Review

A Review of Machine Learning Techniques in Analog Integrated Circuit Design Automation

Rayan Mina [1,*], Chadi Jabbour [2] and George E. Sakr [3]

[1] Department of Electrical and Mechanical Engineering, Saint-Joseph University of Beirut, Beirut 1004 2020, Lebanon
[2] COMELEC, Institut Mines-Télécom Paris, 91120 Palaiseau, France; chadi.jabbour@telecom-paristech.fr
[3] Virgil Systems, Toronto, ON M5G 1E2, Canada; gsakr@virgilsystems.com
* Correspondence: rayan.mina@usj.edu.lb

Abstract: Analog integrated circuit design is widely considered a time-consuming task due to the acute dependence of analog performance on the transistors' and passives' dimensions. An important research effort has been conducted in the past decade to reduce the front-end design cycles of analog circuits by means of various automation approaches. On the other hand, the significant progress in high-performance computing hardware has made machine learning an attractive and accessible solution for everyone. The objectives of this paper were: (1) to provide a comprehensive overview of the existing state-of-the-art machine learning techniques used in analog circuit sizing and analyze their effectiveness in achieving the desired goals; (2) to point out the remaining open challenges, as well as the most relevant research directions to be explored. Finally, the different analog circuits on which machine learning techniques were applied are also presented and their results discussed from a circuit designer perspective.

Keywords: automated circuit sizing; machine learning; analog IC design; deep neural networks

check for updates

Citation: Mina, R.; Jabbour, C.; Sakr, G.E. A Review of Machine Learning Techniques in Analog Integrated Circuit Design Automation. *Electronics* **2022**, *11*, 435. https://doi.org/10.3390/electronics11030435

Academic Editor: Kiat Seng Yeo

Received: 14 January 2022
Accepted: 28 January 2022
Published: 31 January 2022

1. Introduction

Since its beginning, integrated circuit (IC) technology has evolved significantly in several aspects: First, the range of circuits has been expanding continuously to target digital and analog, wired and wireless/RF, and low-power and medium-power applications. Second, the complexity of the realized functions have never stopped growing, which adds difficulties and more technical challenges for the circuit designer. Finally, the fabrication of ICs has diversified significantly with more than a dozen different processes currently available in bipolar, MOS, and BiCMOS technologies [1].

On the other hand, digital integrated circuits have increased in integration density and speed, reaching several billions of transistors in one mm^2 of area and achieving Tera floating point operations per second (TFLOPS). Although they realize less complex functions than digital circuits, analog circuits are considered more challenging since their performance is more dependent on the dimensions of the transistors and passive components (i.e., width and length), as well as on the intended fabrication process and bias conditions. Consequently, long design cycles are the main bottleneck in analog IC design. Digital circuits benefit from many flexible and substantially automated electronic design software tools, in addition to pre-sized ready-to-use logic libraries that speed up their design phase [2], whereas analog ICs still lack such a high degree of automation and rely significantly on the experience and skills of the engineer designing the circuit. One interesting way to alleviate this challenge is to design analog circuits by relying only on digital logic gates [3,4]. This allows the use of digital flow for analog circuit design, which drastically reduces the design time and effort. However, the performance obtained by these approaches, especially in terms of power consumption, are not yet as competitive as in classical analog design approaches.

As a matter of fact, once the system-level architecture and specifications are determined, analog design generally consists of choosing first the topologies for the basic building blocks such as amplifiers, oscillators, or comparators and then sizing the devices (transistors and passive components) of these building blocks to achieve a set of performance specifications.

This sizing step is challenging due to three major reasons: First, there are always multiple solutions to the same problem, which are usually spread in the space of possibilities. Second, the circuit performance is sensitive to uncontrollable random variations of the fabrication process [1,5] and temperature. Third, analog circuits are usually difficult to specify with a few performance metrics, which leads in some cases to more than a dozen of specifications to meet [6–8] before a circuit designer is fully satisfied. Finally, some specifications are dynamic in the sense that they depend on the circuit conditions [9], and therefore, some performance metrics could be degraded during the IC life-cycle, which renders the design even more complex.

In the past ten years, a considerable amount of research work has been conducted to enhance the automation level of analog IC design using machine learning (ML). The corresponding prior state-of-the-art can be split into three main categories.

1. Neural networks (NNs) of different complexities (shallow and deep) using both supervised [10–18] and reinforcement [19–21] learning techniques have been proposed in the past six years. These methods have been boosted by the progress in the development of high-performance computing hardware and are gaining attention in the scientific community thanks to their ability to model complex nonlinear problems [22];
2. Multi-objective optimization with several heuristic- and stochastic-assisted strategies to explore the search space efficiently: Bayesian [23,24], particle swarm [25], the Gaussian process [26], simulated annealing [27], and genetic algorithms [28–30]. These are simulation-based optimizations that rely on a circuit simulator to find a solution. Although global optimization is not an ML technique in the strict sense of the word, it remains a powerful mathematical tool for the automation of circuit design. In contrast, older equation-based techniques as in [31] are no longer considered;
3. Hybrid methods combining global optimization and NN were proposed in [32–37]. These techniques achieve enhanced results compared to pure multi-objective optimization methods, at a lower computation time.

In this work, we aimed to survey and investigate the recent progress made in the past six years in applying ML methods to analog IC front-end design automation exclusively. Techniques that use surrogate modeling [38] instead of SPICE-like simulators were not included. Moreover, contributions targeting device modeling, physical back-end IC design [39], and fault testing were not included in this paper. Instead, the focus was on the existing methods and innovations that have been proposed in the fields of supervised and reinforcement deep learning for front-end analog IC design. The working principles of the employed techniques, as well as some of their pros and cons are also discussed. The contributions of this work were: (1) to provide answers on which circuits and to which extent ML techniques were effectively used to help analog designers achieve good results; (2) to elaborate on the existing limitations that require further investigations in the field, as well as relevant future research directions.

There are some existing surveys in the literature that have tackled IC design automation [40–42]. Nevertheless, they differ from the proposed survey in this paper. The authors in [42] pointed out the recent advancements in the usage of deep convolutional neural networks and graph-based neural networks to accelerate the digital IC design workflow. Their survey covered ML-assisted micro-architectural space exploration (from high-level RTL design), physical back-end design (layout) for power optimization and estimation, critical IR drops in routes, and efficient placement. Thus, they focused on the digital VLSI flow and did not target analog circuits.

In [40], a general high-level review of ML applications was provided in multiple areas of IC design from device modeling, front-end synthesis, and layout (back-end) design to fault testing. The work is well organized and provides a wide overview of the prior state-of-the-art without focusing particularly on one area.

The work in [41] is another example of a high-level paper that surveyed a wide spectrum of artificial intelligence (AI) methods including knowledge-based and equation-based optimization techniques. In addition to sizing, the work covered performance modeling, which predicts the circuit performance dependency on device-level variations (e.g., ΔV_{TH}, Δg_m), as well as IC yield optimization under process variation (i.e., corners). Finally, the paper spanned two decades of analysis in its presentation and covered digital along with analog design.

Therefore, the current work is, to the best of our knowledge, the first survey in the literature to focus particularly on recent (post-2015) ML-assisted analog front-end IC design automation, propose a profound level of understanding of the problem, and discuss the details of the key contributions from prior state-of-the-art and the effectiveness of the employed methods.

This paper is organized as follows: The existing NN-based automation approaches for analog IC design are detailed in Section 2. Hybrid techniques are explained in Section 3. Section 4 provides a comparative analysis of the prior state-of-the-art, while Section 5 covers the remaining open challenges and the most promising research directions to be explored in the field.

2. NN-Based IC Design Automation Methods

In this section, we elaborate on the various approaches that exist in the literature from the past six years to automate analog IC front-end design. Those techniques rely exclusively on ML, namely supervised and reinforcement learning using NNs. For completeness, we also expose the papers that present hybrid techniques. The explanatory details, effectiveness, range of applicability, and advantages of each method are given.

The use of NNs to solve complex practical problems has proven to be very efficient in numerous fields [22]: speech recognition, medical imaging diagnosis, robotic vehicles, natural language processing, and image recognition. For such problems, mathematical modeling by equations is not possible because the designer cannot anticipate all situations, nor their changes over time, and in some cases, the designer even lacks knowledge on how to build the model itself.

The general idea of NNs is to build a model that approximates the real solution for a problem based on limited, yet sufficient, available dataset (training phase). At the same time, the model should be able to generalize and predict new outputs for previously unseen data (testing phase) with sufficient accuracy. NNs are very efficient at solving nonlinear problems due to their structure, which mimics the human brain's activity, and their flexibility in adjusting the model's complexity by removing or adding layers and neurons [43].

The idea of using NNs to assist in analog circuit design was validated in 2003 for the case of operational amplifiers (opamp), as shown in [44]. In the coming sub-sections, we elaborate on the use of NNs for analog IC front-end design and sizing in the two cases of supervised and reinforcement learning methods.

2.1. Supervised Learning

The core idea here is to gather a collection of input/output examples for the studied problem (the dataset), before trying to build an approximate solution. The efficiency of supervised learning is usually acquired through a large and labeled training dataset, which might be a limiting factor in problems where large labeled datasets are not available.

Previous attempts have been made to apply the NN supervised learning approach to design basic circuits. For example, in [11], a small five-transistor inverter-based current comparator was designed using 0.18 µm CMOS technology, using a small network with 20 neurons that was trained with just 1500 examples to predict a few device sizes. In [10], the authors developed an NN that can re-design an existing 4 bit current-steering DAC for the same target specifications, but in a novel technology node (0.35 µm CMOS). Although scalable, this solution cannot be generalized to predict sizes for new, different specifications. The transient power consumption of a relaxation oscillator was modeled in [18] using a 60-neuron time-delay NN, to enhance the circuit's functional model during system-level verification. This approach was focused on power consumption and is well suited for mixed-signal blocks, which are usually modeled in behavioral languages.

Analog ICs implementing more complex functions (e.g., amplification, filtering) require more elaborate studies, larger training datasets, and networks with higher complexities in terms of layers and neurons.

In order to discuss the existing work in the literature, which applies supervised learning to analog IC design automation, a set of criteria for evaluating the prior state-of-the-art should be defined. In this work, the following criteria are proposed for this purpose:

- Dataset generation technique;
- Feature selection;
- NN complexity;
- IC fabrication process used;
- Types of circuits targeted;
- Result validation method.

2.1.1. Dataset Generation Technique

The dimensions of all transistors and passives (W, L), as well as reference currents and voltages (I_{ref}, V_{ref}) in an analog block are referred to, in this paper, as the design parameters X_i, while the circuit's performance metrics (e.g., gain, power consumption, etc.) are called the specifications Y_j. To build a model capable of predicting X_i, the initial step in supervised learning is to generate a dataset of circuit simulation results that covers the design space sufficiently. Analog designers must define the performance specifications, then simulate the circuit using a computer-aided design (CAD) tool for a significant number of possibilities of the design parameters, as shown in Figure 1. The generation of such a dataset is a time-consuming task that involves people with a decent level of experience and knowledge in IC CAD tools. References [13,16] used SPICE circuit simulators to generate their datasets.

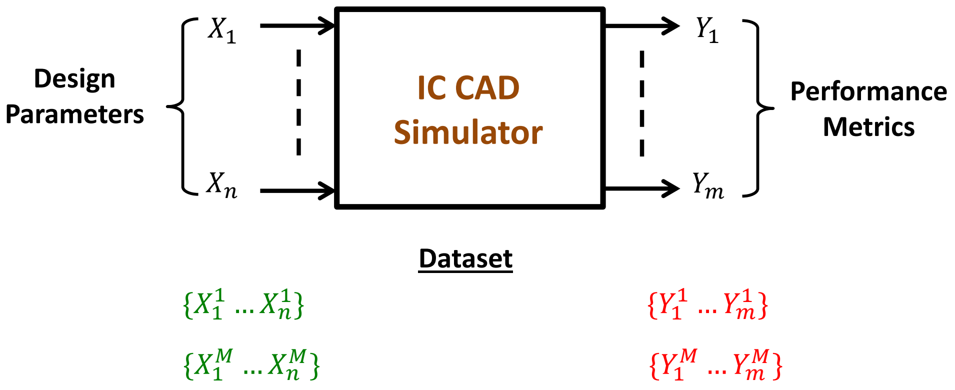


Figure 1. Generation of the training dataset using integrated circuit CAD simulators.

To illustrate how a dataset looks, we considered a basic source-coupled differential pair, with an active load, tail current mirror (equal-length devices), and a known-value resistor for biasing (see Figure 2). Due to symmetry, only four transistors need to be sized (width, length), and thus, there are seven design parameters $X_i = \{W_1, L_1, W_2, L_2, W_3, L_3, W_4\}$.

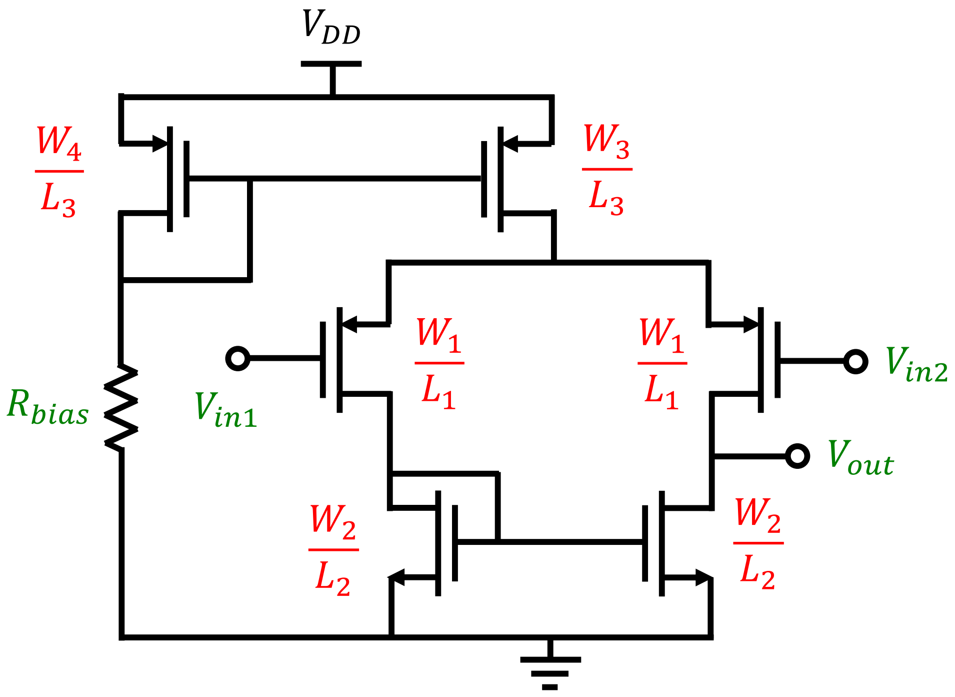


Figure 2. Simple CMOS differential pair amplifier with resistive biasing.

Assume for simplicity that the amplifier to be designed is specified only for the following performance: differential voltage gain and power consumption. Consequently, $Y_j = \{A_d, I_c\}$, and the combined vector $\{X_i, Y_j\}$ represents a single entry in the dataset, which is an $N \times 9$ matrix where N is the number of examples. Table 1 shows two examples from the dataset of the previous circuit (dimensions in µm, I_c in mA).

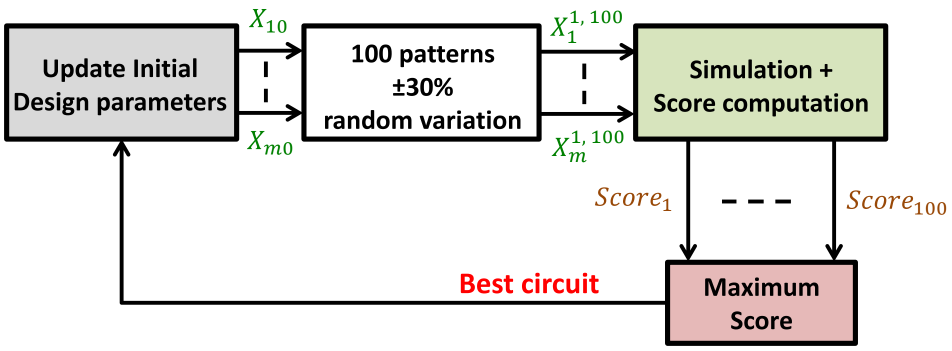
Table 1. An example of a dataset with X_i and Y_j.

	W_1	L_1	W_2	L_2	W_3	L_3	W_4	A_d	I_c
1	5.2	0.13	10	0.2	2.4	0.3	0.24	56	2.3
2	4.1	0.18	11	0.6	2.1	0.35	0.4	112	4
…	…	…	…	…	…	…	…	…	…

On the other hand, there are no strict rules that define how many data are required for each type of circuit. Usually, more data are helpful (since this increases the model accuracy); nevertheless, in the case of analog IC design automation, the designer will generate only a reasonable amount of data to keep the task practical and time manageable. For example, 20,000 examples (dataset size) were needed in [13] to train the NN and predict results for a similar circuit that was designed in [14,15] using only 9000 and 16,600 different examples, respectively. On the other hand, only 1600 data points were sufficient in [12]. However, the previous numbers cannot be directly compared since the way the datasets were generated out of circuit simulators differed significantly from one study to the other. It is hence insightful to look into the different methods used to generate the training examples.

In [14], the authors manually adjusted, prior to simulation, all MOS transistors sizes in order to obtain an initial valid solution that achieved proper performance (from a circuit designer's perspective). This initial adjustment was made by circuit designers. Only then, the initial design parameters were slightly varied by 5%, and the different corresponding circuits were simulated one by one to extract their performance and build the dataset. Moreover, all transistors' lengths L were fixed to 1 µm to reduce the design space and avoid short-channel effects. Therefore, the dataset in [14] was too dependent on a pre-sized state of the circuit.

Reference [13] proposed a similar, yet different approach for dataset generation: Initial values for the design parameters (X_{i0}) were first determined by a circuit designer. Next, 100 different patterns were produced by randomly varying X_{i0} in the range $\pm 30\%$. The obtained circuits were then simulated, and a score was calculated to select the best circuit that had the highest score among all. The score was defined as the ratio of higher-the-better performance metrics in the numerator and lower-the-better in the denominator. This process was iterated as many times as needed to build the dataset, with the best circuit from the previous 100 patterns being the starting circuit for the generation of the next 100 patterns. The whole process is depicted in Figure 3.



X_{10}
X_{m0}
Update Initial Design parameters
100 patterns ±30% random variation
$X_1^{1,100}$
$X_m^{1,100}$
Simulation + Score computation
$Score_1$
$Score_{100}$
Maximum Score
Best circuit

Figure 3. Description of the training-dataset-generation approach used in [13].

This approach had better exploration capabilities than [14] thanks to its higher range of variation (30%); nevertheless, the selection of a solution at each step was dictated by a single score that compressed the valuable information contained in a set of performances to a unique metric. Moreover, the dataset used in training the NN depended somewhat on a pre-defined state of the circuit.

The authors in [15] presented two novel ideas in data generation: First, they used an existing dataset from previous optimizations made by analog designers, which contained optimal or quasi-optimal circuits. The motivation here was to make the training data as accurate as possible by having in the examples only the design parameter values that guaranteed good performance. Second, the dataset was augmented by exploiting the fact that some specifications correspond to inequalities. When a circuit is required to have a power consumption below a specified threshold, all solutions resulting in a lower power consumption values were considered compliant and hence added to the set of examples. Therefore, the authors in [15] artificially increased the dataset size by randomly generating additional examples with the same design parameters, but lower or higher (inequality) performance values.

In [16], the dataset was generated from a special software framework (MaxFit GA-SPICE), based on a genetic multi-objective optimization algorithm that relied on a set of predefined circuit equations, which were developed by analog circuit designers. Therefore, the circuit performances for all examples had a high level of confidence and were highly accurate, since they were produced by a previous study. Nevertheless, this approach entirely relied on the accuracy and availability of the aforementioned software framework. Moreover, the generated dataset was further filtered to remove any outlier values, which were values exceeding three-times the standard deviation from the mean value (as defined in [16]).

The work in [17] proposed an interesting approach to widen the prediction range for the target performance, by generating a fully random dataset from a normal distribution on all design parameters. Some restrictions were rightly added to ensure quality examples and improve the model's accuracy.

Table 2 summarizes the details of the different dataset-generation techniques used in the aforementioned studies.

Table 2. Different dataset-generation methods used in prior state-of-the-art.

	Starting Point	Generation Technique	Dataset Size
[12]	1 random point	linear sweeps	1600
[13]	1 expert design	random variation ±30%	20,000
[14]	1 expert design	random variation ±5%	9000
[15]	existing database	–	16,600
[16]	–	MaxFit software	10,000
[17]	1 expert design	normal random distribution	40,000

2.1.2. Feature Selection

After dataset generation, the aim is to make the NN compute and converge towards a model that predicts the design parameters X_i from the existing performance specifications Y_j. The neural network is asked, in this sense, to solve an inverse problem using a dataset generated from a direct problem definition.

The training phase is a standard ML process that requires a proper definition of the features or inputs from which the NN will learn. Normally, these can be the circuit performance metrics directly taken from the CAD simulators such as differential and common-mode gains in amplifiers, unity-gain–bandwidth, and power consumption, as in [13,14,16]. A typical block diagram showing the input and output variables for the NN core used for analog IC design is represented in Figure 4.

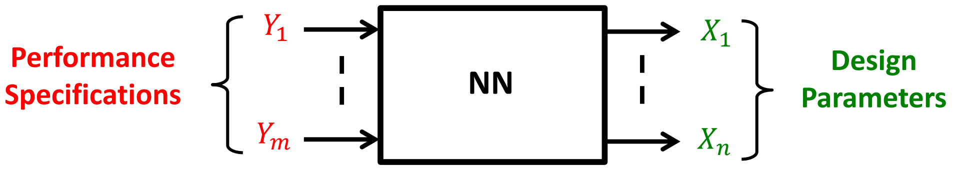


Figure 4. Inputs/outputs of a neural network using the performance metrics directly as features [14,16].

The NN features may also be expanded by nth-order polynomial combinations as in [15], where fourteen different combinations of second-order terms were generated from four simulated performance metrics only. In [18] also, the same feature was sampled d times before feeding it to the NN. A two-step approach wherein three behavioral features were considered separately for training, then recomposed into a single output in a later stage was adopted in [12].

It is crucial that feature selection be left to experienced analog circuit designers who know well what, how, and to what extent each performance is linked to the design parameters. This step dictates how accurately the model will converge. To illustrate this, assume that a designer is building a model for an opamp using its open-loop voltage gain A_v without including the performance features that are related to the high-frequency gain behavior (phase margin and gain–bandwidth product). Thus, the predicted design may experience instability once the opamp circuit is used in a closed-loop configuration. Another aspect that is worth discussion is the optimal number of features that need to be included in a dataset. From an ML perspective, more relevant features are usually welcomed since this pushes the neural network to build a more informed model. From a circuit designer perspective, performance metrics have different levels of importance. For example, a passive mixer's total harmonic distortion (THD) might be more important to learn than its noise performance. Nevertheless, adding relevant and uncorrelated circuit features will certainly help the NN build a more accurate model, regardless of their importance.

2.1.3. NN Complexity

The NN structure and complexity must be carefully defined with the optimal number of layers and neurons per layer, to start the training process. There is no defined rule for this choice. It may vary substantially depending on the realized circuit function, the transistor count, the number of features (Y_i) and their polynomial combinations (if any), as well as the size of the generated dataset. In IC design automation, the first layer has a number of inputs equal to the number of features, while the number of neurons in the output layer must match the number of design parameters to be predicted. In the middle, hidden layers are inserted to model complex nonlinear combinations of the circuit features, and this is where each NN should be customized for the intended circuit. Some good guidelines in this sense are the following: (1) for complex circuits (i.e., transistor count), a high number of neurons is required in each Layer; (2) when the combination of several performance metrics in a circuit contains useful information that needs to be included in the model, a high number of layers should be used to allow the NN to learn those effects. Nevertheless, adding more neurons and/or layers always comes with a longer training time. Table 3 summarizes the prior state-of-the-art NN complexities implemented in supervised learning.

Table 3. Complexities in terms of number of neurons for the different NN structures used in prior state-of-the-art.

References	Inputs	Hidden Layers	Output Layer
[12]	3	24 + 200	1
[13]	12	Missing	7
[14]	5	15 × 1024	6
[15]	4 (expanded: 14)	120, 240, 60	8
[16]	6	1 × 9	12
[17]	5	1 × 50	9

For example, the work in [15] used three hidden layers with 120, 240, and 60 neurons, respectively, while in [16], the authors opted for a single hidden layer with just nine neurons because the training examples were pre-sized ones. On the other hand, and since the dataset was randomly generated in [14], which led to widely spread values in the design space, a much more complex network composed of 15 layers with 1024 neurons each was designed to obtain accurate results (Table 3). This major difference stems from the simple observation that a low-complexity NN training on highly accurate datasets can learn as good as complex networks do on random data.

2.1.4. IC Fabrication Process

The training dataset depends heavily on the models of the transistors and passives that were used in the CAD simulations, which differ significantly from one fabrication process to another. Consequently, the same circuit topology with the same target specifications will result in different sets of design parameters depending on the intended IC fabrication technology. In [13,14], circuits were designed and verified in sub-micron CMOS processes (<1 µm), although the exact transistor's minimum length was not stated in the papers. On the other hand, the authors in [15,16] worked on 0.13 µm and 0.18 µm CMOS standard technologies, respectively. All previous work included both NMOS and PMOS transistor types in the circuits; however, there is no available study that investigates bipolar transistors and non-standard IC technologies such as SOI or BiCMOS.

2.1.5. Target Circuits

Analog circuits are extremely diversified currently, and they are designed to realize a variety of functions ranging from signal amplification, filtering, shaping, and clipping, to frequency mixing and many others. Therefore, the choice is very wide for the target circuit on which to apply the supervised ML approach. An intuitive decision is to opt for a structure that is easily reused in numerous systems, such as opamps, since their theory of operation is well known and they have the widest application range in analog systems [1]. In fact, opamps are usually specified and designed as single open-loop circuits to be reused extensively in closed-loop systems. The automatic sizing of all the opamp components provides a huge benefit to circuit designers since they can focus more on the optimization of their macro-system performance and architecture decisions.

In [14,16], the same two-stage opamp circuit was used in the supervised ML approach. It corresponds to an NMOS source-coupled pair followed by a PMOS common-source. The biasing was realized by a simple resistor, thus exhibiting large process, supply voltage, and temperature variations.

In [13], the authors used a very similar circuit, only replacing each transistor with its complementary type (PMOS vs. NMOS). Both circuits are valid opamps, although analog designers have a slight preference for the PMOS input devices, in order to minimize noise and optimize unity-gain frequency [45]. All three previously studied circuits have differential inputs, but single-ended output structures.

A slightly different circuit was investigated in [15], having differential outputs and using voltage combiners for biasing.

An interesting case is the CMOS band-gap voltage reference studied in [12]. It includes an amplifier, with additional MOS transistors, diodes, and resistors. It represents a different type of analog circuit compared to [13,16] with the particularity of providing a supply- and temperature-independent DC voltage.

It is worth noting that previous work focused mainly on opamps and all had moderate circuit complexity (transistor count < 12).

2.1.6. Result Validation Method

Usually, after the NN has been trained on all examples, the standard way to validate the results is to generate a small set of new input/output examples that are not part of the original dataset. The new outputs, called the test-set performance metrics, are fed to the inputs of the NN, which in turn computes the corresponding design parameters according to the model built during the training phase. Next, those predicted NN outputs are fed back to the circuit simulator to obtain the predicted performance metrics. At this stage, a comparison is possible between the initial test-set performance and the predicted one (for each metric), then the mean relative error on all test examples is calculated to quantify the accuracy of the results. Table 4 summarizes the accuracy of the results obtained in [13,16]. A comparison among the common specifications that were computed is theoretically sound, since all previous papers studied very similar circuits (two-stage CMOS opamp). The prediction error for each specification is computed using Equation (1).

$$\overline{err_j(\%)} = 100 \times Mean\left[\frac{|Y_{j,pred} - Y_{j,true}|}{Y_{j,true}}\right]. \tag{1}$$

$Y_{j,pred}$ is the jth performance metric estimated using a circuit simulator from the schematic that contains the NN-predicted design parameters $X_{j,pred}$. On the other hand, $Y_{j,true}$ is the jth performance metric that was initially input to the NN to make the prediction. For example, the gain and slew rate metrics in [13] had a mean prediction error of 1.1% and 19.1%, respectively, according to Table 4. The overall performance of a circuit is the combined effect of each of its predefined metrics, and therefore, it is interesting to compute both the average μ_e and standard deviation σ_e of the mean error for each performance. All prior state-of-the-art works achieved a low average error $<$10%; however, from a circuit designer's perspective, it is mandatory that each metric reaches a low mean error, and hence, a low σ_e is highly desirable. The work in [13] had the highest μ_e and σ_e computed on 12 target specifications because the training dataset was randomly generated with 30% variation and the selection of examples was based on a single score. On the other hand, the work in [16] had the lowest values thanks to high-accuracy pre-sized circuit examples provided to the NN by a special software framework.

Table 4. Prediction error statistics in % per specification in prior state-of-the-art calculated using Equation (1).

	[13]	**[14]**	**[15]**	**[16]**
Power consumption	14.5	–	9.8	3.7
Gain	1.1	7.2	18.2	3.7
Gain-BW product	18.1	0.1	1.3	5.9
Phase margin	5.8	5.8	1.6	3.5
Common-mode rej. ratio	3.7	1.6	–	–
Power-supply rej. ratio	2.8	7.5	–	–
Slew rate	19.1	–	–	2.1
Average—Std. Dev.	9.3–7.7	4.4–3.4	7.7–8	3.8–2.5

2.2. Reinforcement Learning

Reinforcement learning (RL) is a sub-field of machine learning where the agent learns to perform a specific task by reward and penalty. Unlike supervised learning, RL does not require a labeled dataset to train. The training is performed by making predictions and obtaining rewards for those predictions. Rewards should be higher when the prediction is closer to the real value and lower otherwise. The goal of the training phase is to learn a policy that maximizes the sum of all rewards.

In IC design, RL is not extensively used yet. This section presents the previous work of four different teams who applied RL to solve analog IC design sizing problems. Before going into the details, it is worth pointing out the basics of RL in IC design. A vector of performance features represents the state of a circuit. The RL agent takes this vector as the input, executes an action by changing the design parameter values, computes the reward, then moves to the new state if the reward has increased and explores different available actions otherwise (Figure 5). The reward is computed from the circuit simulator results.

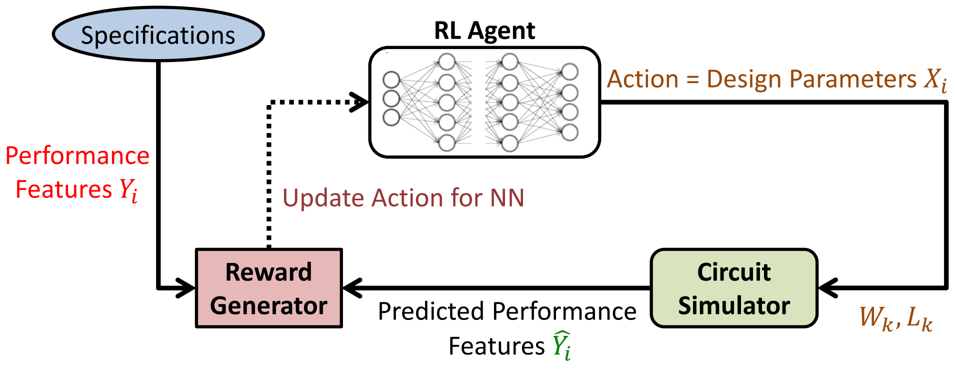

Specifications
RL Agent
Action = Design Parameters X_i
Performance Features Y_i
Update Action for NN
Reward Generator
Circuit Simulator
Predicted Performance Features $\hat{Y}_i$
W_k, L_k

Figure 5. Reinforcement learning design flow as applied to analog IC design automation.

Reference [19] trained an RL agent for 30–40 h and achieved comparable or better performance than human experts on several circuits. The algorithm would output at each time step a set of values corresponding to the dimensions of the transistors and passive components (i.e., design parameters). The circuit simulation environment would then take those values and compute the performance metrics of the circuit. A reward function was defined based on hard design constraints, as well as good-to-have targets. The reward was formulated to be high if the simulated performance was close to the real values and low otherwise. Their approach was validated on two circuits: two- and three-stage transimpedance amplifiers in CMOS 0.18 µm. Five performance features were targeted:

noise, gain, power consumption, AC response peaking, and bandwidth. In both circuits, RL showed promising results, where it outperformed the human experts in the time taken to find the optimal values.

On the other hand, a deep reinforcement learning framework was used in [20] to find the circuit design parameters, as well as to gain knowledge of the design space. The algorithm found relevant solutions 40-times faster than regular genetic algorithm optimization. The approach was tested on a simple transimpedance amplifier and a two-stage opamp in CMOS 45 nm, then on another opamp with negative-gm load in FinFET 16 nm. On the first circuit, the algorithm converged 25-times faster than regular algorithms and was able to generalize well with 97% correct predictions out of 500 different target specifications. For the second and third circuits, it showed a speedup of 40-times over regular algorithms and predicted correctly 96% of 1000 and 100% of 500 different specifications.

The work in [21] used deep RL to predict circuit design parameters as in the previous papers. However, the authors introduced a novel technique based on a symbolic analysis that rapidly evaluated the values of the DC gain and phase margin (PM) without invoking the circuit simulator. This process filtered out the most-likely bad states of the circuit and helped reduce the number of states that needed to be sent to the simulator for accurate performance prediction. Any set of design parameters that did not satisfy the DC gain and PM specifications were taken out of the loop early. Their system was tested on a folded-cascode opamp in CMOS 65 nm. The results did not show values for convergence speedup; nevertheless, the work was able to find sizing solutions correctly.

A promising work using a graph NN (GNN) and RL was presented in [46]. The GNN achieved good prediction in applications where understanding the dependencies between related entities was important: chemistry (properties of molecules) and social networks (human interactions). The authors in [46] exploited the fact that a circuit is not different from a graph where nodes are components and edges represent connections (i.e., wires). They implemented a convolutional GNN that processed the connection relationship between IC components and extracted features in order to transfer knowledge from one circuit to another. The proposed technique was capable of transferring knowledge between two slightly different topologies when they had some common design aspects (e.g., two-stage and three-stage amplifiers) or between two different fabrication process nodes of the same topology (e.g., CMOS 65 nm and 45 nm). For instance, the proposed RL algorithm learned to change the gain of a differential amplifier by tuning the dimensions of the input pair transistors. A successful demonstration on four circuits in CMOS 0.18 µm including amplifiers and low-dropout voltage regulators was presented. The problem was formulated to maximize a figure of merit (FoM), which was defined as the weighted sum of several performance metrics. Finally, porting of two sized circuits from one larger process node to several lower ones was achieved with shorter runtime.

3. Hybrid IC Design Automation Methods

In this section, existing work where two different automation techniques were used is presented. Since a combination of several methods leads to hybrid solutions that differ significantly, it is difficult to define common criteria to discuss them.

Reference [33] proposed a two-step method targeting the re-design of existing analog circuits in new contexts. A context is a group of high-level conditions that impact the performance of an already pre-sized circuit (e.g., load, supply voltage values). The core idea is to use first a multivariate polynomial regression to estimate the performance trade-offs of the optimization and predict the circuit performance for the new context (which differ only moderately from the original ones). Then, the outputs are fed to an NN that trains itself from those updated examples to predict device sizes that correspond to the new performance (Figure 6).

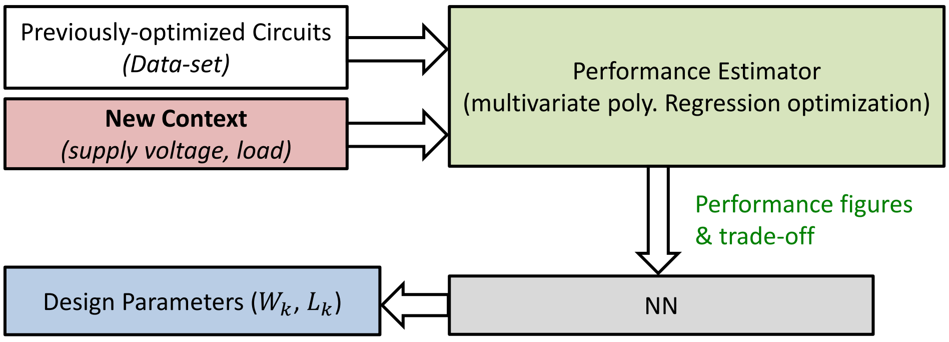

Previously-optimized Circuits
(Data-set)
New Context
(supply voltage, load)
Performance Estimator
(multivariate poly. Regression optimization)
Performance figures & trade-off
Design Parameters (W_k, L_k)
NN

Figure 6. Conceptual bloc diagram of the hybrid technique used in [33].

The previous two-step hybrid approach is a rapid tool for predicting (given a new circuit context) the sizing solutions corresponding to a near-optimal performance trade-off. A major aspect of this study was to benefit from existing designs and make an efficient reuse of them to predict the circuit performance under new conditions. The method was tested on a folded-cascode opamp with V_{th}-biasing designed in CMOS 0.13 µm. A re-design was made for new load capacitor values.

In [32], the core idea was to delegate part of the NN design to a genetic algorithm (GA) that iterated to decide on the following: the neural network type (multi-layer perceptron or radial-basis function) and which performance metrics and which design constraints to include as the input features during training. The GA was actually given a set of performance specifications and some additional design constraints: the algorithm worked by feeding consecutively several subsets of the two previous variables to the NN, thus removing and adding input elements depending on the training and validation results. The best combination of NN type and set of features/constraints was selected as the optimal solution to perform the prediction of the circuit. One novelty in the proposed technique was that the constraints on the design parameters (to be predicted) were actually presented as additional inputs to the GA-NN combination to help reduce the level of under-determination. This hybrid approach was tested on a wide-band low-noise amplifier (LNA) based on a common-source cascode circuit with inductive degeneration, working at 3 GHz with input and output matching networks (see Figure 7).

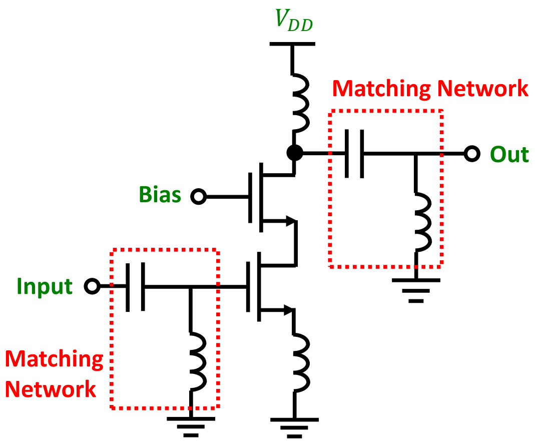


Figure 7. Schematic of the cascode low-noise amplifier circuit used in [32].

A novel learning framework based on evolutionary-algorithm optimization with an embedded neural network was proposed in [34]. At each step of the optimization, a new design was derived from the previous (population-based method). The NN came into action only to assist in the sampling process that selected the best candidate design for the next generation. Predicting whether an offspring design was better than its parent was realized by a Bayesian NN that mimicked the circuit simulator with an additional benefit of being faster. In fact, it did not predict the circuit performance exactly; instead, it only classified two design samples (parent and current offspring) and output which one was better for each specification separately. The NN was trained progressively during the optimization and converged to stable weights quickly, since it only helped in increasing the sample efficiency (see Figure 8). This novel method was tested on three different circuits: an amplifier similar to [14], a two-stage opamp with negative-gm loads, and a mixed-signal system comprising an optical link receiver front-end. Moreover, layout parasitic effects were included in the prediction, which constituted a clear advantage.

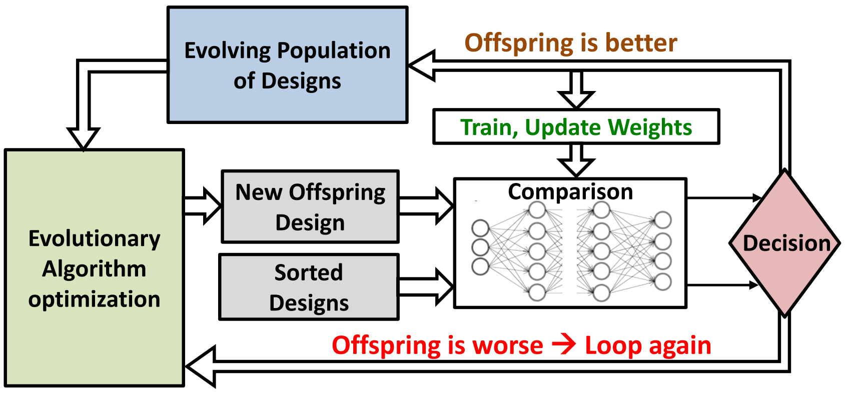

Evolving Population of Designs
Offspring is better
Train, Update Weights
Evolutionary Algorithm optimization
New Offspring Design
Sorted Designs
Comparison
Decision
Offspring is worse → Loop again

Figure 8. Bloc diagram of the hybrid approach adopted by [34].

A significantly different approach was used in [35], wherein the objective was to reduce the convergence time of the evolutionary algorithm that predicted the circuits' design parameters. Data produced from on-going optimization iterations were used to train an NN in parallel, in order to replace the time-consuming SPICE circuit simulators. Once the training was complete, the simulator was no longer used, and the NN would estimate the circuit performance rapidly, thus achieving quicker optimization results. The particularity of this work was that a separate NN was used for each performance specification. This hybrid method was successfully tested on a single-stage PMOS coupled-pair amplifier and a folded-cascode opamp with cascode loads and Vth biasing designed in 0.13 µm CMOS while achieving an ∼65% reduction in the execution time.

A time-efficient approach based on a genetic algorithm (GA) global optimization was adopted in [37]. SPICE simulations were first used with the GA to cover the large space encountered in analog circuits, while a local minimum search for neighboring design regions was computed using a trained NN. The latter was progressively trained during the GA process, and once completed, it replaced the time-consuming circuit simulators for local regions only. The method was tested on a two-stage rail-to-rail opamp in CMOS 0.18 µm with a fifth-order Chebyshev active filter achieving <2% average error and a four-times speed improvement compared to classic evolutionary algorithms.

With the objective of decreasing the computation cost, the authors in [36] used a Bayesian neural network (BNN) to model multiple circuit specifications by means of a novel training method (differential variational inference). The trained BNN was then introduced within a Bayesian optimization framework to capture the performance trade-off efficiently. The method was tested on a charge-pump circuit and a three-stage PMOS input device folded-cascode amplifier in 55 nm CMOS technology.

4. Discussion and Analysis

4.1. Analysis of the Previous Learning Approaches

Tables 5 and 6 provide a comparative summary of the studied prior state-of-the-art. Fabrication technology, input features, output design parameters, dataset size, and NN complexity were among the considered criteria. Another interesting aspect was the overall design time, which quantifies how long an ML agent needs to complete a circuit design as compared to humans. This time can be split into two parts: the dataset-generation and the NN training times. When applicable and available, these metrics are included in Table 5 to provide some insights into the required design time. However, the hardware platform used for computation was rarely stated, which somewhat limits the conclusions. It is worth noting also that a straightforward estimation of the reliability of the presented techniques was difficult to obtain since each work had a different circuit, fabrication process, dataset size, and design parameters. Nevertheless, several important aspects of the comparison are presented in the coming paragraphs.

Usually, hybrid techniques rely on neural networks to assist the main optimization algorithm that actually performs the circuit-sizing task, either by reducing the operation time (replace the circuit simulator) or by helping to decide which exploration path to undertake. The software framework that hosts the whole process is hence complex, since a circuit simulator, a neural network, and a global optimization algorithm are required to communicate at the same time. In contrast, supervised and reinforcement methods rely on the NN to achieve the objective, and therefore, the framework is simpler to build, with the circuit simulator feeding its results to the NN in one step (supervised) or progressively (reinforcement). Moreover, the network complexity varies among the existing solutions for the same circuit being designed (Tables 5 and 6).

The number of different circuit simulations that should be performed in the analog IC design (e.g., DC, AC, transient) is a bottleneck for the computation time. Some simulations are particularly time consuming such as the transient, while others are time efficient such as AC. Therefore, adding a new performance might lead to excessive simulation time and should be considered upfront during the problem definition. Circuits that require heavy transient simulations to be characterized are not well adapted to hybrid techniques that employ evolutionary algorithms, which are known to take an excessive time to converge. This point is worth considering when choosing between NN-based or hybrid techniques for a particular circuit.

Table 5. Comparative summary of supervised and reinforcement learning techniques in the prior state-of-the-art.

	[12]	[13]	[14]	[15]	[16]	[17]	[19]	[20]	[21]
ML Method	superv.	superv.	superv.	superv.	superv.	superv.	reinforc.	reinforc.	reinforc.
Circuit	bandgap reference	2-stage opamp	2-stage opamp	volt. combi. diff. opamp	2-stage opamp	2-stage opamp	3-stage opamp	2-stage opamp-gm	fold.-casc. opamp
Fabrication technology	CMOS missing	CMOS <1 µm	CMOS <1 µm	CMOS 0.13 µm	CMOS 0.18 µm	CMOS 0.18 µm	CMOS 0.18 µm	CMOS 45 nm	CMOS 65 nm
Perform. features	3	12	5	4 (exp. 14)	6	5	6	4	4
Design parameters	1	7	6	8	12	9	17	11	16
NN complexity	24 + 200 + 1	100 + 200 + 7	15 × 1024 + 6	120 + 240 + 60 + 8	1 × 9 + 12	50 + 9	missing	3 × 50 + 11	3 hidden layers
Average error (%)	0.25	7.8	4.4	7.7	3.7	<10	missing	3.7	missing
Dataset size	∼1600	20,000	9000	16,600	10,000	40,000	N/A	N/A	N/A
Generation time	missing	28 h	missing	0 (existing)	3.6 h	16 h	N/A	N/A	N/A
Training time	missing	19 min	18 min	1 h	1 h	30 min	30 h	1.6 h	missing

Table 6. Comparative summary of hybrid techniques in the prior state-of-the-art.

	[32]	[33]	[34]	[35]	[36]	[37]
Circuit	cascode LNA induc. deg.	fold.-casc. opamp Vth-bias	2-stage opamp-gm	fold.-casc. opamp Vth-bias	charge-pump and fold.-casc. opamp	rail-to-rail opamp 5th-ord. filter
Fabrication technology	CMOS 0.18 µm	CMOS 0.13 µm	CMOS 45 nm	CMOS 0.13 µm	CMOS 40/55 nm	CMOS 0.18/0.13 µm
Perform. features	6	5 (expanded 20)	3	4	6/3	4/3
Design parameters	10	19	11	19	16/13	9/8
NN complexity	17 + 10	$1 \times 100 + 19$	$5 \times 20 + 3$	missing	(25 + 10 + 13)/(10 + 13)	(8 + 9)/(16 + 3)
Average error (%)	6.6	∼1	missing	0.4	missing	0.5/0.8

The origin and reliability of a training dataset are crucial when using supervised learning in analog circuit design automation. How are the examples gathered? To what extent do they represent the full reality of the circuit? Answers to these questions determine the generalization capability of the obtained model, which guarantees that for new target specifications, we will obtain a set of relevant design parameters.

Some of the supervised learning techniques as in [13,14] constructed their dataset by randomly varying the design parameters in a close region around an initial solution corresponding to a pre-sized circuit with good performance. Thus, the NN model is highly correlated with this region within the overall design space, which raises questions on how good the predictions will be for solutions that are geometrically far in the design space from the initial one. Therefore, such a generation method creates local prediction models that are acceptable only if the intended application limits the circuit performance to a small region of the design space (opamps targeting audio applications always have low GBW values).

On the other hand, in [15,16], the dataset was collected from an existing database of solutions and, hence, depended totally on the availability and the validity of such an optimized set of examples. The dataset size may not be sufficient, nor is it possible to obtain on a wide range of circuits and fabrication technologies. Consequently, the computed model will have moderate generalization capability. To overcome this limitation, we recommend building a fully random dataset in order to cover the widest region in the overall design space.

On the other hand, hybrid methods rely on the wide exploration capability of the global optimization algorithms they employ (e.g., genetic) to cover the majority of the design space. Nevertheless, this comes at the cost of excessive simulation time and possible convergence issues.

A classification of all the studied machine learning techniques used in analog IC design in this paper is presented in Figure 9.

The supervised learning technique is further divided into three sub-categories depending on how the dataset is being generated: using an existing pre-sized circuit database, by random generation in a close region around a predefined initial solution, or from a global normal distribution. On the other hand, hybrid techniques employ NNs for different purposes: replace the time-consuming IC simulators that slow down the global optimization or assist the algorithm itself by sampling or search exploration speedup. Finally, some hybrid methods use the genetic algorithm to design the optimal NN structure and to select the features for training.

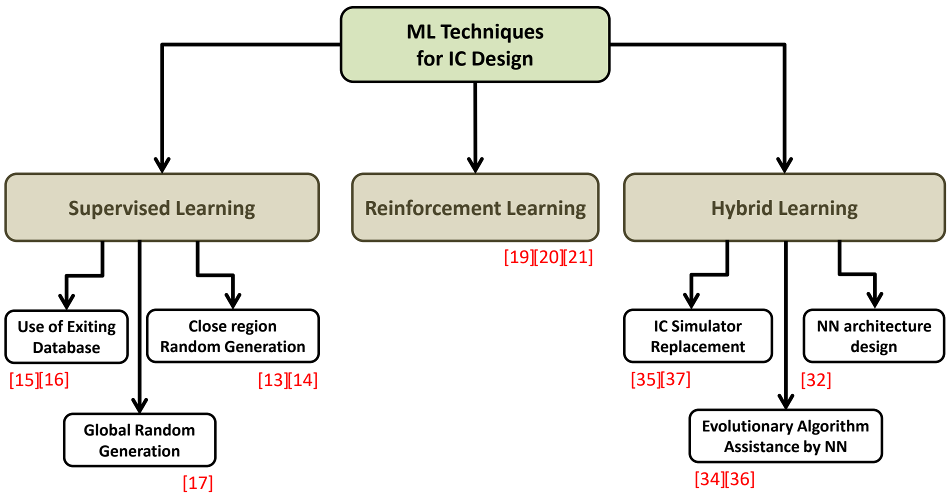


Figure 9. Classification of all the studied techniques in the prior state-of-the-art.

Figure 10 shows the different exploration capabilities of the supervised learning and hybrid techniques studied in this work. Without loss of generality, a simplified two-dimensional representation of the design space was adopted. The wider the exploration within the design space, the higher the generalization of the computed model will be. Ideally, fully random dataset generation would cover the whole white space in Figure 10 by sampling in multiple small regions spread over the entire design space, which is practically very difficult. Some randomized methods start by sampling over the whole design space before zooming-in on promising areas, which would help address this issue.

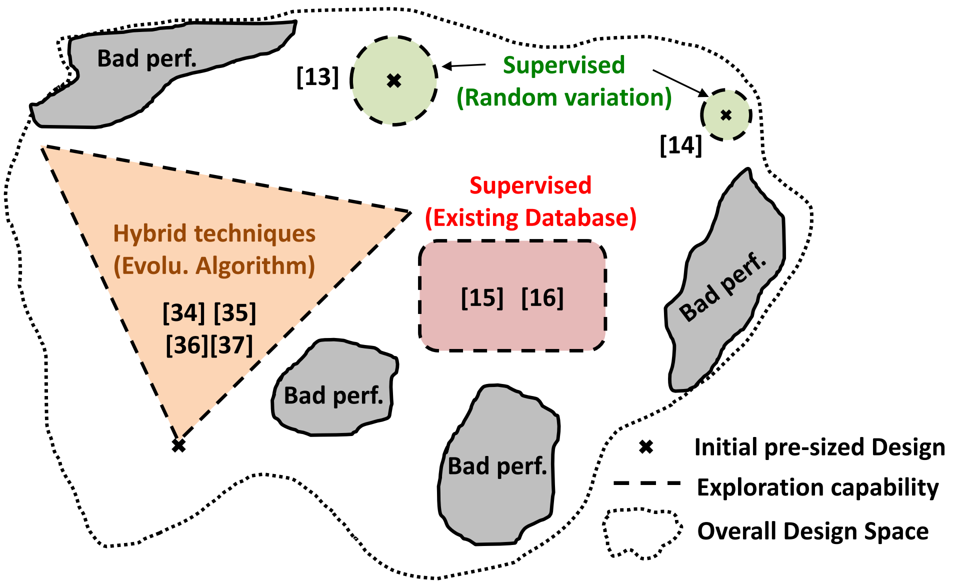


Figure 10. Exploration and generalization capabilities of all the studied techniques in the prior state-of-the-art.

4.2. Analysis of Circuit Features

In analog circuit design, performance metrics are rarely equivalent in importance, nor are they similar in definition. For instance, an opamp's phase margin (PM) is linked to its stability, and thus, no ML method has to match the specifications exactly to make the circuit stable. A higher PM than the target fixed by the circuit designer is highly desirable. On the other hand, a low-noise amplifier's (LNA) gain specification must be met by equality, since lower gain values create poor noise behavior, whereas higher values result in bad linearity performance. Therefore, it is recommended to take into consideration this point during an

ML algorithm's implementation, by classifying the metrics into three types: lower, higher and hard feature. For example, in an LNA, the voltage gain must be processed as a hard feature, while its power consumption as a lower feature, meaning that reaching below the target is acceptable. Such domain-specific insights can be incorporated dynamically using logically inferred rules, as in [47] or manually by domain experts, as is the case in the medical field.

In some cases, it is relevant to define two ways to measure the same performance depending on the input signal's magnitude. For instance, the linearity of a voltage buffer should be measured by computing the full-scale (FS) THD in addition to the IIP3 figure. Thus, both metrics should be included in the training dataset; otherwise, the ML model will be limited to the small-signal behavior of the circuit and cannot predict accurate designs that meet FS requirements.

4.3. Transistor Biasing Considerations

When simulating to collect the training dataset, design parameters are often generated randomly, which leads to a high number of bad examples (poor performance) due to the inaccurate biasing of the transistors (see Figure 10). Therefore, it is difficult for an ML technique to build a good model that contains such high variations in the input features. A way to overcome this limitation is to filter out (prior to training) all the examples for which the transistors are not in their intended region of operation, thereby reflecting the behavior of circuits accurately and allowing faster model convergence.

5. Open Challenges in the Field

Although the use of ML techniques to automate the analog IC design process has been a subject of intensive research recently, there are still topics that require further investigation in the field. In this section, we outline those research directions and point out relevant open challenges.

Despite the promising results obtained so far, there are very few studies in the prior state-of-the-art (if any) that deal with switched circuits. ML-assisted approaches to automatic sizing have not yet been explored for analog circuits in which transistors operate as switches. Mixers (both passive and Gilbert cell) and switched-capacitor filters (passive and opamp based structures) are widely used analog blocks that are interesting to investigate. Furthermore, from a circuit designer's perspective, some aspects are still understudied. For example, in fully differential circuits, common-mode feedback (CMFB) loops are systematically used to control and adjust the common-mode output voltage for the optimal output swing. Only one paper in the prior state-of-the-art focused on fully differential structures, yet without including the CMFB loop. The latter poses additional constraints on the circuit stability and increases the overall transistor count and the number of design parameters, as well as the NN's complexity.

One other open challenge is achieving an optimized multi-corner robust design. As is well known, process, voltage, and temperature (PVT) variations can lead to a significant performance drop and, in some cases, to a non-functional circuit. In a conventional approach, designers usually take margins with respect to the target specifications when sizing their blocks and then verify that their design is valid in all the corners. This approach requires fixing accurate margins for all specifications, since too large margins lead to over-sizing the circuit, while too small ones lead sometimes to redesigning. This approach could be used in the case of an ML-based design; however, it remains sub-optimal because the margins are chosen manually by the circuit designers. In [48], the authors addressed the PVT challenge using a novel gravitational search algorithm. Nevertheless, their approach was able to find the most robust and feasible solution for an existing design only. Including this aspect in the ML algorithm from the beginning of the design phase would constitute an interesting contribution that would help reduce the design time.

Referring to Tables 5 and 6, all the designed circuits in the prior state-of-the-art targeted CMOS technology exclusively. Moreover, they were designed using one library of transistors. The application of ML techniques to the design automation of circuits using multiple libraries such as low/standard/high V_{th} could be explored to widen the range of applicability of those methods. Applying ML techniques to design RF and analog circuits in BiCMOS, SOI, and other non-standard processes is another open challenge in the field. In the same line of thought, advanced CMOS technologies often exhibit many new effects that were not significant for older nodes. For example, an aspect such as shallow trench isolation should be integrated into the ML algorithm since it has a significant impact on the transistor performance, especially on the mobility and, as a consequence, on the threshold voltage [27]. This would require integrating parameters such as the number of fingers and even information about the layout (form factor, number of dummy transistors, etc.) as design features. None of the existing studies have explored this research direction.

In all the previous surveyed papers, except for one example in [34], analog circuits realizing one typical function were studied. An interesting and yet-unexplored research direction is to apply ML techniques to size and design an analog system composed of multiple building blocks. Analog-to-digital converters, digital-to-analog converters, track-and-hold amplifiers, and receiver and transmitter front-ends are good examples. The underlying benefit of such studies is to propose a flat-level optimization of the complete system using ML techniques, in contrast to the human expert approach and other hybrid approaches [49], where a breakdown of the specifications per block is usually performed.

6. Conclusions

This paper reviewed the existing state-of-the-art in front-end analog IC design automation using machine learning. Previous contributions in this field were classified into three main categories: supervised, reinforcement, and hybrid learning techniques. A comparative study was carried out based on relevant criteria such as the type of circuits, the NN complexity, the result validation, and feature selection methods. Different dataset-generation approaches in supervised learning were also discussed, and qualitative recommendations were given on how to build relevant data to enhance the exploration and generalization capabilities of the ML model. Furthermore, this work pointed out several open challenges (that have not been explored yet) when using machine learning for automated analog IC design: the sizing of switched circuits, the impact of including CMFB loops in sizing fully differential structures, the early estimation of PVT variations in the proposed solutions, the impact of specific effects related to advanced CMOS fabrication technologies, and finally, the feasibility of a flat-level optimization when designing complex systems composed of multiple sub-circuits.

Author Contributions: Formal analysis, R.M., C.J. and G.E.S.; investigation, R.M., C.J. and G.E.S.; writing—original draft preparation, R.M.; writing—review and editing, R.M., C.J. and G.E.S.; visualization, R.M. and C.J.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: The authors would like to thank Marc Ibrahim from Saint-Joseph University of Beirut for his insightful comments on the efficiency of global optimization methods and evolutionary algorithms.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gray, P.R.; Hurst, P.J.; Lewis, S.H.; Meyer, R.G. *Analysis and Design of Analog Integrated Circuits*, 6th ed.; Wiley Publishing: Hoboken, NJ, USA, 2017.
2. Lavagno, L.; Martin, G.; Scheffer, L. *Electronic Design Automation for Integrated Circuits Handbook*, 2nd ed.; CRC Press: Boca Raton, FL, USA, 2016.
3. Toledo, P.; Rubino, R.; Musolino, F.; Crovetti, P. Re-Thinking Analog Integrated Circuits in Digital Terms: A New Design Concept for the IoT Era. *IEEE Trans. Circuits Syst. Ii Express Briefs* **2021**, *68*, 816–822. [CrossRef]
4. Aiello, O.; Crovetti, P.; Alioto, M. Fully Synthesizable Low-Area Analogue-to-Digital Converters with Minimal Design Effort Based on the Dyadic Digital Pulse Modulation. *IEEE Access* **2020**, *8*, 70890–70899. [CrossRef]

5. Rémond, E.; Nercessian, E.; Bernicot, C.; Mina, R. Mathematical approach based on a "Design of Experiment" to simulate process variations. In Proceedings of the 2011 Design, Automation and Test in Europe, Grenoble, France, 14–18 March 2011; pp. 1–5.
6. Joet, L.; Dezzani, A.; Montaudon, F.; Badets, F.; Sibille, F.; Corre, C.; Chabert, L.; Mina, R.; Bailleuil, F.; Saias, D.; et al. Advanced 'Fs/2' Discrete-Time GSM Receiver in 90-nm CMOS. In Proceedings of the 2006 IEEE Asian Solid-State Circuits Conference, Hangzhou, China, 13–15 November 2006; pp. 371–374. [CrossRef]
7. Montaudon, F.; Mina, R.; Tual, S.L.; Joet, L.; Saias, D.; Hossain, R.; Sibille, F.; Corre, C.; Carrat, V.; Chataigner, E.; et al. A Scalable 2.4-to-2.7GHz Wi-Fi/WiMAX Discrete-Time Receiver in 65nm CMOS. In Proceedings of the 2008 IEEE International Solid-State Circuits Conference—Digest of Technical Papers, San Francisco, CA, USA, 3–7 February 2008; pp. 362–619. [CrossRef]
8. Nguyen, M.T.; Jabbour, C.; Ouffoue, C.; Mina, R.; Sibille, F.; Loumeau, P.; Triaire, P.; Nguyen, V.T. Direct delta-sigma receiver: Analysis, modelization and simulation. In Proceedings of the 2013 IEEE International Symposium on Circuits and Systems (ISCAS), Beijing, China, 19–23 May 2013; pp. 1035–1038. [CrossRef]
9. Wagner, G.; Mina, R. Amplifier for Wireless Receiver. 2015. Available online: https://patentimages.storage.googleapis.com/ca/21/0b/4bc82b44df4daa/US9077302.pdf (accessed on 13 January 2022)
10. Vural, R.A.; Kahraman, N.; Erkmen, B.; Yildirim, T. Process independent automated sizing methodology for current steering DAC. *Int. J. Electron.* **2015**, *102*, 1713–1734. [CrossRef]
11. Lohit, V.B.; Pandey, N.; Bhattacharyya, A. Modeling and Design of Inverter Threshold Quantization Based Current Comparator using Artificial Neural Networks. *Int. J. Electr. Comput. Eng.* **2016**, *6*, 320–329. [CrossRef]
12. Hasani, R.M.; Haerle, D.; Baumgartner, C.F.; Lomuscio, A.R.; Grosu, R. Compositional neural-network modeling of complex analog circuits. In Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, USA, 14–19 May 2017; pp. 2235–2242. [CrossRef]
13. Fukuda, M.; Ishii, T.; Takai, N. OP-AMP sizing by inference of element values using machine learning. In Proceedings of the 2017 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), Xiamen, China, 6–9 November 2017; pp. 622–627. [CrossRef]
14. Wang, Z.; Luo, X.; Gong, Z. Application of Deep Learning in Analog Circuit Sizing. In Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence, Shenzhen, China, 8–10 December 2018; pp. 571–575. [CrossRef]
15. Lourenço, N.; Rosa, J.; Maryins, R.; Aidos, H.; Canelas, A.; Póvoa, R.; Horta, N. On the Exploration of Promising Analog IC Designs via Artificial Neural Networks. In Proceedings of the 2018 15th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD), Prague, Czech Republic, 2–5 July 2018; pp. 133–136. [CrossRef]
16. Harsha, M.; Harish, B.P. Artificial Neural Network Model for Design Optimization of 2-stage Op-amp. In Proceedings of the 2020 24th International Symposium on VLSI Design and Test (VDAT), Bhubaneswar, India, 23–25 July 2020; pp. 1–5. [CrossRef]
17. Murphy, S.D.; McCarthy, K.G. Automated Design of CMOS Operational Amplifier Using a Neural Network. In Proceedings of the 2021 32nd Irish Signals and Systems Conference (ISSC), Athlone, Ireland, 10–11 June 2021; pp. 1–6. [CrossRef]
18. Grabmann, M.; Feldhoff, F.; Gläser, G. Power to the Model: Generating Energy-Aware Mixed-Signal Models using Machine Learning. In Proceedings of the 2019 16th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design, Lausanne, Switzerland, 15–18 July 2019; pp. 5–8. [CrossRef]
19. Wang, H.; Yang, J.; Lee, H.S.; Han, S. Learning to Design Circuits. *arXiv* **2018**, arXiv:abs/1812.02734.
20. Settaluri, K.; Haj-Ali, A.; Huang, Q.; Hakamaneshi, K.; Nikolic, B. AutoCkt: Deep Reinforcement Learning of Analog Circuit Designs. In Proceedings of the 2020 Design, Automation Test in Europe Conference Exhibition (DATE), Grenoble, France, 9–13 March 2020; pp. 490–495. [CrossRef]
21. Zhao, Z.; Zhang, L. Deep Reinforcement Learning for Analog Circuit Sizing. In Proceedings of the 2020 IEEE International Symposium on Circuits and Systems (ISCAS), Seville, Spain, 12–14 October 2020; pp. 1–5.
22. Russell, S.J.; Norvig, P. *Artificial Intelligence: A Modern Approach*, 3rd ed.; Pearson: London, UK, 2009.
23. Lyu, W.; Xue, P.; Yang, F.; Yan, C.; Hong, Z.; Zeng, X.; Zhou, D. An Efficient Bayesian Optimization Approach for Automated Optimization of Analog Circuits. *IEEE Trans. Circuits Syst. I* **2018**, *65*, 1954–1967. [CrossRef]
24. Lyu, W.; Yang, F.; Yan, C.; Zhou, D.; Zeng, X. Batch Bayesian Optimization via Multi-objective Acquisition Ensemble for Automated Analog Circuit Design. In Proceedings of the 35th International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; Volume 80, pp. 3306–3314.
25. Rout, P.K.; Acharya, D.P.; Panda, G. A Multiobjective Optimization Based Fast and Robust Design Methodology for Low Power and Low Phase Noise Current Starved VCO. *IEEE Trans. Semicond. Manuf.* **2014**, *27*, 43–50. [CrossRef]
26. Liu, B.; Zhao, D.; Reynaert, P.; Gielen, G.G.E. GASPAD: A General and Efficient mm-Wave Integrated Circuit Synthesis Method Based on Surrogate Model Assisted Evolutionary Algorithm. *IEEE Trans. -Comput.-Aided Des. Integr. Circuits Syst.* **2014**, *33*, 169–182. [CrossRef]
27. Martins, R.; Lourenço, N.; Póvoa, R.; Horta, N. Shortening the gap between pre- and post-layout analog IC performance by reducing the LDE-induced variations with multi-objective simulated quantum annealing. *Eng. Appl. Artif. Intell.* **2021**, *98*, 104102. [CrossRef]
28. Rojec, Z.; Bűrmen, Á.; Fajfar, I. Analog circuit topology synthesis by means of evolutionary computation. *Eng. Appl. Artif. Intell.* **2019**, *80*, 48–65. [CrossRef]

29. Okochi, K.; Takai, N.; Sugawara, Y.; Suzuki, K.; Yoshizawa, S.; Kobayashi, H. Automatic design of operational amplifier by combination method of function block. In Proceedings of the 2016 13th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT), Hangzhou, China, 25–28 October 2016; pp. 965–967. [CrossRef]
30. Negishi, T.; Arai, N.; Takai, N.; Kato, M.; Seki, H.; Kobayashi, H. Automatic Synthesis of Comparator Circuit Using Genetic Algorithm and SPICE Optimizing Function. *Key Eng. Mater.* **2015**, *643*, 131–140. [CrossRef]
31. Hjalmarson, E.; Hagglund, R.; Wanhammar, L. An equation-based optimization approach for analog circuit design. In Proceedings of the Signals, Circuits and Systems, SCS 2003, International Symposium on, Iasi, Romania, 10–11 July 2003; Volume 1, pp. 77–80. [CrossRef]
32. Dumesnil, E.; Nabki, F.; Boukadoum, M. RF-LNA circuit synthesis by genetic algorithm-specified artificial neural network. In Proceedings of the 2014 21st IEEE International Conference on Electronics, Circuits and Systems (ICECS), Marseille, France, 7–10 December 2014; pp. 758–761. [CrossRef]
33. Lourenço, N.; Afacan, E.; Martins, R.; Passos, F.; Canelas, A.; Póvoa, R.; Horta, N.; Dundar, G. Using Polynomial Regression and Artificial Neural Networks for Reusable Analog IC Sizing. In Proceedings of the 2019 16th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD), Lausanne, Switzerland, 15–18 July 2019; pp. 13–16. [CrossRef]
34. Hakhamaneshi, K.; Werblun, N.; Abbeel, P.; Stojanovic, V. BagNet: Berkeley Analog Generator with Layout Optimizer Boosted with Deep Neural Networks. In Proceedings of the 2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), Westminster, CO, USA, 4–7 November 2019; pp. 1–8. [CrossRef]
35. İslamoğlu, G.; Çakici, T.O.; Afacan, E.; Dündar, G. Artificial Neural Network Assisted Analog IC Sizing Tool. In Proceedings of the 2019 16th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD), Lausanne, Switzerland, 15–18 July 2019; pp. 9–12. [CrossRef]
36. Gao, Z.; Tao, J.; Su, Y.; Zhou, D.; Zeng, X. Efficient Performance Trade-off Modeling for Analog Circuit based on Bayesian Neural Network. In Proceedings of the 2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), Westminster, CO, USA, 4–7 November 2019; pp. 1–8. [CrossRef]
37. Li, Y.; Wang, Y.; Li, Y.; Zhou, R.; Lin, Z. An Artificial Neural Network Assisted Optimization System for Analog Design Space Exploration. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2020**, *39*, 2640–2653. [CrossRef]
38. Passos, F.; Roca, E.; Castro-López, R.; Fernández, F.V. A Comparison of Automated RF Circuit Design Methodologies: Online Versus Offline Passive Component Design. *IEEE Trans. Very Large Scale Integr. (VLSI) Systems* **2018**, *26*, 2386–2394. [CrossRef]
39. Wei, P.H.; Murmann, B. Analog and Mixed-Signal Layout Automation Using Digital Place-and-Route Tools. *IEEE Trans. Very Large Scale Integr.n (VLSI) Systems* **2021**, *29*, 1838–1849. [CrossRef]
40. Afacan, E.; Lourenço, N.; Martins, R.; Dündar, G. Review: Machine learning techniques in analog/RF integrated circuit design, synthesis, layout, and test. *Integration* **2021**, *77*, 113–130. [CrossRef]
41. Fayazi, M.; Colter, Z.; Afshari, E.; Dreslinski, R. Applications of Artificial Intelligence on the Modeling and Optimization for Analog and Mixed-Signal Circuits: A Review. *IEEE Trans. Circuits Syst. Regul. Pap.* **2021**, *68*, 2418–2431. [CrossRef]
42. Khailany, B.; Ren, H.; Dai, S.; Godil, S.; Keller, B.; Kirby, B.; Klinefelter, A.; Venkatesan, R.; Zhang, Y.; Catanzaro, B.; et al. Accelerating Chip Design With Machine Learning. *IEEE Micro* **2020**, *40*, 23–32. [CrossRef]
43. Aggarwal, C.C. *Neural Networks and Deep Learning*, 3rd ed.; Springer International: Berlin/Heidelberg, Germany, 2018.
44. Wolfe, G.; Vemuri, R. Extraction and use of neural network models in automated synthesis of operational amplifiers. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2003**, *22*, 198–212. [CrossRef]
45. Carusone, T.C.; Johns, D.; Martin, K. *Analog Integrated Circuit Design*, 2nd ed.; John Wiley and Sons Inc.: Hoboken, NJ, USA, 2012.
46. Wang, H.; Wang, K.; Yang, J.; Shen, L.; Sun, N.; Lee, H.-S.; Han, S. GCN-RL Circuit Designer: Transferable Transistor Sizing with Graph Neural Networks and Reinforcement Learning. 2020. Available online: https://arxiv.org/abs/2005.00406 (accessed on 13 January 2022)
47. Nassif, H.; Costa, V.S.; Burnside, E.S.; Page, D. Relational Differential Prediction. In *Machine Learning and Knowledge Discovery in Databases*; Flach, P.A., De Bie, T., Cristianini, N., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; pp. 617–632.
48. Dehbashian, M.; Maymandi-Nejad, M. An enhanced optimization kernel for analog IC design automation using the shrinking circles technique. *Eng. Appl. Artif. Intell.* **2017**, *58*, 62–78. [CrossRef]
49. Ding, M.; Harpe, P.; Chen, G.; Busze, B.; Liu, Y.H.; Bachmann, C.; Philips, K.; van Roermund, A. A Hybrid Design Automation Tool for SAR ADCs in IoT. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2018**, *26*, 2853–2862. [CrossRef]