Received 29 January 2024, accepted 10 February 2024, date of publication 19 February 2024, date of current version 29 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3367598

APPLIED RESEARCH

A Graph Attention Network Based System for Robust Analog Circuits' Structure Recognition Involving a Novel Data Augmentation Technique

ALI DEEB[1], MOHAMED SALEM[2], ABDALRAHMAN IBRAHIM[3], JOACHIM PICHLER[2], SERGII TKACHOV[2], KARAJ ANJEZA[2], FADI AL MACHOT[4], (Member, IEEE), AND KYANDOGHERE KYAMAKYA[1]

[1]Institute of Smart Systems Technologies, University of Klagenfurt, 9020 Klagenfurt, Austria
[2]Infineon Technologies Austria, 9500 Villach, Austria
[3]AGILOX Services GmbH, 4671 Neukirchen bei Lambach, Austria
[4]Faculty of Science and Technology, Norwegian University of Life Sciences (NMBU), 1430 Ås, Norway

Corresponding author: Ali Deeb (Ali.Deeb@aau.at)

This paper is funded by Infineon Technologies Austria.

ABSTRACT This paper addresses the issue of reliable and automated analog circuit's structure recognition (ACSR). First, it presents a comprehensive critical review of the related state-of-the-art. Then, essentially, this study proposes and validates a novel approach for realizing a dependable structure recognition system for analog circuits through a comprehensive ontology definition, some transformation tools, scene modeling through graph models, and then, by involving a graph attention neural network (GAT) model. Knowingly, identifying sub-circuits requires, in the brute-force mode, extensive screening of the numerous alternative substructures that can be constructed from the basic elements (i.e. the transistors) in presence, especially w.r.t. the actual connectivity topology amongst them; this is evidently a very complex and challenging endeavor. The relevant state-of-the-art has involved mostly unsupervised learning approaches to tackle this analog circuit's structure recognition-related challenging task, whereby they have reached a clear limitation of not (or only very hardly) reaching beyond 90% to 93% accuracy/precision. But in this work, we develop, for the first time, a comprehensive supervised-learning approach that demonstrates its clear superiority by enabling the reaching of a recognition accuracy or precision that is reliably in the range beyond 99% (or even 100%) for each subblock of the analog circuit. The supervised clustering approach developed and validated in this study consists of a comprehensive modeling and conceptual pipeline that is implemented around and through a Graph Attention Neural Network model, which ensures a reliable recognition of sub-blocks of analog circuits and their related internal adjacency connectivity topology. Besides the modeling pipeline, this study also develops a set of tools for mapping an analog circuit's schematics into a graph model. To overcome the limited and unbalanced samples for training the graph neural model, this study proposes a special novel augmentation strategy based on a graph sub-cropping technique. This augmentation technique is embedded in a smart stepwise augmentation protocol that leads at the end, through iterative additional dedicated training steps to a significant progressive increase of the recognition accuracy by the graph neural model until reaching more than 99%, better 100%, for each of the subblocks of the analog circuits as demonstrated in the demonstration case-study presented. Indeed, this paper's quintessence represents a significant breakthrough in view of the clear outperforming of the currently relevant related state-of-the-art.

INDEX TERMS Analog circuit's structure recognition, graph clustering, deep learning, graph attention neural networks, robust data augmentation.

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

I. INTRODUCTION

Analog and mixed-signal circuits form the foundation of many electronic devices in use today. As demand for smaller and faster devices grows, the need for higher-speed and lower-power consumption has become increasingly urgent. This has led to the use of more complex and powerful analog/mixed-signal (AMS) systems. However, as these systems become more complex, the design and verification tasks become more challenging and time-consuming. One of the main difficulties in working with analog and mixed-signal circuits is their complexity, which can make it challenging to understand the overall structure of the circuit and how it works [1]. In order to automate the design and verification of AMS circuits, it is essential to recognize the circuit structures used in AMS design. Machine learning has become an active research area in recognizing circuit structures in analog and mixed-signal circuits [1], [2]. Several approaches have been proposed and studied. Some common approaches include using a supervised learning algorithm to learn a set of features from a labeled dataset of circuits and then using those features to recognize circuits in new circuit's data [1]. Other approaches include unsupervised learning algorithms, which learn features from unlabeled data, and through transfer learning, which uses features learned from one dataset to recognize structures in a new dataset [3].

Identifying structures in a hierarchical schematic represents one of the most challenging and time-consuming tasks during the development of an analog-mixed-signal integrated circuit (AMS-IC). Manual structure identification requires a significant amount of domain-specific knowledge [3]. Currently, no commercially available technology can automatically detect structures at the schematic level. The goal of the research reported in this paper is to automatically perform structure recognition in analog circuits while ensuring an accuracy close to or higher than 99%. Therefore, we have used graph neural network approaches to solve the challenging issue of automated subblock detection within a given analog circuit, even across hierarchical borders without making any assumptions about the job beforehand. The experiments conducted in this paper provide results that confirm that the proposed approach, based on Graph Attention Networks, is very promising. Indeed, this work does automatically categorize the functional labels of the identified subblocks. These categories include, for the demonstration use-case, just to name a few, current mirror, differential pair, cascode current mirror, and other analog building blocks of the same level.

The notable contributions of this paper can be summarized around satisfactorily reaching the following core objectives:

1) Determining and discussing how important analog circuit's structure (i.e. subblock) recognition is in various important application domains within the AMS circuits engineering processes.
2) A comprehensive critical review of the related state-of-the-art.
3) A comprehensive ontology in the frame of the converting process of any analog circuit into a graph model, this coupled with a comprehensive modeling workflow that involves machine learning methods, in this case, graph neural networks, to reliably resolve the analog circuit's structure recognition problem or its equivalent task in the graph domain, that is the so-called "graph clustering" problem.
4) Suggest a comprehensive protocol for analog circuit structure (sub-blocks) recognition using Graph Attention Network (GAT). In the frame of a comprehensive holistic modeling protocol/methodology, we build and optimize a Graph model that represents the AMS circuit at hand.
5) Create and validate a novel dataset augmentation strategy that does enhance progressively the GAT-based graph clustering model's performance, this in consideration of practical real-world scenarios where initial real/native training datasets may be or are rather mostly very limited in size and unbalanced w.r.t. the possible subblock labels.
6) Suggest a comprehensive protocol/methodology to ensure a detection accuracy of 100% (or very close to 100%) for all subblocks of a given analog circuit's family (i.e. for a known list of possible subblocks).
7) Demonstrate through a conceptual and qualitative comparison with findings from related works the novelty and evident superiority of the overall concept developed in this paper.

Our overall modeling pipeline assumes that the analog circuit to be clustered will be available as a SPICE circuit netlist generated using Cadence Virtuoso, which is the tool used in analog circuits engineering processes. Cadence Virtuoso can produce a description of the circuit entity (the schematics in Cadence Virtuoso) in a text form called "netlist". Thus, a parser is needed to interpret the netlist and build out of it a "graph model" describing the circuit entity. This graph model, the output of the parser, is the one to be processed by a graph attention node classifier model.

In circuits design, a netlist is a description of the connectivity of an electronic circuit. A netlist contains amongst other things a list of the electronic components contained and a list of the nets connecting these components. A net is a collection of two or more interconnected components. A supervised learning-based analog substructure recognition concept has been designed and is comprehensively explained in this work. More than 99% clustering accuracy has been achieved, as validated by the extensive experiments conducted and reported in this paper. This clearly and significantly outperforms competing unsupervised learning approaches from related works. For example, just for illustration, a K-Means followed by (coupled to) a graph neural network (GCN) model was used in an unsupervised learning approach, see Ref [4], to perform structure recognition and find the subcircuits in an analog circuit. However, this last-named concept requires the (guessed or known) number of subcircuits as input to K-Means to achieve high accuracy, which is a drawback for automation.

In the core case study discussed in the paper, the initial dataset (processed and built using Cadence Virtuoso) was used for the first comprehensive initial training of the graph attention neural network (GAT) model of our novel concept. However the initial results, which were in the level of those of most related works, needed to be still significantly improved. Hence, a novel and smart augmentation technique has been developed for that purpose, this is based on what we call the "subgraph cropping method" (explained later in this paper), and the subsequent accuracy was effectively stepwise, progressively, and significantly improved. Essentially, as will be closely explained later below, the progressive novel dataset augmentation process is guided by a progressive label-aware dedicated step-wise dataset balancing strategy. The process is successful as it leads to achieving more than 99% accuracy (i.e. 100%) for the structure (i.e. subblocks) recognition endeavor.

This paper's remaining sections are structured as follows; The importance and related a requirements engineering for a comprehensive analog circuits' structure recognition are covered in Section II. A comprehensive critical review of the related state-of-the-art is provided in Section III. In Section IV, we comprehensively describe our novel method for subblock recognition and it is handled through its equivalent problem of "graph clustering", whereby a comprehensive ontology is developed for a comprehensive mapping of an analog circuit into an equivalent graph model. Thereby, a general methodology for building a "graph clustering" pipeline is introduced.

In the following section, Section V, the dataset collection, its transformation, and its general preprocessing are discussed. Section VI presents the concept of a Graph Attention neural network model (the so-called GAT) that we suggest for structure recognition, its design, and its validation. Now, addressing the challenge related to size limited and unbalanced datasets is addressed through one novel innovative data augmentation strategy that is coupled to the GAT model; this is explained in section VII. Further, the experiments' anatomy for the suggested pipeline is presented in section IX. Section X presents and discusses a global qualitative performance evaluation of the suggested GAT-based model when compared to the most relevant recent related works. Finally, Section XI contains comprehensive concluding remarks describing the quintessence of the research reported in this paper.

II. IMPORTANCE AND REQUIREMENTS OVERVIEW OF AN ANALOG CIRCUITS STRUCTURE RECOGNITION

Analog circuits can be viewed as hierarchical structures having different levels. At each level of the hierarchy, one has different building blocks or entities that carry specific names depending on their respective pattern. Thus, every building block of a particular level is made up of lower-level building blocks. And this repeats so all the way down up to the very lowest-level consisting of transistors and other primitives (e.g.: nets, passive elements, diodes, etc.) [2], [4], and [5]. Therefore, analog circuits structure recognition enables a greater understanding of the properties and requirements of a given complex circuit (viewed from various level-related perspectives). Indeed, being able to classify analog circuits and, more specifically, recognize their internal structure without human supervision/intervention does accelerate several processes in the semiconductor industry. For example, a layout engineer could benefit from the structure recognition for layout routing of the subcircuits without inspecting the design in simulation. These work elements help reduce development time. Another practical use is to detect subblocks within a given analog circuit for symmetry detection and component placement in layouts [2]. In general, an efficient structure recognition of analog circuits and of the related topology does significantly impact a lot of semiconductor applications. Figure 1 presents a representative list of the many use-cases (within the circuits engineering business) for which analog subblocks recognition is very needed and is of precious value in the context of analog/mixed signals circuits' engineering.

An analog circuits designer needs knowledge about the subblocks' identities to appropriately translate the design into a layout via layout placement and routing. Therefore, knowing the names of the subblocks and, therefore, their functions will greatly speed up layout simulation and coupling in automated analog circuit generators. Also, in other contexts, to get precise results for analog circuit design, one needs to solve differential equations through simulators. However, doing so either requires a lot of computing time or relies on already pre-solved differential equations, which reduces the accuracy margin. For maintaining a high level of accuracy, analog circuit structure recognition may assist the simulator in finding a better selection of the differential equations that have already been solved and thus support maintaining a high level of accuracy. This is done in much less time than it would take to obtain a precise solution. Analog circuit structure recognition further helps in the so-called "analog block-level identification," which is helpful for block-level verification and thereby lessens the workload of the verification engineer. Analog circuit designers may also gain from understanding the subblocks of the circuit or schematic by selecting the proper test bench to produce stimuli, carry out verification, and carry out additional measurements. Guaranteeing the circuit's functionality at the IP level lowers the amount of work the designer needs.

The importance of structure recognition being so far clear, let us formulate now a minimum requirements engineering dossier (i.e. specification book) for a robust structure recognition model for analog circuits. A genuinely robust analog circuit structure recognition system must meet the following 10 requirements (RQs):

1) REQ-1 (Accuracy Baseline – WITHOUT POST-PROCESSING): The core model (essentially neural model) should guarantee a structure recognition

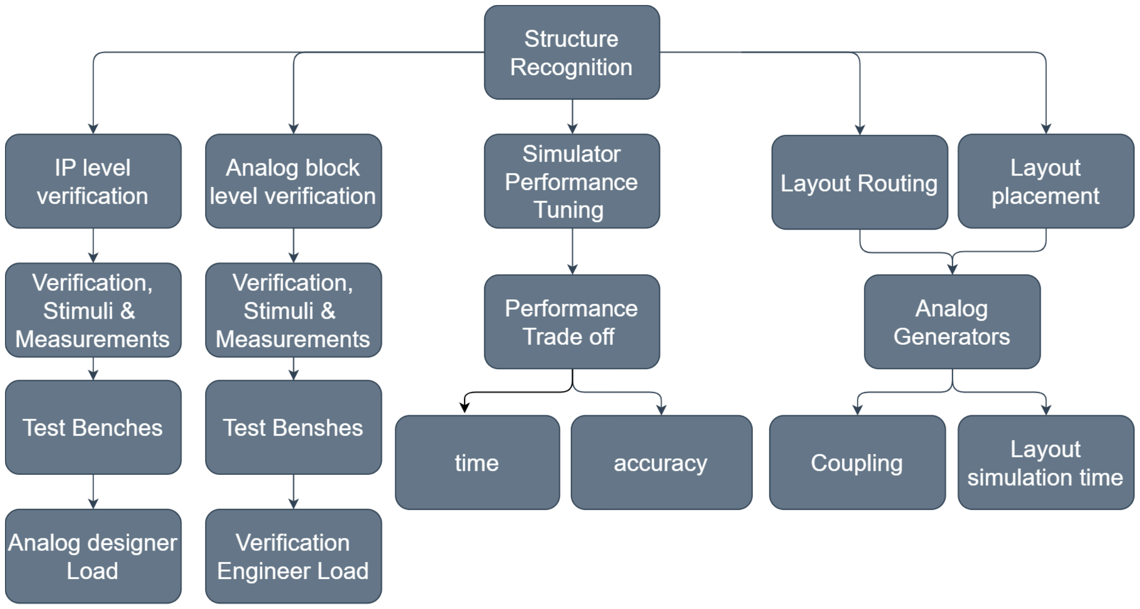


FIGURE 1. Flow diagram of structure recognition importance.

accuracy of at least 97%, with a target to reach an optimal 100% accuracy for all subblocks, including challenging cases such as those outlined in REQ-8. The methodology to ensure reaching the target performance through the core model must be explicitly presented.

2) REQ-2 (Topology Variability Robustness): The model should maintain a high recognition robustness across various circuit topologies, encompassing differences in component values, placements, and wiring configurations. Specific common topologies the model should recognize must be identified.
3) REQ-3 (Transistor Type Variability Robustness and MULTI-NODES TRANSISTOR REPRESENTATION): The model ensure high robustness when faced with circuits that utilize different or mixed transistor types (e.g., bipolar, NMOS, and PMOS) within the same subblock. Future-oriented transistor types should also be taken into consideration. Hereby, it is important to check with how many nodes a transistor is represented. Essentially each transistor shall be represented by multiple nodes in the equivalent graph model. This has also an impact on the vertical scalability described in REQ-5. All models representing a transistor with only one node will surely be limited w.r.t. REQ-5.
4) REQ-4 (Dataset Resilience): The model and subsequent additional concepts should be designed to handle dataset challenges like limited size and "subblock labels"-related imbalance. For example, it should integrate some advanced augmentation techniques to mitigate these challenges.
5) REQ-5 (Hierarchical Scalability): The model must be applicable across multiple levels of analog circuits, from the simplest to the most complex. It should adeptly handle multi-level hierarchical circuits where sub-blocks might have intricate internal sub-circuits. (Eventually, a proof of concept must validate performance on at least the four initial levels).
6) REQ-6 (Standardized Representation of entities of any level): Advocate for compatibility with a standardized representation of circuits, or ensure the model is adaptable to widely accepted methods. Special focus should be on the representation of transistors and high-level subblocks, addressing issues like the number of nodes required for representation. The issue is that entities of all levels must be capable of being represented with multiple nodes.
7) REQ-7 (Size Scalability): The model should be versatile enough to analyze circuits of varied sizes, from those with a handful of components to those with thousands. The model shall be capable of handling varying input-graph sizes.
8) REQ-8 (Similar Topology Handling): The model should be capable at recognizing and distinguishing between multiple subcircuits with that have the same label (i.e. similar topologies) within a given circuit, thus avoiding errors like incorrect clustering or merging of distinct sub-circuits.
9) REQ-9 (Adaptive Training): The model must support progressive "smart/context-aware" supervised learning, allowing performance improvements over time as more data or feedback is available. This is coupled to an adaptive/smart/ context-aware dataset augmentation.
10) REQ-10 (Model Extendibility - a.k.a. "HORIZONTAL SCALABILITY"): Ensure the model can be progressively trained to incorporate new subblock labels or patterns. This includes adding new layers, nodes, or patterns, as well as the potential to forget;

or deprioritize obsolete patterns. We call this "HORIZONTAL SCALABILITY". But the introduction of a new sub-block requires the adaptive training mentioned in REQ-9. Thus, a model that does not fulfill REQ-9 cannot satisfy REQ-10 either.

11) REQ-11 (Operational Environment Compatibility): The model should be compatible with prevalent EDA (electronic design automation) tools and platforms and be efficient enough for real-time or near-real-time operation during circuit design processes. This requirement is for me of low priority. This requirement becomes only relevant if the model has proven to satisfy first all 10 previous requirements. The model development in the "RESEARCH" phase tries to satisfy all 10 requirements first. If this succeeds, then for the "ENGINEERING" phase, for the application in specific industrial processes later on, this requirement (REQ-11) becomes then relevant.

Ensuring a detection high accuracy (i.e., higher than 97%) for an automated analog circuit structure recognition is not trivial. According to the related works, it is an extremely difficult task to reliably tackle [3], [6]. Still, it is an industrial requirement in view, amongst others, of the various use-cases (see Figure 1) needing this functionality. But in this work, it is crucial to put on the table a comprehensive methodology/protocol that shows and validates that a 100% detection accuracy can be reached for all subblocks of the analog circuit even for the case described in REQ-8 above.

III. COMPREHENSIVE CRITICAL REVIEW OF THE RELATED STATE-OF-THE-ART

Knowingly, the analog design and verification processes are time-consuming and require tremendous effort and high human experience (e.g., running test simulations, using behavior models for block functionality checks, and setting proper model test benches) [7]. These two last-named processes need specific design knowledge to ease the identification of structures, sub-structures, and related features. Most of this knowledge production is done manually by virtue of experienced analog design engineers [4], [8], [9]. Because of these needed experiences, several research efforts have started focusing on the automation structure and/or substructure recognition/identification by using different procedures.

In this section, we review related work in AMS schematic structure recognition, focusing on the strengths and weaknesses of existing approaches. Several approaches have been proposed to address the challenge of recognizing the complex structure of analog-mixed-signal (AMS) circuits. These approaches include several methods, which we will discuss as follows:

A. TEMPLATE-BASED METHODS

The success of Template-based methods [5], [10], or Library-based methods, depends on including all possible designs in one library, or in other words, building a comprehensive library of structures.

In this case, structure recognition means searching for the exact structure in this library. when it is found, then the procedure is successful. However, any slight change in the design of the circuit structure, which is a common thing and sometimes necessary, leads to a recognition failure. therefore, this method is not robust since the matching with the library elements is strict, and there is no sense of measuring the accuracy of this method for new test samples.

B. SEARCH-ORIENTED BASED METHODS

The works presented in [11] and [12], have introduced search-oriented algorithms for sub-circuits recognition by involving subgraph isomorphism [11] and pattern matching, [12]. These methods were efficient in subcircuit recognition but are very time-consuming, especially for large circuits, as they use trial and error techniques. The circuits mentioned in these papers were clustered correctly, but no accuracy was stated for a test dataset, and clustering time was the evaluation metric for these methods.

C. GRAPH-BASED METHODS METHODS

Reference [13] proposed a circuit graph coding to discover and verify subcircuits. This coding identifies each subcircuit by generating a unique code for each one of the subcircuits. This unique code depends on the fact that each transistor has specific features and is connected to different nodes. the drawback of this approach is that the circuit partitioning is done before the recognition. Hence, the post-processing is a must to ensure high accuracy. Although the approach was applied on large circuits, which include up to 15000 transistors, no accuracy was mentioned for a test dataset.

In [14], the so-called "bipartite graph labeling" algorithm for the subcircuit recognition was suggested and combined with the "probabilistic match assignment algorithm", [15], [16], for recognizing sub-circuits. In essence, a probabilistic method has also been integrated with a nonlinear function that calculates a structure matrix for matching between devices and nets in different levels (e.g., circuits and subcircuits). However, the structure recognition accuracy varies from 76% to 100%, depending on the test schematic.

The primary concept technique, developed in [15], involves creating match matrices, which define how the subcircuit and circuit nets and devices correspond to one another, respectively. During the optimization phase, the matrix elements are calculated by a productive bipartite graph labeling algorithm and the GA matching method. Besides in [16], an approach to recognize graphs based on optimization for the structure recognition problem. The error propagation and soft (delayed) decision-making concepts from pattern recognition theory are combined with the self-annealing optimization strategy in this approach. In both references, runtime was the evaluation metric.

The probabilistic match assignment method was faster than search-based approaches, but it struggled to keep up with the requirements of CAD software containing a template library of a large number of large circuits, which necessitates more user/tuning effort to reduce the computation cost of the probabilistic match assignment method [17]. In order to overcome the high computation complexity struggle, [18] used a nonlinear graph optimization strategy of second-order terms instead of first-order linear optimization techniques of references [15] and [16].

D. UNSUPERVISED LEARNING-BASED METHODS

The topological (structural) properties of analog circuits can now be automatically identified and extracted thanks to a revolutionary technique presented in [19]. The study offers unsupervised feature extraction algorithms. The following key issues are addressed by the method: hierarchical structures, repeated structures, and overlapping amongst building blocks. Thereby 34 modern analog circuits were the subject of experiments that examine feature extraction. The accuracy and efficiency of operations like circuit synthesis, scaling, and design knowledge description depends on the ability to find structural features. Nevertheless no concrete accuracy was provided for a test dataset.

Using GCN proprieties the studies in [20] play a key role in circuit elements classification which, in turn, sums up these elements to make higher-level known building blocks (e.g., current mirror, differential pair) to be used in various circuitry applications. Also, [21] presents a structure recognition algorithm that is "library-free" to extract the basic blocks of any circuit relying on some rules given to the algorithm.

In [22], a system that bridges schematic and layout creation is presented. It achieves this by identifying key subcircuit structures using Graphical Convolutional Neural Networks (GCNNs) and an unsupervised graph clustering approach.. To our knowledge, this framework fully automates clustering. It achieves over 90% accuracy in under 1 second across six analog circuits of varying size and complexity. The maximum accuracy, which varies from 91.3% to 100%, was the only accuracy provided. Meanwhile, [3] proposed a "K-Means + GCN"-based recognition framework for circuit recognition. The experimental findings demonstrate the potential of the suggested approach based on the K-Means and GCNs algorithm.

E. SUPERVISED LEARNING-BASED METHODS

Reference [23] uses a convolutional neural network (CNN) for the identification and recognition of building blocks seeking circuit layout automation without considering human inputs or experience. The accuracy reached was 97.2% ± 1.4%. Similar to [24], a Graph Convolutional Network (GCN) based technique is used for structure identification using a transistor-level netlist of the circuit. A unique strategy is presented in [24], for clustering analog circuits utilizing library-based primitive matching and GCN-based machine learning. The GCN-based technique can handle diverse subblock design topologies, as shown on two hand-crafted circuits. The technology is scalable and successfully clusters circuits into subblocks and creates circuit hierarchy trees. The accuracy of clustering varies from 79.8% to 98.2% depending on the circuit type. However, postprocessing was performed to increase the accuracy to 100% for all test cases. Postprocessing was used to avoid overfitting and to help the GCN model with the limited dataset. Two classes of heuristics were used in the final stage after the GCN. The first class uses a graph-based heuristics in which nodes that belong to the same channel-connected component (CCC), a group of transistors connected at the sources and drains, are assigned to a subblock. We then identify all primitives within a CCC using graph-based approaches to extract primitives within a subblock. All primitives in a CCC that are integral parts of a subblock (e.g., a differential pair in an OTA) are added to the hierarchy tree at the same level. An independent primitive, which can be considered an independent entity (e.g., an input buffer for an oscillator, as occurs in our phased array example), is separated and listed as an independent primitive in the hierarchy tree. The second class refers to circuit-specific knowledge and is based on information about the connections to the input/output ports.

Table 1 shows an overview of the most relevant related works with respect to an assessment of how far they do satisfy or not the eleven hard requirements formulated and outlined in section II (see Requirements Engineering dossier consisting of 11 Requirements)

Table 1's main message is fundamentally and factually a form of "gap analysis" for all works that performed structure recognition regardless of the method. It is true that none of the relevant works entirely meet the specified requirements dossier. However, and this is the main goal, the novel concept developed and validated in this paper is expected to fulfill all of the defined requirements. This will highlight its innovative nature in comparison to the most current and relevant state-of-the-art.

Reference [3] mentions an accuracy of over 95%, but it doesn't reach the 97% baseline. Hence, REQ-1 was not fulfilled. Regarding REQ-2, the paper does discuss the use of different schematics. The used model can recognize different circuit topologies and adapt to different configurations. The paper explicitly states that MOSFETs were considered, without mention of other transistor types like bipolar, NMOS, or PMOS. Therefore, REQ-3 was not met. Regarding REQ-4, there is no clear mention of the model handling dataset challenges. This paper fulfills REQ-5 since the used scheme appears to handle hierarchical schematics. Regarding REQ-6, it is unclear if the model supports a standardized representation of circuits. The paper doesn't explicitly mention varied sizes of circuits, and the model is GCN-based. Hence, REQ- 7 was not met. While the methodology identifies structures, it is unclear if it can distinguish between similar topologies within a given circuit. Therefore, REQ-8 was not fulfilled. Regarding REQ-9: the paper doesn't specify progressive supervised learning based on a step-wise augmentation technique, and the introduction

TABLE 1. An overview of the most relevant related works w.r.t. an assessment of how far they do satisfy the five requirements formulated and outlined in section II (see Requirements Engineering dossier.

Most relevant Works	REQ-1	REQ-2	REQ-3	REQ-4	REQ-5	REQ-6	REQ-7	REQ-8	REQ-9	REQ-10	REQ-11
Rituj et al. [3]	Yes	Yes	No	No	Yes	No	No	No	No	No	Yes, possibly
Meissner et al. [10]	No	Yes	No	No	No	No	No	Yes	No	No	No
Massier et al. [5]	No	No	Yes	No	Yes	No	Yes	Yes	No	No	Yes, possibly
Ohlrich et al. [11]	No	Yes	Yes	No	No	No	Yes	Yes	No	Yes	Yes, possibly
Pelz et al. [12]	No	Yes	yes	No	Yes	No	No	Yes	No	Yes, possibly	Yes, possibly
Huang et al. [13]	No	Yes	Yes	No	No	Yes	Yes	Yes	No	No	Yes, possibly
Rubanov et al. [14]	Yes	No	No	No	No	No	Yes	Yes	No	No	Yes, possibly
Rubanov et al. [15]	Yes	Yes	No	No	Yes	No	Yes	Yes	No	No	Yes
Rubanov et al. [16]	Yes	Yes	No	No	Yes	No	Yes	Yes	No	No	Yes, possibly
Conn et al. [17]	No	Yes	No	No	No	No	Yes	Yes	No	No	No
Rubanov et al. [18]	Yes	Yes	No	No	No	No	Yes	Yes	No	No	Yes
Li et al. [19]	Yes	Yes	No	No	Yes	No	No	Yes	No	No	No
Neuner et al. [21]	Yes	No	No	No	No	No	Yes	Yes	No	No	No
Settaluri et al. [22]	No	Yes	No	No	Yes	No	No	Yes	No	No	No
Liou et al. [23]	Yes	Yes	Yes	No	No	Yes, possibly	Yes, possibly	No	No	No	Yes
Kunal et al. [24]	No	Yes	No	No	No	No	No	No	No	No	No

REQ-1: Accuracy Baseline
REQ-2: Topology Variability Robustness
REQ-3:Transistor Type Variability Robustness
REQ-4: Dataset Resilience
REQ-5: Hierarchical Scalability
REQ-6: Standardized Representation
REQ-7: Size Scalability
REQ-8: Similar Topology Handling
REQ-9: Adaptive Training
REQ-10: Model Extendibility
REQ-11: Operational Environment Compatibility

of a new subblock requires the adaptive training mentioned in REQ-9. Thus, a model that does not fulfill REQ-9 cannot satisfy REQ-10. Regarding REQ-11, the compatibility with prevalent EDA tools or platforms is not mentioned nor discussed.

By applying the same judgment on all other related references that performed analog circuits' structure recognition, Table 1 has been filled, and the justifications of each of the answers posted in Appendix C.

IV. GENERAL METHODOLOGY FOR REALISING OUR STRUCTURE RECOGNITION PIPELINE

In this paper, we develop and validate a robust novel neural ML model to solve the analog circuits' structure recognition problem. Our overall pipeline assumes that the analog circuit to be clustered will be available as a SPICE netlist, as Cadence Virtuoso is the tool used in analog circuits engineering processes. Cadence Virtuoso can produce a description of the circuit entity (the schematics in Cadence Virtuoso) in a text form called "netlist". Thus, a parser is needed to interpret the netlist and build out of it a "graph model" describing the circuit's entities and how they are interconnected. This graph model, the output of the parser, is the one to be processed by a graph attention neural network based node classifier model.

The parser, in its essential role of processing the netlist and generating a graph model, is fundamentally reliant on a comprehensive ontology. This ontology encompasses various aspects, including how to appropriately represent each component of the analog circuit within the graph model. Moreover, the challenge extends beyond determining the representation of individual electronic elements (e.g., 3-pin transistors, 5-pin transistors, diodes, passives like resistors, capacitors, inductors, etc., as well as nets) within the circuit, and encompasses defining how to articulate the distinctive characteristics associated with each of these elements.

The comprehensive ontology is the foundation of the pipeline. Table 2 shows the most common primitives that can be found in a schematic. Nevertheless, passive elements are not considered for the proof of concept of this paper and can be considered in a later generalized approach. In this paper we chose to use solely transistors and represent each of them with several nodes: Transistor NMOS-1 ((3ports, 3 nodes)), Transistor PMOS-1 ((3ports, 3 nodes)), Bipolar Transistor ((3ports, 3 nodes)), and Nets (1 node) for our proof of concept case-study in this paper.

Figure 2.a displays a schematic illustrating a Level L2 circuit constructed using the Cadence Virtuoso design tool. In Figure 2.b, you can observe an equivalent graph, where each transistor is denoted by three nodes, while each net is symbolized by a single node. For instance, the transistor M0 is indicated by nodes M0s, M0g, and M0d, and it is linked to net7, net6, and net3, respectively.

In circuit design and modeling, a netlist is a description of the structure and connectivity within an electronic circuit. Indeed, a netlist is a list of all the components and inter-connections in an electronic circuit which define the electrical connectivity and relationships between components. The netlist includes information about components such as transistors and diodes, as well as passive elements like resistors, capacitors, and inductors, including their electrical characteristics and connections to other compo-

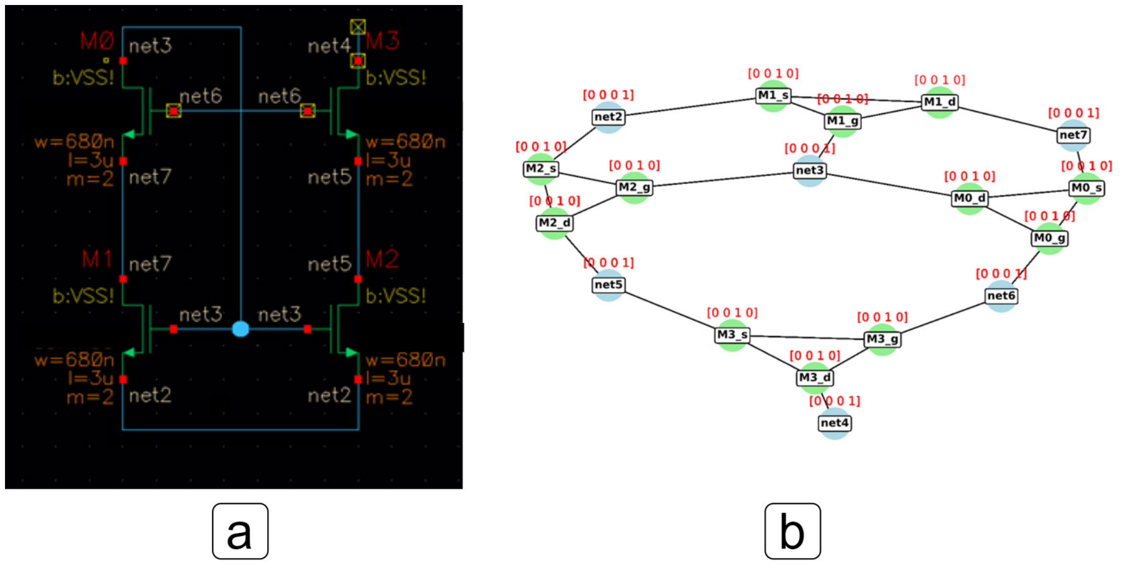


FIGURE 2. **a. A schematic representing one illustrative L2 circuit (i.e. an analog circuit of level 2) and exported from the Cadence Design tool as a netlist b, the equivalent graph where each one of the transistors is represented by three nodes, and each net is represented by one node.**

TABLE 2. **Primitives which can be found in schematics and the number of needed nodes to represent them in the graph domain.**

Primitive name	Number of representing graph nodes
Transistor NMOS-1 ((3ports))	3
Transistor NMOS-2 ((5ports))	5
Transistor PMOS-1 ((3ports))	3
Transistor PMOS-2 ((5ports))	5
Bipolar Transistor ((3ports))	3
Diode	2
Resistor	2
Inductor	2
Capacitor	2
Voltage Source	2
Current Source	2
Nets	1

TABLE 3. **Four unique types of elements are considered for this clustering problem.**

Node Type	Feature Vector
BIPOLAR	[1 0 0 0]
NMOS [CMOS]	[0 1 0 0]
PMOS [CMOS]	[0 0 1 0]
N-Net	[0 0 0 1]

nents. In an analog circuit netlist, passive elements like resistors, capacitors, and inductors are represented by their electrical properties and connections. The netlist includes information about the value and type of each passive element, such as the resistance value of a resistor, the capacitance value of a capacitor, or the inductance value of an inductor. The netlist also specifies the connections between each passive element and other components in the circuit.

Similarly, transistors are represented by their electrical properties and the connections between their three (or more) terminals. The netlist may specify the type of transistor, such as NPN or PNP or NMOS/PMOS, and provide information about the transistor's electrical characteristics such as the maximum current rating and the voltage gain. The netlist also defines the connections between the transistor and other components in the circuit.

Figure 4 shows all the tested features (i.e. features finally selected to be useful after extensive tests) in order to represent the nodes in the best way possible. However, the one-hot

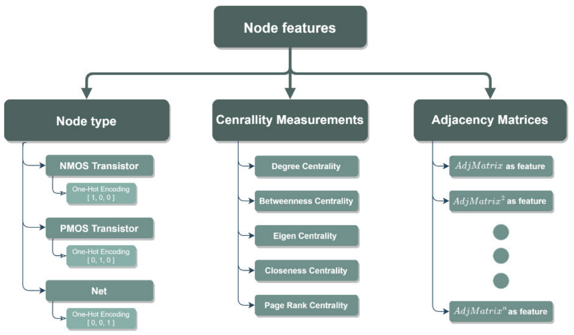


FIGURE 3. **The tested node features of the equivalent graph are one-hot encoding, the centrality features, and the adjacency matrices features.**

encoding features were sufficient to reach the needed high accuracy.

For this problem, we consider four unique types of elements, and we chose one-hot-encoding to represent them as shown in Table 3. These encodings are used to constuct the feature matrix as each one of them is the feature vector for the node it represents.

Figure 3.a is the schematic's equivalent graph (of the circuit displayed in Figure 2.a) and it is represented in Figure 3.b by its related adjacency matrix and features matrix. The chosen features (at the end, found to be the best and sufficient), besides the adjacency matrix, are the one-hot encoding for both transistors and nets (see Fig.3).

The structure recognition concept suggested in this paper is centered around a comprehensive ontology, machine learning techniques, and a spatial graph neural network processor. Figure 5 describes the overall system pipeline architecture suggested in this work. Here, the input analog circuit's schematic is first exported as a netlist in Cadence virtuoso. Then, a parser that we have developed, integrating the knowledge provided by the comprehensive ontology described above, converts the netlist file into a graph model comprising an adjacency matrix and a feature matrix.

Furthermore, the graph model is used as an INPUT of a graph attention neural model GAT model, which shall cluster (after training) the analog circuit sample (represented by the corresponding input graph model) into the building subblocks contained that corresponding analog circuit.

A GAT neural model is indeed an excellent choice compared to other types of graph neural networks and various other machine learning tools. A Graph Attention Network (GAT) is a specific type of graph neural network that effectively models the structure and relationships between components in a circuit. GATs can learn to focus on different components and connections in a circuit, capturing complex dependencies and interactions among them. GATs are particularly well-suited for recognizing the structural patterns in analog circuits, as they can handle variable-size inputs and capture the hierarchical structure of circuits. For example, a GAT can learn to attend to individual components and their connections, and then aggregate this information to make higher-level decisions about the circuit's overall structure. Compared to other types of graph neural networks, such as graph convolutional networks (GCNs), GATs have several advantages for the task of analog circuit structure recognition. First, GATs can handle variable-sized graphs, while GCNs require fixed-size inputs. This is important because the size of an analog circuit can vary greatly depending on its complexity, and GCNs would require padding or truncating the inputs to a fixed size. Second, GATs can capture long-range dependencies in a circuit more effectively than GCNs.

This is because GATs use attention mechanisms that allow the network to learn to attend to different components and connections in the circuit, regardless of their distance from each other. In contrast, GCNs typically only capture information from a node's immediate neighbors. In a previous work [25], we did use GCN for solving a circuit classification problem and could see and feel this limitation of GCNs.

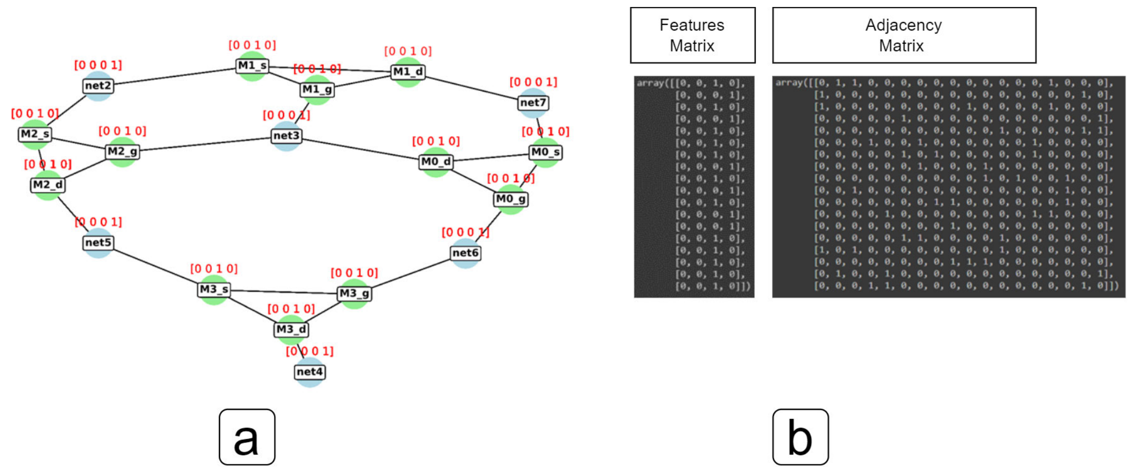
In summary, a graph attention network is a good choice to solve the structure recognition problem of analog circuits because it can handle variable-sized inputs, capture long-range dependencies, and is robust to noise and variability in the circuit components. These advantages make GATs a powerful tool for solving this challenging problem.

A. PROOF OF CONCEPT SCENARIO

This work addresses a structure (subblock) recognition problem for analog circuits. Contrary to some most prominent related work which involved some unsupervised learning components, in this work our approach to solve this problem based fully on supervised learning. This our approach is centered around a graph attention network GAT. This requires, amongst others, an appropriately labeled dataset for proper training. The dataset used in the proof of concept is comprehensively explained in Section V.

In view of the very comprehensive Specification Book presented (see requirements engineering: RQ-1, up to RQ-11) in Section II above, the best model to tackle the challenging endeavor addressed in this paper should fulfill satisfactorily the formulated 11 requirements. In the following, we briefly explain, from a philosophical and conceptual perspective, why and how far our concept developed/suggested in this paper will/does fulfill almost all of the 11 tough requirements of the specification book; this underscore how far it does significantly enrich the related state-of-the-art:

1) Consider REQUIREMENT-1 - A comprehensive protocol showing how an accuracy of 100% can be reached for all known subblocks within analog circuit, eventually progressively: this paper shows and demonstrate that our concept does fully satisfy this requirement as shown in section VIII.
2) Consider REQUIREMENT-2 - A high robustness w.r.t. topology/architecture variations within circuits/subblocks having the same label: this paper clearly demonstrates, through the proof of concept study, that this requirement is fully satisfied as shown in section VIII.
3) Consider REQUIREMENT-3 - A high robustness to variations involving different transistor types: this paper clearly demonstrates, through the proof of concept study, that this requirement is fully satisfied. See section IV-C1.
4) Consider REQUIREMENT-4 - A high robustness to most pervasive real-world imperfections of the training datasets, namely w.r.t. limited dataset size, and strong dataset unbalance: this paper clearly demonstrates, through the proof of concept study, that this requirement is fully satisfied. See Section VII.
5) Consider REQUIREMENTS-5 - Applicability of the concept (structure recognition model) of this paper to all levels (from the lowest to the highest ones) of analog circuits IPs (intellectual property): this paper has a dedicated sub-section that explains thoroughly, at a conceptual level, how the ontology and the comprehensive protocol of this paper can be applied, after respective simple adaptations, to all layers of an industrial IP stack of analog circuits (i.e. mixed signal circuits). We have called this requirement "vertical scaling". For a multi-level analog circuit (e.g.., a circuits covering the levels 1, 2, 3, and 4, a straightforward procedure is to organize the clustering of subblocks structures



```
array([[0, 0, 1, 0],
       [0, 0, 0, 1],
       [0, 0, 1, 0],
       [0, 0, 0, 1],
       [0, 0, 1, 0],
       [0, 0, 1, 0],
       [0, 0, 1, 0],
       [0, 0, 0, 1],
       [0, 0, 1, 0],
       [0, 0, 0, 1],
       [0, 0, 1, 0],
       [0, 0, 0, 1],
       [0, 0, 0, 1],
       [0, 0, 1, 0],
       [0, 0, 1, 0],
       [0, 0, 1, 0],
       [0, 0, 1, 0],
       [0, 0, 1, 0]])
```

```
array([[0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
       [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
       [1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0],
       [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1],
       [0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0],
       [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0],
       [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0],
       [1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0],
       [0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
       [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]])
```

FIGURE 4. **The schematic equivalent graph is represented by the adjacency matrix and the features matrix. The chosen feature are the one hot encoding for the used transistors and nets.**

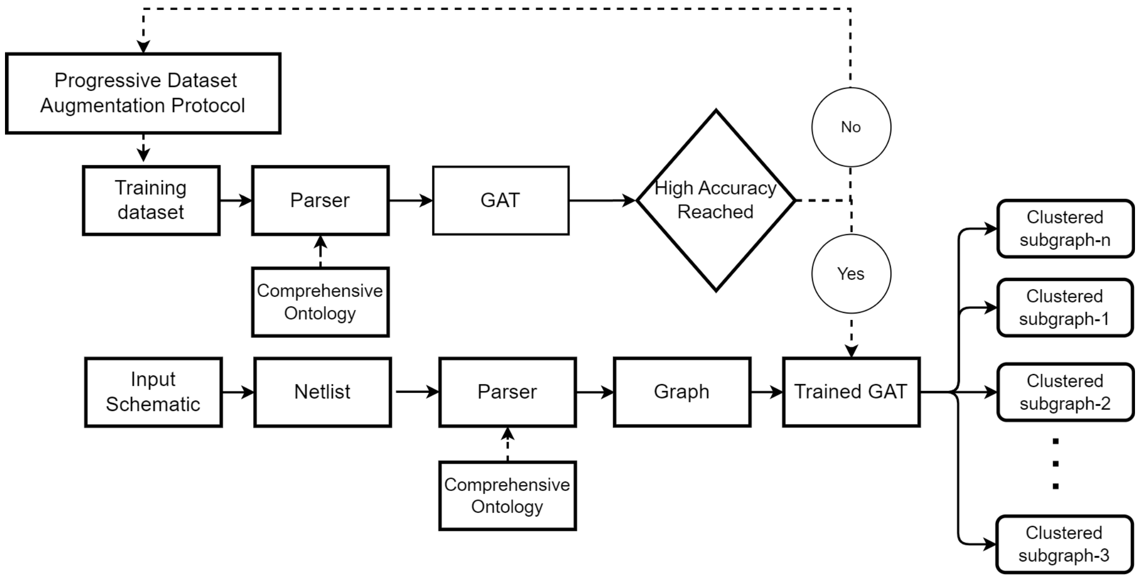


FIGURE 5. **Overall system pipeline of the developed Analog Circuit clustering GAT-based system: The progressive dataset augmentation protocol is applied to the training dataset, which is transformed into the graph domain using a parser that considers a comprehensive ontology until high accuracy is reached after performing the needed iterations. When the GCN model is trained, each test schematic is fed to the final pipeline as follows; An input schematic is exported as a netlist by cadence Virtuoso and then converted to a graph model through a parser we have developed. Then, the equivalent graph model is fed into the trained GAT to be clustered into subgraphs. Each one of these subgraphs represents an analog subblock.**

in steps, level-to-level, from the lowest to the highest level. This lastly described procedure may be justified by the fact that the analog circuit's schematics is generally (if not always) provided in form of a level-0 circuit. Another possible/alternative procedure is to identify at once the hierarchical structure, covering all four of the analog circuit. The concept presented in this paper, as it is centered around a GAT model, can also be adapted (w.r.t. to both ontology and protocol) and accommodated to implement this named alternative procedure for detecting the hierarchical structure. A justification for this capability is the fact that GATs are particularly suited for the structure recognition problem of analog circuits as they can knowingly handle variable-size inputs and can capture the hierarchical structure of analog circuits.

6) Consider REQUIREMENT-6 - A comprehensive representation, i.e. comprehensive ontology, for all basic electronic components and all subblocks at all levels of the analog circuits' hierarchy where the transistor in all subblocks of higher levels must be represented by three nodes. As already indicated in the comments above on REQ-5, this requirement is fully satisfied in the work presented in this paper.

7) Consider REQUIREMENT-7 - A scalability of the model w.r.t. the size of the input analog circuit to be analyzed. As already indicated in one of the comments above, the GAT model, which is central piece of our model and concept, can knowingly handle variable-size inputs. This is an important capability of high practical relevance. Thus, the concept suggested in this paper does fully satisfy this requirement.
8) Consider REQUIREMENT-8 - Capability to perform well, even when there are multiple sub-circuits with similar topologies in the input circuit. This is a very tough requirement that most related works fail to solve. But the case study presented in this paper, proof-concept, shows experimentally that our concept and the model therein does fully solve this requirement too.
9) Consider REQUIREMENT-9 - Capability for the model to be progressively (extended/tuned) trained to improve performance through a supervised learning process: this paper has a dedicated subsection where the innovative protocol/methodology is comprehensively presented, which solves this concern with full satisfaction, as also demonstrated by the different relative performance results. Thus, this requirement is also fully solved in this paper.
10) Consider REQUIREMENT-10 - Capability for the model to be progressively (extended/tuned) trained to integrate new (previously not considered in the previous/current GAT model training) subblock (s) through a supervised learning process: in this paper, a dedicated sub-section, where a comprehensive conceptual protocol is presented to solve this issue. We have called this requirement "horizontal scaling". Thus this requirement is also fully solved in this paper. See Subsection VII-D.
11) Consider REQUIREMENT-10 - (Operational Environment Compatibility): our model is compatible with prevalent EDA (electronic design automation) tools and platforms. It is efficient enough for real-time or near-real-time operations during circuit design processes.

B. HIERARCHICAL STRUCTURES AND LEVELS DEFINITION

In Analog/Mixed-Signal (AMS) design, abstraction is the process of hiding the details of a design at one level of representation in order to simplify the description of the design at a higher level to enable the designer to access the functionality of the design more than the details of the implementation. There are three levels of abstraction in AMS design: circuit level, behavior level, and system level. As the lowest level of abstraction, circuit-level abstraction is concerned with the physical implementation of the concept. The designer focuses on the individual components and their interconnections at this level. As the lowest level of abstraction, circuit-level abstraction is concerned with the physical implementation of the concept. Behavior-level abstraction is the next level of abstraction and is concerned with the behavior of the design. The emphasis is on the input- output behavior of the design rather than on the individual components. The highest degree of abstraction, system-level abstraction focuses on the overall behavior of the system. At this stage, the designer concentrates on the system as a whole instead of on its constituent parts. The focus is on system-level behavior rather than implementation particulars [26]. This work focuses on the circuit level, which has several entities inside. Every circuit can contain a hierarchy which helps in understanding the model.

Analog circuits are often designed with hierarchical structures in order to improve performance and/or reduce/handle complexity. In a hierarchical structure, a circuit is divided into smaller subcircuits, each of which performs a specific function. The subcircuits are then interconnected to form the overall circuit. The hierarchical structure of an analog circuit is a top-down approach, where the circuit is divided into smaller and more manageable blocks. This approach is often used in the design of integrated circuits (ICs). In this approach, the overall circuit is first divided into major blocks, and each block is further divided into smaller subblocks. This approach allows the designer to focus on each block separately and then integrate them together. This results in a more efficient design process and a higher-quality circuit [27]. The most common type of hierarchical structure is the two-level hierarchy. In this structure, the overall circuit is divided into two major blocks: the upper level and the lower level. The upper level contains the more complex blocks, while the lower level contains the simpler blocks.

Another type of hierarchical structure is the three-level hierarchy. In this structure, the overall circuit is divided into three major blocks: the upper hierarchy level, the middle hierarchy level, and the lower hierarchy level. The upper level contains the more complex blocks, while the middle level contains the less complex blocks. The lower level contains the simplest blocks. The fourth type of hierarchical structure is the multi-level hierarchy. In this structure, the overall circuit is divided into two major layers/levels: the upper level/layer and the lower level/layer. The upper level contains the more complex blocks, while the lower level contains the simpler blocks. Another type of hierarchical structure is the three-level hierarchy. In this structure, the overall circuit is divided into three major blocks: the upper hierarchy level, the middle hierarchy level, and the lower hierarchy level. The upper level contains the more complex blocks, while the middle level contains the less complex blocks. The lower level contains the simplest blocks. The fourth type of hierarchical structure is the multi-level hierarchy. In this structure, the overall circuit is divided into multiple levels, each containing a different level of complexity.

Using a hierarchical structure in analog circuits has numerous advantages. First, it can improve the performance of the circuit by allowing each subcircuit to be optimized for its specific function. Second, it can raise the circuit's complexity without increasing the circuit's number of components or its size. Lastly, it might make testing and debugging the circuit

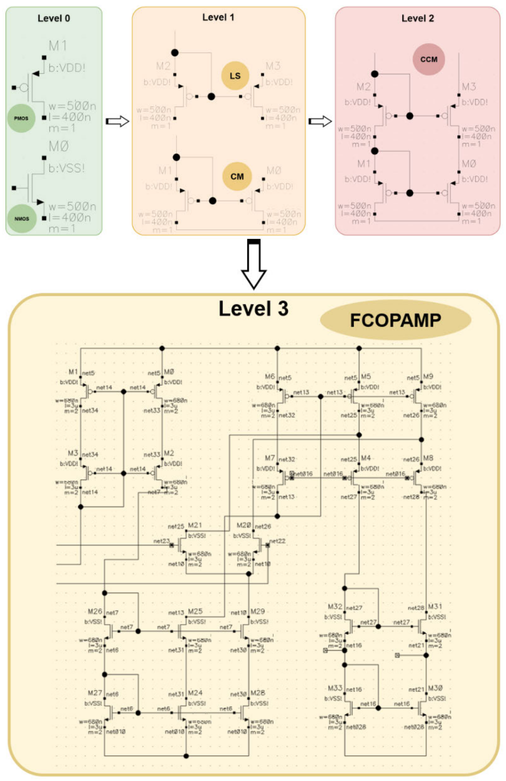


FIGURE 6. Example for the three levels of the hierarchy of analog/mix signal circuits IPs.

easier. There are some trade-offs to using a hierarchical structure in analog circuits. First, it can make the circuit more difficult to design and understand. Second, it can increase the cost of the circuit due to the need for more components and/or higher-quality components. Finally, it can make the circuit more sensitive to environmental changes (such as temperature or humidity). Despite the trade-offs, hierarchical structures are often used in analog circuits because the benefits can outweigh the costs.

In this work on analog circuit structure recognition, it was a must to define the levels of the schematics in order to cluster from it the subblocks of the lower level correctly. As mentioned, every block in analog design could be a building block of another more significant building block (of a higher level), which may be confusing for an AI based recognition model.

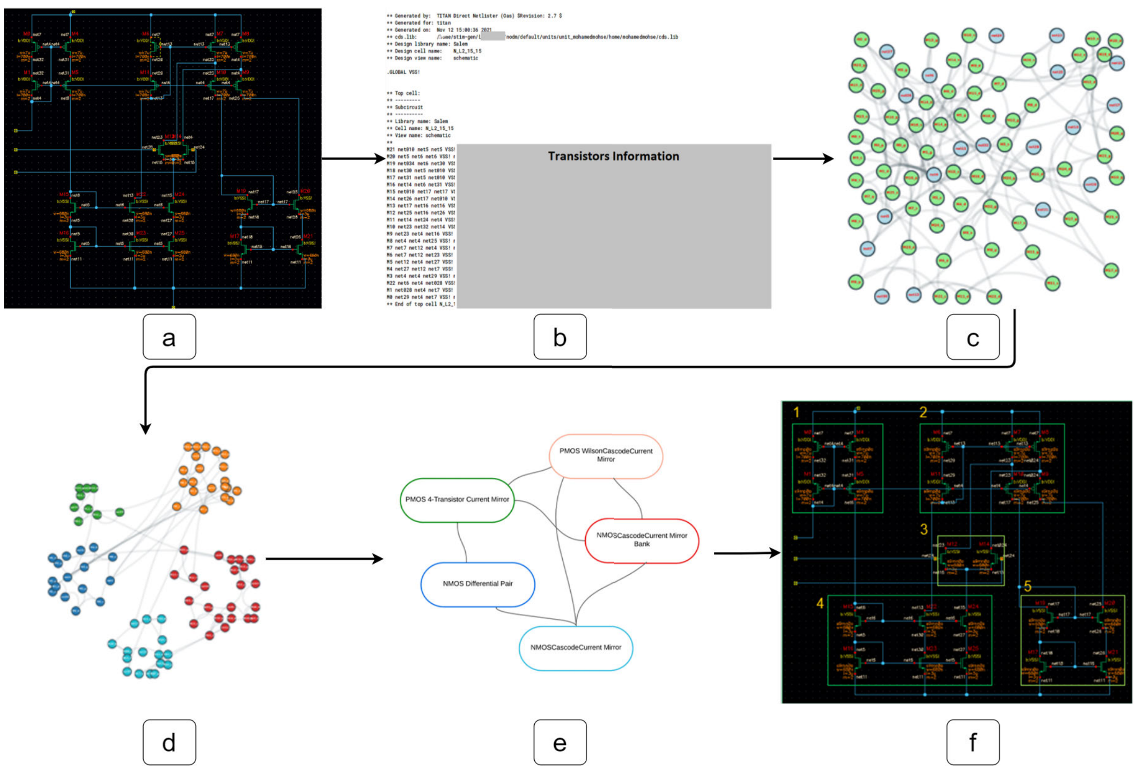
In this paper, as shown in Figure 6, The very lowest level basic building blocks of the analog circuit are transistors, which could be (PMOS, NMOS, or Bipolar), and all passives are neglected for this proof of concept stage. Every number of transistors, according to how they are interconnected, can form some groups of specific block(s). Those blocks are unique from the interconnections between the transistors inside the block. For example, as shown in Figure 6, transistors M0 and M1 are forming the current mirror (CM). Transistors M2 and M3 form another block, the level shifter (LS). In addition, other blocks may be formed out of two transistors, such as differential pair (DP), voltage reference (VR), current mirror load (CML), and flip-flop (FF). Those blocks contain some transistors as a lowest-level building blocks; we consider them as level 1 and the transistors level as level 0. The same idea related to the relation from level 0 to level 1 applies to level 1 to level 2. We can see that level 1 blocks can build higher levels of blocks too. For example, CM and LS form the cascode current mirror (CCM), and VR and CML form the 4-transistor current mirror (4TCM). Typically, other level 2 blocks can be formed, such as the Wilson current mirror (WCM), the improved Wilson current mirror (IWCM), and the wide swing cascode current mirror (WSCCM). After level 2 blocks are defined, combinations of the level 2 blocks can construct level 3 blocks, such as the folded cascode operational amplifier (FCOPAMP), the operational amplifier (OPAMP), the transconductance amplifier (OTA), and the Buffer amplifier (BAMP). The AMS circuits could contain many levels until they reach the IP level. Every block of level $i - 1$ is (may be) a building block of a level-i block. Figure 6 provides an example of different levels and how every level builds up to a higher level. Level 0 shows two types of transistors (PMOS and NMOS) as the building blocks of the very lowest level of all the schematics. When we combine two (or more) transistors, we can have different types of level 1 blocks, for example, LS and CM. The LS and CM could be combined and increase the level to a higher level. The combination of LS and CM results in having the CCM. CCM is one block of many that combine schematics of level 3 and higher schematics. In this work we are using and considering different blocks from different levels to be clustered. We call level 1 and level 2 sub-schematics and level 3 as the level main schematic.

C. PREPROCESSING: THE PARSER

In this subsection, we answer a very important question of how to project a given circuit from schematic space to graph space. A parser was developed with the main task of providing a proper numerical representation (ontology) of a graph out of a given netlist representing the schematics.

As shown in Figure 7.a, the input is a schematic representing the L2 circuit as exported from the Cadence Design tool as a netlist (See Figure 7.b), then the tool Net2Graph IV-C1 exports an undirected graph (See Figure 7.c) represented mathematically by both an Adjacency Matrix and a Feature matrix. Both matrices form the input of the graph attention network (GAT), the GAT model is comprehensively described in section VI, and the output is a clustered graph as shown in Figure 7.d and 7.e. This result can be visualized in the schematics domain as shown in Figure 7.f.

The parser reads and identifies the electronic elements and the so-called "nets" from the Netlist and generates a graph model with nodes and edges based on the connection provided by the Netlist. This yields an equivalent graph model that adequately describes the original circuit. The equivalent graph is represented by the graph's adjacency matrix and



```
** Generated by:  TITAN Direct Netlister (Oas) $Revision: 2.7 $
** Generated for: titan
** Generated on:  Nov 12 15:00:36 2021
** cds.lib:       /home/stim-gen/L[redacted]nodm/default/units/unit_mohamedmohse/home/mohamedmohse/cds.lib
** Design library name: Salem
** Design cell name:    N_L2_15_15
** Design view name:    schematic

.GLOBAL VSS!


** Top cell:
** ---------
** Subcircuit
** ----------
** Library name: Salem
** Cell name: N_L2_15_15
** View name: schematic
**
M21 net010 net5 net5 VSS!
M20 net5 net6 net6 VSS! r
M19 net034 net6 net30 VS
M18 net30 net5 net010 VS
M17 net31 net5 net010 VS
M16 net14 net6 net31 VSS!
M15 net010 net17 net17 V
M14 net26 net17 net010 V
M13 net17 net16 net16 VS
M12 net25 net16 net26 VS
M11 net14 net24 net4 VSS!
M10 net23 net32 net14 VS
M9 net23 net4 net16 VSS!
M8 net4 net4 net25 VSS! r
M7 net7 net12 net4 VSS! r
M6 net7 net12 net23 VSS!
M5 net12 net4 net27 VSS!
M4 net27 net12 net7 VSS!
M3 net4 net4 net29 VSS! r
M22 net6 net4 net028 VSS!
M1 net028 net4 net7 VSS!
M0 net29 net4 net7 VSS! r
** End of top cell N_L2_1
```

Transistors Information
a
b
c
PMOS WilsonCascodeCurrent Mirror
PMOS 4-Transistor Current Mirror
NMOSCascodeCurrent Mirror Bank
NMOS Differential Pair
NMOSCascodeCurrent Mirror
1
2
3
4
5
d
e
f

FIGURE 7. **a. the input is schematic representing the L2 circuit and exported from the Cadence Design tool as a netlist b, then the tool Net2Graph exports an undirected graph c represented mathematically as Adjacency and Feature matrices. Both matrices are the input of the graph attention network GAT, and the output is clustered graph as shown in figure d and e. This result can be visualized in the schematics domain as shown in f.**

features matrix. A sequence/series of properties of each of graph nodes is described in an appropriate coding in the feature matrix. The feature matrix, for example, specifies whether a node is a net or an electronic element or a part of an electronic element (Since transistors are represented by 3 nodes in this work). The precise type of electronic element is also coded. The GNN (specifically GAT) model is fed by the two matrices: the adjacency matrix and the feature matrix.

1) NET2GRAPH

As described in IV-B, we define three levels for graphs in this work. "Basic building blocks" are represented in level-1 graphs, "functional building blocks" are represented in level-2 graphs, and "modules" are represented in level-3 graphs. The parser Net2Graph was developed to recognize level 1 within a level 2 schematics and to scale it to higher levels. The parser also considers a "label file" depending on the requirements of the experiments. For each netlist and label, the parsed output consists of the following elements:

- Adjacency matrix A: describes the connectivity of the graph nodes.
- Feature matrix X: contains the node features of the graph.
- Labels L: vectors of labels: each node has a label that identifies the class of subblocks that it belongs to.
- Graph object G: a networkX Python object that contains all the information (graph related information, connectivity information) from the netlist.

The algorithm used to convert the netlist to graphs is described in Appendix A

V. DATASET COLLECTION, TRANSFORMATION, AND PREPROCESSING

To train the GAT model to be used for structure recognition we need an appropriate dataset.

A. DATASET COLLECTION

Designs for analog/mixed signals originated from a vast body of knowledge that expands daily. Schematics for this proof of concept work were collected from the relevant literature in the field. These schematics cover a variety of levels, each with its unique functionality and distinct architectural approaches to the same functionality. The dataset was collected in PMOS, NMOS, and BIPOLAR transistors, the three most common types. The collection of datasets is carried out in a very structured manner by using a container file, which is used to store the gathered schematics, and by categorizing these schematics according to the different levels and types of transistors.

The dataset uses the ID labeling strategy to organize the data more logically. The ID of every sample shows the information needed to be known about the samples collected.

The form of labeling is **[Transistor type, Schematic level, Schematic type No., Schematic of the same type No., Schematic name, subschematic name]**. For example, **[N, L1, 1, 1, FCOPAMP1, CCM]**. The ID labeling strategy allowed all users involved in the project implementation to explore the collected data. In addition, the strategy of ID labeling is helpful in the statistics of the data, as well as for the completeness, diversity, and comprehension of the data. In addition, the data collected is considered the first step toward understanding the common structures of analog schematics and defining the levels of those structures to have powerful AI/ML model models trained on these data.

B. DATASET TRANSFORMATION AND ANNOTATION

The collected data provide valuable insights into the design of analog/mixed signals. The next step was implementing these collected schematics on the standard design software Cadence Virtuoso. Since the collected data are many and in different levels, types, and transistor technologies, only 8 level-3 schematics, listed in Table 4, have been chosen to perform structure recognition (in the proof of concept prototype). Those 8 Samples are in level 3, with ten different level 2 subschematics inside. The eight samples are real-world circuits that are commonly used in analog design. In addition, eight samples are in NMOS and PMOS transistor types, which helps with the diversity of data. ID labeling was a valuable technique in dataset drawing as it improved the implementation organization and the naming system. Cadence Virtuoso software is a circuits CAD software with various entities, allowing you to use different technologies in the software libraries. The software is so user-friendly as it enables connecting the **(transistors)** used together by a connection line called **(net)**. The naming of transistors and nets is done automatically and in an organized way. After sorting and defining the schematics level, all eight schematics with their subschematics were implemented in Cadence with a specific transistor industrial library. Analog/mixed signals schematics contain subschematics of common structures with different functionalities, different transistor numbers, and connections between transistors and every group of transistors sets up a structure. With a slight change in the connection between transistors, it can set up another structure/subschematic. For dataset understanding and analysis, all common subschematics are annotated using yellow boxes around the groups of transistors and nets as shown in Figure 8. Every box is numbered and described according to the type of schematics. The annotation is done for the eight chosen schematics. This annotation has the advantage of giving a better visualization of the structures and is a good step toward labeling the dataset components.

Figure 8 demonstrates an example of FCOPAMP1 annotation of subschematics and how it may be used within the context of the main schematic. The annotation was purposefully abstract so that readers would have a better understanding of analog structures and improved statistical capabilities.

TABLE 4. **schematics list: The schematics which will be the input for asnalog circuit structure recognition to find their subschematics in an automatic way.**

Schematic name	Subschematics number
Folded Cascode Operational Amplifier 1	5
Folded Cascode Operational Amplifier 2	5
Folded Cascode Operational Amplifier 3	5
Folded Cascode Operational Amplifier 4	4
Operational Transconductance Amplifier 1	3
Operational Transconductance Amplifier 2	4
Operational Amplifier 1	5
Operational Amplifier 2	4

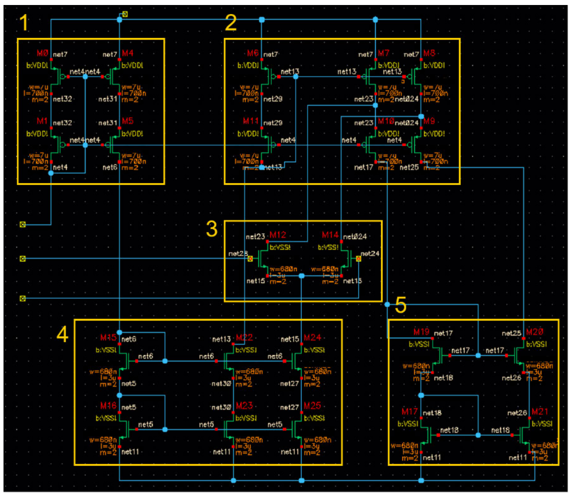


FIGURE 8. **Folded Cascode Operational Amplifier Annotation: the nodes and the connecting nets in this analog circuit are gathered into on of the following groups: 1. 4-Transistor Current Mirror, 2. Wilson Cascode Current Mirror Bank 1, 3. Differential Pair, 4. Cascode Current Mirror Bank 1, and 5. Cascode Current Mirror.**

In addition to this, the annotation highlights the area on the targeted structures that will be clustered afterwards.

C. DATASET LABELING

Data labeling is an essential part of the data preparation phase for machine learning, especially supervised learning. To establish a learning foundation for further data processing, supervised learning uses data that is labeled to establish a correspondence between the input and output forms [28]. Machine learning (ML) and deep learning (DL) require rich data to learn reliably. Information for their training techniques must be labeled to organize data and identify trends. Accurate algorithms require informative, selective, and individual feature labels [29].

The implemented dataset consists of transistors and nets, and the application of supervised learning requires labeling for each component we have. Manual labeling has been tried, but faces serious challenges such as labeling errors and time consumption, as well as lack of improvement, scaling limitations, and low support for generalization.

Given all the challenges mentioned earlier, a novel **Automatic Labeling (AL)** algorithm was implemented and improved. The primary concept of Automatic Labeling (AL) is to iterate at the netlist level to map the transistor names and net names. These names are grouped and show the group number of their structure/sub-schematics and the class names of every component in a .label file. Since the netlists of the main schematics and sub-schematics are in one container file, a generic Python algorithm maps the transistor instances from the subschematics and groups them according to the file naming, which provides the group number and class name. Subschematics were cut from the main schematic, so net names changed when generating a new netlist for new schematics. Both transistor and net instances were hard to map from the main schematics. A new methodology was developed to solve this problem. It allows us to simultaneously operate on main and subschematics files and provides a robust and efficient labeling system for analog/mixed Signals schematics. Automatic Labeling only accepts main and subschematic netlists with the same naming protocol. The Python code works on every netlist, and the first step is to match the technology, as every transistor is different. After matching, the parser splits netlist lines. All transistors in a sub-schematics netlist are grouped and saved in a dictionary with the group number and the class name. For the net name change challenge, grouped transistors are mapped in the main schematics to get the connections for each transistor and add them to the group. The final step is to generate a .label file with all transistors, three nodes (drain, gate, and source), nets, and connections with class names and group number.

D. DATASET AUTOMATIC LABELING USE CASES AND IMPROVEMENTS

Automatic Labeling (AL) is a valuable step for automating analog-circuits-related applications of AI. AL enables us to label main schematics and sub-schematics in two seconds instead of one hour without needing to read the schematics. AL is used through the project flow in different applications, and since it is "automated," it was not a trivial task to adapt it for different cases. For instance, when we want to detect a specific sub-schematics within a schematics or what we call "one vs. all" structure recognition, the group numbers in the labels were not used. Instead, every sub-schematics was activated once, and all other sub-schematics were deactivated. For each activation iteration, the model's target block was the activated block. In other words, the targeted block was activated or deactivated using "1" or "0" to provide the model with further information about the sub-schematic within the main schematic that was targeted. The flexibility of the automatic labeling did not stop here; it also improved and became able to remove the labels of the targeted sub-schematics once the model detects it and start iterating on the other sub-schematics doing the same procedure again and having the new schematics activating other targeted sub-schematics. Last but not least, AL was adaptable to target one class within all the schematics and activate it in the improved version of "one vs. all." For example, during one of the initial tuning steps, a differential pair (DP) was selected to be clustered among all sub-schematics and to achieve this, AL has been configured to activate only the selected sub-schematics. Automatic labeling is thus a precious building brick for all automation processes in the area of analog/mixed signals design and verification automation. In addition, automatic labeling for the dataset has allowed us to go further in the project, specifically for designing and realizing a novel automatic dataset augmentation method.

VI. A ROBUST GAT BASED MODEL FOR STRUCTURE RECOGNITION: DESIGN AND VALIDATION

A. INTRODUCTION AND BACKGROUND

A novel neural network architecture called GAT (Graph Attention Network) uses masked self-attention layers to overcome the drawbacks of previous GNN approaches based on graph convolution or its approximations. It works with graph-structured inputs. The model allows (implicitly) specifying different weights for different nodes in a neighborhood without requiring any expensive matrix operation (such as an inversion) or knowing the graph structure in advance by stacking layers where nodes can attend to the features of their neighborhood. In this way, GAT simultaneously overcomes many of the main problems of so-called spectral-based neural graph networks [30] and makes the model easily adaptable to inductive and transductive applications.

GATs were first proposed in the paper "Graph Attention Networks" by *Veličković et al.* [31]. In a GAT, each node in the graph is assigned a "hidden state" vector. The hidden state vectors are then used to compute attention weights, which are used to determine how much each node should be "attentioned" to by the network. The hidden state vectors and attention weights are then used to compute a final node representation, which can be used for various downstream tasks such as classification or link prediction. GATs are similar to other neural networks, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs). However, GATs are explicitly designed for graph data. This makes them more efficient and effective at learning from and making predictions on graph data.

GATs are built on the idea of self-attention, which was introduced by [32]. The Transformer paper showed that self-attention is an effective way to learn representations of data. GATs use self-attention to learn representations of graph data. A self-attention layer of a neural network selectively processes data, focusing on what's important while ignoring the rest. Self-attention is well-suited for learning from graph data because graphs are often highly structured and have many relationships between the data points.

GATs are composed of two layers: an attention layer and a feed-forward layer. The attention layer is where the self-attention mechanism is applied. The attention layer consists of a series of attention heads. Each attention head attends to a different part of the graph. The output of the attention layer is fed into the feed-forward layer. The feed-forward layer is a standard neural network layer that transforms the input data into a new representation. The output of the feed-forward layer is the final representation learned by the GAT. GATs can be trained using standard neural network techniques, such as backpropagation. GATs have been shown to outperform traditional graph neural network architectures on a variety of tasks, such as node classification [33] and link prediction [34], [35]. Additionally, GATs are computationally efficient, making them well-suited for large-scale graph data.

Graph Attention Networks (GATs) [31], and Graph Convolution Networks (GCNs) [20] are both types of neural networks that can be used for learning on graph-structured data. However, there are several critical differences between GATs and GCNs [36], [37]. GATs are designed to operate on a self-attention mechanism, whereby each node attends to all other nodes in the graph to compute its representation. This contrasts with GCNs, which use a neighborhood aggregation scheme where each node only attends to a limited number of neighbors. Another key difference is that GATs use multi-head attention to allow for different node representations to be learned in parallel. GCNs typically use a single attention head. Moreover, GATs are typically trained using a masked self-attention objective, which encourages the model to focus on local structure. GCNs are typically trained using a node classification objective, which can encourage the model to learn global structure. Finally, GATs can be applied to graphs with arbitrary node connectivity, while GCNs require the graph to be locally connected.

Instead, self-attention is used over the node characteristics and let the attention scores α_{ij} be implicitly specified. This decision was not made arbitrarily, as the Transformer design has already proven self-attention to be sufficient for state-of-the-art outcomes in machine translation [32].

B. MATHEMATICAL MODEL

Graph Attention Networks (GATs) with a single attention layer in [31] is used. The input layer of GAT is the features of nodes $\mathbf{h} = \{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_N\}, \mathbf{h}_i \in \mathbb{R}^F$, whereby N is the number of nodes, and F is the number of features in every node. The layer outputs a fresh set of node characteristics, (maybe with different attribute values, F'), $\mathbf{h}' = \left\{\mathbf{h}'_1, \mathbf{h}'_2, \ldots, \mathbf{h}'_N\right\}, \mathbf{h}'_i \in \mathbb{R}^{F'}$, as the GAT output.

The attention mechanism $a : \mathbb{R}^{F'} \times \mathbb{R}^{F'} \to \mathbb{R}$ is used as a byproduct of an attentional mechanism which computes unnormalized attention coefficients e_{ij} across pairs of nodes i, j, based on their features:

$$e_{ij} = a\left(\mathbf{W}\mathbf{h}_i, \mathbf{W}\mathbf{h}_j\right) \tag{1}$$

where the weight matrix, $\mathbf{W} \in \mathbb{R}^{F' \times F}$, is applied to every node. The weight matrix is a primary step to parametrize the shared linear transformation of input features to higher-level features, providing one needed learnable parameter. The attention coefficient shows the importance of the features of the node j to the node i. The model eliminates all structural information in its most basic version and allows each node to attend to every other node. By doing masked attention, where we only calculate e_{ij} for nodes $j \in \mathcal{N}_i$, where $\mathcal{N}_i$ is some neighborhood of node i in the graph, the graph structure is introduced into the mechanism. i and its first-order neighbors

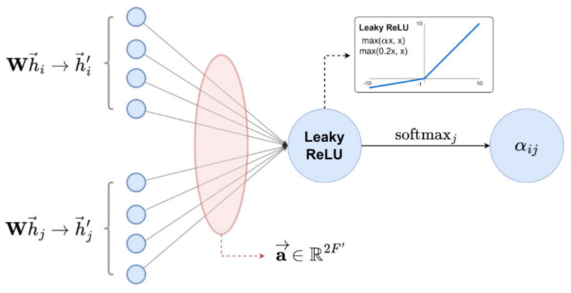

$\mathbf{W}\vec{h}_i \to \vec{h}'_i$
$\mathbf{W}\vec{h}_j \to \vec{h}'_j$
Leaky ReLU
max(αx, x)
max(0.2x, x)
Leaky ReLU
softmax$_j$
α_{ij}
$\vec{\mathbf{a}} \in \mathbb{R}^{2F'}$

FIGURE 9. The attention mechanism: the attention values α_{ij} are computed from the embeddings $\mathbf{h}_i$ and $\mathbf{h}_j$. W, which is a learnable matrix, is multiplied by the old embeddings of node i. A similar multiplication is performed for node j. Both results are concatenated and then multiplied by another learnable vector a. The result is fed to a leaky ReLU activation function to obtain the raw values of attention. Finally, a softmax is applied to normalize the values across the neighborhood and α_{ij} to obtain the normalized attention values.

are normalized across all choices of j using the softmax function to make it easier in comparison through different nodes.

$$\alpha_{ij} = \mathrm{softmax}_j\left(e_{ij}\right) = \frac{\exp\left(e_{ij}\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(e_{ik}\right)} \tag{2}$$

The attention mechanism used by [31] is a single layer feedforward neural network precomputed by the weight vector $\vec{\mathbf{a}} \in \mathbb{R}^{2F'}$.

$$\alpha_{ij} = \frac{\exp\left(\mathrm{LeakyReLU}\left(\mathbf{a}^T\left[\mathbf{W}\mathbf{h}_i \| \mathbf{W}\mathbf{h}_j\right]\right)\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(\mathrm{LeakyReLU}\left(\mathbf{a}^T\left[\mathbf{W}\mathbf{h}_i \| \mathbf{W}\mathbf{h}_k\right]\right)\right)} \tag{3}$$

where the concatenation operation is represented by | and transposition is represented by $\cdot^T$. Features corresponding to the normalized attention coefficients are received after the non-linearity *sigma* is applied, and the final output features for each node are generated by linearly combining these features.

$$\mathbf{h}'_i = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{W}\mathbf{h}_j\right) \tag{4}$$

In [32], multi-head attention is introduced and extended in [31] which mentions that including multi-head attention into the extended method helps stabilize the self-attention learning process. The following output feature representation is the outcome of Equation 5's transformation being executed by "K independent attention mechanisms" and then their features being concatenated:

$$\mathbf{h}'_i = \|_{k=1}^{K} \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}^k \mathbf{W}^k \mathbf{h}_j\right) \tag{5}$$

where '‖' indicates concatenation, α_{ij}^k are normalization of attention coefficients calculated by the k-th attention mechanism $\left(a^k\right)$, and $\mathbf{W}^k$ is the weight matrix of the associated input linear transformation. In this configuration, $\mathbf{h}'$ will produce an intermediate result consisting of KF' features (instead of F') for each node.

If multi-head attention is executed on the last layer (prediction layer) of the network, concatenation is no longer reasonable; instead, the paper suggests averaging and postponing applying the final nonlinearity (often a softmax or logistic sigmoid function for classification tasks) in the following:

$$\mathbf{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^{k}\mathbf{W}^{k}\mathbf{h}_j\right) \tag{6}$$

Figure 9 shows the process of the attention of one node and the aggregation process of Equation 6. It also answers the question of how to compute the attention values α_{ij} from the embeddings $\mathbf{h}_i$ and $\mathbf{h}_j$. $\mathbf{W}$, which is a learnable matrix, is multiplied by the old embeddings of node i. A similar multiplication is performed for node j. Both results are concatenated (denoted as $||$) and then multiplied by another learnable vector $\mathbf{a}$, which represents the direction of attention. The result is fed to a leaky ReLU activation function to obtain the raw values of attention. Finally, a softmax is applied to normalize the values across the neighborhood and α_{ij} to obtain the normalized attention values.

C. GATS FOR INDUCTIVE LEARNING

One of the most significant advantages of the GATs is that they can work on both *inductive learning* and *transductive learning* [31]. Inductive learning is identical to what is conventionally known as supervised learning. We construct and train a machine learning model using an existing labeled training dataset. Then, using this trained model, we predict the labels of an unexplored testing dataset. In contrast to inductive learning, transductive learning approaches evaluate both the training and testing datasets beforehand. We learn from the previously observed training dataset and then predict the testing dataset's labels. While we cannot utilize the labels from the testing datasets, we may still benefit from the patterns and other information included within these datasets as we progress through the learning process [38], [39].

For the AMS dataset, we go through inductive learning for its novelty and generalization potential. Thanks to GATs, all the model was able to recognize and generalize the subschematics within the main schematics using inductive learning.

Figure 10 shows the general architecture of the GAT. The architecture starts with the input graph of nodes and features. The first layer of the GAT network is composed of $K = 8$ attention heads, and each head is responsible for calculating $F' = 8$ features. After that comes the nonlinearity, exponential linear unit (ELU). The classification takes place in the second layer, which consists of a single attention head that calculates C features (where C is the total number of classes) and is then followed by a softmax activation. Regularization is used extensively throughout the model in order to compensate for the very modest sizes of the training sets. In order to avoid overfitting. The dropout with $p = 0.6$ is applied to the inputs of both layers as well as the normalized attention coefficients, exposing each node during each training iteration to a randomly chosen neighborhood. In the output layer each of the nodes clustered to each respective class specified.

D. MODEL ADAPTATION

Recently, attention-based models have seen widespread use in a number of contexts. However, the area of analog circuit design has not conducted much research on this technology. This work proposes a novel analog circuit structure recognition method based on the graph attention network (GAT).

The proposed GAT model for analog circuits structure recognition is adapted from the GAT model proposed in the paper "Graph Attention Networks" (GAT) [31]. The original GAT model is designed for the task of node classification in a graph. In this work, the GAT model is adapted for the task of analog circuits structure recognition. The model takes as input a graph representation of an analog circuit and outputs a label for each node in the graph, indicating the type of analog circuit subcircuit that the node represents. Many different attention parameters that work on pairs of nodes in the graph are learnable in the GAT model. The attention modules are designed to learn the attention between pairs of nodes to enable the model to focus on the relevant parts of the graph when making predictions. In order to adapt the GAT model for the task of analog circuits structure recognition, attention layers are added to operate on the entire graph rather than on pairs of nodes. This model is designed to learn attention between the nodes in the graph and the labels assigned to those nodes. Then the nodes are collected into classes of the subschematics.

The proposed model can be adapted to recognize the structure of analog circuits. The model can be trained on a dataset of graphs of analog circuits and then tested on a new graph to predict/detect the structure of the circuit.

Inspired by the original GAT architecture, the architecture shown in Figure 11 is showing a new machine learning architecture developed to adapt the AMS circuits' characteristics and to be flexible for various problems in the field.

The architecture is based on a series of Attention Layers, which are trained to focus on different parts of the input data. The output of each Attention Layer is fed into a fully connected layer, which is then fed into a softmax layer to produce the final predictions. The advantages of this architecture are that it is more scalable than the GAT architecture and can be trained on much larger datasets.

Since we have a dataset of graphs and every graph has the embeddings of the nodes and nodes' features, the model can input the circuits' graph data to the new proposed GAT-based architecture. For the limitation of the dataset, a Dropout of 0.1 is carried out in the architecture for the input layer and output layer to avoid overfitting.

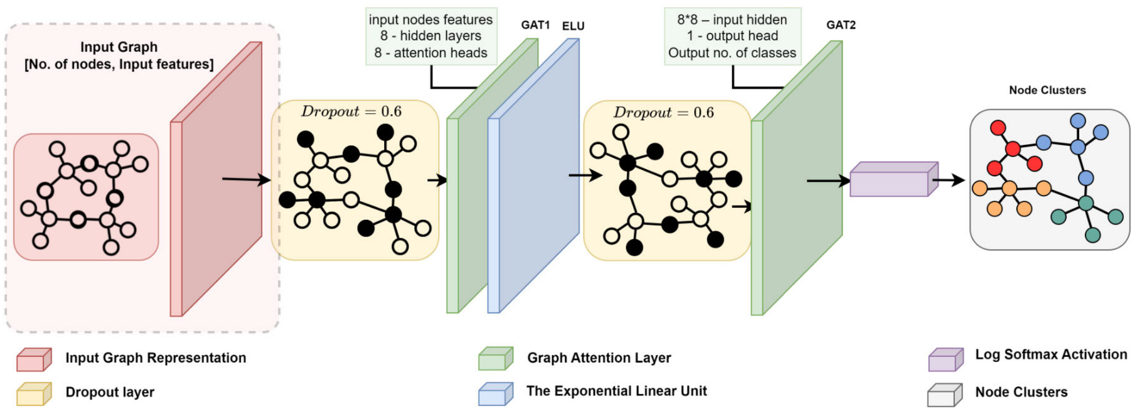


FIGURE 10. **Graph attention networks general architecture.**

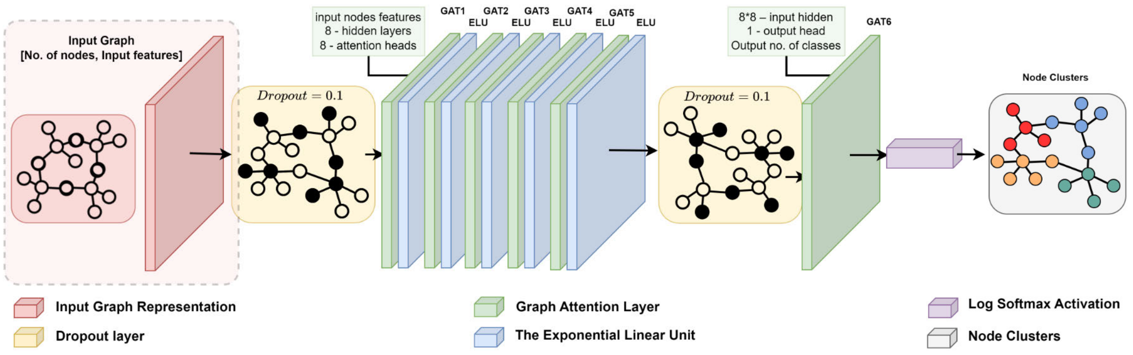


FIGURE 11. **The baseline architecture Using 7 Layers of GAT implemented to get the best results.**

The first layer of a GAT is typically an input layer, which is followed by five attention layers. Each attention layer contains a number of attention heads, which focus on different parts of the input graph. Every new attention layer added to the new architecture means that this layer will make the networks able to increase its attending hubs. In electronics circuits, attending the more hubs of neighborhoods is an efficient way to gather more data about subcircuits information. In addition, every attention layer is followed by an exponential linear unit (ELU) layer. ELU has several advantages over other activation functions, such as rectified linear units (ReLUs). ELUs can help neural networks converge faster and achieve higher accuracy. Finally, the GAT includes an output layer used to make predictions based on the learned representations. The final output of the output layer is a vector with the output class classification from the softmax nodes, followed by the class clustering for all the tested nodes.

E. MODEL OUTPUT VISUALIZED: NODE CLUSTERING PIPELINE OUTPUT

As mentioned earlier, Figure 7 shows how the pipeline works. After the parser generates a graph from the netlist, the features of the nodes are generated based on the node type. The first GAT learning operator receives a fully masked edge list of the input graph and its node's feature matrix. The first layer (multi-head attention) concatenates eight outputs and the second layer output only has 1 head, which produces the final embeddings. The final output, which is a vector, equals the number of classes expected in the circuit times the number of nodes in the input graph or circuit. The output after the activation function is a set of probability scores over the number of classes available in the circuit. Once the network collects all the highest scores of each node, the class of this node is known either, for example, as DP (differential pair) or any other level-1 schematic type. After knowing the class of each node, we group the similar classified nodes to form the class sub-block. Figure 12 is showing an example of output after obtaining the sub-blocs of one given L2 circuit. The red nodes are classified as connectivity nodes, and the other nodes as belonging to each respective specific cluster or L1 block.

VII. ADDRESSING THE CHALLENGE OF LIMITED DATASETS THROUGH INNOVATIVE DATA AUGMENTATION CONCEPTS COUPLED TO THE GAT MODEL

A. DATASET AUGMENTATION BACKGROUND AND LITERATURE REVIEW

Significance in context, quality, quantity, and training data all contribute significantly to a deep learning model's accuracy.

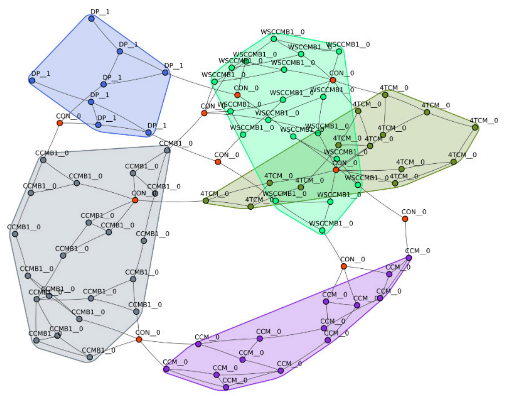


FIGURE 12. Pipeline output of the L2 circuit clustered into its building subschematics of L1.

However, the lack of enough data is a frequent obstacle when it comes to developing deep learning models. In a production setting, gathering this kind of information (i.e., dataset gathering) is a time-consuming and thus expensive task. The term "data augmentation" refers to any method that generates additional artificial data samples from actual real-world data. Data may be enhanced by making minor adjustments or using machine learning models to produce additional data samples in the latent space of the original data [40].

There has been a rise in both interest and demand for data augmentation techniques on graph data in recent years. This is mostly attributable to the widespread use of Graph Machine Learning (GML) methods like graph neural networks. Graph data augmentation (GDA) strategies were not easily comparable to data augmentation (DA) methods in computer vision (CV) and natural language processing (NLP) because of the non-Euclidean and asymmetrical nature of graph data. In addition, GML encounters its issues, such as missing or unreliable feature data, sparse structural data owing to power-law distributions, a lack of labeled data because of expensive annotations, and over-smoothing due to message transmission in GNNs. [40]

There has been a significant focus of research on GDA that attempts to tackle these issues. When faced with similar difficulties on graphs, GML researchers develop graph-specific augmentation strategies. The goal of GDA approaches, when applied to graph-level problems, is to increase generalization by creating labeled training data beyond what is provided in the input. When applied to node-level activities, GDA approaches have improved GML models in several ways. To address the issue of over-smoothing, [41] proposes randomly removing edges during training. To encourage graph homophily, [42] improved the graph's structure. Using adversarial training, [43] adjusted or added node properties.

Since the graph data type is different from other types, such as pictures or text, different types of augmentation are introduced throughout the research in graph data augmentation. Graph data augmentation has four main strategies that target different parts of graphs [40].

- **Nodes augmentation**: which operates on nodes by removing, adding, combining two nodes, or masking only features for selected nodes [44], [45].
- **Edges augmentation**: which changes some graph connectivity by adding or removing edges in random or regular ways [41], [46], [47].
- **Features augmentation**: that manipulates, adds, or masks some features in the feature vector [48].
- **subgraph augmentation**: which depends on augmenting and creating new graphs by cropping in [49] and mixing up subgraphs to create new graphs in [50]

In addition, there are some data augmentation techniques in the analog design field. It has many shortcuts that help a lot in data variety within minimal parts in the schematics.

B. DATASET ELECTRICAL AUGMENTATION

Dataset Electrical Augmentation (DEA) is an efficient way to improve and augment your dataset of electronic circuits. There are many different ways to augment the data electrically. For example, banks are one of the most beneficial ways to augment the dataset. One to ten banks could be added to any subschematic, which keeps all other blocks the same by adding just transistors in a row. Banks could be added in a different number of branches. Another type of DEA is to replace the single transistors on the input or on the output of the circuit with a Current Mirror (CM). The single transistor could be replaced by a CM, in one of the blocks we cluster, and this benefits augmentation by using the schematics that do not fit our model. The third type of DEA is adding passives. Resistors and capacitors are the most common passives that can be added to schematics. The fourth type of DEA is adding the stability compensation (SC) block. SC may contain capacitors, or resistors. The SC blocks are put in to ensure the so-called stability of the circuit's output, which could also be removed without affecting the functionality of the schematics.

DEA methods are efficient in augmenting datasets and adapting the schematics that are not fitting with the AI model. However, DEA methods are time-consuming and need significant experience in analog design. Moreover, it is not efficient when it comes to generalization and operating on a more extensive dataset because all the steps of drawing and netlists generation will be required again. In addition, upscaling the model would not be an easy task to perform, and its automated labeling will not be a trivial task either. Thus, a so-called combinatory data augmentation (CDA), our own new suggested data augmentation concept, is used in order to ensure automation, generalization, and upscaling without the need to have considerable experience in analog design and without consuming much time.

C. ANALOG/MIXED SIGNALS DATASET AUGMENTATION

Analog/mixed signals schematics have unique features and settings. Therefore, any augmentation process needs to consider the functionality of all the circuits. The functionality of the circuits depends on the transistors (nodes) and the connections (edges). In AMS design, just one connection change, one transistor removal, or adding components not belonging to a typical structure alters the whole circuit functionality. Other than functionality, time consumption, generalization, upscaling, and lack of experience in analog design for electrical augmentation should be considered. Hence, not all of the above-mentioned methods of augmentations are the efficient way to augment the data we have in order to keep the functionality, save time, automate labeling, and ensure generalization and upscaling of the dataset. In the structure recognition model, every transistor has three nodes which represent (drain, gate, and source), and an edge represents every connection. In this case, using either **nodes augmentation** and **edges augmentation** is not efficient as it changes the concept or functionality of the respective circuits. Features augmentation is used in two different ways. The first feature augmentation consists of adding some new features to the feature vector used, such as centrality measurements on the graph level and using adjacency matrices as the features and adding them to feature vectors.

While analyzing and applying the **subgraph cropping** [49] on the AMS dataset, some challenges are encountered. Cropping the graph can affect the functionality of the circuit, since the transistors instances and classification is crucial for the supervised clustering process. In addition, the connecting nets between subschematics are classified as connecting nets, which makes them altered by the cropping on the graph level. Moreover, converting the new cropped graphs again to netlist is time-consuming and challenging. Finally, every small connection change in the dataset generates a totally new graph, which leads to the lack of generalization for the problem we have.

To address all the challenges facing the **subgraph cropping**, [49], a new novel augmentation approach is here suggested (our own novel concept) and implemented, which augments the dataset on the netlist level, not on the graph level. The new approach called combinatory data augmentation (CDA). CDA is a powerful solution for the challenges face by subgraph cropping. All issues related to circuits functionality, connectivity, and nets classification challenges are all solved by CDA. Also, generalization and upscaling for the structure recognition problem are insured by using the CDA concept.

D. DATASET AUTOMATIC COMBINATORY AUGMENTATION CDA

Inspired by the automatic labeling (AL), this automatic augmentation (CDA) for the limited dataset is a cutting-edge, novel tool/approach that benefits to all the electronic design field. A generic python algorithm has been developed and used for this automatic augmentation that operates on netlists of the main schematics and sub-schematics (or the original real-world dataset). The code generates a number of netlists equal to the combination of the number of sub-schematics within the main schematic. All the generated netlists are functional blocks that represent different combinations of sub-schematics. The generated netlists are labeled using the automatic labeling tool to have a ready dataset for a supervised clustering machine learning model.

This dataset augmentation is a combination of two augmentation methodologies. The first one is the netlist reduction, and the second one is the sub-schematics combinations that generate the combinatory dataset. The following sections will describe netlist reduction, sub-schematics combinations, dataset preparation, methodology and algorithm, and some use-cases of the augmented data.

This automatic augmentation benefits from the prepared data in AL and use both the main schematics and the sub-schematics. A different case for data augmentation is that every type of single schematic is saved in a single folder with other single folders for the new, generated netlists. Since we had to fix one basis sub-schematics to be in all schematics, an analysis of the sub-schematics is done for the dataset we have. Current Mirror (CM) is the subschematic that presents the highest number of all schematics we have. However, although CM is the most considerable quantity in the dataset, it has two drawbacks. The first one is that its high number is only distributed within a small number of main schematics. The second is that CM may reflect an overlapping problem between the Current Mirror Bank (CMB) and Cascode Current Mirror (CCM) regarding the level definition. Accordingly, Differential Pair (DP) is the most distributed subblock amongst most of all the schematics, as it refers to (i.e. is contained in) most circuits' inputs. By targeting DP, the problem of overlapping is addressed, and we can augment more schematics due to its excellent distribution.

The netlist reduction methodology idea came from the "one vs. all" scenario of targeting one sub-schematic and then removing it from the netlist. In the netlist reduction, it is opposite to the "one vs. all" scenario because we target one sub-schematic in this case and keep it in all generated samples. One sub-schematic, other than the targeted sub-schematic, is removed at every iteration to have a different type of connection with the different sub-schematics.

Sub-schematics combination is whenever you make a combination in the linking of all blocks together. For instance, we can take all varieties between a specific sub-schematic and all other blocks without repetitions. The combination mechanism is efficient in mapping and putting all blocks together. This method avoids bias (can ensure dataset balancing) since all the blocks have the same chance to be present in the dataset.

The first step for the automatic augmentation is the netlist reduction, as it is done once when we have more than two sub-schematics. The model operates on the netlist and maps all sub-schematics' transistors without using the target block, Differential Pair (DP). The code removes one of those blocks, and then comes the combination process. The combination process, the second process, deals with all remaining sub- schematics and makes all possible combinations of blocks together. The third process is the generation process, which generates a new netlist of all possible combinations in the empty folder assigned before generation. The process goes again for netlist reduction, and another block is removed. The combinations of the other sub-schematics still existing is made and one generate the combinatoric schematics again. At the end of this process, the main schematic will have only two sub-schematics (the targeted DP + another block), and no further reduction happens anymore. The python code implemented for this automatic augmentation is using the sub-schematics and the main schematic at the same time. It maps the transistors from the sub-schematics and the whole lines from the main schematics and builds the new lines of the new netlist.

This combinatory data augmentation (CDA) is one of the most valuable methods/concept in this work. CDA can significantly increase the dataset schematics using different algorithms, such as netlist reduction and sub-schematics combinations. CDA is a novel way of augmenting a limited dataset for analog electronic circuits. The CDA method works on the netlist level, making the dataset avoid many challenges. CDA also solved many drawbacks of other methods on the graph level or the schematics netlist level. For companies, e.g. for Infineon, their Infineon open-source schematics are very limited and confidential, hampering the research efforts in that area. However and thus, CDA is a cutting-edge tool that opens the door for new inventions in the area without altering the confidentiality of the company's schematics. In addition, CDA has a good asset in the time aspect. Five seconds are enough to generate 14 new netlists from one schematics containing five subblocks. Not only time-efficiency, but also it does not need experience in analog design. Moreover, CDA is a flexible algorithm that can be extended easily for upscaling to higher levels of structures. Generalization is one of the advantages of the CDA, as it keeps all the functionality of all schematics. The flexibility of the CDA did not stop here; it also enables to target specific sub-schematics, which in turn elaborates on the dataset balancing aspect after augmentation. Finally, CDA opens a big room for automation in the field of analog design and verification research and development. CDA can be used by anyone who wants to have a dataset out of the avaible limited, unique, and complex analog/mixed signal designs.

Figure 13 shows the distribution of the sub-schematics (of Level 1) over the whole available original dataset. The low number of most of the sub-schematics was the reason and motivation of creating more samples to balance the sub-schematics in the training dataset.

The best systematic way for increasing the training samples without considering the test dataset is the dataset augmentation. In this subsection we provide a comprehensive definition of what we call dataset automatic combinatory augmentation (DACA). As shown in Figure 14, one sub-schematics is fixed, in this case it is a differential pair DP, and the other sub-components are omitted one by one, over the iterations,

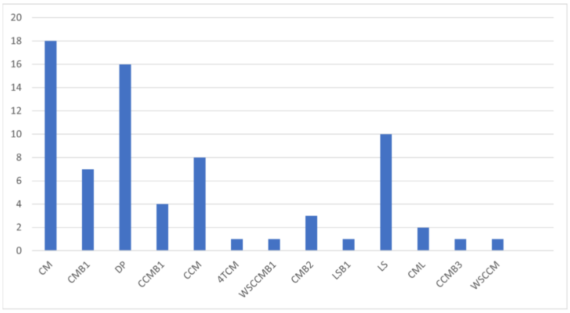


FIGURE 13. The subschematics histogram shows the unbalanced dataset before applying the augmentation method. The subschematics are: current mirror CM, current mirror bank 1 CMB1, differential pair DP, cascode current mirror bank 1 CCMB1, cascode current mirror CCM, 4 transistror current mirror 4TCM, wide swing cuscode current mirror bank 1 WSCCMB1, current mirror bank 2 CMB2, level shifter LS, current mirror load CML, cascode current mirror bank 3 CCMB3, wide swing cascode current mirror WSCCM.

to create non-functional (artificial) schematics (from the electronic circuit's perspective) but very useful for training the AI model with the new sub-schematics-balanced dataset.

Since the DP was fixed in all augmented samples, we call this augmentation step a DP-based augmentation. It resulted in creation of 14 combinatory samples. We have trained the GAT model with these 14 samples and the test dataset was the Folded Cascode Operational Amplifier FCOPAMP1, shown in Figure 8, and the task is to cluster all nodes to the following sub-schematics: 4-Transistor Current Mirror, Wilson Cascode Current Mirror Bank 1, Differential Pair, Cascode Current Mirror Bank 1, and Cascode Current Mirror.

Table 5.1 shows that 11 nodes out of 86 nodes were clustered incorrectly. Therefore, new samples were needed, containing new sub-schematics, in order to train the model better and to correct the confused samples. Since 6 out of 14 samples of CCM were detected wrongly, as shown in Table 5.1, a CCM-based augmentation was performed, In addition to the DP-based augmentation, to increase the number of CCM samples in the training dataset, and it resulted in 28 samples in total in the new training dataset. The confusion matrix, shown in Table 5.2, presents an improvement as only 5 out of 86 nodes were clustered incorrectly.

The final step of improvement was done by applying a CCMB1-based augmentation since the results needed improvement and the most class with the wrong classified samples is cascode current mirror bank 1 CCMB1. After adding 14 new combinatory samples and reaching 42 samples to the training dataset. Table 5.3 shows that the three augmentation steps we have done were sufficient to ensure a 100% accuracy of the clustering, or in other words, successful analog circuit structure recognition as the FCOPAMP1 was clustered to either its building subblocks and/or as a connectivity node (CON).

VIII. RESULTS AND DISCUSSION

In this sub-section we do present the results obtained by our comprehensive novel structure recognition concept when

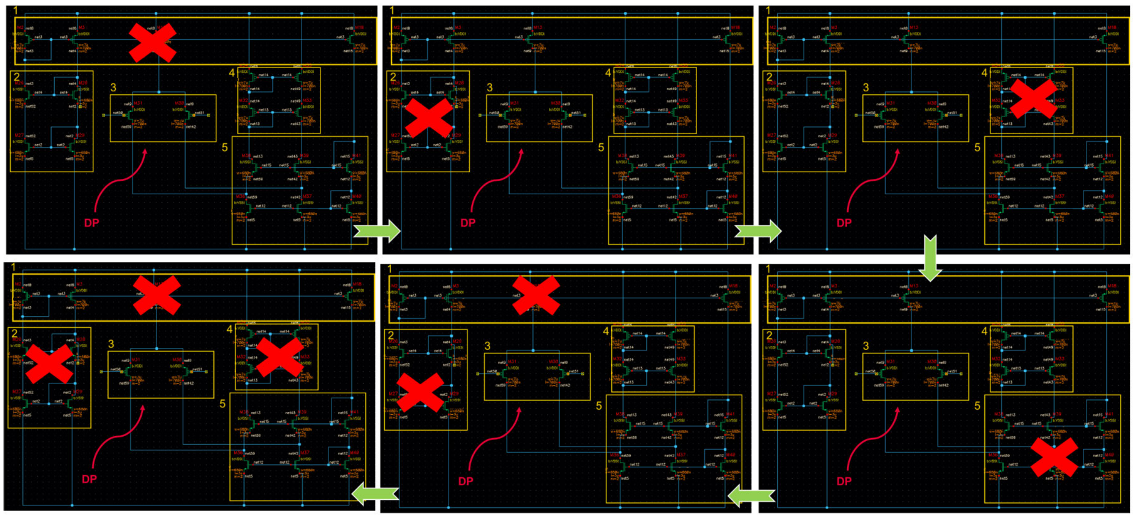


FIGURE 14. Dataset automatic combinatory augmentation applied on FCOPAMP1: DP-based augmentation was applied were all other subschematics were omitted one by one, except for DP, and new sample was created for each step to enrich the training dataset.

TABLE 5. (a). Confusion matrix of FCOPAMP1 for each of the following cases: 1. Augmented dataset based on DP only which results in 14 subschematics, 2. Augmented dataset based on DP and CCM which results in 28 subschematics, and 3. Augmented dataset based on DP, CCM, and CCMB1 which results in 42 subschematics.

(1)

True Labels		CCM	WSCCMB1	4TCM	CON	DP	CCMB1
	CCM	8/14	0	0	0	0	6/14
	WSCCMB1	0	19/19	0	0	0	0
	4TCM	0	0	14/14	0	0	0
	CON	0	0 1/10	1/10	7/10		1/10
	DP	0	0	0	0	8/8	0
	CCMB1	2/21	0	0	0		19/21

Predictions

(2)

True Labels		CCM	WSCCMB1	4TCM	CON	DP	CCMB1
	CCM	14/14	0	0	0	0	0
	WSCCMB1	0	19/19	0	0	0	0
	4TCM	0	0	14/14	0	0	0
	CON	1/10	0	0	8/10	1/10	0
	DP	0	0	0	0	8/8	0
	CCMB1	3/21	0	0	0		18/21

Predictions

(3)

True Labels		CCM	WSCCMB1	4TCM	CON	DP	CCMB1
	CCM	14/14	0	0	0	0	0
	WSCCMB1	0	19/19	0	0	0	0
	4TCM	0	0	14/14	0	0	0
	CON	0	0	0	10/10	0	0
	DP	0	0	0	0	8/8	
	CCMB1	0	0	0	0		21/21

Predictions

tested on a proof-of-concept setting (of scenarios) or framework that shall illustratively validate the concepts developed in this work and demonstrate their efficiency.

A. TEST DATASET

Here, we consider different test scenarios where we take one single circuits from the test dataset. We first demonstrate how the trained GAT model performs without applying DACA and then show how the test performance progressively improves with the gradual application of DACA to the training set until it reaches 100%.

TABLE 6. (a). Confusion matrix for FCOPAMP2: a 100% accuracy of clustering was achieved by applying DP-based augmentation, CCM-based augmentation, and CCMB1-based augmentation.

True Labels		CCM	CMB2	CON	DP	CCMB1
	CCM	28/28	0	0	0	0
	CMB2	0	12/12	0	0	0
	CON	0	0	10/10	0	0
	DP	0	0	0	8/8	0
	CCMB1	0	0	0	0	19/19

Predictions

1) FOLDED CASCODE OPERATIONAL AMPLIFIER FCOPAMP2

The Folded Cascode Operational Amplifier FCOPAMP2, explained in Appendix B-B, consists of 5 subschematics. As shown in Figure 17, it consists of the following subschematics:

1) Current Mirror Bank 2
2) Cascode Current Mirror
3) Differential Pair
4) Cascode Current Mirror
5) Cascode Current Mirror Bank 1

After applying respectively a DP-based (combinatorial) augmentation, a CCM-based (combinatorial) augmentation, followed by a CCMB1-based (combinatorial) augmentation, a 100% accuracy of the clustering (i.e. structure recognition) was achieved, as shown in Table 6, and all nodes were clustered either into the correct respective subschematics or as connectivity nodes (CON) between two subschematics.

2) FOLDED CASCODE OPERATIONAL AMPLIFIER FCOPAMP3

The Folded Cascode Operational Amplifier FCOPAMP3, explained in Appendix B-C, consists of 5 subschematics.

TABLE 7. (a). Confusion matrix for FCOPAMP3: a 100% accuracy of clustering was achieved by applying DP-based augmentation, LSB1-based augmentation, and CCM-based augmentation.

True Labels / Predictions	CCM	CMB1	CON	DP	DP
CCM	14/14	0	0	0	0
CMB1	0	18/18	0	0	0
LSB1	0	0	9/9	0	0
CON	0	0	0	9/9	0
DP	0	0	0	0	8/8

TABLE 8. (a). Confusion matrix for FCOPAMP4: a 100% accuracy of clustering was achieved by applying DP-based augmentation, CCMB1-based augmentation, and CCM-based augmentation.

True Labels / Predictions	CCM	CMB1	CON	DP	CCMB1
CCM	14/14	0	0	0	0
CMB1	0	9/9	0	0	0
CON	0	0	8/8	0	0
DP	0	0	0	8/8	0
CCMB1	0	0	0	0	19/19

As shown in Figure 18, it consists of the following sub-schematics:

1) Current Mirror Bank 1
2) Level Shifter Bank 1
3) Cacode Current Mirror
4) Current Mirror Bank 1
5) Differential Pair

After applying respectively a DP-based (combinatorial) augmentation, followed by a LSB1-based (combinatorial) augmentation, and then followed by a CCM-based (combinatorial) augmentation, a 100% accuracy of the clustering was achieved, as shown in Table 7, and all nodes were clustered into either the correct sub-schematics or as a connectivity node (CON) between two subschematics.

3) FOLDED CASCODE OPERATIONAL AMPLIFIER FCOPAMP4

The Folded Cascode Operational Amplifier FCOPAMP4, explained in Appendix B-D, consists of 4 subschematics. As shown in Figure 19, it consists of the following sub-schematics:

1) Cascode Current Mirror Bank 1
2) Cacode current mirror
3) Current Mirror Bank 1
4) Differential Pair

After applying respectively a DP-based (combinatorial) augmentation, followed by a CMB1-based (combinatorial) augmentation, and then followed by a CCM-based (combinatorial) augmentation, a 100% accuracy of the clustering was achieved, as shown in Table 8, and all nodes were clustered into either the correct sub-schematics or as a connectivity node (CON) between two subschematics.

B. BENCHMARK DATASET

To test our model we used an independent dataset of level-3 schematics and we have performed our method and showed the results in the previous subsection. In this subsection, we perform structure recognition on a new set of schematics to identify the effectiveness of our method and we call this new group of samples by the benchmark dataset. This dataset contains two samples of Operational Transconductance Amplifiers OTAs and two samples of Operational Amplifiers OPAMPs.

TABLE 9. (a). Confusion matrix for OTA2: a 100% accuracy of clustering was achieved by applying CMB1-based augmentation.

True Labels / Predictions	CMB1	DP	CM	CON
CMB1	10/10	0	0	0
DP	0	8/8	0	0
CM	0	0	6/6	0
CON	0	0	0	4/4

TABLE 10. (a). Confusion matrix for OTA1: a 100% accuracy of clustering was achieved by applying CMB1-based augmentation.

True Labels / Predictions	CMB1	DP	CM	CON
CMB1	9/9	0	0	0
DP	0	8/8	0	0
CM	0	0	12/12	0
CON	0	0	0	6/6

1) OPERATIONAL TRANSCONDUCTANCE AMPLIFIER OTA 1

The Operational Transconductance Amplifier OTA 1, explained in Appendix B-E, consists of 3 subschematics. As shown in Figure 20, it consists of the following subschematics:

1) Current Mirror Bank 1
2) Differential Pair
3) Current Mirror

After applying CMB1-based (combinatorial) augmentation, a 100% accuracy of clustering was achieved, as shown in Table 9. All nodes were clustered into either the correct subschematics or as a connectivity node (CON) between two subschematics.

2) OPERATIONAL TRANSCONDUCTANCE AMPLIFIER OTA 2

The Operational Transconductance Amplifier OTA 2, explained in Appendix B-F, consists of 3 subschematics. As shown in Figure 21, it consists of the following subschematics:

1) Current Mirror Bank 1
2) Differential Pair
3) Current Mirror

After applying CMB1-based (combinatorial) augmentation, a 100% accuracy of clustering was achieved, as shown in Table 10, and all nodes were clustered into either the correct subschematic or as a connectivity node (CON) between two subschematic.

TABLE 11. Confusion matrix for OPAMP1: a 100% accuracy of clustering was achieved by applying LS-based augmentation.

True Labels	DP	CM	LS	CON	CMB2
DP	8/8	0	0	0	0
CM	0	12/12	0	0	0
LS	0	0	6/6	0	0
CON	0	0	0	8/8	0
CMB2	0	0	0	0	13/13
			Predictions		

TABLE 12. Confusion matrix for OPAMP2: a 97% accuracy of clustering was achieved by applying CCM-based augmentation.

True Labels	CCM	DP	CM	CON	CMB2
CCM	14/14	0	0	0	0
DP	0	8/8	0	0	0
CM	0	0	6/6	0	0
CON	0	1/6	0	5/6	0
CMB2	0	0	0	0	13/13
			Predictions		

3) OPERATIONAL AMPLIFIER OPAMP 1

The Operational Amplifier OPAMP 1, explained in Appendix B-G, consists of 5 subschematics. As shown in Figure 21, it consists of the following subschematics:

1) Current Mirror Bank 2
2) Level Shifter
3) Current Mirror
4) Current Mirror
5) Differential Pair

After only applying LS-based (combinatorial) augmentation, a 100% accuracy of clustering was achieved, as shown in table 11, and all nodes were clustered into either the correct subschematics or as a connectivity node (CON) between two subschematics.

4) OPERATIONAL AMPLIFIER OPAMP2

The Operational Amplifier OPAMP 2, explained in B-H, consists of four subschematics. As shown in Figure 21, it consists of the following subschematics:

1) Current Mirror Bank 2
2) Cascode Current Mirror
3) Current Mirror
4) Differential Pair

After applying only one CCM-based (combinatorial) augmentation, a 97% accuracy of clustering was achieved, as shown in Table 12. Except for one node, all nodes were clustered into the correct subschematics or as a connectivity node (CON) between two subschematics.

C. DISCUSSION

The analog circuit structure recognition problem has been converted into a graph clustering problem as shown earlier. In other words, the task of finding the correct sub-schematics in a schematic has been converted into clustering the circuit graph into the equivalent subgraphs. By performing node classification and correctly classifying each node to its proper subgraph, the equivalent sub-schematics are recognized perfectly.

The classification model chosen to solve this classification problem is the graph attention network GAT. Hence the evaluation metric is the overall classification accuracy. A confusion matrix is needed to visualize the classification performance by showing how many nodes were classified correctly and how many weren't.

In general, the test set results, discussed in described in Section VIII-A, are fully satisfactory w.r.t. our specification book. The model was then used to perform structure recognition on totally new dataset, as described in VIII-B. Figure 15 summarizes the results of the test dataset, in blue, and those of the benchmark dataset in green. The x-axis represents how many subschematics were used for subschematic-based augmentation, and the y-axis represents the overall accuracy. Figure 15 shows that a one-step single subschematic-based combinatorial augmentation is not always enough for reaching 100% accuracy of clustering. In the OTAs cases, it is necessary to perform two steps of subschematic-based combinatorial augmentation to reach 100%, and for other cases we needed to perform three steps of subschematic-based combinatorial augmentation. The main reason for low accuracy lies in the unbalanced subschematics in the original dataset w.r.t. some of the relevant subschematics as described in Subsection VII-D.

In prior exploratory experiments, we have tried a model combining K-means followed by GCN for an unsupervised clustering as recommended by [3]. One of the main drawbacks of that (unsupervised learning) approach is the need for indicating the number of clusters as an input of the k-means algorithm. Moreover, even if the number of clusters were correct, the clusters were not always detected correctly.

The implication of a wrong structure recognition is indeed quite serious in the analog circuit domain since it might lead to wrong verification or design.

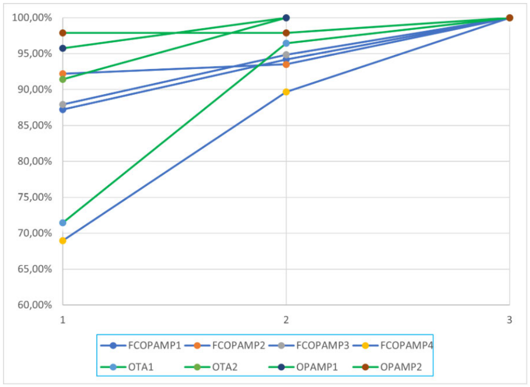


FIGURE 15. Clustering performance Enhancement through progressive novel augmentation: The number on the X-axis refers to how many subschematic-based step-wise augmentations were performed, and the percentage on the Y-axis refers to the overall clustering accuracy. The blue lines represent the test dataset and the green ones represent the benchmark dataset. Both datasets samples show enhancement after applying more augmentation steps.

TABLE 13. A comprehensive summary of how far all requirements of the specification dossier have been fully fulfilled, and a brief comparison to the respective related works.

REQ-ID from the requirements Engineering Dossier	What this paper has specifically done w.r.t. to REQ-ID	A brief commenting of a sample of relevant related work(s)	General comments
REQ-1	In this work, we managed to reach 100% accuracy of clustering	[21] [3] achieved above 97% accuracy of clustering but [21] did not	our paper performed clearly better than the relevant related works w.r.t. REQ-1
REQ-2	Our model is robust w.r.t. topology variations of circuits having the same label.	Some models are robust against differences in circuit topology and are able to accurately identify the subcircuits regardless of the variations in their structure that occur within a single class, this could be achieved by [3], and [21], [22], if their pipelines were adapted accordingly.	This paper does as well as or better than the relevant related works w.r.t. REQ-2
REQ-3	Our model is not sensitive to transistor technology changes in dataset samples classified as the same class (NMOS, PMOS, or Bipolar).	This was fulfilled in some sources such as [3], [21], and others.	This paper mostly outperforms the relevant related works w.r.t. REQ-3
REQ-4	The most recent research findings were presented, each of which was derived from a larger dataset than the one we used.	None of the relevant references fulfilled this requirement	This paper outperforms the relevant related works w.r.t. REQ-4 when a dataset is used for training
REQ-5	A proof of concept was presented in this paper, and the pipeline is scalable to function in a higher level	this could/may be achieved by [3], and [21], [22], if their pipelines were adapted accordingly.	This paper outperforms the relevant related works w.r.t. REQ-5
REQ-6	A comprehensive representation of transistors was presented in this paper where drain, gate, and source were represented in three unique nodes in the graph domain	All mentioned references represented a transistor with a single node except for [13]	This paper outperforms the relevant related works w.r.t. REQ-6
REQ-7	The used GAT model is adaptable w.r.t the graph input's size which is equivalent to the analog circuit's size	This is achieved in some sources, such as [23], [24] and others.	This paper outperforms the relevant related works w.r.t. REQ-7
REQ-8	An automatic labeling was performed to give similar subcircuits different labels and therefor be clustered separately	This could be achieved in some sources such as [23], [24] and others	This paper outperforms the relevant related works w.r.t. REQ-8
REQ-9	The GAT Model can be progressively trained (through a progressive balancing augmentation) to improve performance	This was not fulfilled in any of the related references	This paper outperforms the relevant related works w.r.t. REQ-9
REQ-10	The GAT Model can be trained to integrate new subblock(s) by simply adding training samples of the new subblock	This requirement cannot / was not be fulfilled in any of the related works.	This paper outperforms the relevant related works w.r.t. REQ-10
REQ-11	Our GAT-based model is compatible with prevalent EDA tools and platforms and is efficient enough for real-time or near-real-time operation during circuit design processes	This could be achieved by some sources such as [23], and [24] if their pipelines were adapted accordingly	This paper outperforms the relevant related works w.r.t. REQ-11

Thus, the novel concept presented in this paper represent a very significant improvement because this is a library-based approach and the number of subschematics is limited. Hence, it is possible to build a dataset of all analog subschematics for a fully supervised structure recognition. On another level, a potential systematic generalization of the approach for higher functional schematics is possible.

IX. EXPERIMENTS ANATOMY FOR THE PIPELINE DEVELOPED - A COMPREHENSIVE RECAPITULATION

In order to perform the experiments, an extensive preprocessing has been done. For instance, the dataset was prepared by performing various tasks such as collecting real-world schematics from literature, dataset drawing, dataset annotation, dataset netlists generation, dataset Automatic Labeling (AL) and dataset Automatic Augmentation, dataset nodes feature extraction, and dataset graphs generation. In addition, model selection, model adaptation, and feature adaptation processes were achieved for AMS structure recognition.

Different experiments were planned in order to have real (extensive and stress-test) tests of the developed model on the AMS schematics. The experiments are done on five types of schematics:

1) Experiments on the test dataset samples after applying DP-based combinatorial augmentation.
2) Experiments on a single type of level 2 schematics after applying DP-based combinatorial augmentation.
3) Experiments on single type of level 2 schematics after applying at least one-subschematic-based combinatorial augmentation till reaching a minimum overall accuracy of more steps than 97%. More combinatorial augmentations have the potential of reaching an overall accuracy of 100%.
4) Experiments on totally new schematics considered as a benchmark dataset.

X. GENERAL EVALUATION OF THE FINAL DATA-AUGMENTATION ENHANCED GAT MODEL THROUGH A QUALITATIVE COMPARISON WITH RELEVANT RECENT WORKS FROM THE LITERATURE

The comprehensive and extensive hard engineering requirements addressed/formulated in Section II were totally satisfied by the final best pipeline that has been progressively constructed and presented in this work and by the advanced GNN model that supported it. We were able to get a classification accuracy of 100%. Therefore, the conditions for a very high accuracy were fully satisfied.

In Table 13 we do comprehensively summarize how far all elements of the comprehensive requirements dossier have been fully satisfied and thereby also put the finger on the respective limitations of the corresponding related most relevant related works or concepts from the literature.

XI. GENERAL CONCLUSION

The three initial tiers of the hierarchy of analog/mixed signal circuits IPs, described in Figure 6, makes it crystal clear why a proof of concept that has been verified for level-1/level-2 is readily adaptable to handle the other higher levels. The use of the same protocol or process developed is possible. In fact, all entities at levels beginning at level 1 and upwards have the traits of being expressed or described as a level-0 graph. Only their respective individual size and shapes greatly differ from one immediate upper layer to the next.

The comprehensive experiments conducted show that our model, after applying the (combinatorial) augmentation technique based on the subgraph cropping [30], can cluster schematics correctly into the basic building sub/blocks of the analog circuit schematics and does guarantee, always, the reaching of a 100% for all subblock labels. Further, this can be upscaled (adapted) towards a library-based analog substructure recognition, for simulator performance tuning, analog block-level verification, IP-level verification, layout routing, and layout placement.

Also, this study has evaluated the significance of the clustering of analog circuits in a variety of application domains. We have shown how analog circuit clustering without human supervision accelerates many applications in the semiconductor industry. This mainly saves time in layout placement and layout routing while generating any analog circuit design. After constructing the analog subblocks, an analog designer should connect these blocks to make an analog circuit. Layout placement and routing are the next steps in properly converting the design into a layout. Therefore, layout simulation and coupling in automated analog generators will be significantly sped up by knowing the names of the subblocks and, consequently, their functions. However, it is crucial to use simulators to solve differential equations in order to obtain correct results for analog circuit design. However, doing so either takes a lot of time or relies on differential equations that have previously been solved, which lowers the accuracy margin. Circuit classification could help the simulator locate a better level of differential equations that have already been resolved while retaining a high level of accuracy. It takes much less time to accomplish this than it would take to find a precise solution. Understanding the type of circuit or schematics can help analog designers choose the best test bench to generate stimuli, conduct verification, and conduct extra measurements. This reduces the amount of effort required by the designer by guaranteeing the circuit's functionality at the IP level.

A critical and comprehensive analysis of the current state-of-the-art in the relevant field was also provided in this paper. We have reviewed subcircuit detection or what is called subblock/structure recognition from 1995 till recent works in 2022 using graph convolution networks. We, however, have used the newest versions of graph neural networks (GNNs), which are graph attention networks (GATs.) A comprehensive comparative presentation of capabilities of this concept, augmented by a novel smart combinatorial dataset augmentation was provided in this paper.

In addition to conducting a literature review, we have formulated a comprehensive ontology, and then a pipeline, beginning with the SPICE netlists generated by the used tool Cadence Virtuoso, for the purpose of transforming a given analog circuit into a corresponding graph model. Additionally, we have developed a modeling workflow that makes use of graph attention networks in order to robustly/reliably9/scalably solve the problem of structure recognition in complex analog circuits.

A methodology has been created to identify L1 subblocks within L2 circuits using a graph attention network neural model. This method is a core component of the supervised graph learning approach we have developed. We first project the circuit into graph space using a parser. The input is a graph that shows the L2 circuit and is labeled for each node. The node type and degree centrality measures make up the distinctive features vector for each node. The Graph Attention Layers will use these features. A node list, which consists of transistors-related nodes and nets-related nodes, and an edge list are used to numerically build the graph. The edges take no features into account. We just utilize the node features while learning. The cluster or subblock to which each node belongs is labeled in its label. The basis dataset collection we have used contains 14 different L1 circuits that serve as labels. Every node in the circuit graph is labeled and expected to belong to a specific class in the pipeline, which is an end-to-end fully supervised node classification scheme. Multiple Graph Attention learning layers or processors make up the Graph Attention Network model.

Furthermore, we have suggested a novel augmentation strategy (in the face of the potentially size-limited and unbalanced training dataset(s) in practical settings): we develop and validate one innovative dataset augmentation concept, which has a clear and robust positive boosting effect on the performance of the suggested GAT-based system. The technique used mainly focused on creating new netlists out of the original netlist, providing new artificial samples to enrich the training dataset. This method is equivalent to the subgraph cropping method for graph data augmentation. For example, one L2 schematics in the dataset is a Folded Cascode Operational Amplifier, and it has 5 L1 subblocks. By applying our method of augmentation, we have created 14 complementary L2 samples on the netlist level. A total of 76 samples were thus created out of 8 L2 schematics. Furthermore, we made sure that the L1 subblocks are balanced in the dataset samples for each test use-case. The core findings were summarized in a conceptual and qualitative comparison with the related state-of-the-art competing concepts from the literature. Throughout the illustraive experiments multiple scenarios were described, and they were gradually improved through gradually optimizing/tuning/augmenting the datasets in each consecutive scenario. We have shown how and why we designed each scenario in order to reach 100% overall accuracy. This was done in order to better emphasize the novel nature, strong robustness and the accuracy-related full scalability of the concept that was developed in this paper.

APPENDIX A
Net2Graph ALGORITHM

We define 3 levels for graphs to be considered in this work. The so-called "basic building blocks" are represented in level-1 graphs; the so-called "functional building blocks" are represented in level-2 graphs, and the so-called "modules" are represented in level-3 graphs. The parser Net2Graph was utilized to recognize level-1 elements within a level-2 circuit to scale it to higher levels. The parser also considers a label file depending on the requirements of the experiments. For each netlist and label, the parsed output is "noitemsep" and consists of the following elements:

noitemsep

- Adjacency matrix A: describes the connectivity of the graph nodes.
- Feature matrix X: contains the node features of the graph.
- Labels L: vectors of labels depending on the problem formulation
- Graph object G: networkX python object contains all the information from the netlist

The algorithm utilizes a regular expression functionalities, a regular expression is a sequence of characters that specifies a search pattern in the text. Usually, such patterns are used by string-searching algorithms for "find" or "find and replace" operations on strings, or for input validation. Internally the parser maintains dictionaries describing each component and its connectivity also the features from the netlist. The feature of each node in the graph or component is its type.

APPENDIX B
DATASET DESCRIPTION

This appendix consists of a detailed description of the dataset samples which were used either as a test dataset or in the benchmark dataset.

Algorithm 1 Net2Graph

```
Data: Titan Netlist,Netlist Label
Result: A,X,L,G
initialization;
read netlist file;
read netlist label file if available;
while not at the end of netlist file do
    read the current line and match transistor library;
    if match is found then
        get elements and connected nets;
        if element is transistor then
            generate drain, gate, and source nets;
            added to the components dictionary;
        else
            consider an element as net;
            add it to the components dictionary;
        end
    else
        file, not a netlist exit;
    end
end
for N Number of Isomorphic graph do
    Initialized graph new G variant
    for component in components dictionary do
        add node randomly to new initialized graph G;
        add edges between the current node and other
          nodes in graph object;
        set node attributes from features based on
          type;
        if node label available then
            add node labels as attributes;
            append node label to labels vector L;
        else
            none;
        end
        append node to nodes list; append features
          vector to feature matrix X
    end
    append the G to the list of Gs Isomorphic.
end
return N Isomorphic Graphs
```

A. FOLDED CASCODE OPERATIONAL AMPLIFIER FCOPAMP1

The first schematic used in the test dataset is the Folded Cascode Operational Amplifier FCOPAMP1. As shown in Figure 16, it consists of the following subschematics:

1) 4-Transistor Current Mirror.
2) Wilson Cascode Current Mirror Bank 1.
3) Differential Pair.
4) Cascode Current Mirror Bank 1.
5) Cascode Current Mirror.

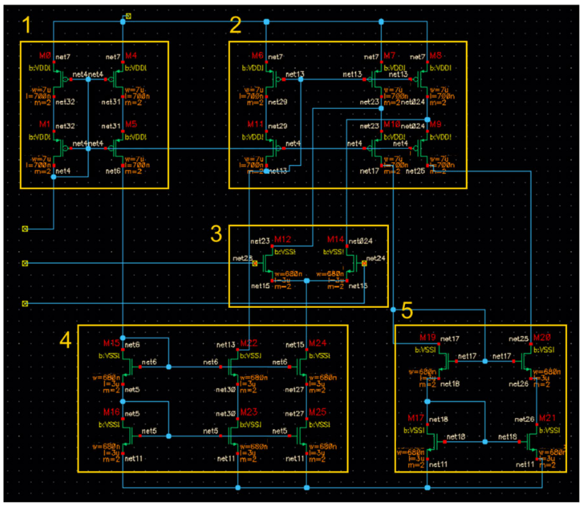


FIGURE 16. **Folded Cascode Operational Amplifier FCOPAMP1 components clustered as follows: 1. 4-Transistor Current Mirror, 2. Wilson Cascode Current Mirror Bank 1, 3. Differential Pair, 4. Cascode Current Mirror Bank 1, and 5.Cascode Current Mirror.**

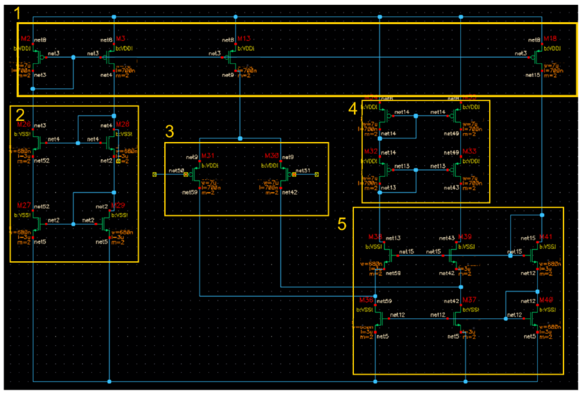


FIGURE 17. **Folded Cascode Operational Amplifier FCOPAMP2 components clustered as follows: 1. Current Mirror Bank 2, 2. Cascode Current Mirror, 3. Differential Pair, 4. Cascode Current Mirror, and 5.Cascode Current Mirror Bank 1.**

B. FOLDED CASCODE OPERATIONAL AMPLIFIER FCOPAMP2

The second schematic used in the test dataset is the Folded Cascode Operational Amplifier FCOPAMP2. As shown in Figure 17, it consists of the following subschematics:

1) Current Mirror Bank 2
2) Cascode Current Mirror
3) Differential Pair
4) Cascode Current Mirror
5) Cascode Current Mirror Bank 1

C. FOLDED CASCODE OPERATIONAL AMPLIFIER FCOPAMP3

The third schematic used in the test dataset is the Folded Cascode Operational Amplifier FCOPAMP3. As shown in Figure 18, it consists of the following subschematics:

1) Current Mirror Bank 1

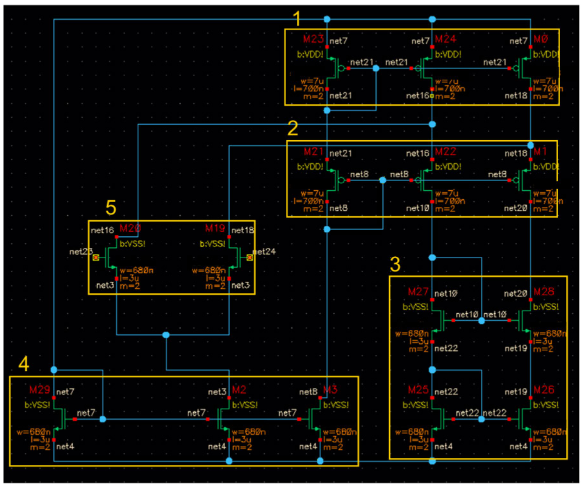


FIGURE 18. **Folded Cascode Operational Amplifier FCOPAMP3 components clustered as follows: 1. Current Mirror Bank 1, 2. Level Shifter Bank 1, 3. Cascode Current Mirror, 4. Current Mirror Bank 1, and 5. Differential Pair.**

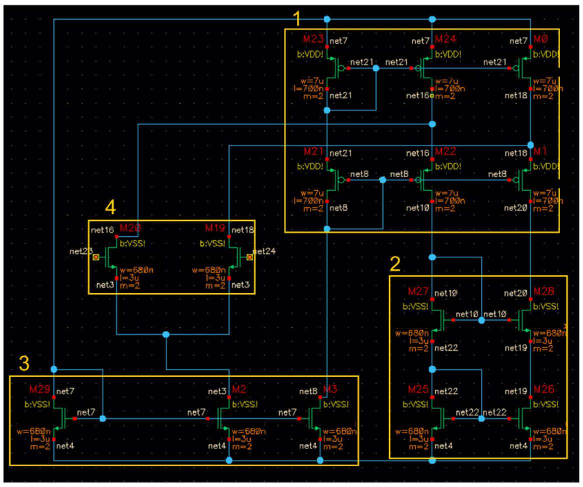


FIGURE 19. **Folded Cascode Operational Amplifier FCOPAMP4 components clustered as follows: 1. Cascode Current Mirror Bank 1, 2. Cacode current mirror, 3. Current Mirror Bank 1, and 4. Differential Pair.**

2) Level Shifter Bank 1
3) Cacode Current Mirror
4) Current Mirror Bank 1
5) Differential Pair

D. FOLDED CASCODE OPERATIONAL AMPLIFIER FCOPAMP4

The forth schematic used in the test dataset is the Folded Cascode Operational Amplifier FCOPAMP4. As shown in Figure 19, it consists of the following subschematics:

1) Cascode Current Mirror Bank 1
2) Cacode current mirror
3) Current Mirror Bank 1
4) Differential Pair

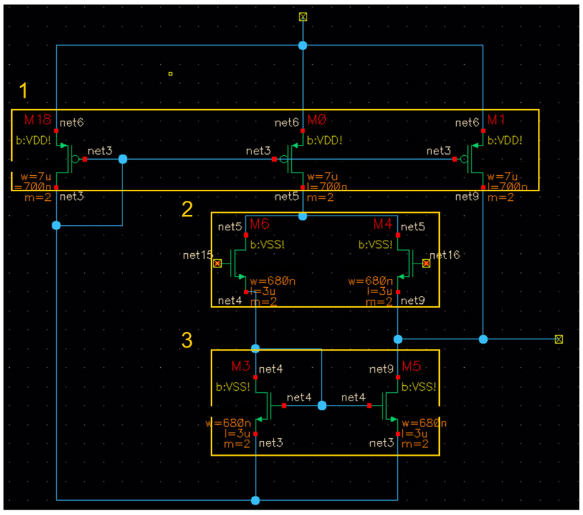


FIGURE 20. **Operational Transconductance Amplifier OTA 1 components clustered as follows: 1. Current Mirror Bank 1, 2. Differential Pair, and 3. Current Mirror.**

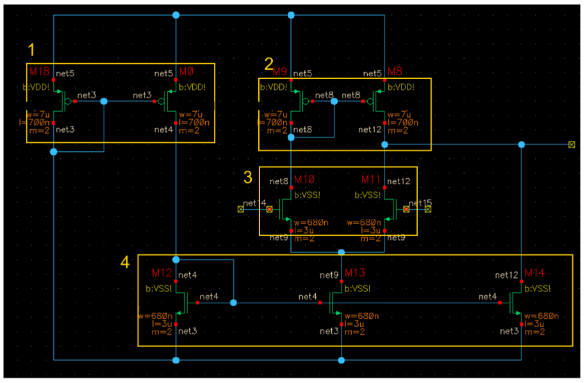


FIGURE 21. **Operational Transconductance Amplifier OTA 2 components clustered as follows: 1. Current Mirror, 2. Current Mirror, 3. Differential Pair, and 4. Current Mirror Bank 1.**

E. OPERATIONAL TRANSCONDUCTANCE AMPLIFIER OTA 1

The fifth schematic, was used in the benchmark dataset, is the Operational Transconductance Amplifier OTA 1. As shown in Figure 20, it consists of the following subschematics:

1) Current Mirror Bank 1
2) Differential Pair
3) Current Mirror

F. OPERATIONAL TRANSCONDUCTANCE AMPLIFIER OTA 2

The sixth schematic, was used in the benchmark dataset, is the Operational Transconductance Amplifier OTA 2. As shown in Figure 21, it consists of the following subschematics:

1) Current Mirror
2) Current Mirror
3) Differential Pair
4) Current Mirror Bank 1

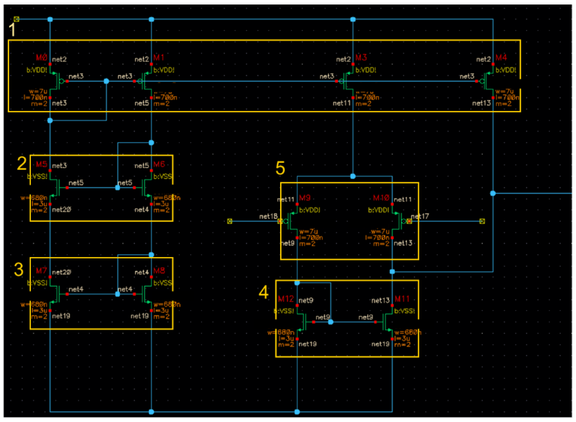


FIGURE 22. **Operational Transconductance Amplifier OTA 2 components clustered as follows: 1. Current Mirror Bank 2, 2. Level Shifter, 3. Current Mirror, 4. Current Mirror, and 5. Differential Pair.**

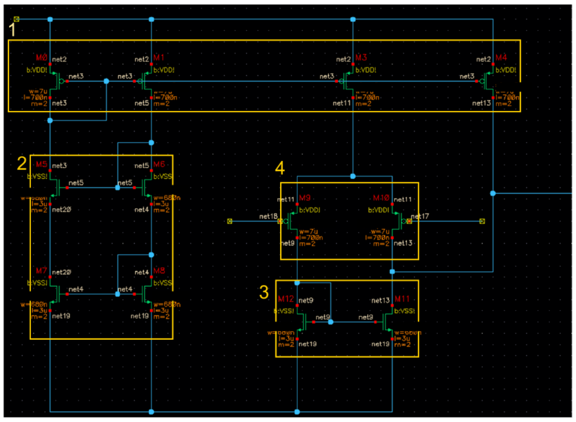


FIGURE 23. **Operational Transconductance Amplifier OTA 2 components clustered as follows: 1. Current Mirror Bank 2, 2. Cascode Current Mirror, 3. Current Mirror, and 4. Differential Pair.**

G. OPERATIONAL AMPLIFIER OPAMP 1

The seventh schematic, was used in the benchmark dataset, is the Operational Amplifier OPAMP 1. As shown in Figure 22, it consists of the following subschematics:

1) Current Mirror Bank 2
2) Level Shifter
3) Current Mirror
4) Current Mirror
5) Differential Pair

H. OPERATIONAL AMPLIFIER OPAMP2

The eighth schematic, was used in the benchmark dataset, is the Operational Amplifier OPAMP2. As shown in Figure 23, it consists of the following subschematics:

1) Current Mirror Bank 2
2) Cascode Current Mirror
3) Current Mirror
4) Differential Pair

APPENDIX C

TABLE 1 GAP ANALYSIS JUSTIFICATION

A. MEISSNER ET AL

In this subsection we justify how we filled the line of Meissner and Hedrich [10] in table 1. We simply analyze how this work performed analog circuits' structure recognition based on our requirements engineering shown in section II.

- REQ-1 (Accuracy Baseline): The paper does not seem to mention a specific accuracy baseline.
- REQ-2 (Topology Variability Robustness): The methodology addresses flexibility and aims to handle multiple circuit classes.
- REQ-3 (Transistor Type Variability Robustness): No specific mention of different transistor types or their variability robustness.
- REQ-4 (Dataset Resilience): No mention of datasets or resilience in that context.
- REQ-5 (Hierarchical Scalability): no good device representation which makes it difficult to upscale to higher levels in the AMS hierarchy.
- REQ-6 (Standardized Representation): it's not clear if it adheres to any standardized representation.
- REQ-7 (Size Scalability): There's no direct mention of how this methodology scales with size, so No
- REQ-8 (Similar Topology Handling): The paper mentions the development of an isomorphism algorithm which essentially handles the problem of similar topologies by finding isomorphic subgraphs.
- REQ-9 (Adaptive Training): There doesn't appear to be any adaptive training involved.
- REQ-10 (Model Extendibility): No. The introduction of a new subblock requires the adaptive training mentioned in REQ-9. Thus a model that does not fulfill REQ-9 cannot satisfy here in REQ-10
- REQ-11 (Operational Environment Compatibility): No direct mention of compatibility with different operational environments.

B. MASSIER ET AL

In this subsection we justify how we filled the line of Massier et al. [5] in table 1. We simply analyze how this work performed analog circuits' structure recognition based on our requirements engineering shown in section II.

- REQ-1 (Accuracy Baseline): The paper does not seem to mention a specific accuracy baseline. It fails to provide more than 71.1% for CMOS buffer amplifier as shown in table 7
- REQ-2 (Topology Variability Robustness): it is mentioned that the model fails to maintain high accuracy when changing topology.
- REQ-3 (Transistor Type Variability Robustness): The paper discusses both CMOS and bipolar transistors, indicating an ability to handle variability in transistor types.
- REQ-4 (Dataset Resilience): There's no mention of datasets or resilience in the context.
- REQ-5 (Hierarchical Scalability): different circuit levels were mentioned in the paper. See fig.3.
- REQ-6 (Standardized Representation): The paper does not show a transistor representation.
- REQ-7 (Size Scalability): The paper's core theme revolves around sizing, indicating an emphasis on scalability in that dimension.
- REQ-8 (Similar Topology Handling): it is shown in figure 4 how the model handled 2 different current mirrors. The model is rule based and able to find similar topologies in the same circuit.
- REQ-9 (Adaptive Training): There's no mention of adaptive training.
- REQ-10 (Model Extendibility): No. The introduction of a new subblock requires the adaptive training mentioned in REQ-9. Thus a model that does not fulfill REQ-9 cannot satisfy here in REQ-10
- REQ-11 (Operational Environment Compatibility): The paper does not provide specific information on the compatibility with different operational environments

C. OHLRICH ET AL

In this subsection, we justify how we filled the line of Ohlrich et al. [11] in table 1. We simply analyze how this work performed analog circuits' structure recognition based on our requirements engineering shown in section II.

- REQ-1 (Accuracy Baseline) The paper highlights that their algorithm is fast in practice for real circuits, but an explicit accuracy percentage isn't given.
- REQ-2 (Topology Variability Robustness) The technology-independent nature of the proposed algorithm and its focus on hierarchical matching across different subcircuit structures implies a robustness to different circuit topologies.
- REQ-3 (Transistor Type Variability Robustness) The paper present technology independent method which means it is valis for all types (technologies of the transistors)
- REQ-4 (Dataset Resilience) There's no mention of dataset challenges or any techniques
- REQ-5 (Hierarchical Scalability) The algorithm was not clearly applied on different levels of the hierarchy.
- REQ-6 (Standardized Representation) entities of all levels were not capable to be represented with multiple nodes.
- REQ-7 (Size Scalability) The typical running time for large CMOS circuits is mentioned to be approximately linear in relation to the total number of devices within the subcircuits being matched, indicating scalability with respect to circuit size.
- REQ-8 (Similar Topology Handling) it is shown in some figures in the paper multiple similar topologies such as CM were in the same circuit.
- REQ-9 (Adaptive Training) The paper doesn't discuss an adaptive or progressive training mechanism for the algorithm.

- REQ-10 (Model Extendibility) The ability to describe constructs as circuits in a library which can be easily extended suggests a level of extendibility.
- REQ-11 (Operational Environment Compatibility) There's no explicit mention of compatibility with prevalent EDA tools and platforms or real-time operation during circuit design processes.

D. PELZ ET AL

In this subsection, we justify how we filled the line of Pelz and Roettcher [12] in table 1. We simply analyze how this work performed analog circuits' structure recognition based on our requirements engineering shown in section II.

- REQ-1 (Accuracy Baseline): The abstract and provided sections don't give explicit information regarding the structure recognition accuracy.
- REQ-2 (Topology Variability Robustness): The method is described as independent of circuit technology and design style, indicating that it has robustness across different circuit topologies.
- REQ-3 (Transistor Type Variability Robustness): The paper includes circuits with different transistor types.
- REQ-4 (Dataset Resilience): There's no explicit mention of handling dataset challenges or employing advanced augmentation techniques.
- REQ-5 (Hierarchical Scalability): The core of this paper revolves around hierarchical processing and matching, making it evident that it has a hierarchical approach.
- REQ-6 (Standardized Representation): No clear representation for the transistors in the paper.
- REQ-7 (Size Scalability): There's no clear mention of the model being versatile for both small and large circuits.
- REQ-8 (Similar Topology Handling): The focus on pattern matching of arbitrary subcircuits in larger circuits suggests that the model has capabilities in this domain.
- REQ-9 (Adaptive Training): The provided content doesn't mention progressive or adaptive training mechanisms.
- REQ-10 (Model Extendibility): The hybrid approach, combining pattern matching with traditional refinement, suggests some extendibility, but specifics on adding new patterns or "forgetting" old ones aren't given.
- REQ-11 (Operational Environment Compatibility): The system was run on a SUN SPARC 1, and the internal representation utilizes SPICE format. However, there's no explicit mention of real-time operation or compatibility with contemporary EDA tools.

E. HUANG ET AL

In this subsection, we justify how we filled the line of Huang and Overhauser [13] in table 1. We simply analyze how this work performed analog circuits' structure recognition based on our requirements engineering shown in section II.

- REQ-1 (Accuracy Baseline): The paper mentions an improved recognition algorithm and claims that it provides a unique way to represent circuits, but the exact accuracy isn't mentioned.
- REQ-2 (Topology Variability Robustness):The paper mentions that their method is suitable for mixed-signal, BiCMOS, bipolar, and analog circuits.
- REQ-3 (Transistor Type Variability Robustness):The paper discusses different types of circuits and implies that the approach handles multiple types of transistors. The technology file approach also allows for customization.
- REQ-4 (Dataset Resilience):There's no explicit mention of the system's ability to handle varying datasets or how resilient it is to data inconsistencies.
- REQ-5 (Hierarchical Scalability): The paper doesn't discuss hierarchical designs or the ability to scale across different hierarchy levels.
- REQ-6 (Standardized Representation):The paper introduces a new representation for circuit graphs and unique codes for circuits.
- REQ-7 (Size Scalability):The recognition is described to be device size independent, and they've tested on circuits with up to 15,000 transistors.
- REQ-8 (Similar Topology Handling):The method is described to generate unique codes for each circuit, implying it can handle and differentiate similar topologies.
- REQ-9 (Adaptive Training): There's no mention of any adaptive training or learning methods in the provided text.
- REQ-10 (Model Extendibility): No. The introduction of a new subblock requires the adaptive training mentioned in REQ-9. Thus a model that does not fulfill REQ-9 cannot satisfy here in REQ-10
- REQ-11 (Operational Environment Compatibility):The paper emphasizes the technology file's adaptability, suggesting compatibility with different operational environments.

F. RUBANOV ET AL

In this subsection, we justify how we filled the line of Rubanov [14] in table 1. We simply analyze how this work performed analog circuits' structure recognition based on our requirements engineering shown in section II.

- REQ-1 (Accuracy Baseline):The paper mentions that the new labeling algorithm improves the SR results in comparison with standard labeling methods.
- REQ-2 (Topology Variability Robustness):The paper doesn't provide a clear indication of topology variability.
- REQ-3 (Transistor Type Variability Robustness):The paper doesn't mention specific transistor types.
- REQ-4 (Dataset Resilience): The paper doesn't discuss how the labeling algorithm handles inconsistent datasets.

- REQ-5 (Hierarchical Scalability): No mention of hierarchical designs or scalability across different levels.
- REQ-6 (Standardized Representation): The paper introduces a new labeling algorithm for BG vertices, implying a one node representation of each transistor. Hence, No.
- REQ-7 (Size Scalability): The experiments cover finding medium-sized cells, small gates, and large subcircuits, indicating scalability.
- REQ-8 (Similar Topology Handling): The paper discusses recognizing subcircuits even when external nets were shorted, indicating it can differentiate and handle similar topologies.
- REQ-9 (Adaptive Training): There's no mention of any adaptive training methods.
- REQ-10 (Model Extendibility):The paper doesn't clearly indicate the ease of extending the model to handle new scenarios.
- REQ-11 (Operational Environment Compatibility):While the technology for SR is discussed, specific operational environments are not mentioned.

G. RUBANOV ET AL

In this subsection, we justify how we filled the line of Rubanov [15] in table 1. We simply analyze how this work performed analog circuits' structure recognition based on our requirements engineering shown in section II.

- REQ-1 (Accuracy Baseline): The paper indicates that the PMAA recognizes 100% of subcircuit instances, meeting the accuracy baseline. This is further emphasized by its outperformance against SubGemini in real-world tests.
- REQ-2 (Topology Variability Robustness): The method is applied to various subcircuits, from combinational gates to memory cells, suggesting robustness across diverse topologies.
- REQ-3 (Transistor Type Variability Robustness): The paper does mention recognizing different logic gate types, including dynamic gates, flip-flops, etc., but there's no explicit mention of different transistor types (like NMOS, PMOS, etc.)
- REQ-4 (Dataset Resilience): The paper doesn't explicitly detail the challenges of the dataset used or the techniques employed to handle dataset challenges.
- REQ-5 (Hierarchical Scalability): Given the application on multi-million transistor circuits, the method likely scales across hierarchical levels. However, proof of four levels isn't provided explicitly.
- REQ-6 (Standardized Representation): One node representation for a transistor in a BG graph.
- REQ-7 (Size Scalability): The paper mentions experiments with circuits consisting of over 500,000 transistors, indicating scalability across various sizes.
- REQ-8 (Similar Topology Handling): PMAA's ability to distinguish between different subcircuits (like different types of gates) in large circuits indicates robust handling of similar topologies.
- REQ-9 (Adaptive Training): There's no mention of any adaptive training or learning methods in the provided text.
- REQ-10 (Model Extendibility): No. The introduction of a new subblock requires the adaptive training mentioned in REQ-9. Thus a model that does not fulfill REQ-9 cannot satisfy here in REQ-10
- REQ-11 (Operational Environment Compatibility): The integration of PMAA into DynaCore, a real-world product by Circuit Semantics, Inc., suggests compatibility with real-world EDA tools.

H. RUBANOV ET AL

In this subsection, we justify how we filled the line of Rubanov [16] in table 1. We simply analyze how this work performed analog circuits' structure recognition based on our requirements engineering shown in section II.

- REQ-1 (Accuracy Baseline): The paper suggests that the proposed method identifies all subcircuit instances, implying high accuracy of 100%.
- REQ-2 (Topology Variability Robustness): The paper discusses the algorithm's application to various subcircuits, such as combinational gates and flip-flops, suggesting robustness across different topologies.
- REQ-3 (Transistor Type Variability Robustness): The paper does mention recognizing different logic gate types, including dynamic gates, flip-flops, etc., but there's no explicit mention of different transistor types (like NMOS, PMOS, etc.)
- REQ-4 (Dataset Resilience): The paper doesn't explicitly detail the challenges of the dataset used or the techniques employed to handle dataset challenges.
- REQ-5 (Hierarchical Scalability): The paper discusses application on circuits with around 500,000 transistors, indicating a scalability across hierarchies.
- REQ-6 (Standardized Representation): One node representation for a transistor in a BG graph
- REQ-7 (Size Scalability): The mention of experiments with circuits consisting of around 500,000 transistors suggests scalability across different sizes.
- REQ-8 (Similar Topology Handling): The paper's methodology emphasizes identifying identical subgraphs in larger circuits, suggesting the capability to handle similar topologies.
- REQ-9 (Adaptive Training): There's no mention of any adaptive training or learning methods in the provided text.
- REQ-10 (Model Extendibility): No. The introduction of a new subblock requires the adaptive training mentioned in REQ-9. Thus a model that does not fulfill REQ-9 cannot satisfy here in REQ-10
- REQ-11 (Operational Environment Compatibility): The integration of PMAA into DynaCore, a real-world

product by Circuit Semantics, Inc., suggests compatibility with real-world EDA tools.

I. CONN ET AL

In this subsection, we justify how we filled the line of Conn et al. [17] in table 1. We simply analyze how this work performed analog circuits' structure recognition based on our requirements engineering shown in section II.

- REQ-1 (Accuracy Baseline): The paper describes the optimization and tuning of circuits, but doesn't provide explicit accuracy metrics for structure recognition.
- REQ-2 (Topology Variability Robustness): The paper seems to handle a wide variety of gates and can be extended to arbitrary custom circuits, suggesting robustness to different topologies.
- REQ-3 (Transistor Type Variability Robustness): The paper does mention accommodating transistor-level schematics, but specific details about handling various transistor types are not clear from the given excerpts.
- REQ-4 (Dataset Resilience): No mention of handling dataset challenges was found in the excerpts.
- REQ-5 (Hierarchical Scalability): The methodology appears to handle combinational circuits, and mentions about extensions for other circuit types. However, the details about hierarchical scalability aren't explicit in the given excerpts.
- REQ-6 (Standardized Representation): No specific mention of a standardized representation or adaptability to one was found.
- REQ-7 (Size Scalability): The paper describes using large-scale, nonlinear optimization, suggesting it can handle circuits of various sizes.
- REQ-8 (Similar Topology Handling): The method seems to handle recognition of a wide variety of gates, including those with similar topologies.
- REQ-9 (Adaptive Training): There was no indication in the excerpts about supporting progressive supervised-learning.
- REQ-10 (Model Extendibility): The introduction of a new subblock requires the adaptive training mentioned in REQ-9. Thus a model that does not fulfill REQ-9 cannot satisfy here in REQ-10
- REQ-11 (Operational Environment Compatibility): The paper does mention the method's implementation and potential for generality with certain tools, but explicit compatibility details with prevalent EDA tools and platforms aren't provided.

J. RUBANOV ET AL

In this subsection, we justify how we filled the line of Rubanov [18] in table 1. We simply analyze how this work performed analog circuits' structure recognition based on our requirements engineering shown in section II.

- REQ-1 (Accuracy Baseline): The algorithm successfully recognized all the subcircuit instances in the circuits. The use of the Gemini II algorithm to justify the SR results further confirms its accuracy.
- REQ-2 (Topology Variability Robustness): The experiments used a mix of combinational gates, sequential gates (latches and flip-flops), dynamic gates, and memory cells. This demonstrates a certain degree of topology variability robustness.
- REQ-3 (Transistor Type Variability Robustness): There isn't explicit mention about handling various transistor types, even if it references transistor counts.
- REQ-4 (Dataset Resilience): There's no mention of handling dataset challenges like limited size or label imbalance.
- REQ-5 (Hierarchical Scalability): No specific mention on hierarchical scalability or on handling multi-level hierarchical circuits.
- REQ-6 (Standardized Representation): One node representation only
- REQ-7 (Size Scalability): The method's efficiency for handling large circuits is indicated, with references to its capability for multimillion transistor or gate-level netlists.
- REQ-8 (Similar Topology Handling): Given the method is aimed at subcircuit recognition, it implicitly requires handling similar topologies, but there's no explicit mention of the challenges in distinguishing similar subcircuits.
- REQ-9 (Adaptive Training): The paper focuses on an optimization algorithm and does not mention any form of training, supervised or otherwise.
- REQ-10 (Model Extendibility): No specific mention about progressive training to incorporate new subblocks or patterns.
- REQ-11 (Operational Environment Compatibility): The paper's method was implemented in C++ and was tested on a standard PC configuration, which indicates some level of operational compatibility, but specifics regarding EDA tool compatibility are not provided.

K. LI ET AL

In this subsection, we justify how we filled the line of Li et al. [19] in table 1. We simply analyze how this work performed analog circuits' structure recognition based on our requirements engineering shown in section II.

- REQ-1 (Accuracy Baseline): The paper presents error corrections from recognizing redundant BBs, unfound BBs, and incorrect BBs, which indicates it emphasizes accuracy.
- REQ-2 (Topology Variability Robustness): The paper has examined multiple types of BBs (basic building blocks) from different circuits, indicating that it's robust to varying topologies.
- REQ-3 (Transistor Type Variability Robustness): The paper doesn't mention any details about handling different transistor types.

- REQ-4 (Dataset Resilience): The dataset consists of 34 analog circuits, but there's no specific mention of dataset challenges or variability.
- REQ-5 (Hierarchical Scalability): The paper discusses recognizing hierarchical BBs and hierarchical structures, indicating it supports hierarchical scalability.
- REQ-6 (Standardized Representation): There's no specific mention of standardized transistor representation used.
- REQ-7 (Size Scalability): While the paper mentions working with state-of-the-art analog circuits, there's no specific discussion about size scalability.
- REQ-8 (Similar Topology Handling): The paper mentions recognizing overlapping BBs and repetitive structures, indicating it can handle similar topologies.
- REQ-9 (Adaptive Training): The paper describes an unsupervised learning method which does not include adaptive training.
- REQ-10 (Model Extendibility): REQ-9 is not fulfilled. Hence, No.
- REQ-11 (Operational Environment Compatibility): The paper does not provide information regarding the operational environment or compatibility with specific EDA tools.

L. NEUNER ET AL

In this subsection, we justify how we filled the line of Neuner et al. [21] in table 1. We simply analyze how this work performed analog circuits' structure recognition based on our requirements engineering shown in section II.

- REQ-1 (Accuracy Baseline): The paper's model outperforms other methods in semi-supervised classification on graph-structured data.
- REQ-2 (Topology Variability Robustness): The paper doesn't specify if the method is robust to varying topologies, but it is evident that the method operates directly on graphs.
- REQ-3 (Transistor Type Variability Robustness): The paper focuses on graph-structured data and does not delve into transistor types or their variability.
- REQ-4 (Dataset Resilience): The paper mentions experiments on citation networks and a knowledge graph dataset but doesn't specify dataset challenges or variability.
- REQ-5 (Hierarchical Scalability): There's no specific mention of hierarchical scalability in the paper.
- REQ-6 (Standardized Representation): The paper does not provide information regarding the standardized circuit representations used.
- REQ-7 (Size Scalability): The paper discusses memory requirements and mentions that their model scales linearly with the number of graph edges. It also talks about the potential for mini-batch stochastic gradient descent for large graphs.
- REQ-8 (Similar Topology Handling): The approach uses a first-order approximation of spectral graph convolutions, which could be adept at handling similar topologies.
- REQ-9 (Adaptive Training): this paper does not introduce a step-wise progressive augmentation technique
- REQ-10 (Model Extendibility): REQ-9 is not fulfilled
- REQ-11 (Operational Environment Compatibility): The hardware used was mentioned, but it's unclear how this method would be compatible with specific operational environments or EDA tools.

M. SETTALURI ET AL

In this subsection, we justify how we filled the line of Settaluri et al. [22] in table 1. We simply analyze how this work performed analog circuits' structure recognition based on our requirements engineering shown in section II.

- REQ-1 (Accuracy Baseline): Only maximum accuracy was provided and it is 91.3% for one kind of circuits
- REQ-2 (Topology Variability Robustness): The method was tested on six different analog circuits and showed its versatility in different circuit topologies.
- REQ-3 (Transistor Type Variability Robustness): The paper does not explicitly mention handling circuits with mixed transistor types.
- REQ-4 (Dataset Resilience): The paper doesn't discuss how the model addresses dataset challenges like limited size or label imbalance.
- REQ-5 (Hierarchical Scalability): 'and our results show over 90% accuracy across six different analog circuits, ranging in size and complexity, while taking just under 1 second to complete.' o Indeed, the method were applied to schematics from different levels and can be easily scaled up when the training dataset for the GCN is adapted accordingly.
- REQ-6 (Standardized Representation): 'To generate the labels automatically, a traditional k-means algorithm is run on the extracted input feature matrix. K-means requires the correct number of clusters as input, however. To prevent our framework from requiring this, we use a linear model that takes as input total number of instances, number of N-type and P-type instances and number of nets in the circuit topology to predict the cluster count.' This means each instance (transistor/net) is represented by one node only. It could be the reason of the low accuracy in some test cases.
- REQ-7 (Size Scalability): The paper shows results on circuits ranging from 4 devices to 35 devices, indicating some scalability, but it doesn't show results for circuits with thousands of components.
- REQ-8 (Similar Topology Handling): The methodology uses GCNNs and k-means for clustering, which can distinguish similar topologies.
- REQ-9 (Adaptive Training): The paper does not mention the ability of adaptive training to enhance the accuracy.
- REQ-10 (Model Extendibility): The paper doesn't explicitly address the possibility of incorporating new sub-blocks or patterns over time.

- REQ-11 (Operational Environment Compatibility): although the algorithm's computing time is mentioned in the publication, compatibility with specific EDA tools or platforms is not specified.

N. LIOU ET AL

In this subsection, we justify how we filled the line of Liou et al. [23] in table 1. We simply analyze how this work performed analog circuits' structure recognition based on our requirements engineering shown in section II.

- REQ-1 (Accuracy Baseline): 97.2 ± 1.4 % was achieved by applying device sorting and feature extraction methods.
- REQ-2 (Topology Variability Robustness): The model demonstrated robustness in handling a wide range of circuit topologies and utilized domain knowledge to aid in recognition.
- REQ-3 (Transistor Type Variability Robustness):different transistors technologies were used in the test samples.
- REQ-4 (Dataset Resilience): While the model was trained on diverse datasets, there was no specific mention of handling challenges such as label imbalance or dataset size constraints.
- REQ-5 (Hierarchical Scalability): It was not mentioned in the paper if the model was designed with the capability to handle multiple levels of hierarchical circuits.
- REQ-6 (Standardized Representation): There is a coding system which considers all transistor terminals Drain, Gate, and Source. It can be simply adapted for other kinds of transistors.
- REQ-7 (Size Scalability): The model was trained on diverse datasets, indicating its adaptability to various circuit sizes. The coding system could be applied for the input schematics, not sure how it will perform with bigger circuits.
- REQ-8 (Similar Topology Handling): The device sorting mechanism gives a unique code for each transistor. However, it was not mentioned in the paper how to deal with similar subblocks. The output of the GCN is the identified subblocks, and the question here if there are two similar subblocks how can the GCN identify them differently if they were not labeled differently.
- REQ-9 (Adaptive Training): There was no explicit mention of the model supporting progressive supervised-learning.
- REQ-10 (Model Extendibility): the model does not seem to be compatible with all transistor technologies REQ-11 (Operational Environment Compatibility): Given the model's intent to aid analog circuit designers, compatibility with prevalent EDA tools is inferred. Also, the mention of the Python environment indicates adaptability.

O. KUNAL ET AL

In this subsection, we justify how we filled the line of Kunal et al. [24] in table 1. We simply analyze how this work performed analog circuits' structure recognition based on our requirements engineering shown in section II.

- REQ-1 (Accuracy Baseline): The paper states that they do not achieve more than 90% for the recognition of each sub-block without post-processing (see Table 2)
- REQ-2 (Topology Variability Robustness): The paper mentions that the model can recognize various circuit topologies and can adapt to different configurations.
- REQ-3 (Transistor Type Variability Robustness): no multiple node representation. Hence, not robust against Transistor Type Variability.
- REQ-4 (Dataset Resilience): No data augmentation mentioned. They used a large dataset in this paper.
- REQ-5 (Hierarchical Scalability): They validated nothing. Their ontology cannot handle multiple levels as they represent an entity at any level with only one node. They just showed a multi-level circuit but they have not proved to recognize multiple levels.
- REQ-6 (Standardized Representation): A transistor is represented with one node only and it is only known which terminal it is connected to by depending on the edge features(binary code for source ls, gate lg, and drain ld)
- REQ-7 (Size Scalability): The model is GCN based.
- REQ-8 (Similar Topology Handling): classes can be detected correctly, but for example, the GCN can only classify nodes to belong to a LNA class, but there is no LNA 1 and LNA 2.
- REQ-9 (Adaptive Training): No augmentation technique mentioned.
- REQ-10 (Model Extendibility): REQ 9 is not fulfilled.
- REQ-11 (Operational Environment Compatibility): The paper did not mention the use of EDA tools.

REFERENCES

[1] 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.

[2] M. Eick, M. Strasser, K. Lu, U. Schlichtmann, and H. E. Graeb, "Comprehensive generation of hierarchical placement rules for analog integrated circuits," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 30, no. 2, pp. 180–193, 2011.

[3] R. Patel, H. Habal, and K. R. Venkata, "Machine learning based structure recognition in analog schematics for constraints generation," *Mirror*, vol. 6, p. N7, Jan. 2021.

[4] H. Graeb, S. Zizala, J. Eckmueller, and K. Antreich, "The sizing rules method for analog integrated circuit design," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design. IEEE/ACM Dig. Tech. Papers*, Nov. 2001, pp. 343–349.

[5] T. Massier, H. Graeb, and U. Schlichtmann, "The sizing rules method for CMOS and bipolar analog integrated circuit synthesis," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 12, pp. 2209–2222, Dec. 2008.

[6] T. Massier, "On the structural analysis of CMOS and bipolar analog integrated circuits," Ph.D. dissertation, Dept. Elect., Electron. Commun. Eng., Technische Universität München, Berlin, Germany, 2010.

[7] C. Liang, Z. Fang, and C.-Z. Chen, "Method for analog-mixed signal design verification and model calibration," in *Proc. China Semiconductor Technol. Int. Conf.*, Mar. 2015, pp. 1–4.

[8] F. Jiao, S. Montano, C. Ferent, A. Doboli, and S. Doboli, "Analog circuit design knowledge mining: Discovering topological similarities and uncovering design reasoning strategies," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 7, pp. 1045–1058, Jul. 2015.

[9] G. Van der Plas, G. Debyser, F. Leyn, K. Lampaert, J. Vandenbussche, G. G. E. Gielen, W. Sansen, P. Veselinovic, and D. Leenarts, "AMGIE–A synthesis environment for CMOS analog integrated circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 20, no. 9, pp. 1037–1058, Sep. 2001.

[10] M. Meissner and L. Hedrich, "FEATS: Framework for explorative analog topology synthesis," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 2, pp. 213–226, Feb. 2015.

[11] M. Ohlrich, C. Ebeling, E. Ginting, and L. Sather, "Subgemini," in *Proc. 30th Int. Design Autom. Conf. (DAC)*, Dec. 1993, pp. 1–7.

[12] G. Pelz and U. Roettcher, "Pattern matching and refinement hybrid approach to circuit comparison," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 13, no. 2, pp. 264–276, Feb. 1994.

[13] K.-T. Huang and D. Overhauser, "A novel graph algorithm for circuit recognition," in *Proc. Int. Symp. Circuits Syst.*, Apr. 1995, pp. 1695–1698.

[14] N. Rubanov, "Bipartite graph labeling for the subcircuit recognition problem," in *Proc. 8th IEEE Int. Conf. Electron., Circuits Syst.*, Sep. 2001, pp. 1285–1288.

[15] N. Rubanov, "SubIslands: The probabilistic match assignment algorithm for subcircuit recognition," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 22, no. 1, pp. 26–38, Jan. 2003.

[16] N. Rubanov, "Fast and accurate identifying subcircuits using an optimization based technique," in *Proc. Signals, Circuits Syst., SCS. Int. Symp.*, vol. 1, Jul. 2003, pp. 69–72.

[17] A. R. Conn, I. M. Elfadel, W. W. Molzen, P. R. O'Brien, P. N. Strenski, C. Visweswariah, and C. B. Whan, "Gradient-based optimization of custom circuits using a static-timing formulation," in *Proc. Design Autom. Conf.*, Jun. 1999, pp. 452–459.

[18] N. Rubanov, "A high-performance subcircuit recognition method based on the nonlinear graph optimization," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 11, pp. 2353–2363, Nov. 2006.

[19] H. Li, F. Jiao, and A. Doboli, "Analog circuit topological feature extraction with unsupervised learning of new sub-structures," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2016, pp. 1509–1512.

[20] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.

[21] M. Neuner, I. Abel, and H. Graeb, "Library-free structure recognition for analog circuits," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Feb. 2021, pp. 1366–1371.

[22] 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.

[23] G.-H. Liou, S.-H. Wang, Y.-Y. Su, and M. P. Lin, "Classifying analog and digital circuits with machine learning techniques toward mixed-signal design automation," in *Proc. 15th Int. Conf. Synth., Modeling, Anal. Simulation Methods Appl. Circuit Design (SMACD)*, Jul. 2018, pp. 173–176.

[24] K. Kunal, T. Dhar, M. Madhusudan, J. Poojary, A. Sharma, W. Xu, S. M. Burns, J. Hu, R. Harjani, and S. S. Sapatnekar, "GANA: Graph convolutional network based automated netlist annotation for analog circuits," in *Proc. Design, Autom. Test Eur. Conf. & Exhib. (DATE)*, Mar. 2020, pp. 55–60.

[25] A. Deeb, A. Ibrahim, M. Salem, J. Pichler, S. Tkachov, A. Karaj, F. Al Machot, and K. Kyandoghere, "A robust automated analog circuits classification involving a graph neural network and a novel data augmentation strategy," *Sensors*, vol. 23, no. 6, p. 2989, Mar. 2023.

[26] C. Ryan, M. Tetteh, J. McEllin, D. Mota-Dias, R. Conway, and E. Naredo, "ADDC: Automatic design of digital circuit," in *Genetic Algorithms*. London, U.K.: IntechOpen, 2022.

[27] R. Martins, N. Lourenço, and N. Horta, "LAYGEN II: Automatic analog ICs layout generator based on a template approach," in *Proc. 14th Annu. Conf. Genetic Evol. Comput.*, Jul. 2012, pp. 1127–1134.

[28] IBM. *B. I. C. Education, What is Data Labeling*. Accessed: Feb. 16, 2024. [Online]. Available: https://www.ibm.com/topics/data-labeling

[29] I. Lee. *Data Labeling and Its Significance*. Accessed: Feb. 16, 2024. [Online]. Available: https://www.ibm.com/topics/data-labeling

[30] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, 2020.

[31] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *Stat*, vol. 1050, no. 20, p. 48550, 2017.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017, *arXiv:1706.03762*.

[33] Z. Sun, A. Harit, J. Yu, A. I. Cristea, and N. Al Moubayed, "A generative Bayesian graph attention network for semi-supervised classification on scarce data," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2021, pp. 1–7.

[34] K. Zhu and M. Cao, "A semantic subgraphs based link prediction method for heterogeneous social networks with graph attention networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–8.

[35] I. B. Ababio, J. Chen, Y. Chen, and L. Xiao, "Link prediction based on heuristics and graph attention," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2020, pp. 5428–5434.

[36] L. Zangari, R. Interdonato, A. Calió, and A. Tagarelli, "Graph convolutional and attention models for entity classification in multilayer networks," *Appl. Netw. Sci.*, vol. 6, no. 1, pp. 1–36, Dec. 2021.

[37] S. Karagiannakos, "Best graph neural network architectures: GCN, GAT, MPNN and more," *AI Summer*, vol. 23, Sep. 2021.

[38] V. Mallawaarachchi. *Inductive Vs. Transductive Learning*. Accessed: Feb. 19, 2024. [Online]. Available: https://towardsdatascience.com/inductive-vs-transductive-learning-e608e786f7d

[39] V. Vapnik, "24 transductive inference and semi-supervised learning," in *Semi-Supervised Learning*. MIT Press, 2006, pp. 453–472.

[40] K. Ding, Z. Xu, H. Tong, and H. Liu, "Data augmentation for deep graph learning: A survey," *ACM SIGKDD Explor. Newslett.*, vol. 24, no. 2, pp. 61–77, Nov. 2022.

[41] Y. Rong, W. Huang, T. Xu, and J. Huang, "DropEdge: Towards deep graph convolutional networks on node classification," 2019, *arXiv:1907.10903*.

[42] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Graph contrastive learning with adaptive augmentation," in *Proc. Web Conf.*, Apr. 2021, pp. 2069–2080.

[43] K. Kong, G. Li, M. Ding, Z. Wu, C. Zhu, B. Ghanem, G. Taylor, and T. Goldstein, "Robust optimization as data augmentation for large-scale graphs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 60–69.

[44] Y. Wang, W. Wang, Y. Liang, Y. Cai, and B. Hooi, "Mixup for node and graph classification," in *Proc. Web Conf.*, Apr. 2021, pp. 3663–3674.

[45] W. Feng, J. Zhang, Y. Dong, Y. Han, H. Luan, Q. Xu, Q. Yang, E. Kharlamov, and J. Tang, "Graph random neural networks for semi-supervised learning on graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 22092–22103.

[46] J. Gasteiger, S. Weißenberger, and S. Günnemann, "Diffusion improves graph learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 11015–11023.

[47] T. Zhao, Y. Liu, L. Neves, O. Woodford, M. Jiang, and N. Shah, "Data augmentation for graph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 12, 2021, pp. 11015–11023.

[48] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, "Graph contrastive learning with augmentations," in *Proc. Neural Inf. Process. Syst.*, 2020, pp. 5812–5823.

[49] Y. You, T. Chen, Y. Shen, and Z. Wang, "Graph contrastive learning automated," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 12121–12132.

[50] H. Guo and Y. Mao, "ifMixup: Towards intrusion-free graph mixup for graph classification," 2021, *arXiv:2110.09344*.

ALI DEEB was born in Hama, Syria, in 1991. He received the B.Sc. degree in communications systems from the Higher Institute for Applied Sciences and Technology (HIAST), Damascus, Syria, in 2014, and the M.Sc. degree in communication engineering from the Carinthia University of Applied Sciences (CUAS), Klagenfurt, Austria, in 2020. He is currently pursuing the Ph.D. degree.

From 2015 to 2017, he was a Radio Access Network Operation Subsystem Engineer (BSS) with Syriatel Mobile Telecom. In 2018, he was a RFID Engineer with FACHHOCHSCHULE KÄRNTEN. From 2018 to 2020, he was a RFS Concept and Application Engineer with Infineon Technologies. Since 2021 he has been a Researcher with the University of Klagenfurt, Klagenfurt, Austria, in collaboration with Infineon Technologies toward the Ph.D. degree. His research interest includes the automation of pre-silicon mixed-signal verification using artificial intelligence techniques, mainly analog circuit structure recognition.

MOHAMED SALEM was born in Dakahlia, Egypt, in 1997. He received the B.Sc. degree in mechatronics engineering from Nile University, Egypt, in 2020, and the M.Sc. degree in autonomous systems and robotics from the University of Klagenfurt, Austria, in 2022. He is currently a Senior Specialist in Chip-Package-Board Codesign Methodology with Infineon Technologies Austria.

In October 2021, he joined the Analog/Mixed Signals Verification Department, Infineon Technologies Austria, as an Intern Student. Before that, he was a Research Assistant with the Transportation Informatics Laboratory, University of Klagenfurt. In 2022, he researched the master's thesis in AI/ML-based structure recognition for AMS circuits with Infineon Technologies Austria.

ABDALRAHMAN IBRAHIM received the B.S. degree in mechatronics from the Misr University for Science and Technology, in 2018, and the M.Sc. degree in autonomous systems and robotics from the University of Klagenfurt (AAU), where he is currently pursuing the Ph.D. degree.

He was a Teaching and Research Assistant with the IRES Laboratory, Misr University for Science and Technology. While working at Infineon Technologies Austria, he received the M.Sc. degree. He is currently an AI and Robotics Research Engineer with AGILOX Systems. His research interests include graph theory, machine learning, deep learning, and pattern recognition for robotics applications.

JOACHIM PICHLER was born in Villach, Austria, in 1975. He received the M.S. degree in electronic engineering from the Carinthian Institute of Applied Science, Villach, in 2001.

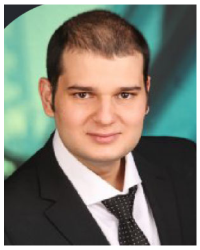
In 1996, he joined Siemens Semiconductors, Villach, (now Infineon Technologies Austria). In the course of his career he gained experience in the fields of layout, laboratory validation, analog design, technology development, concept engineering, AMS design, modeling, AMS verification, and lately leads a team in charge of developing disruptive methods addressing challenges in many of those fields. He holds 11 patents related to semiconductor products used in communication, automotive, and power management systems.

SERGII TKACHOV was born in Khabarovsk, Russia, in 1990. He received the B.S. degree in avionics and the M.S. degree in aircraft control systems from the National Technical University of Ukraine, Kyiv, Ukraine, in 2012 and 2018, respectively.

Since 2014, he has been with Infineon Technologies Austria, Villach, Austria. He started his career as a Laboratory Validation Intern for dc–dc converters. From 2015 to 2022, he was a Pre-Silicon Mixed-Signal Verification Engineer with a focus on verification flow automation and machine learning enablement. Since 2022, he has been with the Disruptive Mixed-Signal Development Group. His research interests include IC design and verification, design flow automation, waveform data analysis and outliers detection, simulation techniques, and engines.

KARAJ ANJEZA was born in Kruje, Albania, in 1992. She received the Dipl.-Ing. degree in communication engineering and the M.S. degree in circuit design from the Carinthia University of Applied Sciences, Klagenfurt, in 2017 and 2020, respectively.

Since 2018, she has been with Infineon Technologies Austria, Villach, Austria. From 2018 to 2022, she was a Pre-Silicon Mixed-Signal Verification Engineer with a focus on verification flow automation and machine learning enablement. Since 2022, she has also been with the Disruptive Mixed-Signal Development Group. Her research interests include IC design and verification, design flow automation, and simulation technics.



FADI AL MACHOT (Member, IEEE) received the M.S. degree in computer science from the University of Potsdam, Potsdam, Germany, in 2010, the Ph.D. degree in computer science from the University of Klagenfurt, Klagenfurt, Austria, in 2013, and the Habilitation degree in applied computer science from the University of Lübeck, Lübeck, Germany, in 2020. From 2013 to 2016, he was a Postdoctoral Researcher with the University of Klagenfurt, Klagenfurt, Austria. From 2016 to 2020, he was a Senior Data Scientist with the Leibniz-Lung Center, Borstel (FZB), Germany. He is currently an Associate Professor in machine learning with the Norwegian University of Life Sciences (NMBU), Norway. He patented and published more than 60 papers in peer-reviewed international conferences and journals. He actively reviews well-known IEEE, ACM, and Springer journals. His research interests include deep learning, neural-symbolic learning, video understanding, cognitive modeling, and zero/few-shot learning.

KYANDOGHERE KYAMAKYA is currently a Full Professor of transportation informatics and the Deputy Director of the Institute of Smart Systems Technologies, University of Klagenfurt, Austria. He is actively conducting research in modeling, simulation, and test-bed evaluations for a series of concepts amongst others in the context of intelligent transportation systems. In the research addressing transportation systems, a series of fundamental and theoretical tools from the fields of applied mathematics, electronics, and computer science is either extensively exploited or a source of inspiration for innovative solutions and concepts, including nonlinear dynamics, systems science, machine learning/deep learning, nonlinear image processing, and neuro computing. He has co-edited more than six books. He has published more than 100 journal articles and more than 100 conference papers.