupying less than 20% of modern system-on-chip (SoC) area, analog blocks determine over 40% of product reliability and power efficiency metrics (Sansen 2007). The design of analog circuits confronts challenges rooted in their sensitivity to physical characteristics. Unlike digital counterparts operating in well-defined Boolean domains, analog components are perpetually exposed to parasitic, process variation, and layout-dependent effects. Thus, analog circuit design, from schematic to layout, remains primarily a manual, time-consuming, and error-prone task (Xu et al. 2024b) at present. The inherent electrical complexities of analog circuits create heavy reliance on human expertise, forming a critical bottleneck in automation.

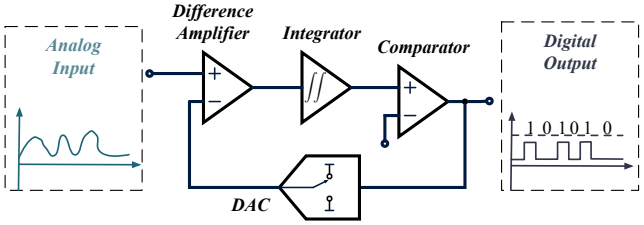

Figure 1: $\Delta\Sigma$ analog-to-digital converter (ADC), a typical analog circuit type. Unlike digital circuits, analog circuits process continuous signals, which poses challenges for representation learning. Effective analog representation learning requires careful attention to the electrical properties.

Emerging machine learning paradigms, particularly graph neural networks (GNNs) (Hamilton 2020; Wu et al. 2021), exhibit substantial potential in decoding the design complexity inherent to analog circuits. Naturally, the analog circuits can be viewed as graphs that consist of devices and nets, and their connections. By converting analog circuits into graphs, contemporary methodologies have demonstrated significant progress across critical analog design sub-tasks (Chen et al. 2021; Gao et al. 2021; Kunal et al. 2020; Settaluri et al. 2020; Wang et al. 2020; Dong et al. 2023; Hou et al. 2024; Tu et al. 2025; Xu et al. 2024a, 2025). For analog topology classification, graph-level embeddings are utilized to retrieve the targeted layout templates of analog circuits, such as amplifiers and filters in (Kunal et al. 2020). For subgraph identification, parallel advancements in layout synthesis leverage subgraph-aware GNN architectures to reduce symmetry violations compared to manual design as in (Chen et al. 2021; Gao et al. 2021). At the transistor level, sizing optimization frameworks employing node-centric GNNs reduce simulation iterations (Settaluri et al. 2020; Wang

[*]These authors contributed equally.

Copyright © 2026, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

et al. 2020; Dong et al. 2023; Hou et al. 2024). Additionally, analog circuit embeddings are also employed in various stages of analog circuit design, including large-scale subgraph matching during testing and performance optimization during routing (Tu et al. 2025; Xu et al. 2024a, 2025).

Despite widespread use of task-specific circuit embeddings, general analog circuit representation learning remains deficient in physical prior integration and lacks dedicated research as a standalone field. Some efforts attempt to develop multi-modality analog pretraining frameworks, where name-based text embeddings and auxiliary layout data are incorporated to incorporate domain knowledge (Ren et al. 2020; Zhu et al. 2022). While these multi-modality frameworks have demonstrated capability in the analog layout aspect, they have not demonstrated general representation capability at the analog circuit level. As the counterpart of analog circuits, digital circuit representation has demonstrated significant potential in leveraging physical laws, with Boolean algebra operations (Jónsson 1988) becoming a cornerstone of digital circuit representation learning (Li et al. 2022; Shi et al. 2023, 2024; Zheng et al. 2025; Wang et al. 2024; Wu et al. 2025). The intrinsic physical characteristics of analog devices are still neglected. Incorporating physical laws into analog circuit representation learning is crucial to match the success achieved in digital representation learning.

To address these limitations, we propose an electrical physics-inspired representation learning framework, named **KCLNet**, which incorporates fundamental electrical principles through *Kirchhoff's current law (KCL)*. Kirchhoff's Current Law states that the algebraic sum of all currents flowing into a node (junction) is equal to the algebraic sum of all currents flowing out (Paul 2001; Rewieński 2011; Athavale 2018). It is a fundamental principle in electrical circuit theory that forms the cornerstone of classical circuit analysis. The KCL is integrated through an electrically simulated message passing and a novel contrastive learning scheme. Our contributions are summarized as follows:

- To honor the electrical current flow, we convert analog circuits to directed acyclic graphs (DAGs) and propose an asynchronous message passing scheme with layer-wise propagation from voltage to ground nodes.
- Based on that, a physics-informed contrastive objective is designed where depth-wise embeddings enforce Kirchhoff's current conservation positives and node masking creates electrically inconsistent negatives. We theoretically justify that the proposed contrastive objective can preserve the electrical principle of KCL.
- The experimental results show that the analog circuit embeddings learned by our proposed KCLNet can benefit a variety of downstream tasks, with 20.77% improvement in Acc@1 gain in analog circuit classification, 43.36% mAP gain in analog subcircuit detection, and 1.6% MAE gain in analog circuit graph-edit-distance prediction.

Related Work and Preliminaries

Graph Neural Networks (GNNs)

GNNs consist of two major components, where the $\mathrm{aggregation}$ step aggregates node features of target nodes' neighbors, and the $\mathrm{combination}$ 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:

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

where $\mathcal{N}(v)$ denotes the neighbours of v.

Table 1: Typical Analog Device Types

Device	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-

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 (Kunal et al. 2020; Settaluri et al. 2020; Wang et al. 2020; Dong et al. 2023; Hou et al. 2024). We give the formal definition of an analog circuit graph as follows:

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

Definition 2 (Net) *Each net $v_n \in \mathcal{V}_n$ is a junction where multiple devices connect, with topological metrics decided by the connected pins of devices as shown in Table 1.*

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 $\mathcal{E}$, where different edge types correspond to different pin types, presenting connection information. Hence, the analog circuit graph has a form of bipartite graph as $\mathcal{G} = \{\mathcal{V}_d, \mathcal{V}_n, \mathcal{E}\}$.*

Method

Electrically-Simulated Async Message Passing

Directed Acyclic Circuit Graph Conversion. In traditional analog representation learning, existing methods frequently disregard the directions of the analog circuit graph (Chen et al. 2021; Kunal et al. 2020; Tu et al. 2025). This omission leads to neglecting current flow directionality during the learning process, resulting in a loss of crucial physical information. Although behavioral-level analog circuits are used to construct directed graphs in (Dong et al. 2023) and (Hou et al. 2024), this approach is specifically tailored for specific analog devices, e.g., operational transconductance amplifier

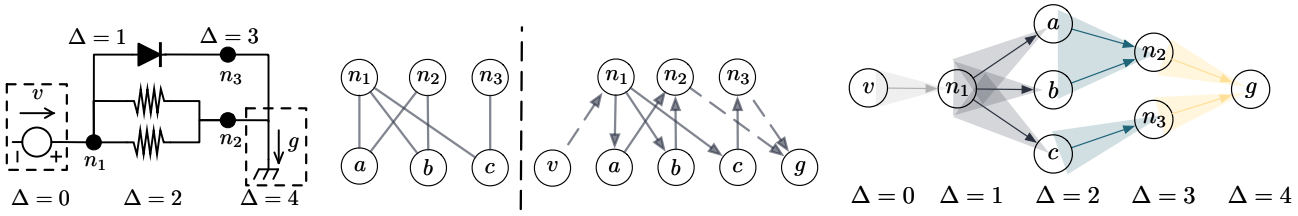

Figure 2: Directed acyclic circuit graph representation: (a) analog circuit; (b) convert bipartite graph representation (left) to DAG via topology sorting (right); (c) electrically-simulated asynchronous message passing scheme. To allow more flexibility in modeling, we define the direction of edges here but permit the current flow through the circuit to be zero, positive, or negative.

(OTA), rather than being general-purpose. Notably, in analog circuits, current typically originates from the power supply and traverses along the signal path to the ground, as noticed in (Gao et al. 2021). Consequently, to incorporate the physical characteristics of current flow into representation learning, this paper introduces a novel approach by designating the *voltage* and *ground* nodes as special nodes. The original analog circuit graph, a bipartite graph, is thus transformed into a directed acyclic graph (DAG) next.

We first discuss the conversion from undirected to directed graphs via topology sorting (Kahn 1962). The *voltage* nodes and the *ground* nodes are added as the start nodes and the end nodes. Based on the current flow direction in the circuit, assign directions to each edge in $\mathcal{E}$. Typically, current flows from voltage nodes to the ground nodes, so the edge directions should align with this current flow. Given that the analog circuit graph is a bipartite graph with edges only between devices and nets, we make the following theorem by topologically traversing from voltage nodes to ground nodes:

Theorem 1 (Alicyclic Guarantee after Conversion) *The original graph is a bipartite graph where edges only exist between devices and nets. Assume voltage and ground nodes are special devices added to the graph: 1. Voltage nodes have only outgoing edges with connected nets; 2. Ground nodes have only incoming edges with connected nets. The converted directed graph will be acyclic by traversing the bipartite graph via topology sorting, with the voltage and ground nodes as the start and end points.*

The proof of this theorem is provided in Appendix A.6. Following the theorem, the resulting graph becomes a DAG after conversion with voltage nodes as the start nodes and ground nodes as the terminal nodes, as shown in Figure 2a and Figure 2b. All paths in this DAG, pointing from voltage nodes to ground nodes, are consistent with the physical characteristics and signal flow of the circuit. This conversion process formalizes the structure of an analog circuit into a DAG, providing a foundation graph format for subsequent analog circuit representation learning.

Depthwise Message Passing Scheme. Conventional graph neural networks (GNNs) operate synchronously as illustrated in (Bruna et al. 2014; Defferrard, Bresson, and Vandergheynst 2016; Hamilton, Ying, and Leskovec 2017; Velickovic et al. 2018; Xu et al. 2019). In *synchronous message passing*, all messages flow simultaneously along edges during each iteration.

For better capturing current flow direction as in (Dimo 1975; Wedepohl and Jackson 2002), we propose an **asynchronous GNN (AGNN)** architecture that simulates depthwise message passing from voltage nodes to ground nodes. We take *voltage nodes* as the root nodes of the first layer through topological ordering while fixing *ground nodes* as the terminal layer nodes. The electrically-simulated message passing process initiates from the voltage source node at depth 0, propagates sequentially from depth 1 to d, and ultimately reaches the *ground nodes*. At each depth level, only the vertices that have received messages from the previous depth propagate messages to their direct successors. The Figure 2c demonstrates an example of embedding nodes using this asynchronous GNN.

Formally, for a target vertex v, the aggregation scheme of depth-asynchronous GNN at the $\boldsymbol{\Delta}$-th depth can be described as:

$$\begin{aligned} \boldsymbol{e}^{(l)}_{\{\boldsymbol{i}:\mathcal{D}(\boldsymbol{i},\boldsymbol{v})=\boldsymbol{\Delta}-\boldsymbol{l}\}} &= \mathrm{AGG}\left(\left\{\boldsymbol{h}_u^{(k-1)} : u \in \mathcal{N}(i)\right\}\right), \\ \boldsymbol{h}^{(l)}_{\{\boldsymbol{i}:\mathcal{D}(\boldsymbol{i},\boldsymbol{v})=\boldsymbol{\Delta}-\boldsymbol{l}\}} &= \mathrm{COMBINE}\left(\boldsymbol{e}_i^{(l)}, \boldsymbol{h}_i^{(\mathbf{0})}\right), \end{aligned} \tag{2}$$

where $\mathcal{D}(i, v)$ denotes the distance between vertices i and v in the graph, and $\boldsymbol{h}_i^{(0)}$ represents the initial feature vector of vertex i. We used the same AGG and COMBINE functions as GCN, with the sum of normalized neighbor embeddings in our implementation (Kipf and Welling 2017). Thus, our AGNN can be regarded as a GCN variant. Specifically, during the k-th iteration of a depth-Δ, aggregation occurs exclusively at vertices located at a distance of $\Delta - k$ from the root node v, which selectively integrates messages from their topological predecessors. This aggregated output is then fused with the vertex's original features to generate its updated representation vector.

Electrically Equivalent Contrastive Learning

Kirchhoff's Current Law Analysis. KCL is one of the most fundamental theorems in analog circuit analysis (Paul 2001; Rewieński 2011; Athavale 2018). It describes a crucial principle: the total current entering a node always equals the total current exiting it, as shown in Figure 3a. Formally, it can be defined as:

$$I_{in1} + I_{in2} + \cdots \Leftrightarrow I_{out1} + I_{out2} + \cdots . \tag{3}$$

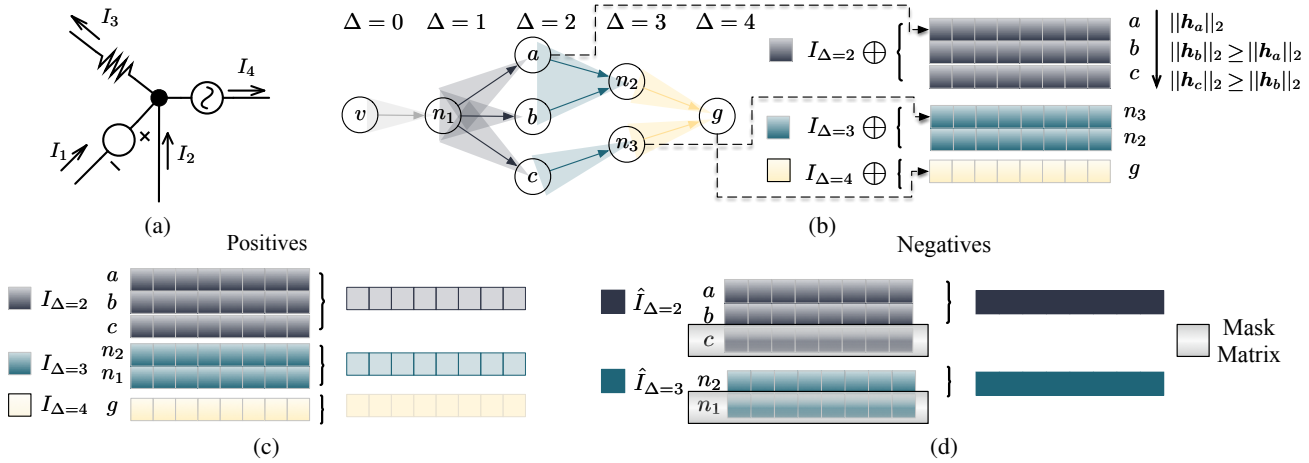

Figure 3: The framework of the physics-guided contrastive learning scheme, named the KCL Loss: (a) KCL states that the current entering any junction is equal to the current leaving that junction, e.g. $I_1 + I_2 = I_3 + I_4$; (b) Illustration of the AGNN encoding process. The summation of the aggregated features at each depth is treated as the current embedding $\boldsymbol{I}_{\boldsymbol{\Delta}}$ for depth $\boldsymbol{\Delta}$; (c) Positive embeddings generated by following the KCL; (d) Negative embeddings generated by breaking the KCL.

A key extension of the KCL is that the algebraic sum of input and output currents of multiple circuit components, i.e., a supernode, remains equal (Zeng and Zeng 2021; Zhu, Chen, and Yang 2024), which is frequently applied in the analysis of the current values at different depths. Our key idea is to preserve this equivalence in the analog circuit embedding space at different depths:

$$\begin{aligned} \boldsymbol{I}_{\boldsymbol{\Delta}} &= \boldsymbol{I}_{\boldsymbol{\Delta'}}, \\ \sum \boldsymbol{h}^{(l)}_{\{i:\mathcal{D}(i,v)=\boldsymbol{\Delta}-l\}} &= \sum \boldsymbol{h}^{(l)}_{\{j:\mathcal{D}(j,v)=\boldsymbol{\Delta'}-l\}}, \end{aligned} \tag{4}$$

where $\boldsymbol{\Delta}$ and $\boldsymbol{\Delta'}$ are different depths of current analog circuit graph. The electrical equivalence relation "$\Leftrightarrow$" is an equivalence relation under the constraint of Equation (4). For all device sets within one equivalent depth, the sum of the embeddings of all devices they consist of should be equal. This equation finds a natural equivalence that exists in universal analog circuits with the same current inputs and outputs.

Electrical Equivalence as Positives. For an analog circuit DAG graph $\mathcal{G}_i = \{\mathcal{V}_d^i, \mathcal{V}_n^i, \mathcal{E}^i\}$, we first use the GNN encoder to process all nodes from different depths in this graph and get their embeddings. As shown in Figure 3c, the subcircuit embedding pairs $(\boldsymbol{I}_{\boldsymbol{\Delta}}, \boldsymbol{I}_{\boldsymbol{\Delta'}})$ from different depths are treated as positive pairs, whose embedding discrepancy will be minimized. According to Equation (4), a straightforward loss function is therefore:

$$L = \sum_{\boldsymbol{\Delta'} \neq \boldsymbol{\Delta}}^{d} \text{sim}(\boldsymbol{I}_{\boldsymbol{\Delta}}, \boldsymbol{I}_{\boldsymbol{\Delta'}}),$$

where $\text{sim}(\boldsymbol{v}_1, \boldsymbol{v}_2)$ is the cosine similarity function $\frac{\boldsymbol{v}_1^\top \boldsymbol{v}_2}{\|\boldsymbol{v}_1\| \cdot \|\boldsymbol{v}_2\|}$ as in (He et al. 2020), and d is the maximum depth.

This simple constraint is critical for enhancing the quality of analog circuit embeddings. We show that by enforcing such constraints, the trained neural network can thus preserve the electrical characteristic of KCL with the following theorem:

Theorem 2 (Kirchhoff's Current Law preservation) *Let $\{\boldsymbol{I}_{\Delta_1}, \boldsymbol{I}_{\Delta_2}, \ldots, \boldsymbol{I}_{\Delta_n}\}$ and $\{\boldsymbol{I}_{\Delta'_1}, \boldsymbol{I}_{\Delta'_2}, \ldots, \boldsymbol{I}_{\Delta'_n}\}$ be two sets of vectors in $\mathbb{R}^d$, where $n \leq d$ and $0 \leq \Delta_i, \Delta'_i < d$, and d is the longest distance of the the analog circuit graph. Then, there exists a non-trivial linear map $\phi : \mathbb{R}^d \to \mathbb{R}$ such that $\phi(\boldsymbol{I}_{\Delta_i}) = \phi(\boldsymbol{I}_{\Delta'_i})$ for all $i = 1, 2, \ldots, n$.*

The proof of this theorem is provided in Appendix A.7. The theorem shows that for the current embeddings of different layers, there always exists a nonlinear mapping function that determines the input and output current values, with the algebraic sum satisfying KCL. Based on that, we also have the following corollary , with the proof also attached in Appendix A.7:

Corollary 1 *Suppose that for each i, the distance between the corresponding vectors after normalization is sufficiently small, i.e., $1 - \hat{\boldsymbol{I}}_{\Delta_i}^\top \hat{\boldsymbol{I}}_{\Delta'_i} \leq \epsilon$. A smaller ϵ makes constructing the desired map $\phi(\cdot)$ easier.*

Directly minimizing the loss function is ineffective, as the model would become degenerate by producing all-zero vectors for every subcircuit embedding (Chen et al. 2020; Pang et al. 2022). Typical approaches to address this involve using negative sampling or contrastive learning techniques. In our work, we introduce a novel technique for generating negative samples by leveraging the electrical contradiction of KCL.

Electrical Contradiction as Hard Negatives. We introduce a novel technique for generating negative samples $\hat{\boldsymbol{I}}_{\boldsymbol{\Delta}}$ by selectively applying dropout to node embeddings at each depth $\boldsymbol{\Delta}$. Because the sum of incoming currents at each layer is equal, artificially creating an imbalance by discarding node

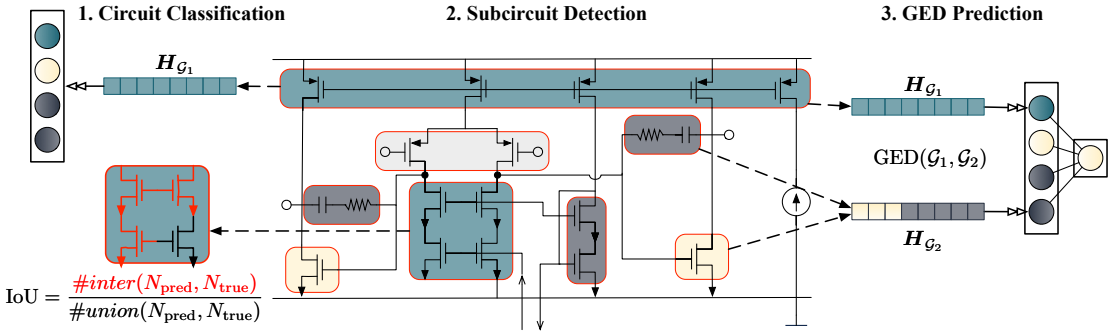

Figure 4: The illustration of the downstream tasks: (1) `Analog circuit classification`: evaluates the ability of models to classify analog circuits into predefined categories; (2) `Analog subcircuit detection`: tests the precision of identifying and labeling fundamental subcircuits within larger analog circuits; (3) `Analog GED preidction`: assess whether learned embeddings capture the structural similarity between circuits with different functions.

embeddings with higher current values naturally generates negative samples from the circuit's perspective.

As shown in Figure 3d, compute the squared L_2 norm of its embedding $\boldsymbol{h}_{\{i:\mathcal{D}(i,v)}$ as a measure of its magnitude and define a binary mask to identify top-k largest nodes:

$$\begin{aligned} \boldsymbol{r}_i^{\boldsymbol{\Delta}} &= \left\| \boldsymbol{h}^{(l)}_{\{i:\mathcal{D}(i,v)=\boldsymbol{\Delta}-l\}} \right\|_2, \\ \boldsymbol{M}^{\boldsymbol{\Delta}}[i] &= \begin{cases} 1 & \text{if } r_i^{\boldsymbol{\Delta}} \in \text{top-}k \text{ of } \{r_1^{\boldsymbol{\Delta}}, \ldots, r_n^{\boldsymbol{\Delta}}\} \\ 0 & \text{otherwise} \end{cases}. \end{aligned} \tag{5}$$

Apply the dropout by element-wise multiplication of the original embeddings with the inverted mask:

$$\hat{\boldsymbol{I}}_{\boldsymbol{\Delta}} = \sum \left(\boldsymbol{h}_{\{i:\mathcal{D}(i,v)\}} \odot \left(\mathbf{1} - \boldsymbol{M}^{\boldsymbol{\Delta}} \right) \right), \tag{6}$$

where $\odot$ denotes element-wise multiplication, and $\mathbf{1}$ is a matrix of ones. This zeros out embeddings for the top-k nodes, creating hard negatives.

KCL Loss. Combine the positives and negatives inspired by KCL, we follow the contrastive framework in (You et al. 2020; Zhu et al. 2020) to derive the KCL Loss, the training objective for $(\boldsymbol{I}_{\boldsymbol{\Delta}}, \boldsymbol{I}_{\boldsymbol{\Delta}'})$ with $N = d \times (d-1)$ pairs is:

$$\mathcal{L} = -\log \frac{e^{\mathrm{sim}(\boldsymbol{I}_{\boldsymbol{\Delta}}, \boldsymbol{I}_{\boldsymbol{\Delta}'})/\tau}}{\sum_{j=1}^{N} e^{\mathrm{sim}\left(\boldsymbol{I}_{\boldsymbol{\Delta}}, \hat{\boldsymbol{I}}_{\boldsymbol{\Delta}'}\right)/\tau}},$$

where τ is a temperature hyperparameter.

Experiments

We conduct experiments to address the following issues: **(1)** *How does KCLNet compare to general GNNs and graph pre-trained methods in terms of performance?* **(2)** *How effective is the proposed KCL Loss?* **(3)** *What is the impact of each KCLNet module on its overall performance?*

Experimental Setup

Dataset and Downstream Tasks. We employ the analog circuits dataset `ANALOG` in (Tu et al. 2025) generated using the analog circuit topology synthesis framework (Zhao and Zhang 2022). The generated circuits contain fundamental topologies ranging from basic building blocks to complex industrial-scale systems. The details of the used analog circuit topologies are in Appendix A.2.

In this paper, we introduce three downstream applications as shown in Figure 4 to verify our pre-trained circuit embedding. Three datasets for sub-tasks were developed for analog circuit-related tasks. For analog circuit classification, the `ANALOG` dataset was expanded by adjusting sizing parameters, resulting in `ANALOG-CLS-428k`; for analog subcircuit detection, following Kunal et al.'s (Kunal et al. 2020) framework, base circuit categories were classified via manual annotation, creating `ANALOG-DET-242k` with 242,320 samples; for analog graph edit distance (GED) prediction, `ANALOG-GED-194k` was generated via mutations on existing data. All datasets emphasize balanced splits and proportional representation of circuit characteristics to ensure robust model evaluation. The details of the definition of subtasks and the dataset partition scheme are provided in Appendix A.2.

Evaluation Metrics. 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), and F1 Score. 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). We utilize common evaluation metrics in regression tasks for the graph edit distance prediction task, including MAE and MSE. The reported results are the average of the best results in 5 runs from different random seeds.

Baseline Methods. The baseline methods we compare include two categories: (1) mainstream GNN encoders without pretraining; (2) mainstream GNN representation learning method combined with different GNN Encoders. For the mainstream GNN Encoders, we adopted the following commonly used GNN frameworks as our analog circuits encoder: GCN (Kipf and Welling 2017), GAT (Velickovic et al. 2018), GIN (Xu et al. 2019), Graphsage (Hamilton, Ying, and Leskovec 2017), GAT_v2 (Brody, Alon, and Yahav 2022). For mainstream GNN pretraining methods, we utilized the general graph pre-training method named

Table 2: Performance comparison on analog circuit classification. Top performers in each category are **bold**.

Method	Acc@1↑	Acc@2↑	Acc@5↑	Recall↑	F1 Score↑
Base Models					
GCN	$0.561_{\pm 0.061}$	$0.784_{\pm 0.040}$	$0.913_{\pm 0.014}$	$0.405_{\pm 0.087}$	$0.404_{\pm 0.121}$
GAT	$0.479_{\pm 0.105}$	$0.681_{\pm 0.148}$	$0.794_{\pm 0.118}$	$0.327_{\pm 0.060}$	$0.275_{\pm 0.069}$
GATv2	$0.498_{\pm 0.100}$	$0.694_{\pm 0.139}$	$0.799_{\pm 0.133}$	$0.322_{\pm 0.051}$	$0.270_{\pm 0.075}$
GIN	$0.786_{\pm 0.077}$	$0.883_{\pm 0.058}$	$0.931_{\pm 0.039}$	$0.648_{\pm 0.101}$	$0.669_{\pm 0.118}$
SAGE	$0.774_{\pm 0.019}$	$0.927_{\pm 0.023}$	$0.938_{\pm 0.004}$	$0.626_{\pm 0.028}$	$0.638_{\pm 0.055}$
GraphCL Variants					
$\text{GraphCL}_{\text{GCN}}$	$0.679_{\pm 0.131}$	$0.795_{\pm 0.120}$	$0.910_{\pm 0.063}$	$0.558_{\pm 0.150}$	$0.564_{\pm 0.186}$
$\text{GraphCL}_{\text{GAT}}$	$0.537_{\pm 0.108}$	$0.794_{\pm 0.046}$	$0.919_{\pm 0.046}$	$0.408_{\pm 0.135}$	$0.368_{\pm 0.173}$
$\text{GraphCL}_{\text{GATv2}}$	$0.493_{\pm 0.112}$	$0.650_{\pm 0.075}$	$0.809_{\pm 0.112}$	$0.401_{\pm 0.104}$	$0.357_{\pm 0.148}$
$\text{GraphCL}_{\text{GIN}}$	$0.875_{\pm 0.083}$	$0.943_{\pm 0.009}$	$0.950_{\pm 0.006}$	$0.741_{\pm 0.083}$	$0.759_{\pm 0.102}$
$\text{GraphCL}_{\text{SAGE}}$	$0.899_{\pm 0.013}$	$0.958_{\pm 0.001}$	$0.962_{\pm 0.001}$	$0.726_{\pm 0.024}$	$0.744_{\pm 0.036}$
Our Methods					
KCLNet	$\mathbf{0.949}_{\pm 0.004}$	$\mathbf{0.958}_{\pm 0.005}$	$\mathbf{0.964}_{\pm 0.004}$	$\mathbf{0.829}_{\pm 0.015}$	$\mathbf{0.853}_{\pm 0.011}$
$w.o$ Pos^{KCL}	$0.938_{\pm 0.005}$	$0.942_{\pm 0.006}$	$0.944_{\pm 0.003}$	$0.794_{\pm 0.013}$	$0.830_{\pm 0.012}$
$w.o$ Neg^{KCL}	$0.939_{\pm 0.004}$	$0.946_{\pm 0.010}$	$0.949_{\pm 0.010}$	$0.801_{\pm 0.015}$	$0.682_{\pm 0.335}$
$w.o$ $\text{Pos+Neg}^{\text{KCL}}$	$0.938_{\pm 0.003}$	$0.941_{\pm 0.005}$	$0.944_{\pm 0.002}$	$0.794_{\pm 0.009}$	$0.822_{\pm 0.018}$

Table 3: Performance comparison on subcircuit detection task.

Method	mAP↑	Recall↑	F1 Score↑	AUC↑	IoU↑
Base Models					
GCN	$0.368_{\pm 0.002}$	$0.260_{\pm 0.004}$	$0.225_{\pm 0.003}$	$0.796_{\pm 0.002}$	$0.144_{\pm 0.002}$
GAT	$0.383_{\pm 0.013}$	$0.290_{\pm 0.033}$	$0.262_{\pm 0.038}$	$0.808_{\pm 0.012}$	$0.170_{\pm 0.032}$
GATv2	$0.382_{\pm 0.017}$	$0.303_{\pm 0.059}$	$0.279_{\pm 0.072}$	$0.807_{\pm 0.022}$	$0.183_{\pm 0.051}$
GIN	$0.553_{\pm 0.029}$	$0.667_{\pm 0.019}$	$0.695_{\pm 0.020}$	$0.932_{\pm 0.005}$	$0.565_{\pm 0.021}$
SAGE	$0.355_{\pm 0.004}$	$0.255_{\pm 0.002}$	$0.211_{\pm 0.002}$	$0.785_{\pm 0.001}$	$0.133_{\pm 0.001}$
GraphCL Variants					
$\text{GraphCL}_{\text{GCN}}$	$0.363_{\pm 0.014}$	$0.222_{\pm 0.005}$	$0.181_{\pm 0.008}$	$0.762_{\pm 0.008}$	$0.115_{\pm 0.005}$
$\text{GraphCL}_{\text{GAT}}$	$0.371_{\pm 0.016}$	$0.231_{\pm 0.027}$	$0.193_{\pm 0.032}$	$0.757_{\pm 0.032}$	$0.121_{\pm 0.023}$
$\text{GraphCL}_{\text{GATv2}}$	$0.389_{\pm 0.010}$	$0.271_{\pm 0.096}$	$0.236_{\pm 0.105}$	$0.779_{\pm 0.052}$	$0.154_{\pm 0.077}$
$\text{GraphCL}_{\text{GIN}}$	$0.434_{\pm 0.033}$	$0.539_{\pm 0.076}$	$0.539_{\pm 0.090}$	$0.898_{\pm 0.017}$	$0.420_{\pm 0.070}$
$\text{GraphCL}_{\text{SAGE}}$	$0.361_{\pm 0.016}$	$0.253_{\pm 0.015}$	$0.213_{\pm 0.013}$	$0.776_{\pm 0.014}$	$0.134_{\pm 0.009}$
Our Methods					
KCLNet	$\mathbf{0.622}_{\pm \mathbf{0.002}}$	$\mathbf{0.721}_{\pm \mathbf{0.002}}$	$\mathbf{0.753}_{\pm \mathbf{0.002}}$	$\mathbf{0.949}_{\pm \mathbf{0.000}}$	$\mathbf{0.634}_{\pm \mathbf{0.003}}$
$w.o$ Pos^{KCL}	$0.374_{\pm 0.027}$	$0.399_{\pm 0.098}$	$0.381_{\pm 0.117}$	$0.845_{\pm 0.042}$	$0.273_{\pm 0.096}$
$w.o$ Neg^{KCL}	$0.602_{\pm 0.039}$	$0.706_{\pm 0.040}$	$0.736_{\pm 0.046}$	$0.944_{\pm 0.010}$	$0.615_{\pm 0.053}$
$w.o$ $\text{Pos+Neg}^{\text{KCL}}$	$0.561_{\pm 0.035}$	$0.654_{\pm 0.038}$	$0.674_{\pm 0.045}$	$0.931_{\pm 0.010}$	$0.545_{\pm 0.051}$

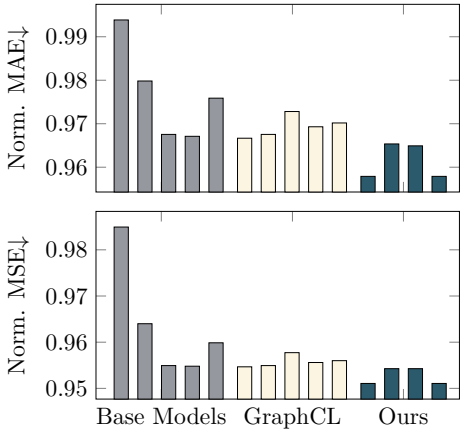

Figure 5: The averaged and normalized performance comparison on the analog circuit GED prediction task.

GraphCL (You et al. 2020) , combined with different graph encoders as comparison methods. All methods are compared based on the same hyperparameter. The baseline methods' implementation details are provided in Appendix A.4.

Experimental Results

Analog Circuits Classification. As shown in Table 2, we comprehensively compared the experimental results of various methods for the analog circuit classification task. Specifically, KCLNet achieves the highest accuracy among all baseline models, showcasing its superior representation ability for downstream classification. Compared to the pre-trained graph model $\text{GraphCL}_{\text{GCN}}$, our method brings about a 39.76% and 51.19% improvement in Acc@1 and F1 Score, respectively. When compared to the best performing methods, $\text{GraphCL}_{\text{GIN}}$ and GIN, it offers a 5.56% and 20.77% improvement in Acc@1 and F1 Score, respectively. These results confirm that KCLNet has better representation compared with existing approaches in the analog circuit classification task.

Analog Subcircuits Detection. As shown in Table 3, KCLNet also outperforms other methods in the subcircuit detection task. Compared to the best performing methods, $\text{GraphCL}_{\text{GIN}}$ and GIN, KCLNet achieves improvements of

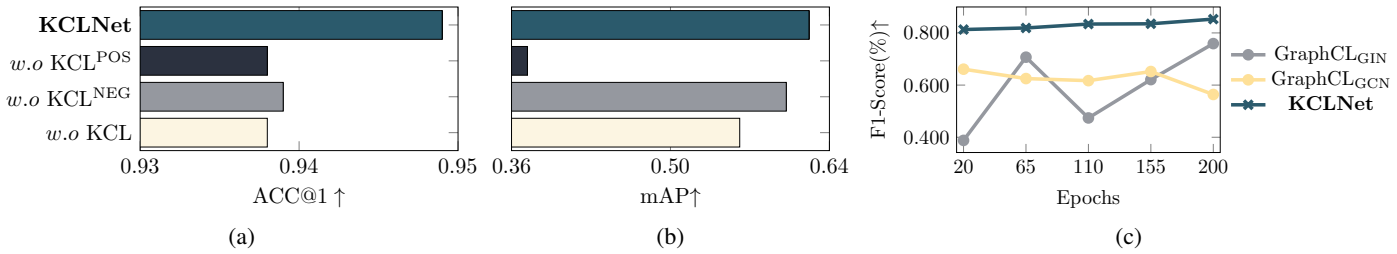

Figure 6: (a) and (b): Comparison between KCLNet with different variants of KCL Loss. The positives and negatives generated by KCL Loss show effectiveness in downstream tasks, compared with the general graph augmentation and negatives generated from different samples in the same batch; (c) The pre-training epochs impact of KCLNet on classification, which outperforms general graph pretraining at earlier epochs.

Table 4: Performance comparison between synchronous and asynchronous message passing (AGNN). AGNN demonstrates a significant performance improvement over other synchronous GNN models.

	Classification		Detection	
	Acc@1↑	F1-Score↑	mAP↑	IoU↑
GCN	$0.561_{\pm 0.061}$	$0.404_{\pm 0.121}$	$0.368_{\pm 0.002}$	$0.144_{\pm 0.002}$
GAT	$0.496_{\pm 0.127}$	$0.306_{\pm 0.101}$	$0.383_{\pm 0.013}$	$0.170_{\pm 0.032}$
GATv2	$0.498_{\pm 0.100}$	$0.270_{\pm 0.075}$	$0.382_{\pm 0.017}$	$0.183_{\pm 0.051}$
GIN	$0.786_{\pm 0.077}$	$0.669_{\pm 0.118}$	$0.553_{\pm 0.029}$	$\mathbf{0.565_{\pm 0.021}}$
SAGE	$0.774_{\pm 0.019}$	$0.638_{\pm 0.055}$	$0.355_{\pm 0.004}$	$0.133_{\pm 0.001}$
AGNN	$\mathbf{0.938_{\pm 0.003}}$	$\mathbf{0.822_{\pm 0.018}}$	$\mathbf{0.561_{\pm 0.035}}$	$0.545_{\pm 0.051}$

up to 43.36%, 33.80%, 39.61%, 5.67%, and 51.02% in the mAP, Recall, F1 Score, AUC, and IoU metrics, respectively. It can be observed that the general-purpose graph pretraining algorithm $\text{GraphCL}_{\text{GIN}}$ fails to enhance the model's performance on downstream circuit-related tasks compared to GIN. The key issue is that these generic pre-training tasks often fail to capture equivalence in circuit diagrams. These improvements in metrics further indicate that KCLNet's representation learning capabilities enhance the detection accuracy, recall, and robustness.

Graph-Edit-Distance Prediction. As shown in Figure 5, the proposed contrastive learning framework reduces up to 3.8% MAE compared to vanilla GNNs. Additionally, compared with methods that have experienced improvement through general pretraining, such as $\text{GraphCL}_{\text{GIN}}$, KCLNet maintains a similar improvement with up to 1.6%. It can also be observed that general graph pretraining methods do not necessarily guarantee an improvement in the performance of GCN, which can be seen from the close MAE and MSE values.

Ablation Studies

KCL-Inspired Loss at Work. A notable advantage of KCLNet lies in its ability to generate positive embeddings in the analog circuits from an electrical perspective. To verify this, Figure 6a and Figure 6b show performance comparisons of different KCL Loss variants. The *w.o* KCL^{POS} uses general graph-augmented positive samples to maximize alignment, *w.o* KCL^{NEG} replaces KCL-based negatives with different graph samples in the same batch, and *w.o KCL* removes KCL Loss entirely, relying solely on asynchronous message passing. The results show that KCL Loss is crucial for final performance, with positive embeddings being the most important.

Asynchronous Message Passing Scheme. In Table 4, we compare the performance differences between the proposed AGNN and synchronous GNN. AGNN outperforms the best synchronous model, GIN, by 19.6% and 22.9% in Acc@1 and F1-score in the cls task. It also improves the mAP by 1.4% in the det task, surpassing all other GNN models. These results confirm the effectiveness of electrically-simulated asynchronous message passing.

The Effect of Training Epochs in KCLNet. As shown in Figure 6c, KCLNet surpasses GraphCL pretrained methods for different epochs ($\text{GraphCL}_{\text{GIN}}$ and $\text{GraphCL}_{\text{GCN}}$) starting from 20 epochs, showing strong early performance. Unlike other frameworks that rely on augmentation for positive samples with more pretraining epochs needed, KCLNet uses KCL's physical prior knowledge to generate positive embeddings without augmentation, reducing the needed pretraining epochs.

Conclusion

In this work, we use an electrically-simulated asynchronous graph neural network as the analog circuit encoder, leveraging Kirchhoff's current law to aid representation learning by enforcing current embedding conservation across depths. Experiments show the model learns vital physical priors, significantly enhancing generalization across analog sub-tasks. We propose three future directions: 1) modeling Kirchhoff's voltage law from a voltage perspective; 2) exploring better encoders like graph transformers; 3) incorporating additional inputs like SPICE codes to assist learning.

Acknowledgments

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), and The Research Grants Council of Hong Kong SAR (No. CUHK14211824 and No. CUHK14201624).

References

Athavale, P. 2018. Kirchoff's current law and Kirchoff's voltage law. *Johns Hopkins University, Retrived*, 6.

Brody, S.; Alon, U.; and Yahav, E. 2022. How Attentive are Graph Attention Networks? In *Proc. ICLR*.

Bruna, J.; Zaremba, W.; Szlam, A.; and LeCun, Y. 2014. Spectral Networks and Locally Connected Networks on Graphs. In *Proc. ICLR*.

Chen, H.; Zhu, K.; Liu, M.; Tang, X.; Sun, N.; and Pan, D. Z. 2021. Universal symmetry constraint extraction for analog and mixed-signal circuits with graph neural networks. In *Proc. DAC*, 1243–1248. IEEE.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A simple framework for contrastive learning of visual representations. In *Proc. ICML*, 1597–1607.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *Proc. NIPS*.

Dimo, P. 1975. *Nodal analysis of power systems*. International Scholarly Book Services, Inc., Forest Grove, OR.

Dong, Z.; Cao, W.; Zhang, M.; Tao, D.; Chen, Y.; and Zhang, X. 2023. CktGNN: Circuit Graph Neural Network for Electronic Design Automation. In *Proc. ICLR*.

Gao, X.; Deng, C.; Liu, M.; Zhang, Z.; Pan, D. Z.; and Lin, Y. 2021. Layout symmetry annotation for analog circuits with graph neural networks. In *Proc. ASPDAC*, 152–157.

Gray, P. R.; Hurst, P. J.; Lewis, S. H.; and Meyer, R. G. 2009. *Analysis and design of analog integrated circuits*. John Wiley & Sons.

Hamilton, W. L. 2020. Graph Representation Learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(3): 1–159.

Hamilton, W. L.; Ying, Z.; and Leskovec, J. 2017. Inductive Representation Learning on Large Graphs. In *Proc. NIPS*.

He, K.; Fan, H.; Wu, Y.; Xie, S.; and Girshick, R. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 9729–9738.

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

Jónsson, B. 1988. Relation algebras and Schröder categories. *Discrete Mathematics*, 70(1): 27–45.

Kahn, A. B. 1962. Topological sorting of large networks. *Communications of the ACM*, 5(11): 558–562.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proc. ICLR*.

Kunal, K.; Dhar, T.; Madhusudan, M.; Poojary, J.; Sharma, A.; Xu, W.; Burns, S. M.; Hu, J.; Harjani, R.; and Sapatnekar, S. S. 2020. GANA: Graph convolutional network based automated netlist annotation for analog circuits. In *Proc. DATE*, 55–60. IEEE.

Li, M.; Khan, S.; Shi, Z.; Wang, N.; Yu, H.; and Xu, Q. 2022. Deepgate: Learning neural representations of logic gates. In *Proc. DAC*, 667–672.

Pang, B.; Zhang, Y.; Li, Y.; Cai, J.; and Lu, C. 2022. Unsupervised visual representation learning by synchronous momentum grouping. In *Proc. ECCV*, 265–282. Springer.

Paul, C. R. 2001. *Fundamentals of electric circuit analysis*. John Wiley & Sons.

Ren, H.; Kokai, G. F.; Turner, W. J.; and Ku, T.-S. 2020. ParaGraph: Layout parasitics and device parameter prediction using graph neural networks. In *Proc. DAC*, 1–6. IEEE.

Rewieński, M. 2011. A perspective on fast-SPICE simulation technology. In *Simulation and Verification of Electronic and Biological Systems*, 23–42. Springer.

Sansen, W. M. 2007. *Analog design essentials*, volume 859. Springer Science & Business Media.

Settaluri, K.; Haj-Ali, A.; Huang, Q.; Hakhamaneshi, K.; and Nikolic, B. 2020. Autockt: Deep reinforcement learning of analog circuit designs. In *Proc. DATE*, 490–495. IEEE.

Shi, Z.; Pan, H.; Khan, S.; Li, M.; Liu, Y.; Huang, J.; Zhen, H.-L.; Yuan, M.; Chu, Z.; and Xu, Q. 2023. Deepgate2: Functionality-aware circuit representation learning. In *Proc. ICCAD*, 1–9. IEEE.

Shi, Z.; Zheng, Z.; Khan, S.; Zhong, J.; Li, M.; and Xu, Q. 2024. Deepgate3: Towards scalable circuit representation learning. *arXiv preprint arXiv:2407.11095*.

Tu, J.; Li, Y.; Li, P.; Xu, P.; Zhang, Q.; Wan, S.; Sun, Y.; Yu, B.; and Chen, T. 2025. SMART: Graph Learning-Boosted Subcircuit Matching for Large-Scale Analog Circuits. *IEEE TCAD*.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *Proc. ICLR*.

Wang, H.; Wang, K.; Yang, J.; Shen, L.; Sun, N.; Lee, H.-S.; and Han, S. 2020. GCN-RL circuit designer: Transferable transistor sizing with graph neural networks and reinforcement learning. In *Proc. DAC*, 1–6. IEEE.

Wang, Z.; Bai, C.; He, Z.; Zhang, G.; Xu, Q.; Ho, T.-Y.; Huang, Y.; and Yu, B. 2024. Fgnn2: A powerful pre-training framework for learning the logic functionality of circuits. *IEEE TCAD*.

Wedepohl, L.; and Jackson, L. 2002. Modified nodal analysis: an essential addition to electrical circuit theory and analysis. *Engineering science and education journal*, 11(3): 84–92.

Wu, H.; Zheng, H.; Pu, Y.; and Yu, B. 2025. Circuit Representation Learning with Masked Gate Modeling and Verilog-AIG Alignment. *Proc. ICLR*.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Yu, P. S. 2021. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1): 4–24.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019. How Powerful are Graph Neural Networks? In *Proc. ICML*.

Xu, P.; Chen, G.; Zhu, K.; Chen, T.; Ho, T.-Y.; and Yu, B. 2024a. Performance-driven analog routing via heterogeneous 3dgnn and potential relaxation. In *Proc. DAC*, 1–6.

Xu, P.; Li, J.; Ho, T.-Y.; Yu, B.; and Zhu, K. 2024b. Performance-driven analog layout automation: Current status and future directions. In *Proc. ASPDAC*, 679–685.

Xu, P.; Tu, J.; Chen, G.; Zhu, K.; Chen, T.; Ho, T.-Y.; and Yu, B. 2025. PARoute2: Enhanced Analog Routing via Performance-Drive Guidance Generation. *IEEE TCAD*.

You, Y.; Chen, T.; Sui, Y.; Chen, T.; Wang, Z.; and Shen, Y. 2020. Graph contrastive learning with augmentations. *Proc. NIPS*, 33: 5812–5823.

Zeng, G. L.; and Zeng, M. 2021. Kirchhoff's Current Law (KCL). *Electric Circuits: A Concise, Conceptual Tutorial*, 31–35.

Zhao, Z.; and Zhang, L. 2022. Analog integrated circuit topology synthesis with deep reinforcement learning. *IEEE TCAD*, 41(12): 5138–5151.

Zheng, Z.; Huang, S.; Zhong, J.; Shi, Z.; Dai, G.; Xu, N.; and Xu, Q. 2025. DeepGate4: Efficient and Effective Representation Learning for Circuit Design at Scale. In *Proc. ICLR*.

Zhu, K.; Chen, H.; Turner, W. J.; Kokai, G. F.; Wei, P.-H.; Pan, D. Z.; and Ren, H. 2022. Tag: Learning circuit spatial embedding from layouts. In *Proc. ICCAD*, 1–9.

Zhu, Y.; Xu, Y.; Yu, F.; Liu, Q.; Wu, S.; and Wang, L. 2020. Deep Graph Contrastive Representation Learning. In *ICML Workshop on Graph Representation Learning and Beyond*.

Zhu, Z.; Chen, Z.; and Yang, S. 2024. A Fast SIE Solver With Cut Set Analysis and Terminals as Supernodes for Interconnects. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 43(9): 2730–2740.