IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 29 December 2024, accepted 16 March 2025, date of publication 24 March 2025, date of current version 8 April 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3553743

SURVEY

Generative AI for Analog Integrated Circuit Design: Methodologies and Applications

DANIAL NOORI ZADEH, (Student Member, IEEE),
AND MOHAMED B. ELAMIEN, (Member, IEEE)
Department of Electrical and Computer Engineering, McMaster University, Hamilton, ON L8S 4L8, Canada

Corresponding author: Mohamed B. Elamien (elamienm@mcmaster.ca)

This work was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada through its Discovery Grant (DG) Program under Grant RGPIN-2024-06826.

ABSTRACT Electronic Design Automation (EDA) in analog Integrated Circuits (ICs) has been the focus of extensive research; however, unlike its digital counterpart, it has not achieved widespread adoption. In this systematic review, we discuss recent contributions in the last five years, highlighting methods that address data scarcity, topology exploration, process-voltage-temperature (PVT) variations, and layout parasitics. Our goal is to support researchers new to this domain by creating a comprehensive collection of references and practical application guidelines. We provide a methodological review of state-of-the-art machine learning (ML) approaches, including graph neural networks (GNNs), large language models (LLMs), and variational autoencoders (VAEs), which have been successfully applied to analog circuit sizing tasks. To the best of authors' knowledge, this is the first review to comprehensively explore the application of generative AI models in analog IC circuit design. We conclude that future research could focus on few-shot learning with domain-adaptation training of generative AI methods to simplify the design tasks such as human-tool interaction or guided design space exploration.

INDEX TERMS Analog integrated circuits (ICs), electronic design automation (EDA), layout-aware sizing, machine learning (ML), large language models (LLM), graph neural networks (GNN), variational autoencoders (VAE), artificial intelligence (AI), machine learning (ML).

I. INTRODUCTION

The increasing complexity and diverse performance requirements of modern analog systems create a high-dimensional design space. In response, full-flow automation has become necessary to handle the intricate trade-offs between numerous performance parameters, as traditional approaches are time-consuming and heavily reliant on expert knowledge [1]. While digital design automation has seen extensive development and adoption across both industry [2] and academia [3], [4], significant challenges remain to automate analog IC design. Researchers have attempted to automate the process at different points in the design flow [5], [6] [7]. Figure 1 shows the three main areas of automation on circuit level, namely topology selection, circuit sizing, and layout generation, in addition to the feedback loops in the design process shown by the dashed lines. Although layout generation tools such as MAGICAL [8] and ALIGN [9] have shown significant advances, scalable circuit sizing solutions remain elusive. A practical automation flow at each stage should account for dependencies across design stages; for example, the selection of a topology should consider the potential layout parasitics and its effect on performance metrics [10].

Analog circuit sizing is often formulated as an optimization problem and is typically addressed using equation-based, simulation-based, or learning-based methods. Equation-based methods rely on closed-form mathematical expressions to describe circuit performance metrics as a function of design parameters [11]. Although these equations provide invaluable intuition for designers, their application in large systems with significant parasitic elements and short-channel effects often breaks down, leading to inaccurate results. In contrast, simulation-based methods use precise but computationally intensive SPICE simulations with optimization

The associate editor coordinating the review of this manuscript and approving it for publication was Jagadheswaran Rajendran.

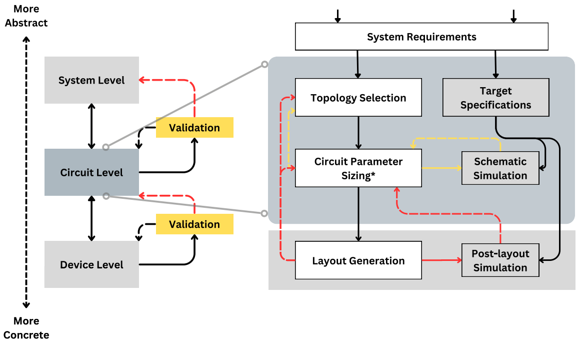

FIGURE 1. Analog design automation flow, focusing on circuit-level automation. Dashed lines indicate design dependence.

engines to iteratively explore optimal designs. Stochastic methods, such as Bayesian Optimization (BO) [12] and Evolutionary Algorithm (EA) [13], introduce randomness to overcome local optima and enhance exploration within the design space. BO is a powerful tool for optimizing expensive-to-evaluate closed-box functions, balancing exploration and exploitation through its probabilistic models. However, stochastic methods can be computationally intensive in high-dimensional design spaces.

While learning-based methods depend on simulation data for training, they enable more efficient design space exploration [14]. Moreover, ML techniques can be applied individually or in combination to support decision-making, function approximation, and closed-box optimization. ML, as a subset of AI, has emerged as a promising solution to many of these challenges, with breakthroughs across fields such as finance [15], chemistry [16], and digital design [3]. Traditional ML models, such as random forests [17], [18] and support vector machines (SVMs) [19], have been applied to circuit design because of their ability to explore and approximate nonlinear functions from limited data. These models excel in handling structured data and offer insight into the relationships between design parameters and performance metrics. However, they are limited by their need for manual feature extraction and their lack of scalability in complex, high-dimensional circuit design problems. Studies such as [20] highlight the advantages of GNNs over traditional models, including multilayer perceptrons, SVMs, and Random Forests, for handling graph-structured data, making GNNs a powerful tool for analog circuit design. Though often overlooked due to its abstract nature, unsupervised learning holds significant potential when properly applied. In the unsupervised learning paradigm, the ML model learns the underlying structure or distribution of data without requiring external labels. This enables the model to perform tasks such as dimensionality reduction, anomaly detection, data imputation, and generative modeling. VAEs, for instance, learn to transform input data (graphs, images, or text) into a lower-dimensional latent space. In this space, the distribution captures the underlying semantic or relational structure between data points [21]. This approach has demonstrated promising results, particularly when combined with GNNs for designing Op-Amps from specification to layout [22], [23], [24]. Furthermore, LLMs have recently received attention for their applications in layout automation [25], [26], [27] and topology generation [28]. Their use extends to optimization tasks [29], [30] and decision-making in circuit design, as demonstrated in [31], where an LLM was used to design ring oscillators and multi-stage amplifiers. Traditionally used for natural language processing, LLMs have proven adaptable to large-scale design problems and cross-domain tasks via techniques such as supervised fine-tuning, zero- and few-shot learning, and retrieval-augmented generation (RAG). However, most of the research in the analog sizing domain has been limited to simple circuits with a small number of parameters.

The main contributions of this paper are as follows:

- Examine and compare recent advancements in circuit sizing with a focus on techniques that consider layout parasitic, topology exploration, and scalability challenges.
- Provide a methodological review of state-of-the-art ML techniques applied in analog circuit sizing automation, including graph neural networks, large language models, and variational autoencoders.
- We collect abundant resources, open-source codes, and application guidelines for researchers new to the field of analog circuit automation.

The remainder of this paper is structured as follows: section II summarizes and compares previous review papers in terms of their automation scope and the ML techniques. section III introduces fundamental IC design challenges and outlines how these challenges shape the automation task. section IV provides the fundamentals of ML relevant to recent research. section V compares significant research works, focusing on their methods and key problems they attempt to solve. Finally, section VI outlines future research directions and challenges for large-scale adoption in the industry.

II. RELATED WORKS

Numerous surveys, reviews, and detailed literature sources have contributed to advancing automatic analog IC sizing, as evidenced by the exponential increase in relevant publications. Figure 2 illustrates the distribution of journal, conference, and review papers on ML-driven analog IC sizing that we have examined in our review.

Rosa et al. provide a comprehensive taxonomy of ML techniques and their potential applications in analog automation, but do not cover more recent developments in the field [6]. Similarly, Budak et al. explore more recent trends in ML methodologies; however, only Chapter 12 of their book discusses analog sizing. Budak et al. categorize prior research into four main approaches: Bayesian optimization (BO), evolutionary algorithms (EA), reinforcement learning (RL), and parasitic-aware sizing [7]. Fayazi et al. review fundamental ML training techniques and conventional models, including deep neural networks (DNNs), convolutional neural networks

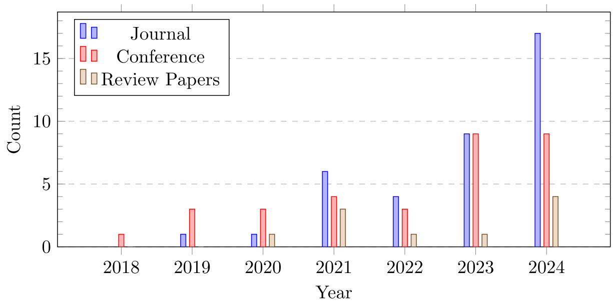

FIGURE 2. Overview of the publications focused on analog IC sizing used in this review paper.

(CNNs), and BO, for circuit parameter sizing. They introduce block-level performance modeling to reduce training data requirements [32]. In their automation framework, AnGeL, Fayazi et al. propose a semi-supervised design flow that leverages DNNs to train both block-level and mapping models with a limited number of labeled samples [33]. However, their methods overlook post-layout performance and require expert intervention for feature engineering during pre-processing. Moreover, applying these techniques remains challenging for more complex circuit blocks.

Chen et al. offer an extensive survey of ML techniques applied in IC design, covering shallow models, CNNs, GNNs, and generative models like VAEs and generative adversarial networks [34]. However, their scope spans both analog and digital design, diluting its focus on analog-specific challenges. Nguyen et al. provide a more application-oriented review, focusing on automation for emerging 6G systems. They evaluate Bayesian-based, meta-heuristic, and RL-based approaches, with a case study on meta-heuristic algorithms for designing a band-gap reference circuit [35]. Maji et al. survey past research on full-flow automation, emphasizing the importance of including layout simulations in the loop and modeling parasitics from the early stages [36]. Their work explores RL agents to improve training efficiency and suggests decoupling topology selection from sizing within a predefined library of topologies. Budak et al. also review parasitic-aware sizing techniques, stressing the performance mismatch between schematic-level and post-layout simulations, especially in analog and mixed-signal designs [37]. They present a case study on a Miller OTA to demonstrate how layout-agnostic methods, while performing well in schematic simulations, experience significant performance degradation post-layout. Tool-generated designs optimized at the schematic level failed to meet key metrics like unity gain bandwidth post-layout. Conversely, an expert-designed circuit is more area-efficient, incorporates dummy devices, and is more resilient to parasitic effects. This case underscores the need for joint optimization across schematic and layout phases to mitigate misleading results.

GNNs have gained significant attention in recent years; for example, [38] focuses on the application of GNNs in digital IC design, while [39] categorizes previous research based on model type and graph data applications. Reviews [40] and [41] provide comprehensive taxonomies, application guidelines, and open-source models for GNN-based EDA. A design pipeline for automation with GNNs is defined in [42], outlining various graph representations, and task definitions. Lastly, [43] and [44] briefly explore reinforcement learning and artificial neural networks for topology selection and transistor sizing. This paper aims to bridge gaps between core concepts while addressing the complexities of analog IC sizing. We present a methodological review on applications of GNNs, LLMs, and VAEs to provide high-level application guidelines, and provide a reference for implementation guidelines.

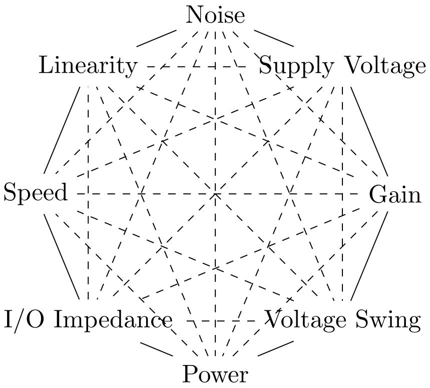

FIGURE 3. Analog design octagon from [1].

III. THE DESIGN PROBLEM

Developing an automation flow requires an understanding of the challenges associated with analog design and automation methodologies. subsection III-A summarizes key challenges that an automation flow needs to address, while subsection III-B introduces various problem formulations for circuit parameter sizing.

A. CHALLENGES IN ANALOG IC DESIGN

1) TIME-CONSUMING SIMULATIONS

Schematic-level simulations provide an initial performance estimate, but post-layout simulations, which account for parasitics and other physical effects, are computationally expensive and time-consuming.

2) INHERENT TRADE-OFFS BETWEEN PERFORMANCE METRICS

A core challenge in analog circuit design is the high interdependence of performance metrics, as depicted in Figure 3. For example, improving noise performance often requires larger device sizes, which can increase parasitic effects and reduce speed. Similarly, enhancing amplifier gain typically involves raising the load impedance or bias current, which can increase power consumption and limit bandwidth. Designers must carefully balance these trade-offs and prioritize metrics based on the target application [1]. The study in [45] offers an interactive tool to further explore these trade-offs.

3) CONTEXT-DEPENDENT FIGURES OF MERIT

Analog IC design is highly application-specific, with different use cases requiring distinct performance metrics. For instance, minimizing power consumption is critical in portable, low-power devices, often at the expense of speed or noise. Conversely, RF circuits in communication systems often prioritize speed and linearity over power efficiency. Designers rely heavily on expert intuition to navigate these trade-offs, using their experience to determine acceptable compromises for a given application.

4) PREDICTING AND MANAGING PARASITIC EFFECTS

As the design progresses from schematic to layout, parasitics, such as capacitance and resistance introduced by interconnects, become significant factors that can degrade performance. Device parasitics represent an additional source of performance degradation. These parasitics, which are not captured in schematic simulations, can limit bandwidth, reduce gain, and increase power consumption [46]. For instance, in high-speed designs, parasitics can severely impair signal integrity, gain, and system's frequency response, reducing overall system performance. Accurate prediction of parasitics is challenging, making layout-level simulations essential to ensure the design meets its performance targets post-layout.

5) TOPOLOGY-AWARE DESIGN

Every application has many possible configurations of circuit building blocks, and the goal is to find the optimal achievable performance; so, a practical automation flow embeds the choice of topology into the design space.

6) ENSURING ROBUSTNESS ACROSS PVT VARIATIONS

PVT corner analysis is critical, as variations in manufacturing processes, supply voltages, and operating temperatures can lead to shifts in performance. For example, a design that performs well under nominal conditions may experience reduced gain, increased power consumption, or even fail to meet specifications at extreme temperature or voltage conditions. Testing at multiple PVT corners helps identify worst-case scenarios, ensuring the design remains functional across a wide range of conditions.

B. FORMULATION OF AUTOMATION TASK

Given the extensive number of design parameters and associated performance trade-offs, circuit sizing should be approached systematically and efficiently to identify viable solutions. The task of circuit sizing can be formulated in the following ways:

1) CONSTRAINED OPTIMIZATION

Circuit sizing can be framed as a single or multi-objective constrained optimization problem, with single-objective formulations being more prevalent in practice.

$$\begin{aligned} &\text{minimize} \quad FoM(x) \\ &\text{s.t.} \quad f_i(x) \leq c_i, \ i = 1, 2, \ldots, m \\ &\qquad x \in \mathbb{R}^n, FoM(\cdot) \in \mathbb{R} \end{aligned} \tag{1}$$

where x is the parameter vector, n represents the number of design variables (e.g., widths, lengths, and bias conditions of a given topology), $f_i(x)$ and c_i denote the i^{th} performance metric and constraint. m is the total number of performance metrics. Typically, a primary performance metric, such as power consumption, is optimized as a single objective, while other metrics like gain, bandwidth, or noise figure serve as constraints. However, multiple metrics can be optimized simultaneously through the figure of merit (FoM), defined using preference vector $\mathbf{w} \in \mathbb{R}^m$:

$$\text{FoM}(\mathbf{x}) = w_1 f_1(\mathbf{x}) + \sum_{i=2}^{z} \min(1, \max(0, w_i f_i(\mathbf{x}))) \tag{2}$$

where, w_i represents the weight assigned to performance metric $f_i(x)$. The min($\cdot$) function prevents single constraint violations from disproportionately affecting the FoM value, while the max($\cdot$) function acts as a clipping mechanism for constraints where $i > 1$.

2) CONSTRAINED SATISFACTION

Krylov et al. suggest that it is often more practical to establish a threshold for performance metrics [47]. Consequently, finding a solution that meets these thresholds is typically sufficient. Hakhamaneshi et al. [10] define a non-negative cost function through the weighted sum of normalized spec errors, where a zero shows a feasible solution. If no solution is feasible, the design with the smallest cost is the closest to satisfying the design requirement.

$$\text{cost}(x) = \sum_i w_i \frac{|c_i - c_i^*|}{c_i + c_i^*} \tag{3}$$

where, c_i denotes the value of constraint i at input x, evaluated using a simulation framework, and c_i^* denotes the optimal value. The weights w_i are tuning factors set by the designer to prioritize certain metrics over others when the design is infeasible. One way to approach sizing under PVT is to impose a limit for each metric at the corners of interest.

3) Bi-LEVEL OPTIMIZATION

Given the high dimensionality of the design space, it is beneficial to decompose the task into multiple optimization problems. A bi-level optimization can be formulated as follows:

$$\begin{aligned} &\min_{x \in X} F(x, y^*(x)) \\ &\text{subject to} \quad h(x) \leq 0, \quad g(x) = 0, \end{aligned} \tag{4}$$

where $x \in \mathbb{R}^m$ represents the decision variables of the upper-level problem, and $y^*(x)$ is the optimal solution to the lower-level problem, defined as:

$$\begin{aligned} &y^*(x) = \arg\min_{y \in Y(x)} G(x, y) \\ &\qquad \text{subject to} \quad h'(x, y) \leq 0, \ g'(x, y) = 0, \end{aligned} \tag{5}$$

where $y \in \mathbb{R}^n$ indicates the decision variables of the lower-level problem. Functions $F : \mathbb{R}^m \times \mathbb{R}^n \rightarrow \mathbb{R}$ and $G : \mathbb{R}^m \times \mathbb{R}^n \rightarrow \mathbb{R}$ serve as the objective functions for the upper and lower levels, respectively, while X and $Y(x)$ describe the feasible regions. For instance, Lu et al. designed a compensation scheme for an op-amp utilizing a bi-level optimization task [48]. In this case, the upper-level task emphasizes topology optimization at the behavioral level, whereas the lower-level task focuses on device sizing at the transistor level. The upper-level optimization establishes the topology vector x, and once set, the lower-level optimization seeks the optimal design vector y through a separate optimization process. An extension of bi-level optimization is presented in [49], where topological information is used to partition analog circuits into sub-blocks, reducing the complexity of the design search space. Each sub-block is then assigned to an agent, and the interactions between agents mimic the optimal design trade-offs made by human experts.

4) MULTI-FIDELITY OPTIMIZATION

This framework leverages models of varying accuracy to optimize a given objective function while balancing computational cost and precision to solve problems in Equation 1. It involves combining high-fidelity (accurate but expensive) and low-fidelity (less accurate but cheaper) models to approximate an optimal solution efficiently [50]. BO incorporates uncertainty estimates in both fidelities to decide when to use the high-fidelity model. Thus, the optimization can be framed as:

$$\min_{x \in X} \mathbb{E}[f_H(x) \mid \mathcal{D}_L, \mathcal{D}_H] \tag{6}$$

where $\mathbb{E}[f_H(x) \mid \mathcal{D}_L, \mathcal{D}_H]$ denotes the expected value of the high-fidelity objective function, conditioned on data from both low-fidelity ($\mathcal{D}_L$) and high-fidelity ($\mathcal{D}_H$) model evaluations. Here, $f_H : \mathbb{R}^m \rightarrow \mathbb{R}$ represents the high-fidelity objective function, and $X \subset \mathbb{R}^m$ is the feasible design space, while the low-fidelity model $f_L(x)$ approximates $f_H(x)$. In analog design, a wide range of simulations, including DC, transient, noise, and offset analysis, can be performed at both the schematic and post-layout stages. Each type of simulation comes with its own level of time complexity. Budak et al. proposed APOSTLE [51], a learning-based multi-fidelity optimization that leverages batch processing and parallelization. They test APOSTLE on folded Cascode OTA (20 design variables), Strong-Arm Latch Comparator (13 design variables), and Programmable Gain Amplifier (58 independent design variables) and achieve a total computation time reduction of 17% to 30% compared to their previous work, DNN-Opt [52].

C. NEED FOR AI IN SIZING

The time complexity of the optimization tasks outlined above can escalate exponentially for industrial-scale problems as the number of design elements increases, rendering traditional methods such as grid search, particle swarm optimization, or classical BO inefficient for convergence. Recent studies have explored the use of multi-fidelity BO to integrate post-layout simulations into the schematic sizing process effectively [12], [53]. Furthermore, some approaches leverage AI to create trust regions or freeze design variables with minimal performance impact, thereby enhancing the efficiency of the design process [54]. These considerations highlight the need for an intelligent agent capable of learning from experience, predicting the behavior of the objective function, and making informed decisions within the design space through guided exploration. Such an agent should not only generalize to unseen tasks but also adapt previous knowledge to similar challenges, emulating the intuition of human designers on a much larger scale. In section IV, we review how machine learning techniques can address these challenges and meet the needs outlined above.

IV. MACHINE LEARNING FUNDAMENTALS IN EDA

Unlike traditional programming, ML enables algorithms to learn from data to perform tasks. Data, regardless of its complexity, is ultimately represented as tensors: text and audio as vectors (1D tensors), images as matrices (2D tensors), and videos as sequences of images (3D tensors). For non-Euclidean data like circuits, graphs are used, represented by adjacency and feature matrices. EDA problems often require a combination of representations and models to capture the full complexity of the design space. ML has traditionally been used for regression and classification of data; however, applications can be extended to performance prediction, closed-box optimization, decision-making, and synthesizing new data. The latter (generative AI) is an important advancement in ML because it allows the models to learn the underlying probability distribution that generated the training data. The following sections summarize key points for understanding ML in EDA.

A. LEARNING PARADIGMS

The choice of a learning paradigm depends on the structure and availability of data, as well as the task at hand. Supervised learning uses labeled input-output pairs for tasks such as regression and classification, allowing models to learn mappings between inputs and known outputs. Unsupervised learning, in contrast, works with unlabeled data to uncover hidden patterns or structures. Techniques such as clustering, dimensionality reduction, anomaly detection, missing data imputation, and generative modeling fall under this paradigm [55]. These methods aim to compress data, identify relationships without explicit labels, or explore the underlying data distribution for insights. Semi-supervised learning bridges supervised and unsupervised approaches by utilizing both labeled and unlabeled data, making it effective in scenarios where labeling is costly. Reinforcement learning (RL), meanwhile, involves agent(s) interacting with an environment and learning through rewards or penalties to optimize a policy. Lastly, transfer learning transfers knowledge from one task to a new, related task, reducing the need for large datasets by leveraging pre-trained models. This can significantly enhance learning efficiency, especially in data-scarce situations.

B. TRADITIONAL ML TO DEEP LEARNING

Principal component analysis, support vector machines, decision trees, and random forests are still used for regression, classification, and visualization of structured data. However, these methods require extensive data preprocessing (feature engineering) and exhibit limited adaptability to unseen data. Deep learning, a subset of ML, overcomes these shortcomings by processing data in a manner inspired by the human brain to perform representation learning.

1) FEED-FORWARD NETWORKS

Multi-layer Perceptrons (MLPs) are one of the foundational architectures in deep learning and belong to the class of feed-forward neural networks. An MLP consists of an input layer, one or more hidden layers, and an output layer, where each neuron is fully connected to every neuron in the subsequent layer. Each layer takes an input vector $\mathbf{h}^{(l)} \in \mathbb{R}^n$ and produces an output vector $\mathbf{h}^{(l+1)} \in \mathbb{R}^m$, the transformation $f_l : \mathbb{R}^n \to \mathbb{R}^m$ expressed as:

$$\mathbf{h}^{(l+1)} = \sigma(\mathbf{W}^{(l)}\mathbf{h}^{(l)} + \mathbf{b}^{(l)}) \tag{7}$$

where $\mathbf{W}^{(l)} \in \mathbb{R}^{n\times m}$ is the weight matrix, $\mathbf{b}^{(l)} \in \mathbb{R}^m$ is the bias vector, and $\sigma(\cdot)$ represents a non-linear activation function. The universal approximation theorem states that MLPs can approximate any continuous non-linear function, given appropriate hyperparameters such as the number of layers, neurons per layer, and choice of activation functions [56]. MLPs are frequently used for regression tasks in constructing a surrogate model for optimization, by learning complex mappings between circuit parameters and performance metrics [57].

2) EMBEDDINGS FOR FEATURE LEARNING

Deep learning techniques often represent data points as dense, continuous vectors in a lower-dimensional space, typically denoted as $\mathbf{v} \in \mathbb{R}^d$, where d is the dimensionality of the embedding vector. These vectors capture semantic relationships between data points, allowing models to encode complex information more efficiently. The similarity between two embeddings $\mathbf{v}_1$ and $\mathbf{v}_2$ is often calculated using *cosine similarity*, defined as $\frac{\mathbf{v}_1\cdot\mathbf{v}_2}{\|\mathbf{v}_1\|\|\mathbf{v}_2\|}$. This metric measures the angle between the two vectors, indicating how similar the data points are in the learned feature space. In analog circuit design, embeddings can compress information about the structure of a circuit, its neighboring components, and potential parasitics caused by circuit dimensions. By capturing these relationships in a compact form, embeddings facilitate tasks such as layout optimization [58], faster sizing [59], and parasitics prediction [60].

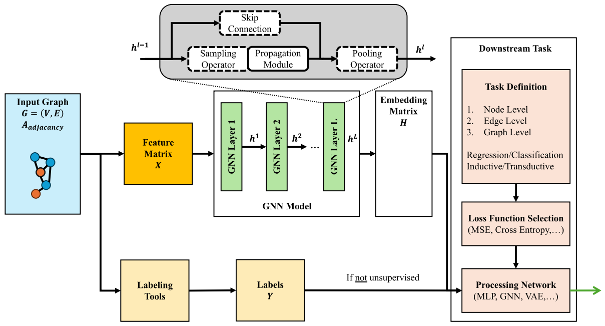

FIGURE 4. GNN design pipeline for EDA application reconstructed from [41] and [42]. SPICE and post-layout simulations can be used to generate labels for a supervised task.

C. GRAPH NEURAL NETWORKS (GNNS)

The motivation behind GNNs is to generate representations that capture both the topological and feature information of graphs, thereby extending deep learning models to permutation-invariant and relational problems. Given an input graph $G = (V, E)$ with a set of node features $X_n \in \mathbb{R}^{d\times|V|}$, the model learns a *neural message passing* function $m_{N(u)}$ to generate node embeddings z_u for each node $u \in V$ by exchanging information with neighboring nodes. Another neural network then processes these embeddings to perform tasks at the node, edge, or graph level in a supervised or self-supervised manner. Most analog EDA tasks can be defined as tasks on graph structures; for instance, layout symmetry extraction as node-level classification [61], [62], topology exploration as edge-level (link prediction) [63], and electromagnetic simulations [64] or parasitic prediction [59] as graph-level regression. Figure 4 shows an overview of the design pipeline, highlighting input graph construction, task definition, loss function selection, model selection, and the task processing network.

1) CIRCUIT CONSTRUCTION AS A GRAPH

The selection of graph types influences the expressiveness and complexity of GNNs in EDA tasks. Attributes such as homogeneity, heterogeneity, directionality, and acyclicity introduce performance trade-offs. Homogeneous graphs have uniform node and edge types and use a single adjacency matrix A, simplifying computations but limiting feature expressiveness. In contrast, heterogeneous graphs employ type-specific adjacency matrices A^{τ} for nodes and edges, allowing more expressive embeddings at the cost of increased overhead that scales with the number of types $|T|$ and the average degree of nodes, resulting in greater message-passing complexity. Directed graphs capture asymmetrical relationships, such as source-to-drain flows, enhancing the model's ability to learn circuit dependencies. However, they require specific aggregation functions per edge direction, adding complexity to message passing. Directed acyclic graphs (DAGs), common in hierarchical circuit structures, constrain propagation paths to improve information flow by preventing cycles. For heterogeneous graphs, the training

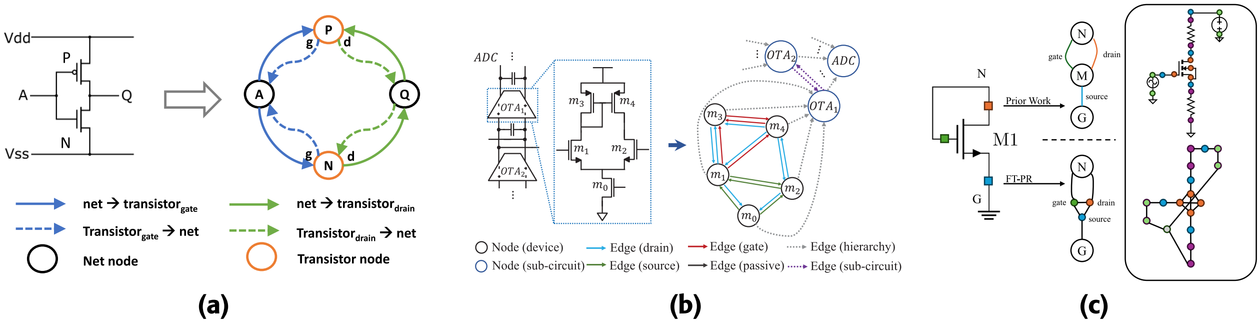

FIGURE 5. Approaches to representing circuits as graphs: (a) ParaGraph [60]. (b) TAG [65]. (c) reconstructed from PT-FR [59].

complexity depends on node count $|V|$, edge count $|E|$, feature size d, and type count $|T|$, generally approximated as $\mathcal{O}(|V|d^2 + |E|d|T|)$.

Selecting the appropriate graph type necessitates a balance between capturing the intricacies of complex tasks (favoring heterogeneous and directed graphs) and maintaining computational efficiency (homogeneous and acyclic structures) [66]. Deeb et al. review methods to convert an analog circuit into a graph model and strategies to augment the graph representation; however, they only focus on structure recognition [67]. Circuit2Graph [68] utilizes a homogeneous graph where each circuit component is represented as a node interconnected by edges, achieving computational efficiency while sacrificing some feature richness. On the other hand, ParaGraph [60] adopts a heterogeneous framework, distinguishing devices and nets as separate nodes with terminal-specific edge types, thereby enhancing parasitic modeling at the cost of added computational complexity due to multiple adjacency matrices. Hakhamaneshi et al. [59] leverage heterogeneity at the terminal level, treating each device's terminal as an individual node. This approach improves expressiveness and facilitates transfer learning across different topologies. Finally, TAG [65] integrates hierarchical heterogeneity with self-attention mechanisms, encoding both circuit structure and instance identifiers from the netlist, and employs hierarchical, directed connections to efficiently address layout-dependent effects. Refer to Figure 5 for a comparison of graph construction methods.

2) MODEL SELECTION

The computation graph of a GNN can be represented as a tree structure that unfolds the neighborhood around the target node. This process is mathematically defined as follows:

$$m_{N(u)}^{(k)} = \text{AGGREGATE}^{(k)}(h_u^{(k)}, \{h_v^{(k)} : v \in N(u)\})$$
$$h_u^{(k+1)} = \text{UPDATE}^{(k)}(h_u^{(k)}, m_{N(u)}^{(k)})$$

where UPDATE and AGGREGATE are differentiable functions (e.g., neural networks), and $m_{N(u)}^{(k)}$ represents the message aggregated from node u's k-hop neighborhood $N(u)$. The vectors $h_u^{(k)}$ and $h_v^{(k)}$ represent the node's and its neighbors' embeddings at layer k of the GNN. The aggregation module uses sampling, propagation, and pooling operators to create the message $m_{N(u)}^{(k)}$, and the update modules can use skip-connections or a recurrent neural network (RNN) to update the node's embedding $h_u^{(k+1)}$. The combination of various UPDATE and AGGREGATE functions creates a GNN model. For instance, convolutional operators are commonly used in the propagation module, forming the ConvGNN family. These models can be enhanced with attention mechanisms to learn the importance weights of neighboring embeddings. For a comprehensive taxonomy and review of GNN methods, we refer readers to [41] and [42].

3) GNN DESIGN CONSIDERATIONS

Unlike other ML models, stacking many GNN layers causes over-smoothing and degrades performance; therefore, various UPDATE strategies such as concatenation and skip-connections of previous layer embeddings, or the use of gated updates using RNNs, such as GRU or LSTM, have been proposed to mitigate over-smoothing [69]. Traditional pooling methods in GNNs tend to lose hierarchical structural information when applied to graph-level tasks. DiffPool, however, introduces a hierarchical pooling mechanism that uses a differentiable cluster assignment matrix to form meaningful node clusters at each layer, progressively coarsening the graph [70]. This approach enables GNNs to learn multi-level structural dependencies, which is critical for capturing complex relationships within the graph, such as subgraph behaviors. Additionally, extensive design space exploration has been conducted on GNNs to balance performance and architecture complexity [71]. Details on the training methodologies of GNNs in much of the existing literature are often inadequately documented, making direct comparisons challenging. To address this, Yamamoto et al. conduct a systematic exploration of hyper-parameter tuning in GNNs, with a focus on maximizing FoM and enhancing transferability across different technology processes [72].

D. VARIATIONAL AUTO-ENCODERS (VAEs)

VAEs are a class of generative models that combine Bayesian statistics with deep learning to perform efficient data compression, generation, and representation learning. Generative models assume each training data x (image, text, or graph) is a sample from an unknown probability distribution $p(x)$. VAEs aim to find a one-to-one mapping

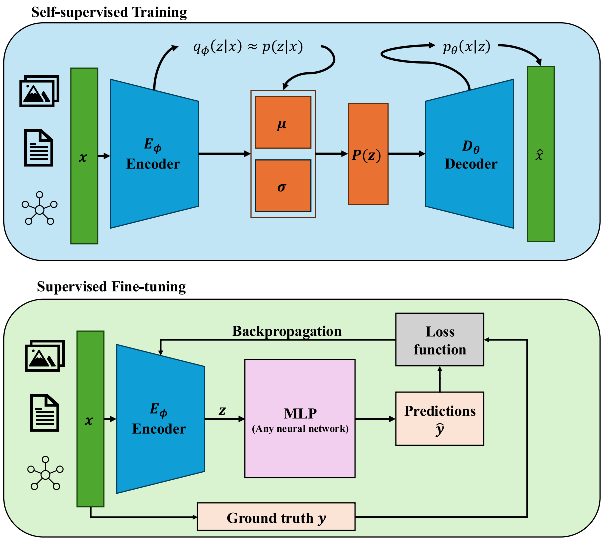

FIGURE 6. The training flow and architecture of VAEs. The fine-tuning can be achieved through limited labeled training data.

from $p(x)$ to a known and structured latent distribution $p(z)$ that can be sampled to produce unseen data points. They consist of an encoder E_ϕ and a decoder D_θ, which are neural networks parameterized by ϕ and θ that can be learned in an unsupervised or supervised manner by minimizing the reconstruction loss. Figure 6 depicts how VAEs are trained and used for inference.

1) ENCODER-DECODER ARCHITECTURE

Both encoder and decoder are MLPs parameterized by ϕ and θ, respectively, and can be employed for various downstream tasks after training converges. The encoder maps input data x to a probabilistic latent representation z, where the latent variables follow a multivariate Gaussian $p(z) \sim \mathcal{N}(0, 1)$. However, the posterior distribution of the latent variables $p(z|x)$ is intractable; so, the encoder uses variational inference thus assuming the transformation to z space $q_\phi(z|x) \sim \mathcal{N}(\mu(x), \sigma^2(x))$ follows a multivariate normal distribution and attempts to estimate its mean and variance. Loosely speaking, the encoder is a Taylor-series-like expansion of $p(x) = \sum_{i=1}^{d} p_\phi(x|z_i)p(z_i)$, where each $p(z_i) \sim \mathcal{N}(0, 1)$ and d is the dimensionality of the latent space vector z. The decoder then reconstructs the input x from a sample z drawn from this latent distribution. It models the likelihood $p_\theta(x|z)$, where θ are the decoder's network parameters, to reconstruct the data as $\hat{x} = p_\theta(x|z)$.

2) LATENT SPACE AND ITS APPLICATIONS

Once trained, VAEs can generate new data points by sampling $z \sim \mathcal{N}(0, I)$ from the latent space and decoding it into the original input space. VAEs are particularly useful in tasks such as synthetic data generation [22], feature learning, and dimensionality reduction for optimization [23]. In the latent space, similar data points tend to cluster together, which can be used to detect relationships and trade-offs in an unsupervised manner. For example, different sets of design parameters for analog ICs that yield similar performance characteristics may cluster in the latent space. Furthermore, Maus et al. [73] have shown that local BO in the latent space of deep VAE models aligns with local optimization in the input space. Moreover, graph VAEs can be combined with GNNs to efficiently explore circuits' design spaces, preserving critical relationships between components while optimizing performance metrics across various stages of the design flow [24]. Therefore, VAEs can uncover latent variables that correlate with performance metrics, enabling more efficient optimization, allowing for the synthesis of novel designs that satisfy the given constraints.

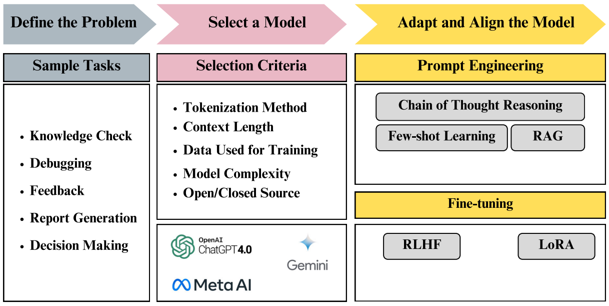

FIGURE 7. Overview of application pipeline using LLMs.

E. LARGE LANGUAGE MODELS (LLMs)

LLMs can process vast datasets and adapt to specific tasks, making them valuable for specialized domains like analog IC design. Understanding pre-training, fine-tuning, and in-context learning is essential for researchers aiming to effectively use LLMs. Figure 7 shows how LLMs can be applied to a task with key design decisions highlighted. subsection V-C examines the latest LLM-based automation tools for circuit sizing.

1) PRE-TRAINING

Most LLMs are generative pre-trained transformers with decoder-only architecture, and are trained in an unsupervised manner. They acquire general knowledge about language patterns, relationships, and structures in the pre-training phase from large amounts of online data. Once an LLM has been pre-trained on general data, it must be fine-tuned to perform well in specific domains or tasks. The same training principle applies to analog design, where datasets could include schematics, netlists, and technical papers. However, pre-training from scratch is often unnecessary for specific tasks because it requires significant computational power, extensive hyper-parameter tuning, and careful dataset preparation [15]. Instead, many applications can be effectively addressed by fine-tuning or adapting existing commercial off-the-shelf models, such as GPT, LLaMA, or BERT, which already possess general linguistic and structural knowledge. These pre-trained models provide a robust foundation, allowing researchers to focus on task-specific adaptation, significantly reducing both time and resource demands.

2) FINE-TUNING

Decoder-only transformers can be fine-tuned to transfer the general knowledge learned to a specific domain by adjusting the transformer parameters. Fine-tuning involves further training the model on a smaller, domain-specific dataset with labeled examples, allowing the LLM to adapt to the new task. Fine-tuning can be primarily additive (i.e., new parameters are added to the transformer), reparameterized, or partial [74]. Full-model fine-tuning is computationally expensive and impractical for very large models, and could result in catastrophic forgetting. This is where Parameter-Efficient Fine-Tuning (PEFT) methods, such as LoRA (Low-Rank Adaptation) [75], come into play. LoRA works by injecting learnable low-rank matrices into the model's layers, enabling the fine-tuning process to modify only a small subset of parameters, while the bulk of the pre-trained model remains frozen. The effectiveness of PEFT approaches, including LoRA, has been demonstrated in the broader context of machine learning, where fine-tuning efficiency is critical. Studies like Lialin et al. [76] and Fu et al. [77] demonstrate that PEFT approaches not only lower computational overhead but also prevent catastrophic forgetting, a common issue where the model loses pre-trained knowledge during fine-tuning. We refer the reader to [78] for a comprehensive review of continued pre-training and supervised fine-tuning domain adaptation strategies.

3) IN-CONTEXT LEARNING

Instead of modifying the model's parameters, LLMs can leverage the information provided in the context of the input (such as examples or instructions) to adapt to the task at hand. The auto-regressive nature of transformer models enables them to predict the next token based on the context, forming a deep understanding of the underlying data distribution. Auto-regressive inference can use the information provided in the prompt, assuming the prompt does not exceed the context length of the attention mechanism in the transformer. In-context learning can be achieved from zero-shot to few-shot learning, chain of thought (CoT) reasoning, and retrieval-augmented generation (RAG). In few-shot learning, the model is provided with a small set of examples or demonstrations within the input prompt, which helps it understand the task and generate the appropriate output. CoT reasoning is an inference strategy that breaks down complex tasks into intermediate steps, mimicking a human-like thinking process. Moreover, RAG further enhances in-context learning by allowing the model to access external databases in real time, improving the accuracy of generated outputs based on the provided context. This method is valuable because it offers flexibility. Therefore, the model can rapidly adapt to new tasks such as designing a novel circuit topology or modifying layout constraints based only on the examples it receives at runtime, without the need for retraining. LLMs have been applied to layout-level automation. For example, Layout CoPilot [26] simplifies human-tool interaction by converting natural language instructions into executable script commands. Similarly, GLayout automates layout generation through fine-tuning and RAG, eliminating the need to use GNNs on netlists [27]. Chen et al. investigate whether LLMs' inherent knowledge and few-shot learning capabilities can enhance crucial aspects of BO in generating analog layout constraints [25]. Chang et al. propose LaMAGIC [29], a topology generation framework for power converter design. subsection V-C investigates the application in analog IC sizing.

V. DISCUSSION: APPLICATIONS IN ANALOG IC SIZING

Our review of past research highlights that ML models cannot solely address the analog sizing problem. Instead, ML should be viewed as an asset for data processing, helping to frame and guide optimization problems, as discussed in subsection III-B. Moreover, ML can be used to tackle the challenges outlined in subsection III-A through unsupervised learning methods, the representation learning of GNNs and generative models, and the ability of LLMs to learn linguistic and domain-specific knowledge. Data scarcity remains a challenge; however, online learning in RL frameworks, transfer learning, and fine-tunability of ML models have been studied as solutions to this problem. We summarize and cluster papers according to their optimization detail and learning objective in Table 1, and provide further references on approaches that utilize GNNs in Table 2.

A. CANDID AUTOMATION METHODS

One key challenge in applying ML to analog sizing is the scarcity and variability of training data. Real-world analog designs are often protected by intellectual property, and generating diverse labeled datasets is infeasible. Different learning paradigms supervised, semi-supervised, self-supervised, and reinforcement learning (RL) have been explored. RL agents learn by interacting with the design environment, and neural networks can enhance sample efficiency by constructing an online surrogate model of the environment [88], or predicting the reward function [43]. RL has been increasingly applied to circuit sizing [90], [97], topology synthesis [98], and topology-aware sizing [95]. Transfer learning offers a promising path to reduce domain-specific training data requirements by leveraging knowledge from related tasks. However, RL presents challenges when applied to continuous and variable action spaces typical of circuit sizing and topology exploration, and defining a reward function that accurately reflects design goals is nontrivial.

1) TACKLING THE CURSE OF DIMENSIONALITY

BO is the optimization engine behind much of the research on automatic sizing, but it suffers from cubic complexity. Some research works attempt to tackle the dimensionality problem by parallelizing the algorithm [82], limiting the

TABLE 1. Clustering of key ML methods of circuit sizing and similar papers.

Method	ML Method	Optimization Framework	Verification Level	Learning Objective	Similar to
Angel [33]	MLP	Single-Objective	Schematic	• A semi-supervised model and database generation • learns to breakdown circuits based on the connectivity of the sub-blocks	[57], [79], [80]
ESSAB [81]	MLP + EA			• A new infill sampling method and surrogate model-assisted evolutionary algorithm • Ranks designs when uncertainty is high.	[82]
TransNet [83]	MLP + RL			• Proposes a one-to-one matrix representation of the circuit netlist using an MLP-mixer • Extract device and net features, without using GNNs.	—
APOSTLE [51]	RL	Multi-Fidelity	Schematic	• Asynchronously parallel online database generation. • A Gaussian Process approximates the rank of designs from cheap simulations. • Guiding the decision to perform more costly simulations.	[84]
BNN-BO [54]	Bayesian Neural Network, MLP		Schematic, Post-layout	• Predicts layout-aware performance with minimal post-layout simulations by modeling. • schematic-to-layout correlations. Uses Magical [8] for layout generation.	[12], [53], [85]
RoSE [86]	BO + RL, GNN, MLP	Single-Objective	PVT Aware, Schematic	• The BO finds an starting point for the RL agent to achieve faster convergence. • The GNN extracts topological information • An MLP processes the intermediate circuit specifications.	[87]–[89]
TRM [90]	MLP, Model-based RL	Constraint Satisfaction	PVT Aware, Schematic	• MLP acts as a SPICE surrogate and estimates the reward agent. • Uses progressive PVT exploration to focus on the most difficult corners first.	[47], [87]
BagNet [10]	Enhanced EA + MLP		PVT Aware, Post-layout-in-the-loop	• Increased sample efficiency • The MLP prunes-out less promising designs before running the post-layout simulation.	[91]
fRL-AD [92]	Adaptive Action-space RL	Multi-Objective	Parasitic Aware, Schematic	• Automation using RL agent and gm/ID methodology. • Optimizes AC parameters while maintaining fixed DC operating points. • States that gm/ID is much better to learn than the device size	[93], [94]
DNN-Opt [52]	RL, MLP, Random Forest	Single-Objective		• A critic network that takes pairs of design vectors as input and outputs performance predictions. • An actor network proposes new design candidates by determining changes in design parameters. • MLParest [91] is used to predict parasitics.	[22], [91]
MACRO [95]	ConvGNN, Multi-Agent RL, VAE	Bi-Level	Behavior Modeling, Schematic	• Unifies topology design and parameter tuning. • Uses the VAE for topology representation. • The GNN to extract features from a behavior-level, directed graph of the circuit.	[49], [96]

number of design variables [87], or using ML techniques to project the task into a low-dimensional latent space [63], [73]. Additionally, recent works have explored how large language models (LLMs) can guide exploration or aid in constraint generation [25], [30].

2) MINIMAL TRAINING DATASET

There are many methods that attempt to minimize training-to-deployment time. For instance, ESSAB [81] is an efficient surrogate model-assisted sizing method that addresses challenges such as cumulative prediction errors across performance metrics and high ML computational costs due to numerous specifications. ESSAB achieves high performance through a candidate design ranking method, a novel ANN construction technique, and an algorithm framework that incorporates an infill sampling criterion to optimize cost and performance. The Circuit Connectivity-Inspired Neural Network (CCI-NN) [80] employs sub-ANNs linked according to the circuit's modular connections, with a final fully connected layer for performance prediction. This architecture effectively models circuit inter-relations, reducing dataset size and SPICE simulation requirements while enhancing accuracy compared to conventional fully connected networks. Fayazi et al. proposed frameworks like AnGeL [33] and FuNToM [79], which leverage semi-supervised learning and modular functional modeling, dividing circuits into manageable E-networks for RF circuit analysis using neural networks. Similarly, the Cascaded Shallow Neural Network (C-SNN) model uses multiple shallow MLPs to size components iteratively, reflecting designer experience and bias while reducing training data requirements. Although currently aimed at generating initial sizes before layout verification, C-SNN can incorporate parasitic effects and constraints like inductance from EM simulations, making it a flexible tool for more comprehensive datasets [57]. Lastly, Shahane et al. [99] take a different approach using GNNs where each circuit is represented as a node in a graph, and leverage the closeness between node embeddings, along with a VAE, to create synthetic data.

3) RL-ENABLED AUTOMATION

Learning to Design Circuits (L2DC) [100] represents an early application of RL to analog IC sizing, predating later advancements such as GCN-RL [97]. The RL framework uses observations derived from simulation tools such as Hspice and Cadence Spectre, including DC operating points, AC magnitude and phase responses, and transistor states. L2DC's training charts show the agent first learns to meet hard-constraints (e.g., gain and bandwidth) and then learns to optimize good-to-have targets (e.g., area, power). Building on these foundations, more advanced methods have emerged. CRONus combines RL with neural simulators based on multiple MLPs to approximate reward functions to accelerate the design process [88]. TransNet enhances knowledge transfer across topologies by representing circuits as netlist-based matrices and utilizing MLPs to capture device and node interactions [83]. MACRO extends RL applications through multi-agent co-training, optimizing topology selection and sizing [95]. The Circuit Attention Network (CAN-RL) refines this approach with an actor-critic RL framework, incorporating stochastic layout-awareness to improve robustness against uncertainties and enable better knowledge transfer [101]. Other contributions, like DNN-Opt, address scalability challenges by introducing a sample-efficient two-stage optimization framework based on RL-inspired algorithms. This method integrates critical device identification to extend its applicability to industrial-scale circuits [52]. Finally, AutoCkt takes a layout-aware approach by combining RL with tools like the Berkeley Analog Generator, successfully validating its designs on advanced technologies such as 16-nm FinFET while accounting for process variations [84]. These advancements highlight the versatility and efficiency of RL-based methodologies in revolutionizing analog IC sizing.

4) PVT AWARE AUTOMATION

Yang et al. reframe the sizing task as a constraint satisfaction problem with trust regions, focusing on efficiently meeting PVT-related constraints rather than achieving a global optimum [90]. By incorporating supervised learning to train model-based agents that mimic SPICE simulations, this method achieves stable convergence and rapid exploration of the design space. Its progressive PVT exploration strategy initially focuses on the most challenging PVT corner, verifying solutions across all corners while minimizing computational overhead. Validated in industrial applications using advanced TSMC 5nm and 6nm technologies, the method outperforms human designers by achieving superior area optimization and reducing design time.

Vaz et al. take a different approach by utilizing artificial neural networks to approximate PVT corner performances. Their approach inputs circuit sizing and nominal performance data into a PVT regressor that estimates corner performances through multiple parallel neural networks [89]. Applied to a state-of-the-art Class C/D voltage-controlled oscillator, this method saves over 16 days of computational effort, balancing high accuracy with computational efficiency. The RoSE [86] Framework combines BO and RL to enhance convergence and sample efficiency in PVT-aware sizing. It uses BO to determine an optimal starting point for the RL agent and employs a variation-aware reward function incorporating sub-rewards for all PVT corners. Architecturally, it employs an MLP to process intermediate circuit specifications under PVT conditions and a GAT to capture device parameters. These features enable robust optimization and superior performance across multiple PVT corners.

5) HYBRID APPROACHES

The integration of knowledge-based methods, such as the gm/ID approach, with advanced learning-based techniques offers a promising direction for analog circuit design automation. Experimental data demonstrate that when neural networks are trained on circuit data, the gm/ID metric is learned more effectively than traditional device sizing parameters [94]. The gm/ID methodology leverages a designer's intuition to define feasible design spaces, making it highly effective for tasks like technology migration. Recent advancements utilize neural network surrogate models to predict performance parameters, enabling the application of automatic differentiation for optimization problems and achieving a fourfold improvement in training data sampling efficiency. [93]. RL-based approaches, such as the fast Reinforcement Learning Analog circuit Designer (fRL-AD) [92], complement this by optimizing AC parameters while ensuring fixed DC operating points. By leveraging Proximal Policy Optimization (PPO), fRL-AD addresses multi-objective optimization challenges and introduces adaptive action spaces for improved efficiency. However, its reliance on small-signal equivalent circuits limits its applicability to non-linear components, suggesting future work should explore novel RL frameworks and incorporate additional performance metrics like noise rejection. Together, these methodologies create a hybrid framework that balances designer expertise with data-driven optimization, driving innovation in analog design automation.

TABLE 2. Summary of GNN-based methods for circuit design tasks.

Method Name	GNN Model	Task Level	Circuit Representation	Learning Objective	Similar to
OPAMP-Generator [24]	GCN+GRU; BO+VGAE	Graph; Edge	• Circuit elements are abstracted as a directed acyclic graph (gm,Rparasitics,Cparasitics)	• Performs bi-level Bayesian optimization and full-flow automation using the MAGICAL [8] layout generator. • Topology optimization occurs in the latent space of the graph VAE, and sizing uses the gm/ID method.	[48], [103], [104]
Li et al. [105]	RelationalGCN	Graph	• Components are modeled as nodes • Edges represent DC or AC wires.	• Focuses on designing low-dropout regulators using the open-source SKY130 PDK. • The GNN is the function approximator for the RL agent.	—
Circuit2-Graph [68]	GraphSAGE; MLP; VGAE	Graph; Node	• Circuit components are abstracted into one-hot encoded vectors for graph construction.	• Studies the impact of different graph construction hyperparameters on task performance.	[29], [62], [67]
FT-PR [59]	Various GNNs; MLP		• Each terminal of a component and circuit net is a node. • No edge types.	• Pre-trains a GNN to predict DC voltages of nodes, • Then fine-tunes to downstream tasks, such as gain and output resistance prediction.	[10]
CktGNN [106]	Nested GNN; VGAE	Graph	• Circuit graphs are represented as combinations of sub-graphs • Each subgraph is a hyper-node.	• Inner GNN embeds sub-graphs into node embeddings • Outer GNN performs message passing for optimization. • Simultaneously, optimizes the topology and sizing of circuits, leveraging hierarchical representations.	[107]
GNN-BLR [63]	GCN	Edge	• Device nodes and nets are modeled as separate nodes. • Directed and typed edges.	• Proposes an improved surrogate model for BO • Focuses on parasitic-aware optimization.	[10], [84], [91]
TAG [65]	GAT; MLP	Graph	• Devices and sub-circuits are modeled as nodes. • Hierarchical nodes representing sub-circuits.	• Introduces a novel circuit representation method • Tested on downstream tasks such as wirelength and parasitic capacitance estimation.	[60]
ParaGraph [60]	Custom GNN; MLP	Node	• Device nodes and nets are represented as separate nodes. • Directed and typed edges.	• Proposes a new GNN architecture to handle heterogeneous graphs with improved accuracy. • Improves parasitics and device parameter predictions using pre-layout simulations to optimize design.	—
CAN-RL [101]	Custom GAT; RL	Graph	• Device pins and circuit nets are represented as nodes. • Connections as directed edges.	• Pre-trains a parasitic prediction GNN to avoid in-loop simulations and improve optimization accuracy. • Introduces a stochastic method to include layout effects • Employs a Circuit Attention Network to enable transfer learning across topologies.	[80], [97], [100]
Lu et al. [23]	GNN+GRU; BO+VGAE	Graph; Edge; Node	• Node types are input, output, ground, or intermediate. • Edges represent various devices.	• Models circuit components behaviorally as sub-circuits and integrates them into the optimization process. • Employs bi-level optimization and post-layout awareness, using an auto-encoder to enhance the Bayesian optimization algorithm.	[24]

B. GNNs IN AUTOMATION

GNNs have demonstrated significant potential in EDA by processing non-Euclidean data, which naturally represents circuit netlists as graphs. GNNs can perform representation learning on circuit structures, eliminating the need for manual feature extraction. GNNs can then apply ML techniques on the learned representations for node, edge, or graph level tasks [42]. This capability makes them well-suited for tasks such as topology exploration [23] and analog circuit labeling [62], [67]. GNNs have also been employed to model the relational data between circuit components and optimize layouts for parasitics [99], [102].

GCN-RL [97] was the first to use GCNs on graphs constructed from circuit netlists for op-amp sizing, demonstrating some transfer learning across technology nodes and similar topologies. However, the design is only verified at the schematic level and does not extend to unseen topologies.

Li et al. provide an open-source framework to design and optimize a low-dropout voltage regulator using RL and a relational GNN in the SKY130 process [105]. Similarly, Hakhamaneshi et al. define another open-source framework, transferrable to unseen topologies, where a GNN trained on inexpensive DC simulation to predict node voltages can be fine-tuned to adapt to more complicated s such as gain of a circuit as a graph-level regression [59].

1) PARASITIC AWARE AUTOMATION

MLParest is an early work that incorporates parasitics into schematic-level sizing, reducing the number of iterations between pre-layout and post-layout design phases [91].

MLParest uses a Random Forest model trained on extracted features from circuit schematic netlists; consequently, it is not easily applicable to unseen circuits. Hakhamaneshi et al. introduce their BagNet: Berkeley Analog Generator with Layout Optimizer to address the challenges of post-layout parasitics [10]. The method reduces the number of simulations required by evolutionary combinatorial optimizers by employing a DNN to filter out generated samples before subjecting them to time-consuming simulations. This DNN acts as an oracle, effectively predicting whether a design will meet the required specifications, thus drastically reducing the number of simulation iterations. BagNet integrates this discriminator into the optimization loop, learning from past designs to improve sample efficiency. Ren et al. propose a custom message passing function for their GNN, named ParaGraph, to predict layout-dependent parasitics [60]. Lastly, Liu et al. avoid layout generation by predicting parasitics using GNNs and BO [63].

2) TOPOLOGY AWARE AUTOMATION

Poddar et al. achieve joint topology optimization using a VAE to build an adaptive dataset of 360 op-amp topologies, as detailed in [22]. However, this approach is not easily extensible to various circuit blocks. CktGNN [106] encodes circuit graphs using a two-level GNN framework (nested GNN). CktGNN represents circuits as combinations of subgraphs in a known subgraph basis. It additionally introduces Open Circuit Benchmark (OCB) as an open-sourced dataset to assess and reproduce the results. Yang et al. have contributed many ideas to topology-aware optimization and dimensionality reduction of the design space using VAEs [23], [48], [104], [107]. Moreover, Yang et al. study the complexity of analog design space exploration and exploitation, and suggest dimension decoupling, dimension embedding, region restriction and region decomposition, and introduce Analog Circuit Optimization Benchmarks (ACOB) [103]. Yang et al. combine all the previous work into a unified framework to generate Op-Amps from specification to layout [24]. Refer to Table 2 for a summary of the methods.

C. LLMs IN AUTOMATION

LLMs represent a sub-domain of generative models, yet their application in analog design remains relatively unexplored. Practitioners should be aware of certain limitations when applying LLMs to this field: LLMs struggle with precise mathematical calculations, often requiring instructions to generate code that can verify or test their outputs. Additionally, they are prone to "hallucination," meaning they may confidently provide incorrect answers if they lack specific knowledge. Despite these challenges, recent research has shown potential in adapting LLMs to analog IC design by employing prompt engineering and selective fine-tuning. In-context learning can be viewed as implicit fine-tuning since it produces meta-gradients through forward computation similar to explicit fine-tuning [108]. This work is the first to integrate large language models into standard cell layout design optimization, leveraging prompt engineering and an iterative ReAct framework to generate high-quality cluster constraints; the proposed methodology not only reduces cell area by up to 19.4% on industrial 2nm benchmarks but also ensures fully routable designs, addressing critical challenges in both performance-power-area optimization and routability. Nonetheless, the following subsections focus on the use of LLMs in circuit sizing.

AnalogCoder: Lai et al. present a training-free, open-source LLM agent that automates analog circuit design through PySpice code generation, achieving rapid initial design with prompt engineering and a feedback-enhanced workflow [28]. AnalogCoder automates schematic-level tasks, yet the absence of topology-aware design and post-layout parasitic management limits its capability to accurately predict or mitigate parasitic or PVT variations. Experimental results show that AnalogCoder successfully generates 20 out of 24 targeted circuits, surpassing the performance of models like GPT-4o, yet it fails in high-complexity designs such as op-amp integrators and voltage-controlled oscillators. LADAC, however, utilizes few-shot learning and iterative prompt engineering to assist in parasitic-aware sizing for improved layout robustness [31]. ADO-LLM addresses complex circuit sizing by integrating BO with LLM in-context learning, making it well-suited for layout and parasitic-aware circuit sizing [30]. The model can infer design principles from a few example circuit layouts and their associated performance metrics. Provided with initial examples, ADO-LLM refines future designs based on performance metrics, achieving efficient, accurate multi-objective optimization without extensive retraining. SPICEPilot [109] is a prompt-steering approach to creating PySpice code based on user specifications using various commercially available LLMs. The paper introduces AmpAgent [110], an LLM-based multi-agent system for efficiently designing multi-stage amplifiers by automating literature analysis, mathematical reasoning, and device sizing. AmpAgent demonstrates significant improvements in design efficiency, iteration reduction, execution time, and circuit performance compared to traditional methods, showcasing the potential of LLMs in complex analog circuit design and performance porting.

VI. FUTURE DIRECTION

Although the analog EDA community has made significant breakthroughs in analog IC design automation, we have identified the following as potential areas to explore in future research.

A. EXTEND TO INDUSTRIAL-SCALE DESIGN

Experimenting with larger-scale design spaces and highly non-linear systems is important; designing circuits in the sub-threshold and sub-micron regions introduces extreme non-linear behavior, complicating the design process. Most research in academia focuses on +45 nm technology nodes with the number of design variables on the order of 10 (partly because smaller technology nodes are exponentially more expensive to tape out), whereas industrial designs are below 5 nm, and much more complicated in their design space.

B. INCORPORATE MANUFACTURING BEST PRACTICES

Most of the research focuses on meeting the design requirements at the schematic level, and those that incorporate post-layout effects rely on automatic layout generation and constraint extraction. Layout generation tools do not yet incorporate the best practices of experienced designers, and thus might lead to suboptimal designs.

C. EXPLORE NOVEL TOPOLOGIES

EDA tools should help designers discover novel topologies that can be sized to achieve cutting-edge performances. Future studies of joint topology exploration and sizing such as [98] could incorporate RL, VAEs, and symbolic modeling to jointly explore topology selection and circuit sizing. The success of circuit sizing automation, therefore, depends on the development of state-of-the-art topology exploration and layout generation tools.

D. CREATE OPEN-SOURCE DATASET AND TESTBENCHES

Establishing standardized datasets and test-benches, similar to other ML fields, would greatly accelerate research. While efforts like those by AICircuit [111], ACOB [103], and AnalogCoder [28] help create a diverse dataset of simple and complex analog circuit blocks, they only include schematic-level simulations and thus lack reliability against parasitics and PVT variations.

E. AUTOMATION OF COMMON CIRCUIT BLOCKS

Majority of the research work focuses on the design of Op-Amps since they are a critical building block in analog and mixed signal systems; however, more complex systems need to be automated. For instance, design of compensation schemes [48], ADCs [112], PLLs, power amplifiers [113], and wide-band impedance matching [114] are relatively under-explored. Krylov et al. [47] experiment with an extensive set of analog and RF circuit blocks.

F. LEVERAGE REPRESENTATION LEARNING

GNNs have proven that they can learn from abundant data such as netlists and hierarchical design of analog systems to predict post-layout performance; however, an optimal graph construction method and message passing function that can handle the range of device types and connections can be a subject for future research. Furthermore, generative models such as auto-encoders can be used to compress high-dimensional design spaces, facilitating efficient exploration without requiring labeled data, or generating synthetic training data. Research could explore how unsupervised methods like these can be better applied to analog circuit optimization.

G. EXPERIMENT WITH NOVEL MULTI-AGENT RL (MARL)

Research has shown that analog design cannot be solved in a zero-shot manner; instead, automation should mimic the approach of an expert engineer. MARL has been used to explore design space, balance between inexpensive and expensive simulations, and learn policies that avoid unreliable designs [49], [96]. Wen et al. achieve co-operative learning between agents by using a multi-agent transformer (MAT) to treat a team of agents as a sequence model [115]. MAT is trainable online and demonstrates high data efficiency. Novel approaches in MARL, such as MAT, enable the parallelization of multi-fidelity problems, as outlined in subsubsection III-B4.

H. MULTI-MODAL AI

Combining multiple data sources (e.g., netlists of open source circuits, research papers and textbooks) can help address data scarcity in analog design. Leveraging multi-modal AI can enhance models' ability to integrate diverse information sources and improve overall performance.

VII. CONCLUSION

Our review summarizes the wide range of research that has contributed to practical AI-driven analog IC sizing. We provide a structured overview of references for various sizing methodologies, and highlight the contributions of MLPs, VAEs, GNNs, and LLMs to analog IC sizing using illustrative examples. Moreover, we review methods that address data scarcity, PVT variations, and layout parasitics, and we suggest future research directions.

REFERENCES

[1] B. Razavi, *Design of Analog CMOS Integrated Circuits*, 2nd ed., New York, NY, USA: McGraw-Hill, 2017.

[2] M. Liu et al., "ChipNeMo: Domain-adapted LLMs for chip design," 2023, *arXiv:2311.00176*.

[3] L. Chen et al., "The dawn of AI-native EDA: Opportunities and challenges of large circuit models," 2024, *arXiv:2403.07257*.

[4] L. Alrahis, J. Knechtel, F. Klemme, H. Amrouch, and O. Sinanoglu, "GNN4REL: Graph neural networks for predicting circuit reliability degradation," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 11, pp. 3826–3837, Nov. 2022.

[5] R. Mina, C. Jabbour, and G. E. Sakr, "A review of machine learning techniques in analog integrated circuit design automation," *Electronics*, vol. 11, no. 3, p. 435, Jan. 2022.

[6] J. P. S. Rosa, D. J. D. Guerra, N. C. G. Horta, R. M. F. Martins, and N. C. C. Lourenço, "Using ANNs to size analog integrated circuits," in *Using Artificial Neural Networks for Analog Integrated Circuit Design Automation*. Cham, Switzerland: Springer, 2020, pp. 45–66.

[7] A. F. Budak, S. Zhang, M. Liu, W. Shi, K. Zhu, and D. Z. Pan, "Machine learning for analog circuit sizing," in *Machine Learning Applications in Electronic Design Automation*. Cham, Switzerland: Springer, 2022, pp. 307–335.

[8] B. Xu, K. Zhu, M. Liu, Y. Lin, S. Li, X. Tang, N. Sun, and D. Z. Pan, "MAGICAL: Toward fully automated analog IC layout leveraging human and machine intelligence: Invited paper," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 2019, pp. 1–8.

[9] T. Dhar, K. Kunal, Y. Li, Y. Lin, M. Madhusudan, J. Poojary, A. K. Sharma, S. M. Burns, R. Harjani, J. Hu, P. Mukherjee, S. Yaldiz, and S. S. Sapatnekar, "The ALIGN open-source analog layout generator: V1.0 and beyond (invited talk)," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD)*, Nov. 2020, pp. 1–2.

[10] K. Hakhamaneshi, N. Werblun, P. Abbeel, and V. Stojanovic, "BagNet: Berkeley analog generator with layout optimizer boosted with deep neural networks," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2019, pp. 1–8.

[11] I. Abel, M. Neuner, and H. E. Graeb, "A hierarchical performance equation library for basic op-amp design," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 7, pp. 1976–1989, Jul. 2022.

[12] J. Huang, C. Wang, Y. Yan, C. Tao, F. Yang, C. Yan, W. Hu, D. Zhou, and X. Zeng, "An analog circuit building block generator via nested multi-fidelity modeling," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 70, no. 8, pp. 3280–3293, Aug. 2023.

[13] R. Zhou, P. Poechmueller, and Y. Wang, "An analog circuit design and optimization system with rule-guided genetic algorithm," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 12, pp. 5182–5192, Dec. 2022.

[14] H.-Y. Liu and L. P. Carloni, "On learning-based methods for design-space exploration with high-level synthesis," in *Proc. 50th ACM/EDAC/IEEE Design Autom. Conf. (DAC)*. New York, NY, USA: Association for Computing Machinery, May 2013, pp. 1–7.

[15] S. Wu, O. Irsoy, S. Lu, V. Dabravolski, M. Dredze, S. Gehrmann, P. Kambadur, D. Rosenberg, and G. Mann, "BloombergGPT: A large language model for finance," 2023, *arXiv:2303.17564*.

[16] J. You, B. Liu, R. Ying, V. S. Pande, and J. Leskovec, "Graph convolutional policy network for goal-directed molecular graph generation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, Jan. 2018, pp. 6412–6422.

[17] A. Al-Ali, B. Maundy, A. Allagui, and A. Elwakil, "Optimum impedance spectroscopy circuit model identification using deep learning algorithms," *J. Electroanal. Chem.*, vol. 924, Nov. 2022, Art. no. 116854.

[18] W. Chen, B. Yan, A. Xu, X. Mu, X. Zhou, M. Jiang, C. Wang, R. Li, J. Huang, and J. Dong, "An intelligent matching method for the equivalent circuit of electrochemical impedance spectroscopy based on random forest," *J. Mater. Sci. Technol.*, vol. 209, pp. 300–310, Feb. 2025.

[19] M. F. M. Barros, J. M. C. Guilherme, and N. C. G. Horta, "Enhanced techniques for analog circuits design using SVM models," in *Analog Circuits and Systems Optimization Based on Evolutionary Computation Techniques*. Berlin, Germany: Springer, 2010, pp. 89–107, doi: 10.1007/978-3-642-12346-7_4.

[20] Y. Ma, Z. He, W. Li, L. Zhang, and B. Yu, "Understanding graphs in EDA: From shallow to deep learning," in *Proc. Int. Symp. Phys. Design*. New York, NY, USA: Association for Computing Machinery, Mar. 2020, pp. 119–126.

[21] T. N. Kipf and M. Welling, "Variational graph auto-encoders," 2016, *arXiv:1611.07308*.

[22] S. Poddar, A. Budak, L. Zhao, C.-H. Hsu, S. Maji, K. Zhu, Y. Jia, and D. Z. Pan, "A data-driven analog circuit synthesizer with automatic topology selection and sizing," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2024, pp. 1–6.

[23] J. Lu, L. Lei, F. Yang, L. Shang, and X. Zeng, "Topology optimization of operational amplifier in continuous space via graph embedding," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2022, pp. 142–147.

[24] J. Lu, L. Lei, J. Huang, F. Yang, L. Shang, and X. Zeng, "Automatic op-amp generation from specification to layout," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 42, no. 12, pp. 4378–4390, Dec. 2023.

[25] G. Chen, K. Zhu, S. Kim, H. Zhu, Y. Lai, B. Yu, and D. Z. Pan, "LLM-enhanced Bayesian optimization for efficient analog layout constraint generation," 2024, *arXiv:2406.05250*.

[26] B. Liu, H. Zhang, X. Gao, Z. Kong, X. Tang, Y. Lin, R. Wang, and R. Huang, "LayoutCopilot: An LLM-powered multi-agent collaborative framework for interactive analog layout design," 2024, *arXiv:2406.18873*.

[27] A. Hammoud, C. Goyal, S. Pathen, A. Dai, A. Li, G. Kielian, and M. Saligane, "Human language to analog layout using GLayout layout automation framework," in *Proc. ACM/IEEE 6th Symp. Mach. Learn. CAD (MLCAD)*. New York, NY, USA: Association for Computing Machinery, Sep. 2024, pp. 1–7.

[28] Y. Lai, S. Lee, G. Chen, S. Poddar, M. Hu, D. Z. Pan, and P. Luo, "AnalogCoder: Analog circuit design via training-free code generation," 2024, *arXiv:2405.14918*.

[29] C.-C. Chang, Y. Shen, S. Fan, J. Li, S. Zhang, N. Cao, Y. Chen, and X. Zhang, "LaMAGIC: Language-model-based topology generation for analog integrated circuits," 2024, *arXiv:2407.18269*.

[30] Y. Yin, Y. Wang, B. Xu, and P. Li, "ADO-LLM: Analog design Bayesian optimization with in-context learning of large language models," 2024, *arXiv:2406.18770*.

[31] C. Liu, Y. Liu, Y. Du, and L. Du, "LADAC: Large language model-driven auto-designer for analog circuits," Authorea Preprints, Tech. Rep., 2024. [Online]. Available: [Online]. Available: https://api.semanticscholar.org/CorpusID:268259740

[32] M. Fayazi, Z. Colter, E. Afshari, and R. Dreslinski, "Applications of artificial intelligence on the modeling and optimization for analog and mixed-signal circuits: A review," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 6, pp. 2418–2431, Jun. 2021.

[33] M. Fayazi, M. T. Taba, E. Afshari, and R. Dreslinski, "AnGeL: Fully-automated analog circuit generator using a neural network assisted semi-supervised learning approach," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 70, no. 11, pp. 4516–4529, Nov. 2023.

[34] T. Chen, G. L. Zhang, B. Yu, B. Li, and U. Schlichtmann, "Machine learning in advanced IC design: A methodological survey," *IEEE Design Test.*, vol. 40, no. 1, pp. 17–33, Feb. 2023.

[35] T. Q. Nguyen, T. Hoang, L. Zhang, O. A. Dobre, and T. Q. Duong, "A survey on smart optimisation techniques for 6G-oriented integrated circuits design," *Mobile Netw. Appl.*, vol. 28, no. 6, pp. 2227–2244, Dec. 2023.

[36] S. Maji, A. F. Budak, S. Poddar, and D. Z. Pan, "Toward end-to-end analog design automation with ML and data-driven approaches (invited paper)," in *Proc. 29th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2024, pp. 657–664.

[37] A. F. Budak, K. Zhu, H. Chen, S. Poddar, L. Zhao, Y. Jia, and D. Z. Pan, "Joint optimization of sizing and layout for AMS designs: Challenges and opportunities," in *Proc. Int. Symp. Phys. Design*. New York, NY, USA: Association for Computing Machinery, Mar. 2023, pp. 84–92.

[38] L. Alrahis, J. Knechtel, and O. Sinanoglu, "Graph neural networks: A powerful and versatile tool for advancing design, reliability, and security of ICs," in *Proc. 28th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2023, pp. 83–90.

[39] D. S. Lopera, L. Servadei, G. N. Kiprit, S. Hazra, R. Wille, and W. Ecker, "A survey of graph neural networks for electronic design automation," in *Proc. ACM/IEEE 3rd Workshop Mach. Learn. CAD (MLCAD)*, Aug. 2021, pp. 1–6.

[40] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[41] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, Jan. 2020.

[42] D. Sánchez, L. Servadei, G. N. Kiprit, R. Wille, and W. Ecker, "A comprehensive survey on electronic design automation and graph neural networks: Theory and applications," *ACM Trans. Design Autom. Electron. Syst.*, vol. 28, no. 2, pp. 1–27, Feb. 2023.

[43] H. Zhu and X. Chen, "A survey of reinforcement learning for electronic design automation," in *Proc. 2nd Int. Symp. Electron. Design Autom. (ISEDA)*, May 2024, pp. 717–721.

[44] G. Huang, J. Hu, Y. He, J. Liu, M. Ma, Z. Shen, J. Wu, Y. Xu, H. Zhang, K. Zhong, X. Ning, Y. Ma, H. Yang, B. Yu, H. Yang, and Y. Wang, "Machine learning for electronic design automation: A survey," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 26, no. 5, pp. 1–46, Jun. 2021.

[45] C. Recker, B. Braswell, P. Drennan, and C. McAndrew, "A web tool for interactive exploration of analog design tradeoffs," in *Proc. IEEE Custom Integr. Circuits Conf.*, Sep. 2006, pp. 631–634.

[46] T. Liao and L. Zhang, "Parasitic-aware gm/ID-based many-objective analog/RF circuit sizing," in *Proc. 19th Int. Symp. Quality Electron. Design (ISQED)*, Mar. 2018, pp. 100–105.

[47] D. Krylov, P. Khajeh, J. Ouyang, T. C. Reeves, T. Liu, H. Ajmal, H. Aghasi, and R. Fox, "Learning to design analog circuits to meet threshold specifications," in *Proc. Int. Conf. Mach. Learn.*, Jan. 2023, pp. 17858–17873.

[48] J. Lu, L. Lei, F. Yang, C. Yan, and X. Zeng, "Automated compensation scheme design for operational amplifier via Bayesian optimization," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, Dec. 2021, pp. 517–522.

[49] J. Zhang, J. Bao, Z. Huang, X. Zeng, and Y. Lu, "Automated design of complex analog circuits with multiagent based reinforcement learning," in *Proc. 60th ACM/IEEE Design Autom. Conf. (DAC)*, Jul. 2023, pp. 1–6.

[50] K. Li and F. Li, "Multi-fidelity methods for optimization: A survey," 2024, *arXiv:2402.09638*.

[51] A. F. Budak, D. Smart, B. Swahn, and D. Z. Pan, "APOSTLE: Asynchronously parallel optimization for sizing analog transistors using DNN learning," in *Proc. 28th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2023, pp. 70–75.

[52] A. F. Budak, P. Bhansali, B. Liu, N. Sun, D. Z. Pan, and C. V. Kashyap, "DNN-opt: An RL inspired optimization for analog circuit sizing using deep neural networks," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, Dec. 2021, pp. 1219–1224.

[53] Z. Gao, J. Tao, F. Yang, Y. Su, D. Zhou, and X. Zeng, "Efficient performance trade-off modeling for analog circuit based on Bayesian neural network," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2019, pp. 1–8.

[54] A. F. Budak, K. Zhu, and D. Z. Pan, "Practical layout-aware analog/mixed-signal design automation with Bayesian neural networks," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD)*, Oct. 2023, pp. 1–8.

[55] E. Bernard and S. Wolfram, *Introduction To Machine Learning*. Champaign, IL, USA: Wolfram Media, 2021.

[56] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, no. 5, pp. 359–366, Jan. 1989.

[57] P.-O. Beaulieu, É. Dumesnil, F. Nabki, and M. Boukadoum, "Analog RF circuit sizing by a cascade of shallow neural networks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 42, no. 12, pp. 4391–4401, Dec. 2023.

[58] 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 *Proc. 26th Asia South Pacific Design Autom. Conf. (ASP-DAC)*. New York, NY, USA: Association for Computing Machinery, Jan. 2021, pp. 152–157.

[59] K. Hakhamaneshi, M. Nassar, M. Phielipp, P. Abbeel, and V. Stojanovic, "Pretraining graph neural networks for few-shot analog circuit modeling and design," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 42, no. 7, pp. 2163–2173, Jul. 2023.

[60] H. Ren, G. F. Kokai, W. J. Turner, and T.-S. Ku, "ParaGraph: Layout parasitics and device parameter prediction using graph neural networks," in *Proc. 57th ACM/IEEE Design Autom. Conf. (DAC)*, Jul. 2020, pp. 1–6.

[61] Q. Xu, L. Wang, J. Wang, L. Cheng, S. Chen, and Y. Kang, "Graph attention-based symmetry constraint extraction for analog circuits," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 71, no. 8, pp. 3754–3763, Aug. 2024.

[62] K. Settaluri and E. Fallon, "Fully automated analog sub-circuit clustering with graph convolutional neural networks," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2020, pp. 1714–1715.

[63] M. Liu, W. J. Turner, G. F. Kokai, B. Khailany, D. Z. Pan, and H. Ren, "Parasitic-aware analog circuit sizing with graph neural networks and Bayesian optimization," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Feb. 2021, pp. 1372–1377.

[64] Z. Guo, H. Hao, and D. Katabi, "Circuit-GNN: Graph neural networks for distributed circuit design," in *Proc. Int. Conf. Mach. Learn.*, May 2019, pp. 7364–7373.

[65] 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 *Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD)*. New York, NY, USA: Association for Computing Machinery, Oct. 2022, pp. 1–9.

[66] A. Said, M. Shabbir, B. Broll, W. Abbas, P. Völgyesi, and X. Koutsoukos, "Circuit design completion using graph neural networks," *Neural Comput. Appl.*, vol. 35, no. 16, pp. 12145–12157, Feb. 2023.

[67] A. Deeb, M. Salem, A. Ibrahim, J. Pichler, S. Tkachov, K. Anjeza, F. A. Machot, and K. Kyamakya, "A graph attention network based system for robust analog circuits' structure recognition involving a novel data augmentation technique," *IEEE Access*, vol. 12, pp. 29308–29344, 2024.

[68] Y. Yamakaji, H. Shouno, and K. Fukushima, "Circuit2Graph: Circuits with graph neural networks," *IEEE Access*, vol. 12, pp. 51818–51827, 2024.

[69] W. L. Hamilton, *Graph Representation Learning* (Synthesis Lectures on Artificial Intelligence and Machine Learning). Cham, Switzerland: Springer, Sep. 2020.

[70] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec, "Hierarchical graph representation learning with differentiable pooling," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 4805–4815.

[71] J. You, R. Ying, and J. Leskovec, "Design space for graph neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, Jan. 2020, pp. 17009–17021.

[72] K. Yamamoto and N. Takai, "Comparison of analog circuit sizing networks and number of steps," in *Proc. 9th Int. Conf. Integr. Circuits, Design, Verification (ICDV)*, Jun. 2024, pp. 258–263.

[73] N. Maus, H. T. Jones, J. Moore, M. J. Kusner, J. L. Bradshaw, and J. R. Gardner, "Local latent space Bayesian optimization over structured inputs," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, Jan. 2022, pp. 34505–34518.

[74] L. Xu, H. Xie, S.-Z. J. Qin, X. Tao, and F. Lee Wang, "Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment," 2023, *arXiv:2312.12148*.

[75] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-rank adaptation of large language models," 2021, *arXiv:2106.09685*.

[76] V. Lialin, V. Deshpande, X. Yao, and A. Rumshisky, "Scaling down to scale up: A guide to parameter-efficient fine-tuning," 2023, *arXiv:2303.15647*.

[77] Z. Fu, H. Yang, A. M. So, W. Lam, L. Bing, and N. Collier, "On the effectiveness of parameter-efficient fine-tuning," in *Proc. AAAI Conf. Artif. Intell.*, Jun. 2023, vol. 37, no. 11, pp. 12799–12807.

[78] W. Lu, R. K. Luu, and M. J. Buehler, "Fine-tuning large language models for domain adaptation: Exploration of training strategies, scaling, model merging and synergistic capabilities," 2024, *arXiv:2409.03444*.

[79] M. Fayazi, M. T. Taba, A. Tabatabavakili, E. Afshari, and R. Dreslinski, "FuNToM: Functional modeling of RF circuits using a neural network assisted two-port analysis method," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD)*, Oct. 2023, pp. 1–8.

[80] M. Hassanpourghadi, S. Su, R. A. Rasul, J. Liu, Q. Zhang, and M. S. Chen, "Circuit connectivity inspired neural network for analog mixed-signal functional modeling," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, Dec. 2021, pp. 505–510.

[81] A. F. Budak, M. Gandara, W. Shi, D. Z. Pan, N. Sun, and B. Liu, "An efficient analog circuit sizing method based on machine learning assisted global optimization," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 5, pp. 1209–1221, May 2022.

[82] S. Zhang, F. Yang, C. Yan, D. Zhou, and X. Zeng, "An efficient batch-constrained Bayesian optimization approach for analog circuit synthesis via multiobjective acquisition ensemble," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 1, pp. 1–14, Jan. 2022.

[83] H.-J. Lee, K.-J. Lee, Y. Choi, K. Lee, S. Kang, and J.-Y. Sim, "Trans-net: Knowledge-transferring analog circuit optimizer with a netlist-based circuit representation," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2024, pp. 1–2.

[84] K. Settaluri, Z. Liu, R. Khurana, A. Mirhaj, R. Jain, and B. Nikolić, "Automated design of analog circuits using reinforcement learning," in *Proc. IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, Oct. 2021, vol. 41, no. 9, pp. 2794–2807.

[85] S. Zhang, W. Lyu, F. Yang, C. Yan, D. Zhou, X. Zeng, and X. Hu, "An efficient multi-fidelity Bayesian optimization approach for analog circuit synthesis," in *Proc. 56th ACM/IEEE Design Autom. Conf. (DAC)*. New York, NY, USA: Association for Computing Machinery, Jun. 2019, pp. 1–6.

[86] J. Gao, W. Cao, and X. Zhang, "RoSE: Robust analog circuit parameter optimization with sampling-efficient reinforcement learning," in *Proc. 60th ACM/IEEE Design Autom. Conf. (DAC)*, Jul. 2023, pp. 1–6.

[87] T. Gu, J. Wang, Z. Bi, C. Yan, F. Yang, Y. Qin, T. Cui, and X. Zeng, "TSS-BO: Scalable Bayesian optimization for analog circuit sizing via truncated subspace sampling," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2024, pp. 1–6.

[88] Y. Oh, D. Kim, Y. H. Lee, and B. Hwang, "CRONuS: Circuit rapid optimization with neural simulator," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2024, pp. 1–6.

[89] P. Vaz, A. Gusmao, N. Horta, N. Lourenco, and R. Martins, "Speeding-up complex RF IC sizing optimizations with a process, voltage and temperature corner performance estimator based on ANNs," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2022, pp. 1570–1574.

[90] K.-E. Yang, C.-Y. Tsai, H.-H. Shen, C.-F. Chiang, F.-M. Tsai, C.-A. Wang, Y. Ting, C.-S. Yeh, and C.-T. Lai, "Trust-region method with deep reinforcement learning in analog design space exploration," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, Dec. 2021, pp. 1225–1230.

[91] B. Shook, P. Bhansali, C. Kashyap, C. Amin, and S. Joshi, "MLParest: Machine learning based parasitic estimation for custom circuit design," in *Proc. 57th ACM/IEEE Design Autom. Conf. (DAC)*, Jul. 2020, pp. 1–6.

[92] S. Hong, Y. Tae, D. Lee, G. Park, J. Lim, K. Cho, C. Jeong, M.-J. Park, S. Hong, and J. Han, "Analog circuit design automation via sequential RL agents and g_m/I_D methodology," *IEEE Access*, vol. 12, pp. 104473–104489, 2024.

[93] Y. Uhlmann, T. Moldenhauer, and J. Scheible, "Differentiable neural network surrogate models for g_m/I_D-based analog IC sizing optimization," in *Proc. ACM/IEEE 5th Workshop Mach. Learn. CAD (MLCAD)*, Sep. 2023, pp. 1–6.

[94] M. Choi, Y. Choi, K. Lee, and S. Kang, "Reinforcement learning-based analog circuit optimizer using gm/ID for sizing," in *Proc. 60th ACM/IEEE Design Autom. Conf. (DAC)*, Jul. 2023, pp. 1–6.

[95] Z. Chen, S. Meng, F. Yang, L. Shang, and X. Zeng, "MACRO: Multi-agent reinforcement learning-based cross-layer optimization of operational amplifier," in *Proc. 29th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2024, pp. 423–428.

[96] J. Bao, J. Zhang, Z. Huang, Z. Bi, X. Feng, X. Zeng, and Y. Lu, "Multiagent based reinforcement learning (MA-RL): An automated designer for complex analog circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 43, no. 12, pp. 4398–4411, Dec. 2024.

[97] 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 *Proc. 57th ACM/IEEE Design Autom. Conf. (DAC)*, Jul. 2020, pp. 1–6.

[98] Z. Zhao and L. Zhang, "Analog integrated circuit topology synthesis with deep reinforcement learning," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 12, pp. 5138–5151, Dec. 2022.

[99] A. Shahane, S. Swapna Manjiri, A. Jain, and S. Kumar, "Graph of circuits with gnn for exploring the optimal design space," in *Advances in Neural Information Processing Systems*, vol. 36, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., Red Hook, NY, USA: Curran Associates, 2023, pp. 6014–6025.

[100] H. Wang, J. Yang, H.-S. Lee, and S. Han, "Learning to design circuits," 2018, *arXiv:1812.02734*.

[101] Y. Li, Y. Lin, M. Madhusudan, A. Sharma, S. Sapatnekar, R. Harjani, and J. Hu, "A circuit attention network-based actor-critic learning approach to robust analog transistor sizing," in *Proc. ACM/IEEE 3rd Workshop Mach. Learn. CAD (MLCAD)*, Aug. 2021, pp. 1–6.

[102] Z. Wu and I. Savidis, "Comparative analysis of graph isomorphism and graph neural networks for analog hierarchy labeling," in *Proc. 25th Int. Symp. Quality Electron. Design (ISQED)*, Apr. 2024, pp. 1–7.

[103] R. Lyu, Y. Meng, A. Zhao, Z. Bi, K. Zhu, F. Yang, C. Yan, D. Zhou, and X. Zeng, "A study on exploring and exploiting the high-dimensional design space for analog circuit design automation: (Invited paper)," in *Proc. 29th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2024, pp. 671–678.

[104] Z. Chen, S. Meng, F. Yang, L. Shang, and X. Zeng, "TOTAL: Topology optimization of operational amplifier via reinforcement learning," in *Proc. 24th Int. Symp. Quality Electron. Design (ISQED)*, Apr. 2023, pp. 1–8.

[105] Z. Li and A. C. Carusone, "Design and optimization of low-dropout voltage regulator using relational graph neural network and reinforcement learning in open-source SKY130 process," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD)*, Oct. 2023, pp. 1–9.

[106] Z. Dong, W. Cao, M. Zhang, D. Tao, Y. Chen, and X. Zhang, "CktGNN: Circuit graph neural network for electronic design automation," in *Proc. 11th Int. Conf. Learn. Represent.*, Jan. 2023, Paper 2308.16406.

[107] J. Shen, F. Yang, L. Shang, C. Yan, Z. Bi, D. Zhou, and X. Zeng, "Topology optimization of operational amplifiers using a performance-aware representation," in *Proc. 2nd Int. Symp. Electron. Design Autom. (ISEDA)*, May 2024, pp. 166–170.

[108] D. Dai, Y. Sun, L. Dong, Y. Hao, S. Ma, Z. Sui, and F. Wei, "Why can GPT learn in-context? Language models implicitly perform gradient descent as meta-optimizers," 2022, *arXiv:2212.10559*.

[109] D. Vungarala, S. Alam, A. Ghosh, and S. Angizi, "SPICEPilot: Navigating SPICE code generation and simulation with AI guidance," 2024, *arXiv:2410.20553*.

[110] C. Liu, W. Chen, A. Peng, Y. Du, L. Du, and J. Yang, "AmpAgent: An LLM-based multi-agent system for multi-stage amplifier schematic design from literature for process and performance porting," 2024, *arXiv:2409.14739*.

[111] A. Mehradfar, X. Zhao, Y. Niu, S. Babakniya, M. Alesheikh, H. Aghasi, and S. Avestimehr, "AICircuit: A multi-level dataset and benchmark for AI-driven analog integrated circuit design," 2024, *arXiv:2407.18272*.

[112] P. Díaz-Lobo, G. Liñán-Cembrano, and J. M. de la Rosa, "On the use of artificial neural networks for the automated high-level design of $\Delta\Sigma$ modulators," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 71, no. 5, pp. 2006–2016, May 2024.

[113] E. A. Karahan, Z. Liu, and K. Sengupta, "Deep-learning-based inverse-designed millimeter-wave passives and power amplifiers," *IEEE J. Solid-State Circuits*, vol. 58, no. 11, pp. 3074–3088, Nov. 2023.

[114] J. Zhang, Z. Wei, K. Kang, and W.-Y. Yin, "Intelligent inverse designs of impedance matching circuits with generative adversarial network," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 43, no. 10, pp. 3171–3183, Oct. 2024.

[115] M. Wen, J. G. Kuba, R. Lin, W. Zhang, Y. Wen, J. Wang, and Y. Yang, "Multi-agent reinforcement learning is a sequence modeling problem," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, Jan. 2022, pp. 16509–16521.

DANIAL NOORI ZADEH (Student Member, IEEE) is currently pursuing the bachelor's degree in electrical and computer engineering with McMaster University. In 2025, he joined Synopsys Inc. as a SerDes IP Digital Verification Intern. He is passionate about using data-driven approaches to solve problems. He joined MacAnalog, in 2024, as a Research Assistant, he is developing machine learning methods to automate analog circuit design. In 2024, he was awarded a Partner in Teaching and Learning Grant to develop an AI-powered teaching team management platform, applying large language models (LLMs) for plagiarism detection with McMaster University. His research interests include high-speed mixed-signal and RF integrated circuits, information theory, and the applications of artificial intelligence (AI) in electronic design automation (EDA).

MOHAMED B. ELAMIEN (Member, IEEE) received the B.Sc. (Hons.) and M.Sc. degrees in electrical and computer engineering from the University of Sharjah, United Arab Emirates, in 2015 and 2017, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Calgary, Calgary, AB, Canada, in 2021, where his research focused on high-performance analog circuits for 5G communication systems. From 2021 to 2022, he was a Postdoctoral Associate with the University of Calgary, developing high-speed digitizers for astronomical telescopes. In 2022, he joined Synopsys Inc. as a Senior Analog and Mixed-Signal Circuit Design Engineer, working on the design of high-speed analog circuits for SerDes IPs. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, McMaster University, Hamilton, ON, Canada. He has authored or co-authored more than 30 journal and conference publications and has written a comprehensive book on CMOS Transconductors. His research interests include circuit theory, analog design automation, high-performance low-power analog, mixed-signal, and RF integrated circuits (ICs) for IoT and biomedical applications. He was a recipient of the Alberta Innovates Graduate Student Award-Technology Category, in 2020, the Alberta Graduate Excellence Scholarship, in 2019 and 2020, and the Schulich School of Engineering Graduate Student Research Excellence Award, in 2022. He serves as an Associate Editor for *International Journal of Circuit Theory and Applications* (Wiley).