International Journal of Scientific Research & Engineering Trends
Volume 12, Issue 5, Sep-Oct-2026, ISSN (Online): 2395-566X

A Review of Machine Learning Techniques for Analog-To-Digital Converter Design Automation

Jai Akkayarajkumar V J
School Of Electronics Engineering Department Of Micro & Nano Electronics Vellore Institute Of Technology (Vit)
Vellore, Tamil Nadu, India
mail: jaiakkayarajkumar.v2025@vitstudent.ac.in

ABSTRACT- ADCs play an important role in converting information from the analog world to the digital one; with modern applications requiring higher resolutions, speed, and power efficiency, it has become very difficult to design them. Traditionally, the design of an ADC follows an iterative design process that includes stages like specification, behavioral modeling, transistor sizing, schematic simulation, layout, and post-layout verification, which is laborious and requires much designer expertise due to the consideration of process-voltage-temperature variations and yield. In this work, we perform a review of the latest advancements in the use of machine learning (ML) to tackle the challenges of ADC design in recent years, with systematic search from 2015 to 2026 in IEEE Xplore, ACM Digital Library, and arXiv. Rather than organizing the literature on different ML algorithms, this paper categorizes ML approaches such as regression, tree-based models, SVM, ANN, CNN, GNN, reinforcement learning, evolutionary algorithms, and Bayesian optimization in ADC design stages such as behavioral modeling, transistor sizing, calibration of comparators and capacitor arrays, calibration of ADC, layout optimization, prediction of PVT/yield variations, and design space exploration. Comparative analysis includes the benefits provided by each technique such as speed and accuracy improvements, as well as their main drawbacks. Open challenges are then presented, including lack of data, poor generalization performance, low explainability, and the largely unexplored topic of co-optimization across multiple stages. Finally, future trends in the field are presented in terms of new directions such as generative AI, GNN-based layout optimization, physics-informed learning, digital twins, and human-AI collaborative design, as well as risks and potential problems such as manufacturability and IP in generative approaches.

Index Terms—Analog-to-Digital Converter (ADC), Artificial Neural Network (ANN), Bayesian Optimization, Design Automation, Design-Space Exploration, Machine Learning, Reinforcement Learning, SAR ADC, XGBoost.

I. INTRODUCTION

Analog-to-Digital Converters (ADCs) are one of the key components in modern electronic systems that act as the interface between the analog world of physical phenomena and the digital domain, which provides for the processing and computation of information. Any system that processes real-world signals, such as temperature sensors, biomedical instruments, audio equipment, wireless receivers, radar systems, or fast data acquisition devices uses an ADC in its chain [1]. As new challenges emerge in these domains, ADC designs face the increasing performance requirements: the higher resolution for critical systems, faster sampling for communication applications and lower power for portable and battery-operated devices [2].

Design of ADCs meeting these requirements has become increasingly difficult over the years. ADCs have to satisfy the tight constraints not only in resolution and speed but also in power and area; all these parameters are often mutually conflicting [3]. Moreover, with the transition to smaller technologies, the issues of transistor mismatch, low operating voltages, and PVT (process, voltage and temperature) sensitivity have to be addressed as well [3]. This makes the conventional analog design methods used for many decades by the industry reach their practical limits.

The conventional ADC design flow is inherently iterative. After defining the specifications, the designer models the circuit behavior, size the transistors, simulates the schematic, performs layout and simulation at that level and, finally, does the post-layout verification. Often the iterations through these steps are needed due to the performance deficiencies observed at certain PVT corners or due to the parasitic effects. All

these steps take considerable amount of design time and require extensive simulations; the design quality is often determined by the designer's experience [4].

Recent advances in machine learning (ML) have provided promising solutions for resolving these issues. In contrast to the complete replacement of designer knowledge by

© 2026 IJSRET

algorithms, ML algorithms try to enhance the design process with the help of pattern recognition from previous design data, performance prediction without performing full simulations, automatic optimization of device parameters, and even circuit layout optimization [5]. Regression methods, decision trees, ANNs, CNNs, GNNs, RL, as well as other advanced techniques like genetic algorithm (GA) and particle swarm optimization (PSO) have been used for different design stages including analog circuits and ADC design [6].

At the same time, the number of studies in this area is not enough for a person new to the field (e.g., graduate student or researcher) to understand the potential of AI in analog circuits design. Existing papers typically describe only one ML method applied to one design stage, e.g., offset prediction using ANN or transistor sizing via RL without providing information about its place in the overall design process. It is difficult for a beginner to understand the current state of the problem and the possible direction for future research in this area.

Thus, this review will try to fill that gap. Instead of grouping the contents of the review depending on different types of algorithms used, in this work we will group the content of the main part of the review by different steps of the ADC design process, namely behavioral modeling, transistor sizing, comparator optimization, capacitor-array optimization, calibration, layout optimization, PVT/yield prediction and design space exploration. Each step will include information about which AI technique was utilized in it, its results and current limitations.

It should be noted that in this review we will consider only application of machine learning in the design and pre-silicon verification stages of ADC development (behavioral modeling up to layout and PVT/yield prediction); assistance of ML in post-silicon ADC testing, characterization and production time calibration is not considered in this work.

The rest of this paper is organized as follows. In Section 2 we give an overview of the fundamentals of ADC design: different architectures and performance metrics. The traditional ADC design flow is discussed in detail in Section 3. The limitations of this process are given in Section 4. Section 5 describes the reasons why there is a need for AI-assisted ADC designs. ML techniques used for this purpose are considered in Section 6. Application of AI in each stage of the ADC design flow is analyzed in Section 7 – the core of this review. Section 8 contains a comparative evaluation of these techniques and their results. Research challenges and gaps are mentioned in Section 9. Future research directions are given in Section 10, and Section 11 is the conclusion of this work.

II. SUMMARY OF REVIEWED LITERATURE

As we proceed to discuss the basics of ADCs and the design flow, it is appropriate to take a brief detour into summarizing the general literature that this review is based on. The surveyed works cover eight main topic areas—ADC architectures, design of sigma-delta and flash, basic ML algorithms (regression, tree-based, SVM), neural networks (ANN, CNN, GNN), optimization approaches (RL, GA/PSO, Bayesian optimization), PVT/yield prediction, ADC calibration, and new emerging directions in future research including generative AI and LLM-assisted design. Table II provides a summary of all these streams of work, stating the advantages, limitations, and major takeaway of each topic area, in order to let the reader recognize which part of literature is more interesting for him/her before moving forward with the rest of the review.

A number of recurring trends can be identified among the literature reviewed. Firstly, the basic ML algorithms (regression, tree-based approaches, SVM) are the most suitable starting point due to limited availability of datasets in analog design, while deep learning methods (ANN, CNN, GNN) prove themselves better when the amount of training data is enough. Secondly, optimization-based methods—RL, GA/PSO, and Bayesian optimization—are dominant in the literature of sizing and design space exploration, since these problems require more active searching compared to prediction. Thirdly, the highest gain in efficiency can be observed for PVT/yield prediction, mainly because of high costs associated with corner simulation which is the most computationally expensive verification step in traditional methodology. Finally, the emerging areas like generative AI, digital twins, and LLM-assisted EDA are relatively immature, with most of the reported works being mostly theoretical and simulated.

While several studies have investigated the topic of ML applications in EDA, the majority of those surveys cover the field quite generally without focusing on specific difficulties that arise when working with analog circuits. Alternatively, those surveys tend to present algorithms according to their classes (for example, one section devoted to reinforcement learning and one devoted to GNNs), failing to establish the position of each method within a design flow. The current study stands out for two reasons: first, because it deals solely with ADCs, thus enabling more thorough consideration of ADC-specific problems such as the issue of comparator offset and capacitor array matching; second, because its main body is organized according to the eight-step ADC design flow, which means that a researcher who, for instance, needs information about layout optimization, can immediately turn to the relevant subsection.

Table I. Literature Survey Summary

© 2026 IJSRET

Topic Category	Ref.	Advantages	Disadvantages	Key Takeaway
ADC Architectures (SAR, Pipeline)	[1]–[6]	SAR: low power, simple, scalable with process node. Pipeline: high throughput, good for high-speed/high-res applications	SAR: comparator offset and capacitor mismatch limit accuracy. Pipeline: needs precise inter-stage gain/offset calibration, sensitive to amplifier nonlinearity	Architecture selection directly determines which design bottlenecks (matching, calibration, speed) AI needs to address later
Sigma-Delta & Flash ADC	[7]–[12]	ΣΔ: exceptional resolution via oversampling + noise shaping, ideal for audio/precision sensing. Flash: fastest architecture, single-cycle conversion	ΣΔ: slow, unsuitable for high-speed applications. Flash: exponential comparator growth causes high power/area at higher resolution	Each architecture's core limitation (speed vs. power vs. resolution) becomes the specific target for AI-driven optimization
ML Techniques — Regression/Tree-based/SVM	[13]–[18], [26]–[29]	Fast training, interpretable outputs, work well with smaller datasets typical in analog design	Struggle with highly nonlinear relationships (e.g., ENOB, SFDR vs. sizing); SVM less effective on large datasets	Good for early-stage screening and baseline prediction, not for complex behavior modeling
Neural Networks — ANN/CNN/GNN	[30]–[38]	ANN captures nonlinear parameter-performance mapping; CNN handles spatial layout data; GNN captures circuit topology/connectivity	Require large labeled datasets (expensive to generate via SPICE); act as black boxes, hurting explainability	Best suited for behavioral modeling, layout, and topology-aware tasks where sufficient data exists
RL / GA / PSO / Bayesian Optimization	[19]–[25], [28]–[35]	Enable active design-space search rather than passive prediction; RL good for sequential sizing decisions; GA/PSO handle multi-objective trade-offs; Bayesian Opt minimizes simulation count for expensive evaluations	High computational cost per training run; RL sample-inefficiency; less interpretable decision paths	Most impactful for automating transistor sizing and multi-block co-optimization
PVT/Yield Prediction	[39]–[43]	Drastically reduces exhaustive corner/Monte Carlo simulation time; enables early yield screening before full verification	Poor generalization across process nodes/architectures; models often need retraining per technology	Shows the largest reported speedup among all reviewed design stages
Calibration (ADC-specific)	[47]–[50]	ML-based calibration adapts in real-time to offset/mismatch drift, unlike static lookup-table methods	Limited validation on fabricated silicon; mostly simulation-based results so far	Promising for lifetime performance tracking, but not yet industry-proven
Future Directions — GenAI, Digital Twin, LLM-assisted EDA	[51]–[60]	GenAI can synthesize novel topologies/synthetic data; digital twins enable lifecycle tracking; LLMs simplify designer-tool interaction via natural language	Mostly early-stage/conceptual; limited real ADC-specific validation; LLMs lack deep circuit-level reasoning currently	Represents the next research frontier rather than current practical solutions

This overview sets the stage for the more detailed, stage-wise treatment that follows: Section 2 covers ADC fundamentals, and Section 7 later revisits many of these same techniques in the context of specific design stages.

III. LITERATURE SEARCH METHODOLOGY

© 2026 IJSRET

The literature used in this paper was collected through a systematic search on IEEE Xplore, ACM Digital Library, arXiv, and Google Scholar databases, which included literature from 2015 to 2026 to ensure both foundational and more recent research is covered. The search keywords consisted of ADC-related keywords ("ADC," "analog-to-digital converter," "SAR," "pipeline," "sigma-delta"), ML-related keywords ("machine learning," "reinforcement learning," "neural network," "Bayesian optimization," "GNN"), and design stages ("sizing," "layout," "calibration," "yield prediction"). Papers meeting the criteria for inclusion in the paper were those using any machine learning approach in one stage of analog/ADC circuit design and published in peer-reviewed conference/journal or pre-print repositories with technical contributions. On the other hand, studies focusing on digital circuit design, studies where there was no methodology for reproducibility given, and studies that were not in English and could not be translated into English were excluded. Out of about 150 candidate papers found initially using keyword search, relevance to ADC/analog design and ML application reduced this number to 60 papers cited in this review.

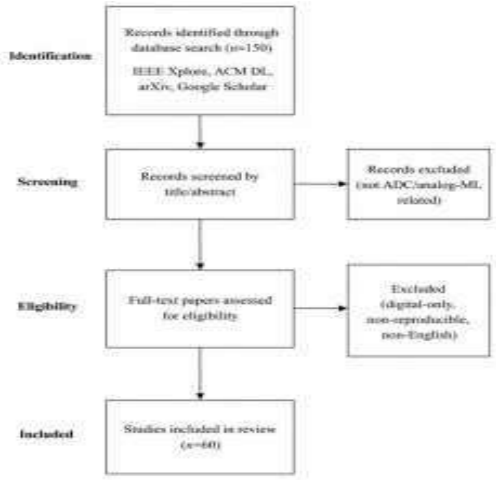


Fig. 1— PRISMA-style Search Methodology Flowchart

- ADC Fundamentals
- ADC Architectures

It is important to understand the advantages and drawbacks of the various ADC architectures before moving onto discussing how the application of AI technologies for designing ADCs can be done. Successive approximation register ADCs implement a kind of binary search, comparing the input signal against a reference voltage produced by capacitive or resistive DAC, and determining the digital output successively bit-by-bit [7]. Advantages of these ADCs are relatively low power consumption, not too complex design, and good scalability

with the progress of process technology, which explains their dominance in medium-speed and medium-resolution applications like biomedical sensing, IoT devices, data acquisition systems etc. Design issues of this type of ADCs are comparator offset, mismatch of capacitors in the DAC array, and the accuracy of the settling.The conversion in a pipeline ADC is split between several stages, each converting a small number of bits and passing an amplified residue of the signal to the next stage [8]. Such architecture gives high throughput, hence the pipeline ADCs are applicable in case when speed and resolution are both required, like for video processing, high-speed instrumentation etc. At the same time, pipeline ADCs require high-accuracy gain and offset calibration between the stages and have high sensitivity to the nonlinearity of amplifiers and capacitor mismatch.

In contrast to other types of ADCs, Sigma-Delta (ΣΔ) ADCs implement a completely different method of operation, based on oversampling and noise shaping, i.e. using the feedback loop to move quantization noise outside the bandwidth of the signal, where it can be filtered out digitally [9]. This allows achieving very high resolution but usually with low conversion speed, and, thus, ΣΔ ADCs are used in cases when high accuracy is the primary requirement rather than high conversion speed, such as audio codecs, sensor interfaces, and instrumentation.Flash ADCs are the fastest converters implementing conversion with a bank of parallel comparators, each corresponding to a certain quantization level, within a single clock cycle [10]. Although this solution provides very high speed, the number of comparators grows exponentially with the resolution, thus imposing area and power limitations for flash ADCs, which limits their usage to low-resolution applications with ultrahigh speeds, such as high-frequency oscilloscopes.

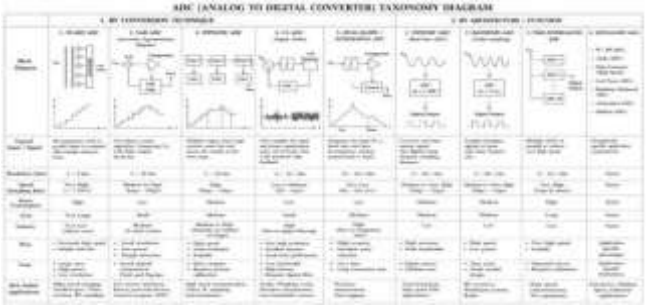


Fig. 2. Taxonomy of ADC Architectures.

© 2026 IJSRET

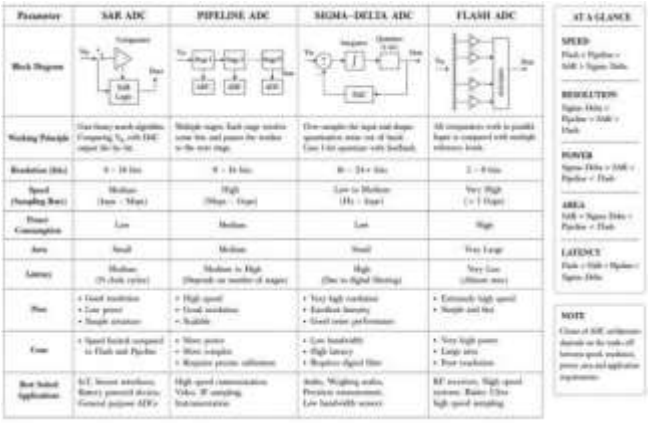


Fig. 3. Comparison of ADC Architectures (SAR, Pipeline, Sigma-Delta, Flash)

Table II. ADC Architecture Comparison

Architecture	Resolution	Speed	Power	Best Use Case
SAR	Medium	Medium	Low	IoT, biomedical
Pipeline	High	High	Medium	Video, high-speed DAQ
Sigma-Delta	Very High	Low	Low-Medium	Audio, precision sensing
Flash	Low	Very High	High	Oscilloscopes, sub-blocks

1. **Performance Metrics**

The performance of the ADC can be quantified using certain parameters which are also used as target variables by several ML models.

1. The Effective Number of Bits (ENOB) is the actual number of bits offered by the ADC after taking into account the noise and distortion and not just the nominal number of bits, and is generally calculated using the value of SNDR.

$$\text{ENOB} = \frac{\text{SNDR} - 1.76}{6.02}$$

where SNDR is expressed in dB.

2. Signal-to-Noise-and-Distortion Ratio (SNDR) is a measure of the impact of both noise and harmonic distortion on the quality of the output signal, and is one of the most common dynamic performance specifications.

$$\text{SNDR} = 10\log_{10}\left(\frac{P_{\text{signal}}}{P_{\text{noise}} + P_{\text{distortion}}}\right)$$

Signal-to-Quantization-Noise Ratio (SQNR):

For an ideal N-bit ADC,

$$\text{SQNR} = 6.02N + 1.76 \text{ dB}$$

- Spurious-Free Dynamic Range (SFDR) determines the ratio between the amplitude of the wanted signal and that of the highest unwanted signal in the frequency domain, and is one of the important measures in radio frequency and wireless communications systems where spectral purity is crucial.
- Integral Nonlinearity (INL) and Differential Nonlinearity (DNL) represent two parameters which determine static accuracy – the deviation of the actual characteristic curve from the linear one (INL) and consistency in step size between successive values of input signal (DNL)[11].
- Power consumption and Figure of Merit (FoM) The energy efficiency is determined by the combination of power consumption and Figure of Merit (FoM), which includes power, sampling rate, and ENOB in one parameter.

1. Walden Figure Of Merit (Fom):

$$\text{FoM}_{\text{Walden}} = \frac{P_{\text{ADC}}}{2^{\text{ENOB}} f_s}$$

2. Where Padc Is The Adc Power Consumption And Fs Is The Sampling Frequency.

Table III. ADC Performance Metrics Summary — Metric, Definition, Typical Unit, Design Significance

Metric	Definition	Typical Unit	Design Significance
ENOB	Effective resolution accounting for noise/distortion	Bits	Reflects real-world accuracy vs. ideal bit count
SNDR	Ratio of signal to combined noise+distortion	dB	Core dynamic performance indicator
SFDR	Gap between signal and largest spurious tone	dBc	Critical for RF/communication purity
INL	Deviation from ideal linear transfer curve	LSB	Static linearity accuracy
DNL	Step-size consistency between codes	LSB	Missing-code/monotonicity risk

© 2026 IJSRET

Metric	Definition	Typical Unit	Design Significance
Power/FoM	Energy per conversion, normalized across specs	mW / fJ/conv-step	Efficiency benchmark across designs

The above mentioned metrics constitute the foundation for any ADC assessment and as we'll see in the following paragraphs, quite often, these metrics themselves become the target of machine learning models which are trained on design variables.

IV. CONVENTIONAL ADC DESIGN FLOW

Traditional ADC design methodology is an extensively developed and refined process that has evolved over many years of analog and mixed-signal design. Although quite reliable, it is rather iterative, and knowing all the details of each of its steps is vital to comprehend where AI can be introduced.

The initial step includes specification and behavioral modeling where the system-level parameters such as resolution, sampling rate, power budget, and the field of application are mapped into a behavioral model which is normally created in MATLAB, Simulink, or other system-level software [12]. At this step, the architecture (SAR, pipeline, ΣΔ, or flash) can be selected and tested against the target specifications without any transistor-level work to see if it needs to be modified to save considerable costs later.

The next step is transistor-level sizing when designers set the parameters of devices – W/L ratios, bias currents, resistors, and capacitors – in each sub-block of an ADC (comparator, DAC, sample-and-hold block, reference generation block) to fulfill the behavioral-level requirements [13]. Transistor-level sizing is probably the most laborious step in the whole design process since designers need to balance various requirements and use their vast knowledge of the process technology.

Schematic-level simulation is used to check whether the circuit meets all the requirements in different corners of process, voltage, and temperature (PVT) and in Monte Carlo mismatch conditions [14]. The failure to do so leads to returning to the transistor-level sizing step, and the loop sizing-simulation may be repeated tens of times.

Upon confirming schematic-level performance, the design goes through the layout step wherein the physical positioning and connection of transistors, capacitors, and interconnections are designed. The layout of an ADC is highly susceptible to parasitic effect and hence needs much consideration in symmetry and matching, especially the differential pair of comparators and capacitor-DAC array because any layout mismatch may lead to significant loss in linearity (INL/DNL) [15].

Next, DRC (Design Rule Check) checks whether the layout satisfies the foundry's manufacturing rules and LVS (Layout Versus Schematic) verifies whether the physical layout corresponds to the schematic connection. After that, the Parasitic Extraction (PEX) process will generate a more accurate netlist that includes the parasitics from the physical layout like the resistive and capacitive ones which do not exist in the schematic.

Finally, post-layout verification will simulate the extracted netlist to check if the design still fulfills the targeted specifications taking into account the parasitic effect of the physical layout. It is typical for the performance of post-layout design to deteriorate compared to the schematic level and hence the designer will need to go back again either to sizing or to layout steps in order to improve it [16].

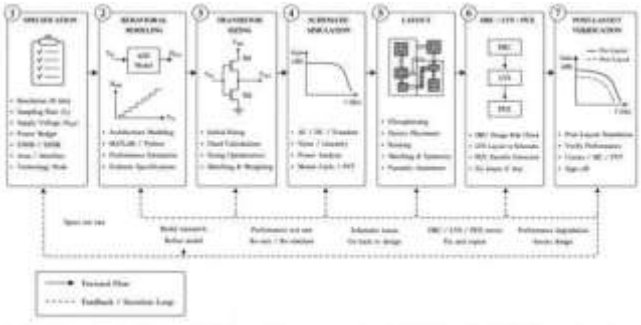


Fig. 4. Conventional ADC Design Flow — Specification → Behavioral Modeling → Transistor Sizing → Schematic Simulation → Layout → DRC/LVS/PEX → Post-Layout Verification, with feedback loops explicitly shown

All this shows that ADC design is a process that is costly in terms of resources since most of the steps in the process have the potential to require going back to a previous step, which incurs costs of its own. This will set the scene for the discussion in the next section.

V. CHALLENGES IN CONVENTIONAL DESIGN

The design flow discussed in section 3, even though it is quite reliable due to years of usage and application, has some drawbacks that have become even more significant due to stringent ADC requirements, scaling down of process nodes, and time-to-market considerations. All of them do not operate independently but interact with each other and therefore create a bottleneck for the adoption of an AI-based methodology.

© 2026 IJSRET

Time-consuming iterations. The essence of the traditional design approach is the sizing/simulation cycle, which is a trial-and-error method. A designer modifies the size of the transistors or the bias condition according to his previous experience, checks the effect with the help of the simulation tool, and then goes through the loop again and again – sometimes tens of times per a sub-block – before finding a design which meets the requirements for all PVT corners [17]. In the case of such interconnected system as the ADC, whose comparator, capacitor-DAC array, sample-and-hold unit, and reference generation unit affect each other, optimization of one particular sub-block does not provide any guarantee of achieving desired system level results. As a result, the sizing loop needs to be performed several times in connection with refinement of different sub-blocks, which leads to the overall simulation time that can reach weeks or even months for a complete design cycle.

High reliance on experience of the designer. Both the process of sizing and the selection of the topology rely heavily on heuristics and the experience of the designer rather than any systematic and reproducible methodology [18]. This fact brings about two significant problems. First, the quality and the speed of design depends heavily on the person working on it, as two engineers of different level of experience will likely come up with entirely different designs when solving the problem with the same specification. Second, it creates a high entry barrier for young engineers joining the industry – a significant part of the "tribal knowledge" of the experienced analog designer is not documented anywhere.

PVT variance and yield difficulties. Current ADC designs have to remain functional over a large number of process-voltage-temperature corners, and in addition, the latter have to be statistically analyzed using Monte Carlo mismatch analysis to estimate manufacturing yield [19]. The variance in the characteristics of transistors at an advanced process node is larger; thus, a circuit design that works well at typical conditions may be failing at worst-case corners or may produce an unacceptably low manufacturing yield due to mismatch. Performing corner and Monte Carlo analysis on each design version can be computationally costly, while skipping it can result in costly silicon failures.

Performance degradation due to layout. As explained in Section 3, moving from a circuit design to a physical layout typically implies the introduction of parasitic resistances and capacitances which have not been taken into account during the process of transistor sizing. Parasitics may lead to a degradation of certain performance metrics, especially those related to linearity (e.g., INL and DNL), since the accuracy of an ADC is highly dependent on capacitor mismatch and symmetry in the DAC array and comparator [20]. Once post-layout simulation shows such degradation, the only remedy is to go back to sizing or layout modification, thereby re-doing parts of a rather long procedure.

Limitations related to scalability. With the resolution of an ADC and its sampling speed increasing, the number of design variables – widths and lengths of transistors, bias currents, capacitor ratios, and many other parameters and their possible combinations – becomes extremely large. It is called "the curse of dimensionality" in the realm of analog design [21]. Exhaustive exploration of such a parameter space in order to find a near-optimal design point can become unfeasible using simulation-driven approaches, especially given a finite period of time allocated for the project.

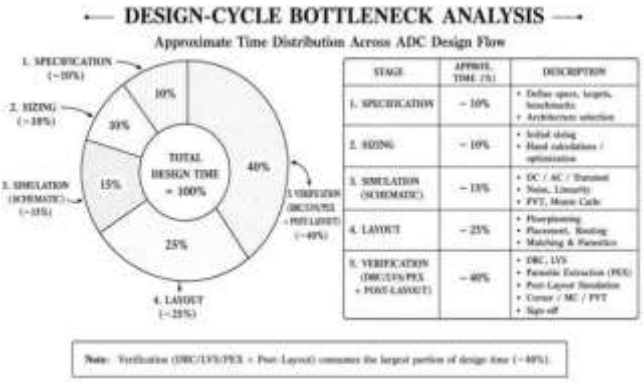


Fig. 5. Design-Cycle Bottleneck Analysis — chart showing approximate time distribution across specification, sizing, simulation, layout, and verification stages.

When all of these five problems are taken into consideration together, it becomes evident that the conventional method used to design ADCs is no longer viewed as a sustainable long-term solution, even though it works reliably, due to the issues listed above. This creates the basis for understanding the motivation behind studying how artificial intelligence can be used to alleviate these problems, which will be addressed further on.

1. Need For Ai

All of the above-mentioned difficulties stem from the fact that current design of ADCs relies on a laborious exploration of the huge, tightly coupled design space mostly through manual trial-and-error, and this paradigm reaches its limit due to the increasing circuit complexity and need to reduce design cycle times. Here lies the main advantage of machine learning over classical design approaches – it provides a fundamentally different way of operating.

Replacement of costly simulation with quick predictions. The first advantage ML gives is an opportunity to learn the relationship between the design parameters (sizes of transistors, bias currents, capacitances) and resulting performance characteristics (ENOB, power consumption, INL/DNL) directly from the existing simulation results [22]. After training on a representative set of examples, the model

© 2026 IJSRET

is able to quickly (in milliseconds) predict the performance metrics for a set of new parameters, compared to the minutes and even hours needed for SPICE-level simulation. It does not replace simulation completely, but significantly reduces the frequency of full simulation runs during the exploratory design stage and allows to examine much larger part of the design space within the same time frame.

Replacement of individual expertise with data-driven solutions. In contrast to conventional design, which relies heavily on the designer's personal experience and intuition, the ML models trained on historical design datasets give a much more systematic way of designing [23]. The experience of a designer gained over many years can be statistically collected from previous design cases and applied on multiple occasions. It does not negate the necessity of good designers, but helps to make the design more consistent and less dependent on the individual knowledge of one person.Faster PVT testing and yield verification. ML models are trainable on Monte Carlo or corner simulation datasets and can predict yield sensitivities and worst-case performance without needing to run all corner simulations oneach candidate design [24]. This makes it possible to perform fast filtering of promising candidates without having to pay the cost of corner sweep verification on many designs while exploring the solution space.

Active optimization rather than prediction. The point of reinforcement learning, genetic algorithms, particle swarm optimization, and Bayesian optimization, for example, is not prediction of design outcomes but searching the design space and proposing solutions [25]. Instead of waiting for the human designer to come up with new design parameters after the simulations are performed, these methods automatically iterate and converge toward some target specification, performing a significant part of the sizing and tuning traditionally done by a human designer.A collaborative tool, not a designer replacement. It is important to note that the purpose of AI-assisted design is to eliminate the need for a human designer to do routine calculations, not to eliminate a human designer from the design process. Human intuition and judgment is very useful in architectural and tradeoff decisions and sign-off processes.

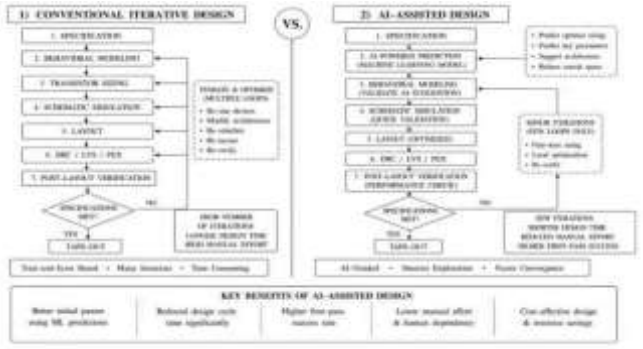


Fig. 6. Conceptual illustration — "Conventional Iterative Design" vs. "AI-Assisted Design" workflow comparison

The remainder of this paper builds on this motivation, first introducing the relevant ML techniques (Section 6), then examining in detail how they are actually applied across each stage of the ADC design flow (Section 7).

2. Ml Techniques Overview

Before we get into how AI is being used through the ADC design process (Section 7), it is helpful to gain an understanding of machine learning itself, what it does, and why some methods work better for some design challenges than others.

3. Regression And Tree-Based Methods

Linear regression (LR) seems to be the most straightforward way for prediction since we should fit a linear model that describes the dependence between the design parameters and performance metrics. It is highly efficient and easy to understand, and it works well when the relation is nearly linear; however, performance metrics of ADCs like ENOB and SFDR are mostly nonlinear functions of transistors sizing [26].

Random Forest (RF) predicts an outcome by constructing an ensemble of decision trees trained on a different random subset of data and averaging their predictions. RF does a much better job with non-linear relationships than LR and is quite robust to noisy data, making it a frequent choice for early performance predictions [27].

XGBoost (Extreme Gradient Boosting) is a further improvement on RF algorithm, training the decision trees sequentially with each subsequent tree learning to correct the error of the previous tree. It usually results in a higher accuracy of prediction with lower overfitting and became one of the most frequently used models in analog/ADC performance prediction literature thanks to that fact [28].

4. Support Vector Machines (Svm)

SVMs use kernel functions for separating or fitting the data into a certain plane of separation (or regression plane, in the case of SVR). SVMs have been applied in the design of ADC

© 2026 IJSRET

for classification tasks such as yield prediction (pass or fail) and regression task for performance prediction where datasets are small [29].

5. Neural Networks (Ann, Cnn, Gnn)

An artificial neural network (ANN) is a series of interconnected nodes which form a deep learning network for complex, non-linear mapping between inputs and outputs. ANN can flexibly model complex, non-linear relationships between transistor-level variables and circuit performance due to its flexibility given sufficient data, hence being a favored method for behavioral modeling and performance prediction applications [30].

A convolutional neural network (CNN), initially designed for images, has found relevance within analog design, especially in layout-related tasks as layout can be treated as an image. CNNs can be applied to predicting the parasitic effects due to layouts or to recognizing certain layout features associated with performance deterioration [31].

A graph neural network (GNN) is a neural network where a circuit can be modeled using nodes representing transistors and components and edges representing connectivity among those nodes. Therefore, GNN is well suited for tasks such as netlist-aware performance prediction and layout optimization [32].

6. Reinforcement Learning And Evolutionary/Optimization Methods

RL casts the problem of circuit sizing into a sequential decision problem wherein the learning algorithm figures out how to modify the design parameters by a process of trial-and-reward until the target specification is achieved without having to explicitly define a performance model initially. RL can thus be applied for automated transistor sizing and design-space exploration [33].

Genetic Algorithms and Particle Swarm Optimization both belong to population-based evolutionary algorithms wherein a collection of candidate designs is evolved using certain evolutionary principles such as mutation and selection in GA or swarm intelligence based evolution in PSO. They have been popularly used for multi-objective optimization problems wherein multiple objectives such as speed, power, and area have to be optimized concurrently [34].

Bayesian Optimization is very useful when each simulation is expensive, as is the case in analog circuits. In Bayesian Optimization, one constructs a probabilistic model of the design space and then intelligently samples new points based on the learned model [35].

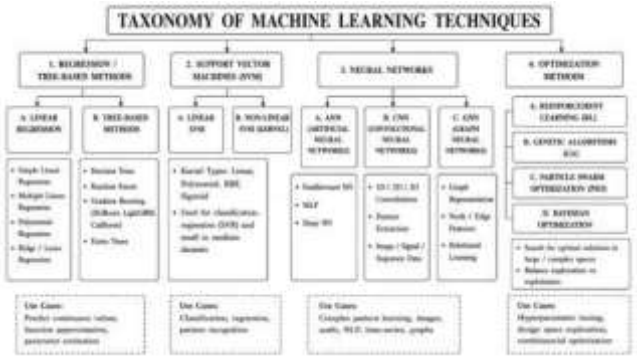


Fig. 7. Taxonomy of Machine Learning Techniques — hierarchical diagram grouping techniques under Regression/Tree-Based, SVM, Neural Networks (ANN/CNN/GNN), and Optimization Methods (RL/GA/PSO/Bayesian)

Each of these approaches has its own set of trade-offs with respect to accuracy, amount of training data needed, computational resources, and explainability. The relationship between these advantages and specific ADC design phases is explored in Section 7.

VI. AI APPLICATIONS ACROSS THE ADC DESIGN FLOW

This part of the work is a core of the review which deals with the application of machine learning approaches described in Section 6 at different stages of ADC development flow discussed in Section 3. Unlike case studies, the aim of the discussion here is to create an overview of the stages where the use of AI has proven itself valuable.



Fig. 8. Conventional vs. AI-Assisted ADC Design Flow — side-by-side flow comparison showing where ML intervenes at each stage

1. Behavioral Modeling

Historically, behavioral modeling was performed through simplified mathematical formulas that approximated the behavior of circuits in system level exploration. Various ML models such as regression and ANN techniques were employed in developing behavioral models through learning

© 2026 IJSRET

of actual transistor-level simulation results, accounting for nonlinearities that cannot be captured by ideal mathematical models [36].

2. Transistor Sizing

This topic is probably the most widely studied area in using AI for circuit design automation. The iterative nature of the transistor sizing process makes the RL method particularly efficient in this case since it treats the task as a sequential optimization problem with an agent learning how to set device dimensions according to target specification [37]. The other interesting methods in this regard are GA and PSO algorithms, which can be applied to perform multi-objective sizing of transistors, while surrogate models such as RF, XGBoost, and ANN techniques are used to replace expensive SPICE simulations.

3. Comparator Optimization

In SAR and pipeline ADCs, the comparator is often the most important element in terms of the circuit performance in view of the fact that the comparator offset, noise, and speed impact significantly on the ADC accuracy. Some studies employ ANN and XGBoost models in predicting comparator offset and delay as a function of transistor-level parameters for fast design space exploration without having to run transient simulations of each design candidate.

4. Optimizing Caparray

The linearity performance (INL/DNL) of SAR ADCs depends on the match and sizing of the capacitor-DAC array. ML models, including both regression and tree-based classifiers, have been applied to predict the effect of capacitor mismatch on linearity to enable sizing and layout optimization before physical realization [39]. GNN-based models have also been employed for this purpose, as they capture the relations between the elements better than the other ML models.

1. Calibration

Both digital and analog techniques have been proposed to compensate for non-idealities such as comparator offset, capacitor mismatch, and gain errors that arise due to process variation. ML-based calibration, including the application of ANNs and adaptive learning approaches, has been considered as a method for calibration different from the standard lookup table and foreground calibration techniques [40].

5. Layout Optimization

Layout optimization is a relatively new area for the application of AI in ADC design. CNN-based models have been used to predict the parasitic and degradation caused by layout design, making use of the inherent spatiality of layout data [41]. GNN-based models extend the approach further and apply the representation of the circuit netlist as a graph for layout optimization and placement recommendations, which is currently a developing area in the literature.

6. Pvt/Yield Predictions

Since the PVT corner and Monte Carlo simulations require significant computational efforts, ML models have been used to predict yield and worst-case performance of the new design point based on a subset of the simulation results [42]. The SVM and RF classifiers are most often used for pass/fail yield predictions, whereas regression-based models predict the value of performance metrics, reducing the simulation effort.

7. Design Space Exploration

On a higher level, several papers have combined multiple ML models—usually surrogate models with optimization techniques, such as Bayesian optimization and RL—to explore the entire design space of ADCs, taking into account multiple sub-blocks simultaneously [43].

Table III. AI Techniques Applied Across ADC Design Stages (Updated)

Design Stage	ML Technique(s)	Key Contribution	Ref.
Behavioral Modeling	Regression, ANN	ANN-based regression modeling for SAR ADC behavior, capturing nonlinearities missed by ideal equations	[13]
Transistor Sizing	RL, GA/PSO, XGBoost	AutoCkt: RL-based automated sizing achieving up to 38× faster runtime than prior methods	[21]
Comparator Optimization	ANN, XGBoost, Bayesian Opt	Fast offset/delay prediction from transistor-level parameters, reducing transient simulation need	[38]
Capacitor-Array Optimization	Regression, Tree-based, GNN	GNN-based parasitic-aware sizing linking capacitor mismatch to layout-level linearity	[35]

© 2026 IJSRET

Design Stage	ML Technique(s)	Key Contribution	Ref.
Calibration	ANN, adaptive learning	ML-based prior-knowledge-free calibration achieving 93.7-dB SFDR in split pipelined-SAR ADC	[49]
Layout Optimization	CNN, GNN	ParaGraph: GNN-based layout parasitics and device parameter prediction	[33]
PVT/Yield Prediction	SVM, RF, Regression	Gaussian Process-based yield optimization for analog/SRAM circuits	[43]
Design-Space Exploration	Surrogate + RL/Bayesian Opt	GLOVA: risk-sensitive RL for global/local variation-aware design	[42]

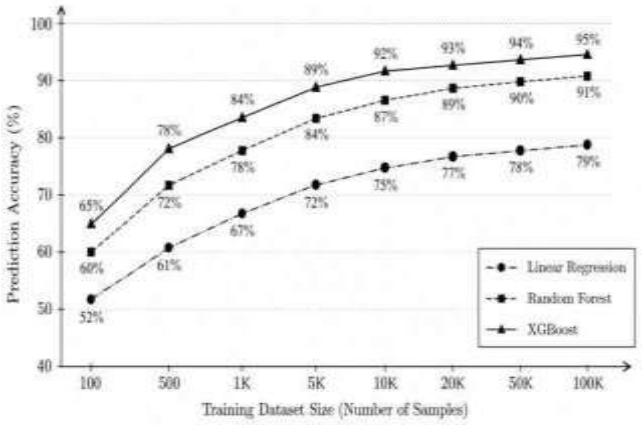
Table IV. AI Techniques Applied Across ADC Design Stages — columns: Design Stage, ML Technique(s) Used, Key Contribution, Reference

In all eight categories, there is a consistent trend: tree-based models and neural networks are mostly used for predictive modeling whereas RL/GA/PSO and Bayesian optimization are used extensively for active searching and sizing problems. While Graph Neural Networks (GNN) based models are still in the early stage of development, they are showing great promise in circuit layout and capacitor array optimization.

VII. COMPARATIVE ANALYSIS

Given the stage-by-stage investigation of AI approaches provided in section 7, it seems appropriate now to take a step back and compare the discussed AI approaches not only based on their capabilities but also with regard to their performance compared to each other and to traditional approaches.

Trade-off between accuracy and computational cost. Tree-based algorithms such as Random Forest and XGBoost are always characterized by a high trade-off between prediction accuracy and computational cost. Such algorithms generally need less training data to achieve a good level of accuracy compared to deep neural networks. That makes tree-based algorithms a good choice for analog design problems, as analog data is typically expensive and sparse [44]. On the other hand, ANNs are expected to be more accurate than tree-based approaches if enough training data is available.



7.1 Fig. 9. Prediction Accuracy Vs. Training Dataset Size For Ml Models

Trade-off between prediction and optimization. Another useful dichotomy that can be observed in the literature involves the classification of methods into those aimed only at prediction (regression, RF, XGBoost, ANN, SVM) and those involving optimization of some criterion (RL, GA, PSO, Bayesian optimization). While the former are usually faster to implement and verify, they still require either human intervention or a loop of an optimizer to make something with the results of prediction. The latter directly perform search for better design points, thus involving designers even less; however, this comes with increased computational expense per learning run and reduced explainability of the selected solution [45].

Speedup. In most of the analyzed papers, ML algorithms for performance prediction and circuit sizing typically deliver reductions in design cycle times by 30%-80%, depending on the design stage being studied, number of examples in the dataset, and the baseline approach used for comparison [46]. The greatest speedups are achieved on yield and PVT-prediction tasks, as corner simulations are always the most computationally intensive part of the standard flow.

© 2026 IJSRET

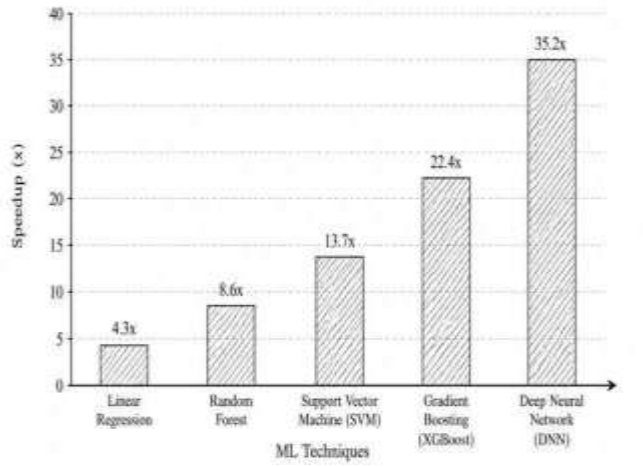


Fig. 10. Reported Speedup Comparison Across ML Techniques

Techniques' limitations that span multiple methods. While there have been many improvements, most of the methods mentioned above suffer from recurring limitations, such as the high reliance on the quality and size of the training dataset, inability to generalize to another node or topology from the one used for training, and decreased interpretability compared to simulation-based verification, which is important in analog design as it is sometimes necessary to explain the failure of a design point [47].

Table V. Comparative Analysis of ML Techniques — columns: Technique, Applicable Design Stage(s), Reported Accuracy/Speedup, Key Limitation

Technique	Applicable Stage(s)	Reported Accuracy/Speedup	Key Limitation
RF/XGBoost	Sizing, Comparator, PVT	High accuracy, ~30–60% faster	Needs moderate labeled data
ANN	Behavioral, Layout, Calibration	High accuracy w/ sufficient data	Large dataset dependency
CNN	Layout	Good spatial parasitic prediction	Needs image-like layout data
GNN	Capacitor-array, Layout	Topology-aware, best for connectivity	Still early-stage, complex training
RL	Sizing, Design-space exploration	Up to 80% faster convergence	High compute cost, sample-inefficient
GA/PSO	Multi-objective sizing	Good trade-off balancing	Slower convergence than RL
Bayesian Opt	Comparator, Yield	Minimizes simulation count	Costly per-iteration for high dimensions

In combination, these observations suggest that there is no single best ML approach; rather, the selection of an approach depends strongly on the availability of data, whether it is for prediction or searching, and on how much interpretability is required in the design process. These issues drive the open challenges described below.

2. Open Challenges & Research Gaps

Despite the promising results presented in Section 8, the present state-of-the-art in the application of ML techniques for ADC design still has some unsolved problems that restrict the applicability of the approach in practice.

Data availability and quality. The majority of models described in the literature are based on a considerable number of labeled examples – design parameters coupled with corresponding performance values. Gathering such data requires running thousands of simulations at the SPICE level, which is computationally expensive and contradicts partially the initial idea behind the use of machine learning in this field [48]. Standardized and publicly available datasets for ADC design are rare, and therefore almost all studies in the field use proprietary datasets, which cannot be reproduced.

Transferability across process nodes and architectures. It is quite challenging to generalize a machine learning model trained for a particular process node (for example, 180 nm) or a particular ADC topology to other nodes and/or other architectures. It significantly decreases the utility of existing trained models – a completely new one should be trained in case of switching to a new technology node [49].

Explanation of predictions and decisions. Analog designers usually have an explanation of why a particular circuit performs in a particular manner. Machine learning models, especially deep neural networks, act like black boxes and provide no information about the decision-making process,

© 2026 IJSRET

which makes the designers reluctant to trust the predictions [50].

Robustness. An ML model trained on a restricted part of the design space can demonstrate unreliable performance when faced with design points that lie outside the train distribution. This becomes important as ADCs are quite sensitive to PVT variations and mismatch.

Integration with existing EDA tools. Most ML-based ADC design solutions mentioned above are stand-alone research scripts or academic prototypes and not integrated into commercial toolchains (Cadence, Synopsys, etc.) used by industry designers. Overcoming this gap and building ML tools that can seamlessly integrate into existing design flows is still an important practical challenge [52].

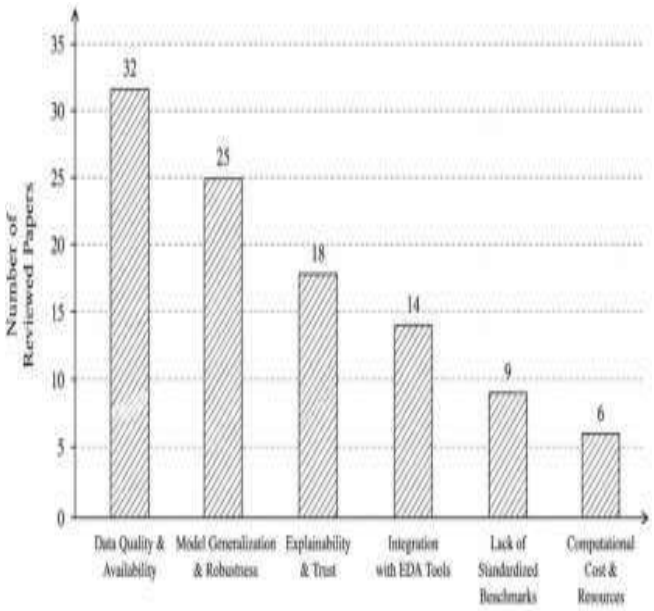
Cost of training models. Even though trained models are computationally cheap, the cost of their training may become a bottleneck and is often underestimated in literature, hence hindering a fair estimation of efficiency improvement of using ML [53].

Cross-stage co-optimization. Most works analyzed in this paper consider optimization of individual design stages (sizing, layout, calibration) as independent optimization tasks and use one ML framework per task. However, sub-blocks of an ADC affect each other: an optimal decision on sizing that helps to reduce the time required for a comparator will probably make worse the layout matching and, for example, a decision on layout that allows minimizing parasitic mismatches will not necessarily be optimal for calibration. This aspect of the problem was not addressed by very few works from those analyzed above; only one stage is optimized, while others are kept constant. It is necessary to develop ML frameworks that can optimize several design stages simultaneously and not separately, but this issue is still poorly addressed and represents one of the important research directions.

Finally, one should note the limitations of this literature review as well. The literature search was limited by English publications; hence, some works written in other languages and possibly related to ML-assisted ADC design were excluded from the review, especially those from East Asia and European countries. Moreover, this review is biased towards publicly available sources (IEEE Xplore, ACM, arXiv). Since a lot of ML-assisted ADC design is performed by semiconductor companies under NDA, there are many such works that are never made public; therefore, this review definitely underestimates the industry usage of ML techniques. Another limitation is connected to accuracy of reported results; since some of them are taken from works that have not been published yet and are not peer-reviewed, results presented in this paper should be considered as reported by authors and not verified independently.

Table VI. Open Challenges Summary

Challenge	Root Cause	Potential Mitigation
Dataset scarcity	Expensive SPICE-level data generation	Synthetic data (GenAI), shared datasets
Poor generalization	Node/topology-specific training	Physics-informed ML, transfer learning
Explainability	Black-box deep models	Physics-informed constraints, hybrid models
EDA integration	Standalone research prototypes	Tooling/plugin development



© 2026 IJSRET



Fig. 11. Distribution of Reviewed Papers Across Open Challenges

Fig. 1. Research Gap Map — visual grouping of challenges (Dataset, Generalization, Explainability, Robustness, EDA Integration, Training Cost)These gaps collectively suggest that while AI-assisted ADC design has demonstrated clear potential, it remains an evolving field rather than a fully mature, production-ready methodology. Section 10 explores the directions researchers are pursuing to address these limitations.

VIII. FUTURE DIRECTIONS

Building on the challenges identified in Section 9, several emerging research directions show promise for addressing the current limitations of AI-assisted ADC design.

1. Generative Ai

Generative architectures like GANs and diffusion models are currently under investigation to generate novel topologies or design points, as opposed to inference of the outcomes for a particular input [54]. This may help circumvent the problem of small-sized datasets since realistic data can be generated using generation techniques, and it may help identify new sizing solutions that would have otherwise been overlooked.Generative AI, with its impressive abilities to generate novel circuit topologies and synthetic training data, also carries some risks that have not been adequately addressed by the current state of literature. First, the question of manufacturability is raised: while a generated topology may be correct in terms of functionality in the simulation environment, it might violate fabrication constraints, matching requirements or specific design rules of the particular foundry that the model was not trained on. Intellectual property concerns arise as well: while a model trained on proprietary circuit designs may generate an output circuit very similar to the original one, thereby raising questions of ownership and originality not thoroughly addressed yet.

2. Gnn-Based Layout Optimization

Since GNNs were introduced in section 7.6, they are quite capable of representing circuit topology. Moreover, efforts have been made to utilize GNNs along with the netlist and physical layout constraints in order to make the process of automatic layout generation more topology-aware and automate the process which is currently done manually for comparators and capacitors [55].

3. Reinforcement Learning

Although RL has already proved itself successful for transistor sizing (Section 7.2), future research is going to focus on approaches based on multi-agent or hierarchical RL which would enable optimization of several ADC components together at once, instead of considering them individually – just like it should be done in actual ADC design [56].

4. Physics-Informed Machine Learning

One potential solution for the problems of explainability and generalization discussed in Section 9 is physics-aware machine learning, in which the models are required to satisfy the laws of physics and the equations of the circuits during the training process instead of learning directly from the patterns in the data [57].

5. Digital Twins

Conceptions of digital twins, where there exists an always-updated, simulation-based virtual version of the circuit through all stages of its design and even after fabrication, have started appearing in the literature on analog design [58]. The idea would allow for real-time monitoring and adaptation to any aging effects in case of ADCs.

6. Human-Ai Collaborative Design

As opposed to achieving autonomy in their design process, some scholars call for interactive designs, whereby AI will be responsible for the tedious work of exploration and predictions while designers continue controlling architectural choices and final design judgments [59]. This interactive approach provides a solution to the problems identified in Section 9.

© 2026 IJSRET



Fig. 12. Future Directions Roadmap for AI-Assisted ADC Design — timeline/roadmap visual grouping near-term (GenAI, RL) vs. longer-term (digital twins, physics-informed ML) directions

All these directions show that research in the domain is increasingly focusing on developing machine learning techniques that not only achieve greater accuracy but also greater explainability, generalizability, and integration into real-life design processes.

IX. CONCLUSION

In this review, an attempt was made to study the role that machine learning plays in improving the ADC design process. Instead of focusing on individual algorithmic implementations, the review has followed

the steps of the design flow from behavioral modeling to layout and verification. It becomes evident that in all areas AI has already made an appreciable contribution towards lowering the burden of simulations and iterations in transistor sizing, comparators and capacitor-array optimizations, as well as predicting PVT/yield by implementing such techniques as XGBoost, ANN, RL, and Bayesian optimization [60].

Nevertheless, it is important to emphasize some aspects of the current problematics which prevent ADC design from being significantly advanced using AI techniques. The lack of dataset, inability to generalize the obtained results across process nodes, poor explainability of the results, and difficulty in integrating algorithms with the existing EDA tools become the main obstacles for the widespread implementation of AI-assisted design. All of those issues are far from being a minor problem, as it explains the absence of a wide industry adoption of this technology despite the long-term research period.

There are several directions which could indicate the future growth of this field including generative AI, GNN-based layout optimizations, physics-informed learning, and human-in-the-loop framework, which could provide a basis for further development of the field and make it much more useful for ADC design industry. For students and researchers who plan to get involved in this sphere, there is still a lot to do.

Ultimately, there is no imminent prospect of replacing the analog designer's judgment with artificial intelligence anytime soon, which is not a goal in itself either. The only benefit provided by the AI tools is to eliminate the repetitive and computationally costly part of the exploration phase allowing the engineer to focus on architectural and creative decisions. With further increase in dataset sizes, interpretability of the models, and development of the toolchains, AI assisted design is set to go from being a novel research topic to a standard part of the analog and mixed-signal design workflow.

In conclusion, let us provide some practical guidance depending on one's role: for students just beginning in the domain, a combination of decision tree based techniques such as XGBoost in application to the comparator and capacitor array synthesis problem seems like the most straightforward route given smaller dataset requirements and ease of implementation when compared to the deep learning techniques. For the researchers looking for a novel area to explore, cross-stage co-optimization is clearly the missing piece in the puzzle providing lots of opportunities for new work given that the majority of existing works consider sizing, layout, and calibration as separate steps.

References

1. C.-H. Chan et al., "ADC trends and impact on SAR ADC architecture and analysis," in Proc. IEEE Int. Conf., 2015. [Online]. Available: https://ieeexplore.ieee.org/document/7338380/
2. R. Wang, “A review on the key optimization techniques of SAR ADC design,” in Proc. 2021 Int. Conf. Electronic Information Engineering and Computer Science (EIECS), 2021, pp. 951–957, doi: 10.1109/EIECS53707.2021.9587905.
3. Z. Tang et al., "Low-Power SAR ADC Design: Overview and Survey of State-of-the-Art Techniques," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 69, no. 6, Jun. 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9761973/
4. Y. Hao, K. Li, M. Gandara, S. Li, and B. Liu, "A Global-Local Optimization Approach for Asynchronous SAR ADC Design," arXiv preprint, 2024. [Online]. Available: https://arxiv.org/pdf/2507.19541
5. J. Monica and P. Kannan, “Challenging factors of pipelined ADC design at higher speed and resolution,”

© 2026 IJSRET

in Proc. 2015 Global Conf. Communication Technologies (GCCT), 2015, pp. 23–28.
6. J. Gao, G. Li, L. Huang, and Q. Li, “An amplifier-free pipeline-SAR ADC architecture with enhanced speed and energy efficiency,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 63, no. 4, pp. 341–345, Apr. 2016, doi: 10.1109/TCSII.2015.2503705.
7. R. Schreier and G. C. Temes (as cited in), "The Delta-Sigma Modulator," IEEE Solid-State Circuits Mag., Jun. 2016. [Online]. Available: https://www.seas.ucla.edu/brweb/papers/Journals/BRSpring16DeltaSigma.pdf
8. R. W. Stewart, “An overview of sigma delta ADCs and DAC devices,” in Proc. IEE Colloquium on Oversampling and Sigma-Delta Strategies for DSP, London, U.K., 1995, pp. 1/1–1/9, doi: 10.1049/ic:19951371.
9. Y. Zhang and Y. Jiang, “Noise shaping sigma delta modulation techniques and models simulation,” in Proc. 2006 8th Int. Conf. Solid-State and Integrated Circuit Technology (ICSICT), 2006, pp. 1471–1473, doi: 10.1109/ICSICT.2006.306240.
10. L. Chacko and G. T. Varghese, “Comparator design for low power high speed flash ADC—A review,” in Proc. 2019 3rd Int. Conf. Computing Methodologies and Communication (ICCMC), 2019, pp. 869–872, doi: 10.1109/ICCMC.2019.8819762.
11. S. Chitti, S. R. Nandikonda, S. Kancha, S. Gollena, et al., “Design and implementation of low power, high speed flash ADC using double tail comparator,” in Proc. 2024 Int. Conf. Inventive Computation Technologies (ICICT), 2024, doi: 10.1109/ICICT60155.2024.10544526.
12.]K. S. Kumar, K. L. Krishna, K. S. Raghavendra, and K. Harish, “A high speed flash analog to digital converter,” in Proc. 2018 2nd Int. Conf. I-SMAC (IoT in Social, Mobile, Analytics and Cloud), 2018, pp. 283–288, doi: 10.1109/I-SMAC.2018.8653782.
13. J. W. Nam and Y. K. Lee, “Machine-learning based analog and mixed-signal circuit design and optimization,” in Proc. 35th Int. Conf. Information Networking (ICOIN), 2021, pp. 874–876, doi: 10.1109/ICOIN50884.2021.9333856.
14. A. Abuelnasr, M. Amer, A. Ragab, B. Gosselin, and Y. Savaria, “Causal information prediction for analog circuit design using variable selection methods based on machine learning,” in Proc. 2021 IEEE Int. Symp. Circuits and Systems (ISCAS), 2021, pp. 1–5, doi: 10.1109/ISCAS51556.2021.9401786.
15. T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 785–794.
16. G. Huang et al., "Machine Learning for Electronic Design Automation: A Survey," ACM Trans. Design Autom. Electron. Syst., vol. 26, no. 5, pp. 1–46, 2021.
17. S. Devi, G. Tilwankar, and R. Zele, “Automated design of analog circuits using machine learning techniques,” in Proc. 25th Int. Symp. VLSI Design and Test (VDAT), 2021, pp. 1–6, doi: 10.1109/VDAT53777.2021.9601131.
18. "RLC Circuit Forecast in Analog IC Packaging and Testing by Machine Learning Techniques," Micromachines, vol. 13, no. 8, 2022. — Includes GNN-based parasitic prediction from netlist graphs. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9413446/
19. Z. Zhao and L. Zhang, “Deep reinforcement learning for analog circuit sizing,” in Proc. 2020 IEEE Int. Symp. Circuits and Systems (ISCAS), 2020, pp. 1–5, doi: 10.1109/ISCAS45731.2020.9181149.
20. Y. Uhlmann, M. Essich, L. Bramlage, J. Scheible, and C. Curio, “Deep reinforcement learning for analog circuit sizing with an electrical design space and sparse rewards,” in Proc. 2022 ACM/IEEE Workshop on Machine Learning for CAD (MLCAD), 2022, pp. 21–26, doi: 10.1109/MLCAD55463.2022.9900083.
21. K. Settaluri et al., "Automated Design of Analog Circuits Using Reinforcement Learning" (AutoCkt), IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9576505/
22. E. Papageorgiou, C. Visan, A. Buzo, G. Pelz, H. Cucu, and T. Noulis, “Comparative evaluation of reinforcement learning and evolutionary algorithms for effective analog circuit sizing,” in Proc. 2025 32nd IEEE Int. Conf. Electronics, Circuits and Systems (ICECS), 2025, pp. 1–4, doi: 10.1109/ICECS66544.2025.11270536.
23. W. Cao, M. Benosman, X. Zhang, and R. Ma, "Domain Knowledge-Based Automated Analog Circuit Design with Deep Reinforcement Learning," arXiv preprint, 2022. [Online]. Available: https://arxiv.org/pdf/2202.13185
24. A. Sasikumar and R. Muthaiah, “Operational amplifier circuit sizing based on NSGA-II and particle swarm optimization,” in Proc. 2017 Int. Conf. Networks and Advances in Computational Technologies (NetACT), 2017, pp. 64–68, doi: 10.1109/NETACT.2017.8076742.
25. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Trans. Evol. Comput., vol. 6, no. 2, pp. 182–197, 2002.
26. R. A. Vural, T. Yildirim, T. Kadioglu, and A. Basargan, "Performance evaluation of evolutionary algorithms for

© 2026 IJSRET

optimal filter design," IEEE Trans. Evol. Comput., doi: 10.1109/TEVC.2011.2112664.
27. G. Rudolph, "Convergence analysis of canonical genetic algorithms," IEEE Trans. Neural Netw., vol. 5, no. 1, pp. 96–101, 1994.
28. B. He, S. Zhang, F. Yang, C. Yan, D. Zhou, and X. Zeng, "An Efficient Bayesian Optimization Approach for Automated Optimization of Analog Circuits," IEEE Trans. Circuits Syst. I, Reg. Papers, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8116661/
29. J. Huang, S. Zhang, C. Tao, F. Yang, C. Yan, D. Zhou, and X. Zeng, "Bayesian optimization approach for analog circuit design using multi-task Gaussian process," in Proc. 2021 IEEE Int. Symp. Circuits and Systems (ISCAS), 2021, pp. 1–5, doi: 10.1109/ISCAS51556.2021.9401205. Available: https://ieeexplore.ieee.org/document/9401205
30. Y. Wang, C. Yan, D. Zhou, and X. Zeng, "High-dimensional Bayesian optimization for automated analog circuit design via add-graph structure," in Proc. 2021 IEEE 14th Int. Conf. ASIC (ASICON), 2021, pp. 423–426, doi: 10.1109/ASICON52560.2021.9620425. Available: https://ieeexplore.ieee.org/document/9620425/
31. S. Zhang, W. Lyu, F. Yang, C. Yan, D. Zhou, and X. Zeng, "An Efficient Multi-fidelity Bayesian Optimization Approach for Analog Circuit Synthesis," in Proc. ACM/IEEE DAC, 2019.
32. B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the Human Out of the Loop: A Review of Bayesian Optimization," Proc. IEEE, vol. 104, no. 1, pp. 148–175, 2016.
33. H. Ren, G. F. Kokai, W. J. Turner, and T.-S. Ku, "ParaGraph: Layout Parasitics and Device Parameter Prediction using Graph Neural Networks," in Proc. IEEE/ACM Design Automation Conf. (DAC), 2020, pp. 1–6.
34. H. Ren, G. F. Kokai, W. J. Turner, and T.-S. Ku, "GCN-RL Circuit Designer: Transferable Transistor Sizing with Graph Neural Networks and Reinforcement Learning," in Proc. IEEE/ACM DAC, 2020.
35. M. Liu, W. J. Turner, G. F. Kokai, B. Khailany, D. Z. Pan, and H. Ren, "Parasitic-Aware Analog Circuit Sizing with Graph Neural Networks and Bayesian Optimization," in Proc. IEEE DATE Conf., 2021, pp. 1372–1377.
36. L. Liu, F. Yang, L. Shang, and X. Zeng, "GNN-Cap: Chip-Scale Interconnect Capacitance Extraction using Graph Neural Network," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 43, no. 4, pp. 1206–1217, 2024.
37. K. Kunal et al., "GANA: Graph Convolutional Network Based Automated Netlist Annotation for Analog Circuits," in Proc. IEEE DATE Conf., 2020, pp. 55–60.
38. Y. Li, Y. Lin, M. Madhusudan, A. Sharma, W. Xu, S. S. Sapatnekar, R. Harjani, and J. Hu, "A customized graph neural network model for guiding analog IC placement," in Proc. 2020 IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD), 2020, pp. 1–9, doi: 10.1145/3400302.3415624.
39. J. Li, Y. Zeng, H. Zhi, et al., "Knowledge Transfer Framework for PVT Robustness in Analog Integrated Circuits," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 71, no. 5, pp. 2017–2030, May 2024.
40. X. Wang, C. Yan, Y. Ma, B. Yu, F. Yang, D. Zhou, and X. Zeng, "Analog Circuit Yield Optimization via Freeze-Thaw Bayesian Optimization Technique," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 41, no. 11, pp. 4887–4900, 2022.
41. A. F. Budak, M. Gandara, W. Shi, D. Z. Pan, N. Sun, and B. Liu, "An Efficient Analog Circuit Sizing Method Based on Machine Learning Assisted Global Optimization," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 41, no. 5, pp. 1209–1221, May 2022.
42. D. Kim, J. Park, C. Shin, J. Jung, K. Shin, S. Baek, S. Heo, W. Kim, I. Jeong, J. Cho, and J. Park, "GLOVA: Global and local variation-aware analog circuit design with risk-sensitive reinforcement learning," arXiv preprint arXiv:2505.11208, 2025. Available: https://arxiv.org/html/2505.11208v1
43. M. Wang, W. Lv, F. Yang, C. Yan, W. Cai, D. Zhou, and X. Zeng, "Efficient Yield Optimization for Analog and SRAM Circuits via Gaussian Process Regression and Adaptive Yield Estimation," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 37, no. 10, pp. 1929–1942, 2017.
44. M. Golzan, H. Nasiri, T. M. N. Ngatched, K. Popuri, and L. Zhang, "Advanced performance estimation of analog layouts using convolutional neural networks," in Proc. 2024 IEEE 20th Int. Conf. Intelligent Computer Communication and Processing (ICCP), 2024, pp. 1–5, doi: 10.1109/ICCP63557.2024.10793041. Available: https://ieeexplore.ieee.org/document/10793041/
45. [45] S. Jeong, M. Kim, and B. Kim, "A CNN-Based Technique to Assist Layout-to-Generator Conversion for Analog Circuits," arXiv preprint, 2025. [Online]. Available: https://arxiv.org/pdf/2512.00070
46. H.-C. Shao, C.-Y. Peng, J.-R. Wu, C.-W. Lin, S.-Y. Fang, P.-Y. Tsai, and Y.-H. Liu, "From IC Layout to Die Photo: A CNN-Based Data-Driven Approach," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., 2020.

© 2026 IJSRET

47. S. Bhanushali, D. Maiti, P. Bikkina, E. Mikkola, and A. Sanyal, “Circuits-informed machine learning technique for blind open-loop digital calibration of SAR ADC,” arXiv preprint arXiv:2412.14051, 2024. Available: https://arxiv.org/pdf/2412.14051
48. S. Bansal et al., "Neural-Network Based Self-Initializing Algorithm for Multi-Parameter Optimization of High-Speed ADCs," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 68, no. 1, pp. 106–110, 2021.
49. T. Zhang et al., "Machine Learning Based Prior-Knowledge-Free Calibration for Split Pipelined-SAR ADCs with Open-Loop Amplifiers Achieving 93.7-dB SFDR," in Proc. IEEE ESSCIRC, 2019, pp. 189–192.
50. A. J. Ginés, E. Peralías, and A. Rueda, "Background Digital Calibration of Comparator Offsets in Pipeline ADCs," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 23, no. 7, pp. 1345–1349, 2015.
51. D. Noori Zadeh and M. B. Elamien, "Generative AI for Analog Integrated Circuit Design: Methodologies and Applications," IEEE Access, vol. 13, pp. 58043next f–58059, 2025, doi: 10.1109/ACCESS.2025.3553743.
52. B. Xu, Y. Lin, X. Tang, S. Li, L. Shen, N. Sun, and D. Z. Pan, "WellGAN: Generative-Adversarial-Network-Guided Well Generation for Analog/Mixed-Signal Circuit Layout," in Proc. IEEE/ACM DAC, 2019.
53. [53]T. Guo, D. R. Herber, and J. T. Allison, "Circuit Synthesis Using Generative Adversarial Networks (GANs)," in Proc. AIAA SciTech Forum, 2019.
54. G. Datta, S. S. Sharif, and Y. M. Banad, “Toward digital twins in 3D IC packaging: A critical review of physics, data, and hybrid architectures,” Electronics, vol. 15, no. 8, Art. no. 1740, 2026, doi: 10.3390/electronics15081740.Available: https://doi.org/10.3390/electronics15081740
55. M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations," J. Comput. Phys., vol. 378, pp. 686–707, 2019.
56. F. Tao, H. Zhang, A. Liu, and A. Y. C. Nee, "Digital Twin in Industry: State-of-the-Art," IEEE Trans. Ind. Informat., vol. 15, no. 4, pp. 2405–2415, 2019.
57. A. Rasheed, O. San, and T. Kvamsdal, "Digital Twin: Values, Challenges and Enablers from a Modeling Perspective," IEEE Access, vol. 8, pp. 21980–22012, 2020.
58. B. Liu, H. Zhang, X. Gao, Z. Kong, X. Tang, Y. Lin, R. Wang, and R. Huang, "LayoutCopilot: An LLM-Powered Multi-Agent Collaborative Framework for Interactive Analog Layout Design," arXiv preprint, 2024. [Online]. Available: https://arxiv.org/pdf/2406.18873
59. J. Pan, G. Zhou, C.-C. Chang, I. Jacobson, J. Hu, and Y. Chen, “A survey of research in large language models for electronic design automation,” ACM Trans. Design Autom. Electron. Syst., 2025.
60. Z. He, Y. Pu, H. Wu, T. Qiu, and B. Yu, “Large language models for EDA: Future or mirage?,” ACM Trans. Design Autom. Electron. Syst., vol. 30, no. 6, Art. no. 90, pp. 1–53, 2025, doi: 10.1145/3736167.

© 2026 IJSRET