International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

A Review of Hybrid AI-Driven Methods for Circuit-Level Design Automation and Performance

Umesh Balkishan Phiske[1], Dr. Goplakrishna D. Dalvi[2]

[1]*Research Scholar, Electronics and Telecommunication Department, ALARD School of Interdisciplinary Research, ALARD University, Pune, India*

[2]*HOD, Electronics and Telecommunication Department, ALARD School of Interdisciplinary Research, ALARD University, Pune, India*

***Abstract*— The rapid advancements in integrated circuit (IC) design have necessitated innovative approaches to optimize performance while meeting multi-objective constraints, such as power consumption, performance, and area (PPA). Traditional methods, relying on rule-based heuristics and simulation-heavy approaches, have become less efficient with the increasing complexity of designs. Hybrid artificial intelligence (AI) systems have been created to deal with these problems. These systems combine machine learning (ML) techniques with traditional circuit optimisation methods. It looks at how supervised learning (e.g., Support Vector Machines, Random Forests), uncontrolled learning (e.g., K-means dustering, Autoencoders), and reinforcement learning (RL) can be used together in circuit-level optimisation. These hybrid systems enhance the design process by automating tasks like transistor sizing, layout planning, and fault detection. The paper provides a detailed analysis of various ML techniques, their applications in circuit optimization, and the performance improvements they offer over traditional methods. Notable advancements include the use of Graph Convolutional Networks (GCN) in conjunction with RL for efficient circuit design, and Bayesian optimization combined with surrogate models for faster convergence. Additionally, the paper presents a bibliometric analysis to track the evolution of hybrid AI research in circuit design, highlighting key publications, co-authorship networks, and emerging research trends. This review shows how important mixed AI systems are becoming in modern circuit design by giving a full picture of the most up-to-date methods. This makes way for better, more flexible, and scalable solutions.**

***Keywords*—Hybrid AI systems, Machine Learning, Circuit Optimization, Reinforcement Learning, Graph Convolutional Networks.**

I. INTRODUCTION

Aspects of integrated circuits' (ICs) design and improvement have become more difficult because of the multi-objective challenges of modern semiconductor devices and Moore's Law's scaling restrictions.

Due to the expanding set of non-linear constraints and parasitic effects, today's circuit designers have to optimise simultaneously across power, performance and area (PPA) in their designs [1]. Although previous methods have been beneficial, traditional electronic design automation (EDA) methods have largely been based on rule-based heuristics and simulations, which are often slow and inflexible, and are ultimately inadequate for today's VLSI challenges. To alleviate these challenges, researchers have developed hybrid AI systems that integrate machine learning and traditional methods to enhance automation, adaptive learning, and optimization speed [2]. The core innovation of these hybrid AI systems lies in the algorithmic integration of ML models into the circuit design pipeline [3]. Numerous people employ supervised learning techniques such as support vector machines (SVMs), random forests, and gradient boosting to predict yield, classify performance and carry out early fault diagnosis [4]. K-means clustering, PCA, and Autoencoders are all examples of unsupervised methods that help identify hidden behaviours in layout data and aid in the exploration of the design space. Recently, strategies that integrate deep learning, coupled with reinforcement learning, have proven highly effective for the automation of placement, floor planning, and routing, all of which have been traditionally viewed as tedious and reliant on heuristics [5]. One notable example is the GCN-RL Circuit Designer, which leverages Graph Convolutional Networks (GCNs) with RL agents to optimize transistor sizing across various circuit topologies [6]. This hybrid algorithm demonstrates the capability to generalise across designs and outperforms traditional Bayesian optimisation in both convergence speed and adaptability. Likewise, Hu et al. presented a Bayesian Optimisation approach supplemented by deep surrogate models for the co-optimization of analogue circuits, showcasing the ability of surrogate learning models to accelerate the convergence of searches [7].

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

Recent studies show that combining ML algorithms with heuristic search methods such as PSO and GA improves efficiency and reduces computing costs simultaneously[8]. For instance, using Artificial Neural Networks (ANNs) as surrogate models within a GA framework allows for efficient prediction and optimization of analog performance metrics [9]. Numerous systematic reviews and survey papers have documented the increasing relevance of these hybrid approaches and have offered extensive taxonomies of the applications of ML in EDA, pointing out that every step from RT-level (RTL) synthesis to layout verification can be optimised with the help of specialized ML algorithms for the specific domain [10].

In parallel, researchers from the University of Minho analyzed optimization strategies that combine metaheuristics with ML models, presenting hybrid systems that exploit both local and global search capabilities for more robust optimization outcomes. Graph Neural Networks (GNNs), in particular, have been explored in Journal of Machine Learning Research (JMLR) as powerful tools for solving complex circuit graph representations through message passing and structural learning [11]. New GNN- based frameworks, such as those proposed combine attention mechanisms with graph learning to support layout generation in physical design flows. These developments underscore the pressing need for a structured review of the algorithms underpinning hybrid AI systems in circuit optimization [12]. This paper presents a comprehensive analysis of ML algorithm types, their integration strategies, comparative performance evaluations, and challenges [13]. It also incorporates a bibliometric analysis to identify key publication trends, co-authorship networks, and emerging research clusters offering a holistic view of this fast-evolving interdisciplinary field.

1.1 Overview of Circuit-Level Optimization Techniques

Optimising at the circuit level is the most challenging phase of the design of integrated circuits (IC), working on the optimisation of the transistors, the bias settings, and the topological arrangement to achieve the desired condition on available design specification requirements for power, delay, gain, and a minimal area. SPICE simulations have been the mainstay for this process along with tuning provided guidance by any of the strategies, either analytical or heuristic. While the process will always offer accurate results to a point, the level of design complexity, the increasingly miniaturised geometries of the design transistors, and advancing to the next level process node will all become increasingly more challenging to achieve higher and tighter design specifications [14].

While using synther technology, we experienced a significant amount of time spent scaling complex circuits involving circuit estimation with analogue design flows based on the designer using iterative simulations. We noticed a considerable time sink and a drop in efficiency. To save this time, we have developed a more efficient system using a combination of machine learning and traditional optimization [15]. Predictive modelling helps in estimating circuit behaviour and, therefore, eliminates the need for a full run simulation. One such study showed that by using a neural network and a random forest regressor to build surrogate models, accuracy was maintained and simulation calls were reduced by 80% or more [16].

Another positive opportunity involves merging Bayesian optimisation alongside neural surrogate models. This fusion permits advanced search capabilities within intricate design domains, gaining the ability to converge rapidly, even with constrained simulation budgets [17]. For higher-dimensional problems, variable selection and subspace optimization techniques such as dropout-enhanced Bayesian frameworks have proven effective in analog circuit sizing tasks [18]. Learning that is based on graphs has also been incorporated into the optimisation of circuits. It is possible to perform efficient optimisation on the layout if GNNs can properly estimate the optimal transistor sizes, and then GNNs can be combined with reinforcement learning to optimise layout transistor sizing [19]. One research implemented a machine learning GCN-RL based framework to optimise multiple circuit parameters across diverse technology nodes and design families, outpacing standard techniques based on black economic Optimisers [20]. Another reinforcement learning framework, AutoCkt, modeled the circuit design process as a Markov Decision Process, resulting in a 40× speed improvement compared to genetic algorithms [21].

We have applied layout-aware surrogate modelling using graph-based embeddings in parasitic-sensitive analogue design, which greatly advances the state of the art in performance prediction under layout constraints [22]. In another approach, surrogate-assisted multi-objective optimization was used to directly evolve Pareto-optimal analog circuit solutions with high diversity and convergence speed [23]. It has been made possible to combine the worldwide exploration power of neural networks with the sample accuracy of Bayesian models in evolutionary Bayesian optimization frameworks. These combination methods give reliable results for analogue size even when there are a lot of design limitations [24].

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

A recent innovation integrated large language models with Bayesian optimization to assist in analog design reasoning and transfer learning, greatly improving the optimization of unseen circuit topologies [25]. At last, there is an all-encompassing assessment that groups these methods into supervision, reinforcement, graphs, and hybrid learning models, demonstrating distinct patterns in surrogate- and adaptive-design techniques in developing analog front-ends [26]. Due to these improvements, there is a movement away from extensive simulation-based workflows, and towards effective, intelligent, and most importantly, hybrid AI systems that optimise for adaptability as well as cost when it comes to computation [27]. This part of the review talks about the classification of the different machine learning methods that make up the core of the mixed optimisation systems.

II. Research Method

Combining improvements of circuits at the hardware level as well as advancements in ML is a new emerging field of research. To obtain a better understanding of how this field operates, the researchers conducted a combination of a detailed scoping review and a bibliometric analysis of the field. Targeting publications from 2015-2024, the researchers performed a comprehensive and systematic search of the top engineering, computer science and AI literature databases. They then meticulously collated the year of publication, frequency and distribution of citations, authors, and country of origin. Publications that specifically focus on the teachings from ML in optimising circuits, as well as the sub-areas of ML such as base learning, reinforcement learning and graph structures, were the focus of this research [28]. A mix of quantitative techniques such as citation examinations and co-authorship analysis were used to determine the value of the investigations and impact. To understand the evolution of the field, temporal changes were mapped and citation analysis was used to assess impact on the field. Co-authorship networks helped to understand the collaborations and determined principal researchers. The researchers used the bibliometric data to determine the significant works and novel patterns along with sites of investigation. This comprehensive method provides an understanding of the extent to which mixed AI systems can be applied to optimisation at the level of circuits [29]. It also gives useful information about where the field might be going in the future.

Table 1:
Year-Wise Publication

Year	Number of Publications
2017	1
2019	4
2020	11
2021	19
2022	35
2023	449
2024	922
2025	1059

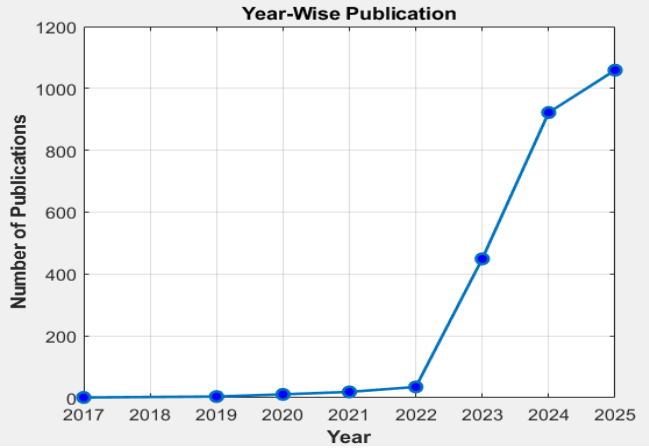

Figure 1: Year-Wise Publication

The annual settlement of publications regarding the valuation of hybrid AI systems in the optimisation of circuit level offers analytics of changes of AI systems in circuit level optimisation. Back in 2017, if one were to consider the correlation between publications and researchers to the interdisciplinary nurtured combination of AI systems and circuit optimisation, one single publication produced suggests the emergence of the combination to have developed only to the incubation levels of opportunity [30]. In 2019, 4 publications were released, and 11 were published the following year, which was a good sign of steady growth. Then, in 2021, 19 publications were released, followed by 35 publications in 2022. That was a sign of strong interest in that field. The most notable jump occurred in 2023, which had 449 publications published [31].

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

This continued for both 2024 and 2025, with 922 and 1059 publications, respectively, and signified the accelerating implementation of hybrid AI systems in the circuit-level optimisation domain. This set of data shows that the field is growing up, with more people working on it and coming up with new ways to make AI-aided circuit design methods more effective and efficient [32].

TABLE 2:
LEADING JOURNALS PUBLISHING RESEARCH ON HYBRID AI AND CIRCUIT-LEVEL OPTIMIZATION

Sr.No	Source	Documents	Citations	Mean Citations
1	ACS Nano	23	1572	68.3
2	Sensors	330	5986	18.1
3	Advanced Materials	32	903	28.2
4	Nature Communications	40	1590	39.8
5	Chemical Reviews	22	1707	77.6
6	Scientific Reports	324	1539	4.8
7	Nature	8	1668	208.5
8	Nano-Micro Letters	18	719	39.9
9	Advanced Science	13	210	16.2
10	Computers in Biology and Medicine	51	714	14.0
11	Materials Horizons	8	22	2.8
12	Science Advances	13	325	25.0

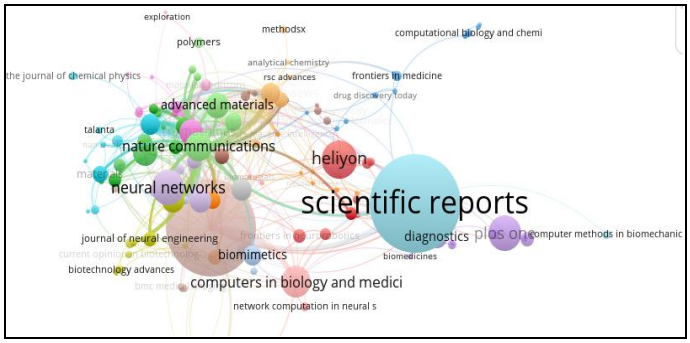

Figure 2: Source Co-Citation Network Visualization of Journals

The following table lists some of the most significant contributors (in terms of quantity and influence) to the literature that pertains to the intersections of hybrid AI systems and circuit-level optimisation and integrated circuit design research. Additionally, the data provide a previously mentioned perspective of the influence each source seems to hold academically, along with some insightful metrics. ACS Nano and Chemical Reviews have published the most documents (23 and 22 documents published, respectively) and have also been mean cited the most per document (68.3 and 77.6 mean citations, respectively). Nature is also noteworthy for being an influential source as it has 8 documents published with a mean of 208.5 citations. The Sensors and Scientific Reports, which published 330 and 324 documents respectively, are also influential; however, their mean citations are low (at least in recent times). Their documents appear to have a wide reach and presumably a less concentrated impact to achieve such an outcome. The Co-Citation Network Visualisation of Journals (as illustrated) enhances the understanding of the interconnections these journals have in terms of their relative research contribution. The figure of the journals showcases co-citation (i.e., collaboration and/or citation), which helps determine key contributors and/or academic research in hybrid AI and circuit optimisation [33].

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

TABLE 3:
AUTHOR-WISE DISTRIBUTION

Sr.No.	Author	Documents	Citations	Mean Citations per Document
1	Li, Guoqi	9	580	64.4
2	Chen, Xiaodong	4	322	80.5
3	Tian, Yonghong	6	508	84.7
4	Xu, Bo	7	213	30.4
5	Yao, Man	3	202	67.3
6	He, Ke	3	293	97.7
7	Su, Jiangtao	3	293	97.7
8	Deng, Lei	4	340	85.0
9	Gao, Wei	3	551	183.7
10	Lee, Chengkuo	10	505	50.5
11	Cauwenberghs, Gert	3	640	213.3
12	Kubendran, Rajkumar	2	635	317.5
13	Fang, Wei	2	263	131.5
14	Chaudhary, Vishal	5	75	15.0
15	Joshi, Siddharth	4	689	172.3

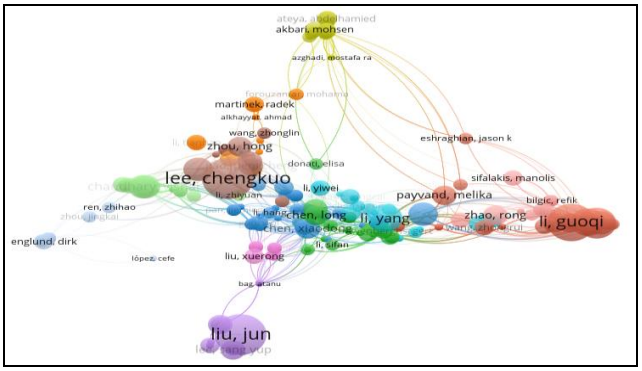

Figure 3: Co-Authorship Network Visualization of Leading Authors

Table 3 illustrates the main contributors to the publications related to hybrid AI systems in circuit level optimisation in the study (by author) that have garnered citations from other publications to help track their influence on the field. It shows how many citations each author got, how many documents they released, and how many citations each document got on average. Based on contributions to the field, Li, Guoqi is the most prolific in the publication of documents with 9 publications with an overwhelming 580 citations, of which the average is 64.4 per publication, and 2nd of 6 publications overall, Tian, Yonghong with 508 citations and an average of 84.7. Gao, Wei made an even lower number of publications, 3, but an even higher 551 citations with an average of 183.7 citations, which speaks to the influence of the work he is doing. Other authors, even those with a lower total of 467 citations, Cauwenberghs, Gert, and Kubendran, Rajkumar are doing the publication of documents with very high average citations of 213.3 and 317.5 respectively. The Co-Authorship Network Visualisation of Leading Authors (figure) along with this information gives credence to the relationships and citation activity for other publications of these authors to illustrate the advancement of the work in AI hybrid for circuit optimisation [34].

TABLE 4:
COUNTRY-WISE DISTRIBUTION OF DOCUMENTS AND CITATIONS

Sr.No	Country	Documents	Citations	Mean Citations
1	China	951	14,785	15.6
2	United States	425	11,522	27.1
3	Saudi Arabia	242	2,798	11.6
4	India	349	3,608	10.3
5	United Kingdom	202	5,888	29.1
6	South Korea	156	2,797	17.9
7	Pakistan	100	1,490	14.9
8	Australia	102	3,328	32.6
9	Germany	101	3,129	31.0
10	Canada	89	2,329	26.2
11	Jordan	50	375	7.5
12	Egypt	89	1,047	11.8
13	Malaysia	82	875	10.7
14	Italy	76	1,575	20.7
15	Spain	63	1,422	22.6

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

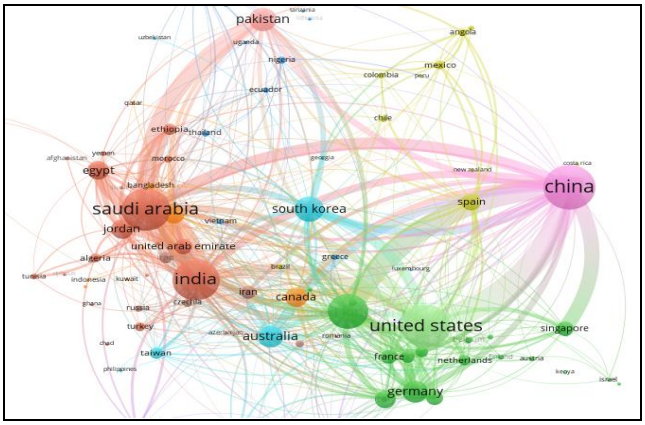

Figure 4: Country-Level Co-Authorship Network in Scientific Publications

The Country-Wise Distribution of Documents and Citations table highlights the global contributions to the field of hybrid AI systems in circuit-level optimization [35]. In the previous year, China produced the most articles, amounting to 951, and garnered the most citations, totaling 14,785, although the country's mean citation per document did remain the most modest at only 15.6. This could mean that the country dominates the quantity of research but that not all of that research has the same level of impact. Following China was the United States, with 425 articles and 11,522 citations, whose mean citation per document was higher at 27.1, meaning that the research produced by the United States was of greater influence. Saudi Arabia and India contributed 349 and 242 articles, respectively, yet their mean citation per document was also lower at 11.6 and 10.6, meaning that while they are contributing to the field, and fairly actively at that, their influence may not be as great. The United Kingdom also produced 202 articles but their mean citation per document was the highest at 29.1, meaning that their research was also of great quality, and then higher still. Together, Australia, Germany, and Canada all contributed and produced a good balance between articles and citations. Australia in particular had a mean citation of 32.6, meaning their contribution was also of great influence. Other countries like Jordan, Egypt, and lesser so, Malaysia and Spain, have recently made some contributions to the field but overall had a much lower number of publications. The Country-Level Co-Authorship Network (figure) visualizes these collaborations and connections across countries, showcasing the global nature of research in hybrid AI for circuit optimization [36].

III. NN-Based IC Design Automation Methods

Here, we talk in more depth about the different ideas that have been put forward over the past six decades for automating the creation of analogue IC the front ends. These methods only use machine learning, specifically supervised and reinforcement teaching with neural networks [37]. To be thorough, we also show the works that talk about mixed methods. Details about how each method works, what situations it can be used in, and its benefits are given. It has been shown that using NNs to answer hard real-world problems works very well in many areas, such as the recognition of speech, medical imaging analysis, driverless cars, language processing, and picture recognition [38]. Equations can't be used to model these kinds of problems because the person designing them can't know what will happen or how things will change over time. Sometimes the person designing the model doesn't even know how to make it. To learn, NNs need a small but useful set of data to make a model that is close to the real answer to a problem [39]. For the testing step, the model also needs to be able to apply and correctly guess new results regarding information that was not available before. NNs are great at handling nonlinear problems because their design is like the way the brain works and they can change how complicated the model is by adding or taking away layers and neurons [40]. NNs were proven to be useful in helping to build analogue circuits for operational amplifiers (opamps) in 2003, as shown in [41]. We will talk more about how supervised as well as reinforcement learning NNs can be used for designing and fitting analogue IC front ends in the parts that follow.

3.1. Supervised Learning

First, get a collection of examples of inputs and outcomes for the issue that is being studied. This is what you should do before you try to come up for a rough answer. For guided learning to work well, it is best to use a large training sample that has been marked. This could be a problem if it's hard to get hold of big files with labels. In the past, people have tried to use NN directed learning to make easy circuits [42]. For example, in [43], Along with 0.18 µm CMOS transistors and a small neural network of 20 neurones that learnt to guess a few gadget sizes from just 1500 samples, it was made. It was used to make a current comparator with five transistors and an inverter [44], the authors made a NN that can change the way a 4-byte current-steering DAC works to achieve the same goals, but in a completely distinct node (0.35 µm CMOS).

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

The size of this method can be changed to fit new needs, however it can't be applied to guess numbers for old ones. The transient power consumption of a relaxation oscillator was modeled in [45] using a 60-neuron time-delay NN, to enhance the circuit's functional model during system-level verification. Not only did it work well to feed mixed-signal bits that carry data, but it also used a lot of power. These are generally talked about in terms of behavioural languages. To make more complex analogue integrated circuits, like those that boost and filter signals, we need more in-depth study, bigger instruction sets, and communication systems with greater number of levels and neurons [46]. First, we need to agree on how to measure the present condition of the art before we may speak about the literature that uses trained students to handle traditional IC design. Because of this, the following constraints are made in this work:

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

3.1.1. Dataset Generation Technique

The factors used for construction Xi in this paper are the standard current and voltage levels (Ire f, Vre f) for an analogous block, as well as the sizes of every single transistor and the passive voice (W, L). The specs Yj tell you how well the electronic device works by telling you things like its gain, how much power it uses, and so on. Getting a set of circuit planning results that include the design area well is the first thing that needs to be done in supervised learning to make a model which may predict Xi. For analogue designers, setting performance goals is the first step [45]. They then use a CAD (computer-aided design) tool to construct the circuit for a lot of different design factors, as seen in the first figure. It takes a long time and people who know a lot about IC CAD tools to put together this kind of information. used SPICE circuit simulators to generate their datasets [47].

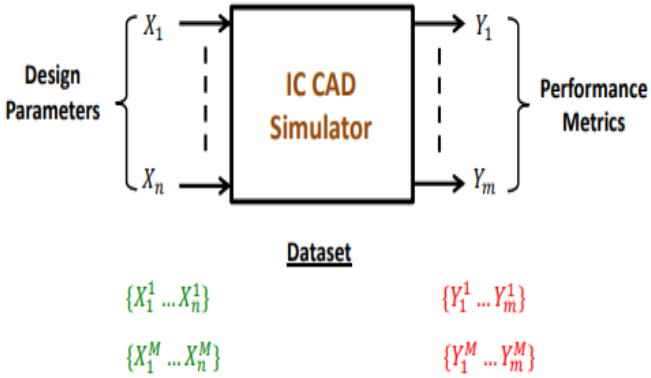

Figure 5: Generation of the training dataset using integrated circuit CAD simulators.

Figure 6 shows a simple source-coupled divergent pair with a running load, an offset current mirror (two of them devices with equal length), as well a known-value inductor for biassing. This was done to show which a dataset looks. Because the transistors are not straight, only the width as well as the length of four of them need to be set. This means that there are 7 design factors. Xi = {W1, L1, W2, L2, W3, L3, W4}.

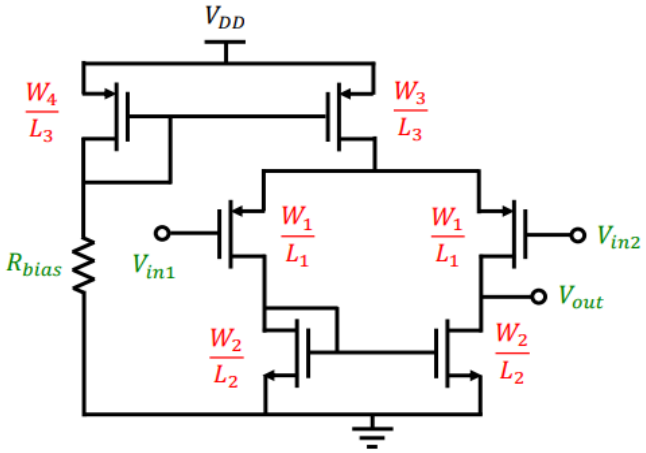

Figure 6: Simple CMOS differential pair amplifier with resistive biasing.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

To keep things simple, let's say that the amplifier that needs to be built only needs to be able to handle differential voltage gain and power usage. So, Yj = {Ad, Ic}, and the vector {Xi,Yj} is a single item in the dataset, which is a 9x9 matrix with N being the number of cases. In Table 1, you can see two examples from the previous circuit's information (sizes are given in µm, current is given in mA).

TABLE 5:
AN EXAMPLE OF A DATASET WITH XI AND YJ

No.	W_1	L_1	W_2	L_2	W_3	L_3	W_4	A □	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	0.40	112	4
…	…	…	…	…	…	…	…	…	…

On the opposite hand, there aren't any hard and fast rules about the amount of information that are needed for each kind of circuit [48]. More data are usually better because they make the model more accurate. But for analogue IC design automated processes, the creator will only make a fair amount of information so as to keep the job realistic and time-effective. For example, 20,000 examples (dataset size) were needed in [49] training the NN and guessing the outcomes of a similar circuit that was created in, with just 9000 and 16,600 cases [50]. On the other hand, only 1600 data points were sufficient in [51]. The earlier numbers, on the other hand, can't be immediately compared because the files from circuit models were made in very different ways for each study. Because of this, it is helpful when looking into the various methods by which the teaching models were made. Before running the simulation, the writers changed the sizes of all the MOS transistors by hand to get a first valid answer that worked well from the point of view of a circuit builder [52]. The people who designed the system made this first change. Then, the basic design factors were changed by 5%, and each of the resulting circuits was simulated one at a time to get information about how well it worked and to create the dataset [53]. Also, the lengths L of all the transistors were kept at 1 µm to save room and avoid effects due to short channels. So, the information in relied too much on a circuit state that had already been set. they suggested a similar but different way to make datasets: The first numbers for the design factors (Xi0) were chosen by the circuit's designer.

Next, Xi0 was changed randomly between ±30% to make 100 different designs. After getting the circuits, they were all recreated, and the best circuit was chosen as the one with the highest score [54]. To get the score, divide the number of performance metrics in the majority by the range of indicators in the numerator. The result gets better as the number goes up. There were as many times of this process as it took to make the collection. With the best circuit from the first 100 shapes as a guide, the next 100 designs were made. This whole process is shown in Figure 7.

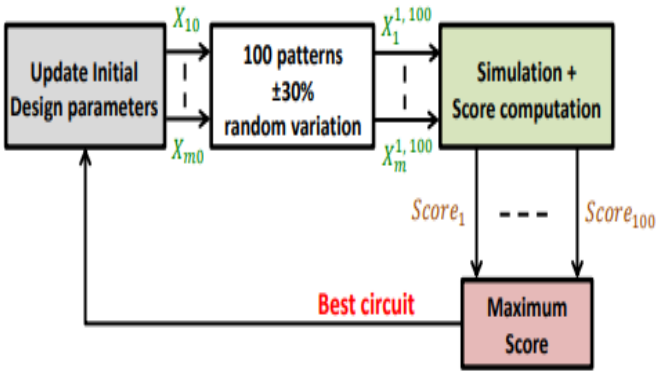

Figure 7: Description of the training-dataset-generation approach

3.2. Reinforcement Learning

An agent learn how to do a job by being praised and criticised for it. This is called RL in machine learning. RL doesn't need a labelled collection to train like supervised learning does [55]. Making guesses and getting awards for those predictions is how the training is done. It makes sense that rewards should go down when the estimate is off and up when it's right. Training's goal is to find a rule that makes the total number of awards as high as possible. RL hasn't been used a lot in IC creation yet [56]. This part shows the past work of four separate teams that used RL to fix problems with the size of analogue integrated circuit designs. Before getting into the specifics, it's important to go over the fundamental elements of RL at IC creation. The state for a circuit is shown by a vector indicating performance traits. The RL agent receives this vector as input, changes the design parameters in order to carry out an action, figures out the reward, and then either goes to the new position if the reward has gone up or looks at other actions that are possible if the reward has not changed (Figure 8). The circuit model data are used to figure out the prize.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

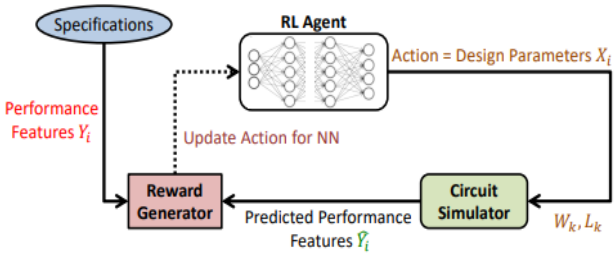

Figure 8: Reinforcement learning design flow as applied to analog IC design automation

After training a real-world agent for 30–40 hours, it did as well as or better than human masters on a number of circuits. At any given step, the executable would send out out a set of digits that showed how big the transistors and idle parts were (these are called design parameters) [57]. They would then be used by the circuit design setting to figure out how efficiently the circuit worked. To make a payment function, hard design boundaries and good-to-have purposes were used. If the success in the game didn't match up with what happened in real life, the prize would be low. If it did, it would be high. They were able to make two-stage and three-stage frequency converters at CMOS 0.18 µm, which shows that their method worked. We looked at noise, gain, power use, AC reaction peaks, and bandwidth to figure out how well it worked. RL had promise in both devices because it found the best deals faster than people could.

Deep reinforcement learning, on the other hand, was used to find the circuit design parameters and learn about the design space. The genetic method took 40 times longer than this one to find good answers. The method was used on both a simple trans capacitive amplifier and a CMOS 45 nm two-stage operational amplifier [58]. Later, it was tested on a different working amplifier with a negative-gm load and a 16 nm FinFET. The method found the right answer 25 times faster on the first try than other methods. Also, it was good at generalisation; out of the 500 different goals it was given, 97% of them were right. It worked 96% of the time on the second and third rounds and 100% of the time on the fourth. It was 40 times faster than normal ways. What was done in [59] used deep RL to guess the design parameters of the circuit, just like in the other works. But the authors came up with a new way to quickly find out what the DC gain and phase margin (PM) numbers were without having to use the circuit model. They did this by using symbolic analysis. More levels didn't have to be sent to the computer to get a good idea of how the circuit would work [60]. This process got rid of the circuit's most likely bad states.

Early on, any set of planning factors that didn't meet the standards for both DC gain and PM were taken out of the loop. An op amp in CMOS 65 nm with a bent cascode was used to test their set up. Even though there were no numbers for convergence speedup in the data, the work was still able to find the right size answers. A hopeful paper was published that used a graph NN (GNN) and RL [61]. In fields like chemistry (things that materials do) and social networks (how people talk to each other), the GNN was good at making predictions because it knew how things that are connected depend on each other. When the writers of used the fact that a circuit is similar to a graph, where points are parts and lines are links (wires), they made their point. It was possible to move data from one circuit to another by making an uncontrolled GNN that looked at how the IC parts were connected and pulled out values [62]. Things like two-stage and three-stage amplifiers were shared by two very different systems that may have learnt from each other. There were also some similarities between two different production process nodes with the same layout. For example, the CMOS 65 nm and 45 nm nodes had the same structure. In this case, the RL method learnt that changing the size of the transistors in the input circuit pair could change the differential amplifier's gain [63]. All four of the CMOS 0.18 µm circuits were shown to work properly. There were amplifiers and low-dropout voltage settings in these systems. Getting the figure of difference (FoM) as high as possible was important. A factor of success (FoM) is made up of several measures of success [64]. Finally, two-sized circuits could be moved from one larger process node to several smaller models with less runtime.

IV. DNN-Opt Framework

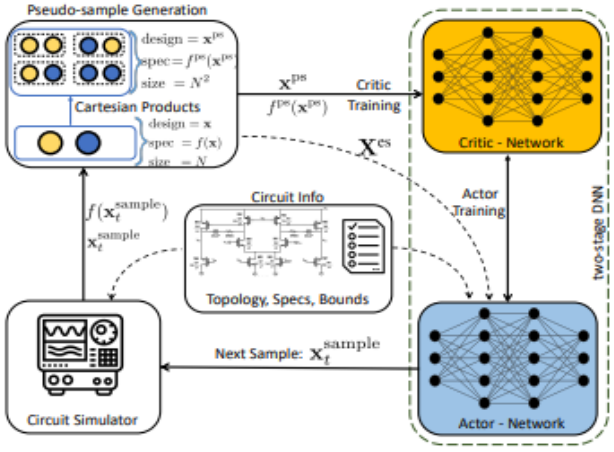

Figure 9: DNN-Opt Framework

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

A. Analog Circuit Sizing: Problem Formulation

The job of sizing an analogue circuit is summed up below as a limited optimisation problem.

$$\begin{aligned} &\text{minimize} \quad f_0(\mathbf{x}) \\ &\text{subject to} \quad f_i(\mathbf{x}) \le 0, \quad i = 1,2,\ldots,m \end{aligned}$$

[1]

Where, x ∈ D d is the parameter vector and d is the number of design variables of sizing task. Thus, D d is the design space. f0(x) is the objective performance metric we aim to minimize. Without loss of generality, we denote i th constraint by fi(x).

B. DNN-Opt Core: RL Inspired Two-Stage DNN Architecture

Figure 1 shows how DNN-Opt is put together as a whole. The DNN-Opt design is built on an a two-stage neural network approach and works with a circuit concept to find the best answer. Samples are generated in the planning area to start the flow. Next, a network of reviewers is used for predicting how well every freshly introduced feature will work. The actor network uses this guess to come up with new people to simulate. This search method works well to copy how BO acts in space exploring [65]. A population control method is also used to improve the sample creation process.

The Deep Deterministic Policy Gradient (DDPG) method is used as a model for our two-stage network design [66], which is a real-life actor-critic program made for action areas with no breaks. In the case of analogue circuit size, however, actor-critic methods can't be used directly because the problem isn't a Markov Decision Process (MDP), which is a requirement for any RL problem [67]. Therefore, we adapt DDPG algorithm with significant modifications tailored for analog circuit sizing.

When it comes to figuring out the size of an analogue circuit, we will retain some of our RL code but change a lot of it to make things easier to understand. Design: An design is a grid of d elements, each of which represents a different design variable [68]. It is made up of circuit factors denoted by x. The goal of optimisation is to find the best xopt that meets Equation 1.

Population: A community is a group of different shapes. In a hybrid AI-based circuit optimization framework, the design population matrix $X \in \mathbb{R}^{N \times d}$ represents a collection of candidate solutions, where each row corresponds to a design vector $\mathbf{x}_i$.

These design vectors encode circuit parameters that form the optimization input [69]. For reinforcement learning (RL)-based optimization, each design is treated as a state in the RL environment: $s_i = \mathbf{x}_i$.

To explore the design space, actions are defined as perturbations $a_i \in \mathbb{R}^d$, resulting in a new state $\mathbf{x'}_i = \mathbf{x}_i + a_i$. The action-guided transitions simulate circuit configuration adjustments during training.

In actor–critic RL algorithms, a critic network works with the reward signal Q(s,a). Here, the critic assesses the fitness or performance of a new design 〖x^'〗_i. For circuit sizing tasks, the critic is trained with supervised learning using the data f(x) in the form of performance metrics such as delay, power, or gain.

To train the critic, pseudo-samples are generated using two sampling strategies: (1) an action-based perturbed vector $\mathbf{x'}_i$, and (2) the corresponding performance score $f(\mathbf{x'}_i)$. This process enables the network to learn the performance landscape across the design space efficiently.

$$\begin{aligned} \mathbf{x}_i^{\text{ps}} &= [\mathbf{x}_i, \Delta\mathbf{x}_i] = [\mathbf{x}_i, \mathbf{x'}_i - \mathbf{x}_i] \\ f^{\text{ps}}(\mathbf{x}_i) &= f(\mathbf{x'}_i) \end{aligned} \qquad [2]$$

To improve the ability of the critic network in performance modelling, the input dimensions are increased from d to 2d by adding the design vector x, along with its action perturbation, Δx. This means the input will now be (x, Δx), which will allow the network to obtain the more granular learnings necessary in the design space. Experimental results using Bayesian analog circuit sizing benchmarks have shown that using these extended pseudo-samples improves the learning quality of the critic network compared to using only original samples [70].

The critic network is trained using a Mean Squared Error (MSE) loss function over a batch of N pseudo-samples, where the predicted value is compared to the known simulated performance values:

$$\mathcal{L}(\phi) = \frac{1}{N}\sum_{i=1}^{N}\left(Q_\phi(\mathbf{x}_i, \Delta\mathbf{x}_i) - f(\mathbf{x}_i + \Delta\mathbf{x}_i)\right)^2 \qquad [3]$$

Here, Q_ϕ is the critic network's prediction, and $f(\cdot)$ is the ground-truth SPICE-simulated performance metric (e.g., gain, bandwidth, delay).

Once the critic network is trained, the actor network is optimized to explore the design space and output perturbations Δx that improve circuit performance. The training of the actor is guided by a Figure of Merit (FoM) function, which scores each sample based on its relative quality with respect to other designs in the batch.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

$$g(f(\mathbf{x}_i)) = w_i f_0(\mathbf{x}_i) + \sum_{j=1}^{N} \min\left(1, \max\left(0, \frac{f(\mathbf{x}_j) - f(\mathbf{x}_i)}{f(\mathbf{x}_j)}\right)\right)$$

[4]

In this formulation:

- w_i is a weight factor for the primary objective f_0,
- The summation component represents a pairwise ranking loss encouraging the actor to produce higher-ranking designs,
- Clipping ensures numerical stability during optimization.

A batch of N samples is ranked using this loss function to guide the actor network's policy toward producing optimal design perturbations $\Delta \mathrm{x}$. The combination of critic-based performance learning and FoM-guided ranking leads to more robust and sample-efficient design space exploration.

The goal of training the actor network is to find changes to parameters Δx_k that make it work better without breaking any design rules. To do this, a loss function is created that blends judging ability with punishing people who break the rules. The actor network is trained by making the following loss function as small as possible for a batch size of N_b:

$$L(\theta^a) = \frac{1}{N_b} \sum_{k=1}^{N_b} [Q(s_k, \mu(\mathbf{x}_k; \theta^a)) + \| \Lambda \cdot \mathrm{viol}_k \|_1] \quad [5]$$

In this equation:

- $\mu(\mathrm{x}_k; \theta^a)$ is the actor network's output perturbation for input design $\mathbf{x}_k$,
- $Q(\cdot)$ is the critic's evaluation function,
- Λ is a diagonal weight matrix controlling the penalty magnitude for constraint violations,
- viol_k measures the amount by which the proposed design exceeds allowable boundaries.

To make sure the search stays in the area where the design is possible, the total border violation for the kth sample is found by:

$$\mathrm{viol}_k = \max(0, \mathrm{x}_k + \Delta \mathrm{x}_k - \mathrm{ub}_{\mathrm{rest}}) + \max(0, \mathrm{lb}_{\mathrm{rest}} - (\mathrm{x}_k + \Delta \mathrm{x}_k))$$

[6]

- $\mathrm{lb}_{\mathrm{rest}}$ and $\mathrm{ub}_{\mathrm{rest}}$ are the lower and upper bounds, respectively, determined from elite designs in the current population.

These bounds are computed as:

$$\mathrm{lb}_{\mathrm{rest}} = \min(\mathrm{x}^e), \quad \mathrm{ub}_{\mathrm{rest}} = \max(\mathrm{x}^e) \quad \forall i = 1, \ldots, d$$

[7]

where x^e are elite solutions and d is the dimensionality of the design space.

The inclusion of the boundary violation term ensures that the actor network explores valid regions of the design space and avoids infeasible solutions. This design-aware optimization framework helps guide the policy network to generate circuit parameters that are both optimal and realizable [71].

C. DNN-Opt: Overall Framework

The DNN-Opt framework is a hybrid actor–critic reinforcement learning architecture designed for analog circuit optimization. First, we perform a sensitivity analysis, which narrows down the most important features to reduce the design space. We randomly sample a population of exact size N_init to initialize the candidates for the design space. We then bring the candidates into a sequence of pseudo-sample generation, critic/actor training, and elite selection for the o iterations of the optimisation[72].

The first step in each iteration is to generate pseudo-samples, followed by critic and actor network training using the previously described loss functions. An elite population X^e is then selected from the updated pool based on a Figure of Merit (FoM) ranking. This elite set represents high-performing circuit candidates and is used to guide the actor network's future predictions.

For every member $\mathrm{x}_k^e \in X^e$, the actor network generates a new design vector $\mathbf{x}_{k'}$, representing a candidate improvement:

$$\mathbf{x}_{k'} = \mathbf{x}_k^e + \mu(\mathbf{x}_k^e) \quad [8]$$

where $\mu(\mathbf{x}_k^e)$ is the learned perturbation from the actor network.

After generating all candidate solutions, the best-performing candidate is selected as the representative sample for the next optimization round. This decision is made using the critic network's score $Q(\cdot)$ and the FoM guidance $g(\cdot)$, formalized as:

$$\mathrm{x}^{\mathrm{sample}} = \left[\mathrm{x}_k^e \text{ for } k = \underset{k}{\mathrm{argmax}}(g(Q(\mathrm{x}_k^e, \mathrm{x}_{k'} - \mathrm{x}_k^e)))\right] \quad [9]$$

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

V. MACHINE LEARNING ALGORITHMS

At its core, intelligence is the ability to learn from experience and pass on specific information from one generation to the next. Machine learning is the study of how to make computer programs (called "learning systems") that get better over time [73]. This time, the experience comes from a process of data analysis that is done by a special program. So, the machine learning method uses mathematical, statistical, optimisation, and knowledge finding methods along with programs to find patterns in a set of data [74]. Although the machine learning term was coined around the 1960s, , it only gained popularity in the 21st century due to the advancement of computational resources. Machine learning is often used to solve problems like numerical prediction (regression), pattern recognition (classification), grouping, optimisation, and control. As an example, these days they can be used in almost every field of study: music , health [75], economics [76],[77], industrial segments [78],[79], education [80],[81], among many others.

We look at three types of machine learning: unsupervised learning, reinforcement learning, and supervised learning [82]. There are different methods within each class based on how they get their information, such as classification, regression, grouping, learning of relationships, relations, differential equations, and so on. Figure 2 shows the machine learning techniques that were thought about.

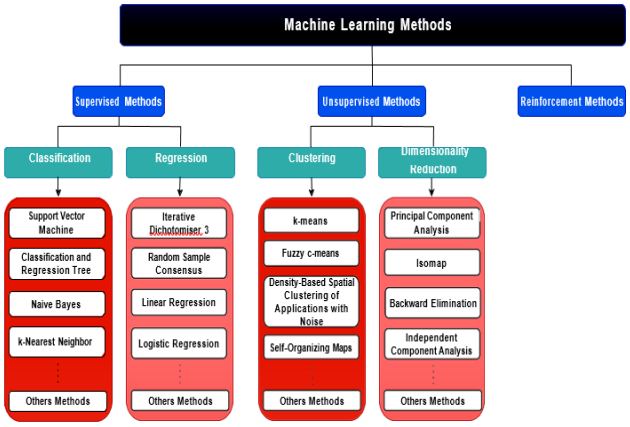

Figure 10: Machine learning methods

The goal of the machine learning process is to come up with a framework (model) that the can use what it learnt from training data on samples it has never seen before [83].

This model needs to be easy to use and good at finding mistakes in the information being collected. You should test an AI system on new data to see how effectively the model design works (mistake rate) before you use it [84]. The assessment methods were chosen given the amount of knowledge that was provided. Most of the time, when a knowledge set is sufficiently large, three sets are looked at: the training set, and these is used to improve the highest model; the set that serves as validation, that makes the model superior by modifying the first model additionally general; and the set for testing, which finds out how often the final model gets things wrong [85]. It's important to remember that you have to pick one of those three sets at random. This means you need a big enough data set.

In some situations, mostly in real life, though, you have to work with limited data because you can't always get three separate and important data sets. Because of this, different ways of evaluating should be used. One choice is to utilise the pause method. There is some material that will be used for tests and the rest will be used for training. People often save a third during the data for tests and use the rest for training. You could also use k-fold cross-validation for small data sets. This tech- nique is very useful in fixed data samples to forecast the success rate of a learning method. In *k*-fold cross-validation, the training and testing process is done *k* times. Thus, consider a given data *D*, which is randomly divided into *k* mutually exclusive subsets D_k, in which $k = 1, ..., k$ each of approximately equal size. In the iteration *k*, the D_k partition is reserved for testing, and the remaining subsets are used to train the model. Thus, in the first iteration, the set $D_2 \cup D_3 \cup \cup D_k$ serves as the training set to attain the first model, which is tested on D_1; the second iteration is trained on $D_1 \cup D_3 \cup \ldots Dk$ and tested on D_2; and so on. In the end, the *k* error estimates received from *k* iterations are averaged to give rise to an overall error estimate. So, the usual number used to guess how often a learning method will make a mistake is k = 10. The three types of machine learning we talked about earlier are explained below: Learning with supervision, learning without supervision, and learning through reinforcement [86].

5.1 Supervised Learning Methods

Assisted learning methods try to figure out how input attributes (also called "independent variables") relate to a goal characteristic (also called "dependent variables"). When it comes to maths, guided learning is a way to look into knowledge that is already known.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

Consider the data set DS used to infer a model of the system, in which each individual instance is represented by x^i, given by

- N is the number of data set elements,
- n_f is the number of attributes (features) of each instance x^i,

$$DS = \begin{bmatrix} x^1, y^1 \\ \vdots \\ x^N, y^N \end{bmatrix} \quad [10]$$

The data set DS lies in state space $\mathbb{R}^N \times \mathbb{R}^{n_f+1}$. The choice of features (or attributes, or parameters) x_j^i, $j = 1, \ldots, n_f$, for a given instance i, significantly affects the output. There are two types of tasks for which supervised learning is used: pattern classification or regression (whose purpose is to predict the value of one or more target attributes).

Classification

Consider the output vector y ∈ Y, where Y stands for M discrete classes. According to the tests in the training set X, the classifier's job is to sort the data into different groups, or to choose which of the M groups each new vector xnew goes to. Many algorithms, including Decision Tree, Support Vector, Machine, k-Nearest Neighbour, Naive Bayes, and others, can be used for the classification job. Figur3 shows a basic classification algorithm process. The goal is to put a set of points (shown by the input vector) into two groups, as shown in the input classes vector: blue and orange.

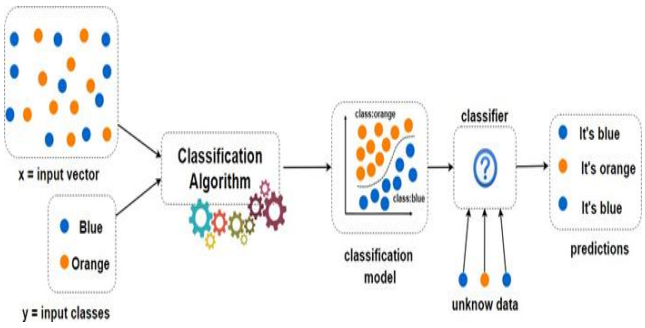

Figure 11: Classification algorithm procedures

Regression

The goal of regression learning is to find out how independent variables, also called features, (input variables x) relate to a dependent variable, also called result (continuous results variable y). Fitting an objective to the The input-output data is what the regression job is all about. The goal is to predict (numerical) outputs for new inputs. There are several forms of regression, such as linear, multiple, weighted, polynomial, nonparametric, and robust [87]. Simple Linear Regression, Logistic regression, Multivariate Regression, and Regression tree are some examples of algorithms that can be used to build regression models [88]. Figure 12 illustrates a general linear regression algorithm procedure, in which the aim is to define a linear function that represents the data set behavior.

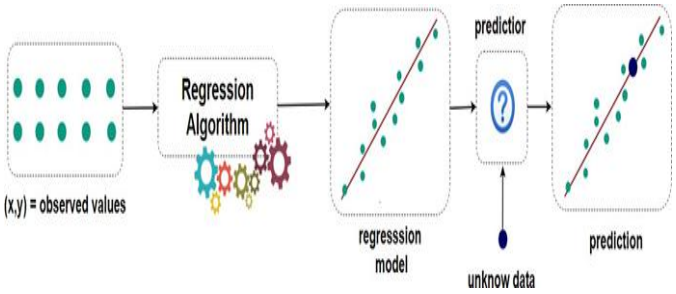

Figure 12: Regression algorithm procedures

5.2 Unsupervised Learning Methods

In some machine learning problems, there is little information about how the qualities of input and output are related [89]. So, the programs have to find things in a data set that are alike or different. The method needs more human insight than guided techniques because the final choice is made by a decision-maker, who could be someone or an entire group of people. Both guided and unguided approaches work with information that we need to explore and understand the data in the application area. However, there are some important differences between the two. One big difference is that there is no output array of the goal variable like there is in supervised methods. Also, independent learning is often linked to creative activities like exploring, understanding, and improving, which don't work well with set steps like guided methods do [90]. Because of this, it can't be automatic. Also, there is not a correct or incorrect response and no easy way to tell from the statistics whether the results are good or bad.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

Stats that describe and show things are important parts of the process. So, unsupervised learning is usually broken down into two groups: approaches to clustering and dimension reduction methods [91].

Clustering Methods

As we already said, sometimes the data set isn't labelled, so it's important to look at the unique qualities of each piece of the information in the set. It can be said that grouping techniques are the most popular unstructured method. Basically, clustering is the process of putting things into groups based on how similar they are to each other and how different they are from each other [92]. This procedure is very useful in engineering, health science, humanities, economics, and other areas [93],[94]. The evaluation of a data set's constituent members' proximity and the division of the data set into groups while taking into account the similarity and dissimilarity between a pair of elements are both essential steps in the clustering process. It is useful to denote the distance between two instances x^i and x^j as $d(x^i, x^j)$ to quantify the similarity between them.

To define the quality of the cluster, it is necessary to use an evaluation criteria measure that is usually divided into two categories: internal and external. The internal quality metrics usually measure the compactness of the clusters using some similarity measure. And, the external measures can be useful for examining whether the structure of the clusters matches some predefined classification of the instances. According to [95] the notion of "cluster" is not precisely defined, for this reason, many clustering methods and algorithms have been developed. These methods can be divided into 5 categories [96]: Partition- ing based, Hierarchical based, Density-based, Grid-based and Model-based. Some examples of clustering algorithms from different types of clustering methods are given below: This list includes the k-means algorithm, the fuzzy c-means algorithm (FCM), the clustering using representatives algorithm (CURE), the density-based spatial clustering of applications with noise algorithm (DBSCAN), the ordering points to identify the clustering structure algorithm (OPTICS), the optimal grid-clustering algorithm (OptiGrid), the gaussian mixture model clustering algorithm, and the self-organising maps clustering algorithm (SOMs) [97].

Figure 5 illustrates an exclusive clustering algorithm procedure, whose objective is to divide the data set into groups according to the data characteristics. In this case, the term exclusive is associated with the idea that each data point exclusively belongs to one cluster.

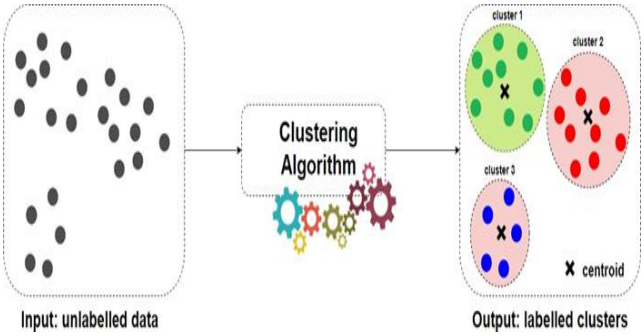

Figure 13: Exclusive clustering algorithm procedures

Dimensionality Reduction

The complexity of an information collection is the number of factors that can be used to learn from it. So, reduction dimension methods pick out the most important factors and leave out the ones that aren't important and could mess up or delay the mining procedure in some way. But the factors that are chosen must keep as much of the original data set's change as possible [98]. There are two main ways to group the reduction dimensionality methods into groups:

- The first factor is whether the technique employs the target variable to select input variables or not.
- The second factor is whether the technique utilizes a subset of the original
  variables or derives new variables from them to maximize the amount of information.

The benefit of keeping the original variables makes sense since the variables that were already in the data are simple to understand than the variables that were made automatically by a particular reduction method. On the other hand, when working with large amounts of data, using reduction dimensionality approaches is the only way to make sure that the machine learning process works well. Certain methods, like Independent Component Analysis (ICA), Rough Sets-Based Feature Reduction, Basic Component Analysis (PCA), and Backward Elimination, can be used to reduce the number of dimensions [99].

3.3 Reinforcement Learning

Contextual learning is a way to teach machines to learn by paying them. It is most often used for dynamic control mechanisms, but it may also be utilised to solve optimisation problems. According to the idea behind reinforcement learning, if an action occurs followed by a good outcome or an enhancement in the outcome, then the desire to do that action gets greater, or is "enforced."

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

Reinforcement learning is a way to teach an independent agent that acts and feels its surroundings how to pick the best actions to reach its goals [100]. The agent is given information about how things are right now in the environment. It needs to use what it already knows by becoming greedy to get the most rewards, but it also needs to look into what it can do better in the future.

With more formalism, reinforcement learning can be formulated as a Markov decision process as presented in Figure 14. At each time step t, given the current state s_t (and current reward r_t), the agent needs to learn a strategy (i.e. the "value function") that selects the optimal decision or action a_t. The action will have an impact on the environment that induces the next reward signal r_{t+1} (which can be positive, negative, or zero) and also produces the next state s_{t+1}. The reinforcement learning continues with a trial-and-error process until it learns an optimal or suboptimal strategy [99].

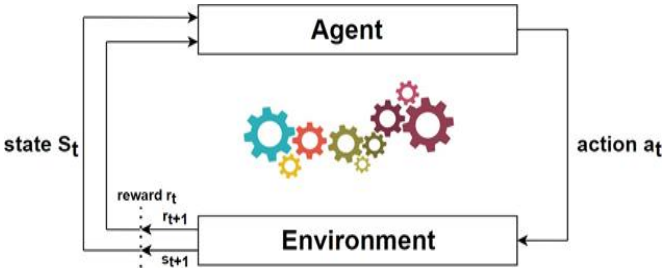

Figure 14: Reinforcement learning schematic. Adapted from

There is a difference between reinforcement learning and guided learning that needs to be made clear. Some people say that in reinforcement learning, the algorithm learns from what it has already done, while in supervised learning, the data set acts as a guide and shows the trends to the algorithms [101]. The reviewer doesn't say anything ahead of time. Because of this, after a number of actions have been completed and awards have been received, it is advisable to look back at each action and figure out which one led to the prize. This makes it possible to record these moves and play them back later.

TABLE 6:
MACHINE LEARNING ALGORITHMS IN CIRCUIT-LEVEL OPTIMIZATION

ML Algorithm	Key Applications	Authors	Description
Supervised Learning			
Support Vector Machines (SVMs)	Classification, Regression	[102]	SVMs are used for classification and regression tasks by finding the hyperplane that best divides data into classes or predicts continuous values.
Random Forests	Classification, Regression	[103]	A decision tree ensemble method used for classification and regression. It creates multiple decision trees and merges their results.
Gradient Boosting Machines (GBMs)	Classification, Regression	[104]	An ensemble technique that builds models sequentially, where each model corrects errors of the previous one, improving prediction accuracy.
Unsupervised Learning			
K-means Clustering	Clustering, Pattern Recognition	[105]	A clustering algorithm that partitions data into K distinct clusters based on distance from the center.
Principal Component Analysis (PCA)	Dimensionality Reduction, Feature Extraction	[106]	PCA is used for reducing the dimensionality of data by projecting it onto a smaller set of orthogonal components while retaining the most significant variance.
Autoencoders	Dimensionality Reduction, Data	[107]	A neural network used to learn efficient representations of data, typically for the purpose of

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

	Compression		reducing its dimensionality or denoising data.
Reinforcement Learning			
Deep Q-Network (DQN)	Game AI, Robotics, Optimization	[108]	DQN uses deep learning to approximate the Q-function, which helps in making optimal decisions in environments requiring sequential action decisions.
Actor-Critic (AC) Algorithms	Optimization, Decision-Making	[109]	A reinforcement learning framework where the "actor" makes decisions and the "critic" evaluates them based on rewards, used in optimizing designs or control systems.
Deep Deterministic Policy Gradient (DDPG)	Continuous Control, Circuit Design	[110]	A deep RL approach used for optimizing continuous action spaces, like analog circuit design, where actions and states are continuous rather than discrete.
Graph-based Learning			
Graph Convolutional Networks (GCNs)	Circuit Design, Molecule Property Prediction, Social Networks	[111]	GCNs are used to process graph-structured data and are particularly effective in applications where relationships between entities (nodes) are important, such as IC design and chemistry.
Hybrid Approaches			
Bayesian Optimization (BO)	Analog Circuit Optimization, Design Space Exploration	[112]	BO is used to optimize black-box functions, often coupled with ML models like Gaussian Processes or Neural Networks to predict performance and optimize parameters iteratively.
Genetic Algorithms (GAs)	Parameter Optimization, Circuit Design	[113]	GAs mimic natural selection and evolve solutions over generations to find optimal or near-optimal solutions, often used in design optimization and search problems.
Particle Swarm Optimization (PSO)	Optimization Problems, Circuit Design	[114]	A population-based optimization algorithm inspired by the social behavior of birds flocking, used to find optimal solutions by adjusting individual "particles" in a swarm.

VI. CONCLUSION

Considering everything, mixed AI systems, especially those integrating machine learning (ML) systems with traditional circuit-level optimisation methodologies, represent a paradigm shift in the way integrated circuits (ICs) are designed and optimised. Even though Moore's Law placed a ceiling on the complexity of next-generation IC designs, optimisation methodologies based on heuristics, rules, and time-consuming models fall short, mainly due to the increasing need to optimise on power, performance and area (PPA).

Alternatives such as hybrid AI models in the context of circuit design, in particular, have shown to be more effective, more flexible and more automated than the traditional models. In particular, guided learning, reinforcement learning (RL), and Graph Neural Networks (GNNs) have been very helpful in improving the design cycle's most important jobs, like device size, layout, placement, and routing.

Of particular note is the way RL and graph models have improved over the years with respect to the more challenging optimisation tasks of a design cycle such as cost of computation, convergence of designs, and generalisation of designs over a varied topological arrangement of integrated circuits, and over frees of designs. Of note are frameworks like DNN-Opt which combine deep learning with architectures inspired by RL in order to enhance the sizing of circuits and design with more efficiency to cover the enormous design constellation.

The increased publication rate, enhanced research, and deeper international integration acknowledge AI as one of the viable techniques to integrate with high modern IC design workflows. His bibliometric analysis shows the integration of AI consolidating underway thrust with notable inputs from China, the United States, and India. The field is actually maturing the next design of IC instruments for hybrid AI systems to be broadened. The fields of optimisation systems for the dynamic problems of next-generation semiconductor devices are smart and incorporate future affordability into the design. The integration of circuit design continues to face the evolving system for absolute efficiency in the integrations of the systems to the overall solutions design.

REFERENCES

[1] M. P. Sarma and K. K. Sarma, "Optimization Techniques in VLSI Circuits: An Analytical Perspective of the Current State of the Art," in *Exploring the Intricacies of Digital and Analog VLSI*, K. Guha, J. Kandpal, and S. Devi, Eds., IGI Global, 2025, pp. 143–182. doi: 10.4018/979-8-3693-8084-0.ch005.

[2] M. Fayazi, Z. Colter, E. Afshari, and R. Dreslinski, "Applications of Artificial Intelligence on the Modeling and Optimization for Analog and Mixed-Signal Circuits: A Review," *IEEE Trans. Circuits Syst. I*, vol. 68, no. 6, pp. 2418–2431, June 2021, doi: 10.1109/TCSI.2021.3065332.

[3] R. Mina, C. Jabbour, and G. E. Sakr, "A Review of Machine Learning Techniques in Analog Integrated Circuit Design Automation," *Electronics*, vol. 11, no. 3, p. 435, Jan. 2022, doi: 10.3390/electronics11030435.

[4] H. Wang *et al.*, "GCN-RL Circuit Designer: Transferable Transistor Sizing with Graph Neural Networks and Reinforcement Learning," 2020, doi: 10.48550/ARXIV.2005.00406.

[5] W. Meng, X. Zhang, L. Zhou, H. Guo, and X. Hu, "Advances in UAV Path Planning: A Comprehensive Review of Methods, Challenges, and Future Directions," *Drones*, vol. 9, no. 5, p. 376, May 2025, doi: 10.3390/drones9050376.

[6] Y. Li *et al.*, "A Circuit Attention Network-Based Actor-Critic Learning Approach to Robust Analog Transistor Sizing," in *2021 ACM/IEEE 3rd Workshop on Machine Learning for CAD (MLCAD)*, Raleigh, NC, USA: IEEE, Aug. 2021, pp. 1–6. doi: 10.1109/MLCAD52597.2021.9531156.

[7] J. Hu, W. L. Goh, and Y. Gao, "AI-Powered Agile Analog Circuit Design and Optimization," 2025, *arXiv*. doi: 10.48550/ARXIV.2505.03750.

[8] D. K. Sarmah, "A Survey on the Latest Development of Machine Learning in Genetic Algorithm and Particle Swarm Optimization," in *Optimization in Machine Learning and Applications*, A. J. Kulkarni and S. C. Satapathy, Eds., in Algorithms for Intelligent Systems. , Singapore: Springer Singapore, 2020, pp. 91–112. doi: 10.1007/978-981-15-0994-0_6.

[9] M. Golzan, T. M. N. Ngatched, K. Popuri, and L. Zhang, "Analog and Mixed-Signal IC Modeling and Optimization: An Artificial Intelligence Perspective," *ACM Trans. Des. Autom. Electron. Syst.*, p. 3754339, July 2025, doi: 10.1145/3754339.

[10] G. Huang *et al.*, "Machine Learning for Electronic Design Automation: A Survey," 2021, *arXiv*. doi: 10.48550/ARXIV.2102.03357.

[11] Q. Cappart, D. Chételat, E. B. Khalil, A. Lodi, C. Morris, and P. Veličković, "Combinatorial Optimization and Reasoning with Graph Neural Networks," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, Montreal, Canada: International Joint Conferences on Artificial Intelligence Organization, Aug. 2021, pp. 4348–4355. doi: 10.24963/ijcai.2021/595.

[12] K. A. Tahir, "A Systematic Review and Evolutionary Analysis of the Optimization Techniques and Software Tools in Hybrid Microgrid Systems," *Energies*, vol. 18, no. 7, p. 1770, Apr. 2025, doi: 10.3390/en18071770.

[13] A.-A.-R. Asif *et al.*, "Performance Evaluation and Comparative Analysis of Different Machine Learning Algorithms in Predicting Cardiovascular Disease," vol. 29, no. 2, 2021.

[14] G. E. Sery, "Approaching the one billion transistor logic product: process and design challenges," presented at the Design, Process Integration, and Characterization for Microelectronics, A. Starikov and K.W. Tobin, Jr., Eds., Santa Clara, CA, July 2002, pp. 254–261. doi: 10.1117/12.475662.

[15] B. F. Azevedo, A. M. A. C. Rocha, and A. I. Pereira, "Hybrid approaches to optimization and machine learning methods: a systematic literature review," *Mach Learn*, vol. 113, no. 7, pp. 4055–4097, July 2024, doi: 10.1007/s10994-023-06467-x.

[16] R. Rashid, K. Krishna, C. P. George, and N. Nambath, "Machine Learning Driven Global Optimisation Framework for Analog Circuit Design," *Microelectronics Journal*, vol. 151, p. 106362, Sept. 2024, doi: 10.1016/j.mejo.2024.106362.

[17] J. Hu, W. L. Goh, and Y. Gao, "AI-Powered Agile Analog Circuit Design and Optimization," May 08, 2025, *arXiv*: arXiv:2505.03750. doi: 10.48550/arXiv.2505.03750.

[18] C. Chen, H. Wang, X. Song, F. Liang, K. Wu, and T. Tao, "High-Dimensional Bayesian Optimization for Analog Integrated Circuit Sizing Based on Dropout and g_m/I_D Methodology," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 41, no. 11, pp. 4808–4820, Nov. 2022, doi: 10.1109/TCAD.2022.3147431.

[19] D. Basso, L. Bortolussi, M. Videnovic-Misic, and H. Habal, "Effective Analog ICs Floorplanning with Relational Graph Neural Networks and Reinforcement Learning," in *2025 Design, Automation & Test in Europe Conference (DATE)*, Lyon, France: IEEE, Mar. 2025, pp. 1–7. doi: 10.23919/DATE64628.2025.10992888.

[20] H. Wang *et al.*, "GCN-RL Circuit Designer: Transferable Transistor Sizing with Graph Neural Networks and Reinforcement Learning," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, July 2020, pp. 1–6. doi: 10.1109/DAC18072.2020.9218757.

[21] K. Settaluri, A. Haj-Ali, Q. Huang, K. Hakhamaneshi, and B. Nikolic, "AutoCkt: Deep Reinforcement Learning of Analog Circuit Designs," Jan. 20, 2020, *arXiv*: arXiv:2001.01808. doi: 10.48550/arXiv.2001.01808.

[22] 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 *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Grenoble, France: IEEE, Feb. 2021, pp. 1372–1377. doi: 10.23919/DATE51398.2021.9474253.

[23] C. Visan *et al.*, "Automated Circuit Sizing with Multi-objective Optimization based on Differential Evolution and Bayesian Inference," June 06, 2022, *arXiv*: arXiv:2206.02391. doi: 10.48550/arXiv.2206.02391.

[24] C. Vişan, M. Boldeanu, G. Nicolae, H. Cucu, C. Burileanu, and A. Buzo, "Evolutionary Bayesian Optimization for automated circuit sizing," *Knowledge-Based Systems*, vol. 318, p. 113483, June 2025, doi: 10.1016/j.knosys.2025.113483.

[25] K. S. N. S and P. Li, "LLM-USO: Large Language Model-based Universal Sizing Optimizer," Feb. 04, 2025, *arXiv*: arXiv:2502.02764. doi: 10.48550/arXiv.2502.02764.

[26] R. Mina, C. Jabbour, and G. E. Sakr, "A Review of Machine Learning Techniques in Analog Integrated Circuit Design Automation," *Electronics*, vol. 11, no. 3, p. 435, Jan. 2022, doi: 10.3390/electronics11030435.

[27] S. M. M. Sajadieh and S. D. Noh, "From Simulation to Autonomy: Reviews of the Integration of Artificial Intelligence and Digital Twins," *Int. J. of Precis. Eng. and Manuf.-Green Tech.*, May 2025, doi: 10.1007/s40684-025-00750-z.

[28] A. Balakrishnan, T. Lange, M. Glorieux, D. Alexandrescu, and M. Jenihhin, "The Validation of Graph Model-Based, Gate Level Low-Dimensional Feature Data for Machine Learning Applications," in *2019 IEEE Nordic Circuits and Systems Conference (NORCAS): NORCHIP and International Symposium of System-on-Chip (SoC)*, Helsinki, Finland: IEEE, Oct. 2019, pp. 1–7. doi: 10.1109/NORCHIP.2019.8906974.

[29] V. Srikanth, P. Aswini, R. Chandrashekar, N. Sirisha, M. Kumar, and K. Adnan, "Machine Learning-Based Analogue Circuit Design for Stage Categorization and Evolutionary Optimization," in *2024 Second International Conference Computational and Characterization Techniques in Engineering & Sciences (IC3TES)*, Lucknow, India: IEEE, Nov. 2024, pp. 1–6. doi: 10.1109/IC3TES62412.2024.10877553.

[30] Y. Xiao, S. Nazarian, and P. Bogdan, "Self-Optimizing and Self-Programming Computing Systems: A Combined Compiler, Complex Networks, and Machine Learning Approach," *IEEE Trans. VLSI Syst.*, vol. 27, no. 6, pp. 1416–1427, June 2019, doi: 10.1109/TVLSI.2019.2897650.

[31] M. A. Ayanwale, R. R. Molefi, and S. Oyeniran, "Analyzing the evolution of machine learning integration in educational research: a bibliometric perspective," *Discov Educ*, vol. 3, no. 1, p. 47, May 2024, doi: 10.1007/s44217-024-00119-5.

[32] K. R, S. N, C. K, G. N, K. S V, and R. S, "Exploring the Frontiers: Innovations in Semiconductor Technology and AI-Driven SoC Design," *SSRN Journal*, 2025, doi: 10.2139/ssrn.5086781.

[33] P. Wei *et al.*, "Bibliographical progress in hybrid renewable energy systems' integration, modelling, optimization, and artificial intelligence applications: A critical review and future research perspective," *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects*, vol. 45, no. 1, pp. 2058–2088, Apr. 2023, doi: 10.1080/15567036.2023.2181888.

[34] D. Pretolesi, I. Stanzani, S. Ravera, A. Vian, and A. Barla, "Artificial intelligence and network science as tools to illustrate academic research evolution in interdisciplinary fields: The case of Italian design," *PLoS ONE*, vol. 20, no. 1, p. e0315216, Jan. 2025, doi: 10.1371/journal.pone.0315216.

[35] M. F. M. Barros, J. M. C. Guilherme, and N. C. G. Horta, *Analog Circuits and Systems Optimization based on Evolutionary Computation Techniques*, vol. 294. in Studies in Computational Intelligence, vol. 294. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. doi: 10.1007/978-3-642-12346-7.

[36] G. Anusha, A. V. V. Sudhakar, R. R. R. Deshmukh, and C. H. H. Basha, "A bibliometric analysis of research trends in electric vehicle power electronics: global perspectives and future directions," *Discov Appl Sci*, vol. 7, no. 5, p. 487, May 2025, doi: 10.1007/s42452-025-06996-1.

[37] L. Wang, W. Zhang, X. He, and H. Zha, "Supervised Reinforcement Learning with Recurrent Neural Network for Dynamic Treatment Recommendation," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, London United Kingdom: ACM, July 2018, pp. 2447–2456. doi: 10.1145/3219819.3219961.

[38] L. A. Kumar and D. K. Renuka, *Deep Learning Approach for Natural Language Processing, Speech, and Computer Vision: Techniques and Use Cases*, 1st ed. Boca Raton: CRC Press, 2023. doi: 10.1201/9781003348689.

[39] B. Dou *et al.*, "Machine Learning Methods for Small Data Challenges in Molecular Science," *Chem. Rev.*, vol. 123, no. 13, pp. 8736–8780, July 2023, doi: 10.1021/acs.chemrev.3c00189.

[40] A. Bielecki, *Models of Neurons and Perceptrons: Selected Problems and Challenges*, vol. 770. in Studies in Computational Intelligence, vol. 770. Cham: Springer International Publishing, 2019. doi: 10.1007/978-3-319-90140-4.

[41] G. Wolfe and R. Vemuri, "Extraction and use of neural network models in automated synthesis of operational amplifiers," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 22, no. 2, pp. 198–212, Feb. 2003, doi: 10.1109/TCAD.2002.806600.

[42] J. E. Rayas-Sanchez, "EM-Based Optimization of Microwave Circuits Using Artificial Neural Networks: The State-of-the-Art," *IEEE Trans. Microwave Theory Techn.*, vol. 52, no. 1, pp. 420–435, Jan. 2004, doi: 10.1109/TMTT.2003.820897.

[43] E. Afacan, N. Lourenço, R. Martins, and G. Dündar, "Review: Machine learning techniques in analog/RF integrated circuit design, synthesis, layout, and test," *Integration*, vol. 77, pp. 113–130, Mar. 2021, doi: 10.1016/j.vlsi.2020.11.006.

[44] R. A. Vural, N. Kahraman, B. Erkmen, and T. Yildirim, "Process independent automated sizing methodology for current steering DAC," *International Journal of Electronics*, pp. 1–22, Jan. 2015, doi: 10.1080/00207217.2014.989924.

[45] M. Grabmann, F. Feldhoff, and G. Gläser, "Power to the Model: Generating Energy-Aware Mixed-Signal Models using Machine Learning," in *2019 16th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, Lausanne, Switzerland: IEEE, July 2019, pp. 5–8. doi: 10.1109/SMACD.2019.8795295.

[46] K. A. Ng, E. Greenwald, Y. P. Xu, and N. V. Thakor, "Implantable neurotechnologies: a review of integrated circuit neural amplifiers," *Med Biol Eng Comput*, vol. 54, no. 1, pp. 45–62, Jan. 2016, doi: 10.1007/s11517-015-1431-3.

[47] H. M.V. and B. P. Harish, "Artificial Neural Network Model for Design Optimization of 2-stage Op-amp," in *2020 24th International Symposium on VLSI Design and Test (VDAT)*, Bhubaneswar, India: IEEE, July 2020, pp. 1–5. doi: 10.1109/VDAT50263.2020.9190315.

[48] Z. Wang, X. Luo, and Z. Gong, "Application of Deep Learning in Analog Circuit Sizing," in *Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence*, Shenzhen China: ACM, Dec. 2018, pp. 571–575. doi: 10.1145/3297156.3297160.

[49] M. Fukuda, T. Ishii, and N. Takai, "OP-AMP sizing by inference of element values using machine learning," in *2017 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, Xiamen: IEEE, Nov. 2017, pp. 622–627. doi: 10.1109/ISPACS.2017.8266553.

[50] N. Lourenco *et al.*, "On the Exploration of Promising Analog IC Designs via Artificial Neural Networks," in *2018 15th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, Prague: IEEE, July 2018, pp. 133–136. doi: 10.1109/SMACD.2018.8434896.

[51] R. M. Hasani, D. Haerle, C. F. Baumgartner, A. R. Lomuscio, and R. Grosu, "Compositional neural-network modeling of complex analog circuits," in *2017 International Joint Conference on Neural Networks (IJCNN)*, Anchorage, AK, USA: IEEE, May 2017, pp. 2235–2242. doi: 10.1109/IJCNN.2017.7966126.

[52] A. Vladimirescu and S. Liu, "The Simulation of MOS Integrated Circuits Using SPICE2:," Defense Technical Information Center, Fort Belvoir, VA, Oct. 1980. doi: 10.21236/ADA606827.

[53] G. G. E. Gielen, H. C. C. Walscharts, and W. M. C. Sansen, "Analog circuit design optimization based on symbolic simulation and simulated annealing," *IEEE J. Solid-State Circuits*, vol. 25, no. 3, pp. 707–713, June 1990, doi: 10.1109/4.102664.

[54] J. D. Lohn and S. P. Colombano, "A circuit representation technique for automated circuit design," *IEEE Trans. Evol. Computat.*, vol. 3, no. 3, pp. 205–219, Sept. 1999, doi: 10.1109/4235.788491.

[55] J. Fan, C. Xiao, and Y. Huang, "GDI: Rethinking What Makes Reinforcement Learning Different From Supervised Learning," 2021, *arXiv*. doi: 10.48550/ARXIV.2106.06232.

[56] C.-Y. Chen and J.-L. Huang, "Reinforcement-Learning-Based Test Program Generation for Software-Based Self-Test," in *2019 IEEE 28th Asian Test Symposium (ATS)*, Kolkata, India: IEEE, Dec. 2019, pp. 73–735. doi: 10.1109/ATS47505.2019.00013.

[57] H. Wang, J. Yang, H.-S. Lee, and S. Han, "Learning to Design Circuits," 2018, *arXiv*. doi: 10.48550/ARXIV.1812.02734.

[58] D. Roohbakhsh, A. Sanaee, and T. Aghaee, "Four- stage CMOS operational transconductance amplifier: Compensated via double differential blocks," *Int J Numerical Modelling*, vol. 36, no. 4, p. e3067, July 2023, doi: 10.1002/jnm.3067.

[59] Z. Zhao and L. Zhang, "Deep Reinforcement Learning for Analog Circuit Sizing," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, Seville, Spain: IEEE, Oct. 2020, pp. 1–5. doi: 10.1109/ISCAS45731.2020.9181149.

[60] B. H. Calhoun *et al.*, "Digital Circuit Design Challenges and Opportunities in the Era of Nanoscale CMOS," *Proc. IEEE*, vol. 96, no. 2, pp. 343–365, Feb. 2008, doi: 10.1109/JPROC.2007.911072.

[61] J. Park, J. Chun, S. H. Kim, Y. Kim, and J. Park, "Learning to schedule job-shop problems: representation and policy learning using graph neural network and reinforcement learning," *International Journal of Production Research*, vol. 59, no. 11, pp. 3360–3377, June 2021, doi: 10.1080/00207543.2020.1870013.

[62] L. Alrahis, J. Knechtel, and O. Sinanoglu, "Graph Neural Networks: A Powerful and Versatile Tool for Advancing Design, Reliability, and Security of ICs," in *Proceedings of the 28th Asia and South Pacific Design Automation Conference*, Tokyo Japan: ACM, Jan. 2023, pp. 83–90. doi: 10.1145/3566097.3568345.

[63] S. Hong *et al.*, "Analog Circuit Design Automation via Sequential RL Agents and Gm/ID Methodology," *IEEE Access*, vol. 12, pp. 104473–104489, 2024, doi: 10.1109/ACCESS.2024.3435331.

[64] A. L. Carsrud, K. W. Olm, and J. B. Thomas, "Predicting entrepreneurial success: effects of multi-dimensional achievement motivation, levels of ownership, and cooperative relationships," *Entrepreneurship & Regional Development*, vol. 1, no. 3, pp. 237–244, Jan. 1989, doi: 10.1080/08985628900000020.

[65] P. Pracht, F. Bohle, and S. Grimme, "Automated exploration of the low-energy chemical space with fast quantum chemical methods," *Phys. Chem. Chem. Phys.*, vol. 22, no. 14, pp. 7169–7192, 2020, doi: 10.1039/C9CP06869D.

[66] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," July 05, 2019, *arXiv*: arXiv:1509.02971. doi: 10.48550/arXiv.1509.02971.

[67] V. R. Konda and J. N. Tsitsiklis, "Actor-Critic Algorithms".

[68] D. L. Stevens, "VARIABLE DENSITY GRID-BASED SAMPLING DESIGNS FOR CONTINUOUS SPATIAL POPULATIONS," *Environmetrics*, vol. 8, no. 3, pp. 167–195, May 1997, doi: 10.1002/(SICI)1099-095X(199705)8:3<167::AID-ENV239>3.0.CO;2-D.

[69] M. S. Dasika and C. D. Maranas, "OptCircuit: An optimization based method for computational design of genetic circuits," *BMC Syst Biol*, vol. 2, no. 1, p. 24, Dec. 2008, doi: 10.1186/1752-0509-2-24.

[70] A. F. Budak, "Efficient optimization methods for analog/mixed-signal integrated circuits via machine learning," The University of Texas at Austin, 2023. doi: 10.26153/TSW/51909.

[71] C. Schaff, "Neural Approaches to Co-Optimization in Robotics," 2022, *arXiv*. doi: 10.48550/ARXIV.2209.00579.

[72] A. F. Budak, P. Bhansali, B. Liu, N. Sun, D. Z. Pan, and C. V. Kashyap, "DNN-Opt: An RL Inspired Optimization for Analog Circuit Sizing using Deep Neural Networks," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*, San Francisco, CA, USA: IEEE, Dec. 2021, pp. 1219–1224. doi: 10.1109/DAC18074.2021.9586139.

[73] F. Rasheed and A. Wahid, "Learning style detection in E-learning systems using machine learning techniques," *Expert Systems with Applications*, vol. 174, p. 114774, July 2021, doi: 10.1016/j.eswa.2021.114774.

[74] C. M. Bishop, *Pattern recognition and machine learning*. in Information science and statistics. New York: Springer, 2006.

[75] A. A. Abarghouei, A. Ghanizadeh, S. Sinaie, and S. M. Shamsuddin, "A Survey of Pattern Recognition Applications in Cancer Diagnosis," in *2009 International Conference of Soft Computing and Pattern Recognition*, Malacca, Malaysia: IEEE, 2009, pp. 448–453. doi: 10.1109/SoCPaR.2009.93.

[76] G. Cicceri, G. Inserra, and M. Limosani, "A Machine Learning Approach to Forecast Economic Recessions—An Italian Case Study," *Mathematics*, vol. 8, no. 2, p. 241, Feb. 2020, doi: 10.3390/math8020241.

[77] M. Buasri, "LEVERAGING AI IN FINANCIAL SERVICES: A GUIDE TO IMPLEMENTING MACHINE LEARNING FOR FRAUD PREVENTION".

[78] Glaucia Maria, "Bayesian Approach to Infer Types of Faults on Electrical Machines from Acoustic Signal," *Appl. Math. Inf. Sci.*, vol. 15, no. 3, pp. 353–364, May 2021, doi: 10.18576/amis/150313.

[79] P. B. Fávero and F. A. M. Zoucas, "REDES NEURAIS PARA PREVISÃO DA PRODUÇÃO INDUSTRIAL DE DIFERENTES SEGMENTOS," *P&P*, vol. 17, no. 2, Aug. 2016, doi: 10.22456/1983-8026.51900.

[80] F. Agrusti, G. Bonavolontà, and M. Mezzini, "University Dropout Prediction through Educational Data Mining Techniques: A Systematic Review," *Journal of e-Learning and Knowledge Society*, pp. 161-182 Pages, Oct. 2019, doi: 10.20368/1971-8829/1135017.

[81] S. Zhu, "Research on data mining of education technical ability training for physical education students based on Apriori algorithm," *Cluster Comput*, vol. 22, no. S6, pp. 14811–14818, Nov. 2019, doi: 10.1007/s10586-018-2420-8.

[82] Southwest Jiaotong University, China, I. Muhammad, Z. Yan, and Southwest Jiaotong University, China, "SUPERVISED MACHINE LEARNING APPROACHES: A SURVEY," *IJSC*, vol. 05, no. 03, pp. 946–952, Apr. 2015, doi: 10.21917/ijsc.2015.0133.

[83] G. Nguyen *et al.*, "Machine Learning and Deep Learning frameworks and libraries for large-scale data mining: a survey," *Artif Intell Rev*, vol. 52, no. 1, pp. 77–124, June 2019, doi: 10.1007/s10462-018-09679-z.

[84] R. Dzhusupova, R. Banotra, J. Bosch, and H. H. Olsson, "Using artificial intelligence to find design errors in the engineering drawings," *J Software Evolu Process*, vol. 35, no. 12, p. e2543, Dec. 2023, doi: 10.1002/smr.2543.

[85] M. D. Dunnette, "A modified model for test validation and selection research.," *Journal of Applied Psychology*, vol. 47, no. 5, pp. 317–323, Oct. 1963, doi: 10.1037/h0047635.

[86] S. Dargan, M. Kumar, M. R. Ayyagari, and G. Kumar, "A Survey of Deep Learning and Its Applications: A New Paradigm to Machine Learning," *Arch Computat Methods Eng*, vol. 27, no. 4, pp. 1071–1092, Sept. 2020, doi: 10.1007/s11831-019-09344-w.

[87] A. Ceselli, "Information Management course".

[88] C. M. Bishop, *Pattern recognition and machine learning*. in Information science and statistics. New York: Springer, 2006.

[89] H. Chen, "Machine learning for information retrieval: Neural networks, symbolic learning, and genetic algorithms," *J. Am. Soc. Inf. Sci.*, vol. 46, no. 3, pp. 194–216, Apr. 1995, doi: 10.1002/(SICI)1097-4571(199504)46:3<194::AID-ASI4>3.0.CO;2-S.

[90] C. Hockings, L. Thomas, J. Ottaway, and R. Jones, "Independent learning – what we do when you're not there," *Teaching in Higher Education*, vol. 23, no. 2, pp. 145–161, Feb. 2018, doi: 10.1080/13562517.2017.1332031.

[91] B. K. Tripathy, S. Anveshrithaa, and S. Ghela, *Unsupervised Learning Approaches for Dimensionality Reduction and Data Visualization: Unsupervised Learning Approaches for Dimensionality Reduction and Data Visualization*, 1st ed. Boca Raton: CRC Press, 2021. doi: 10.1201/9781003190554.

[92] F. Iglesias, T. Zseby, and A. Zimek, "Clustering refinement," *Int J Data Sci Anal*, vol. 12, no. 4, pp. 333–353, Oct. 2021, doi: 10.1007/s41060-021-00275-z.

[93] Y. Zhou, H. Wu, Q. Luo, and M. Abdel-Baset, "Automatic data clustering using nature-inspired symbiotic organism search algorithm," *Knowledge-Based Systems*, vol. 163, pp. 546–557, Jan. 2019, doi: 10.1016/j.knosys.2018.09.013.

[94] N. Albarakati and Z. Obradovic, "Multi-domain and multi-view networks model for clustering hospital admissions from the emergency department," *Int J Data Sci Anal*, vol. 8, no. 4, pp. 385–403, Nov. 2019, doi: 10.1007/s41060-018-0147-5.

[95] V. Estivill-Castro and J. Yang, "Fast and Robust General Purpose Clustering Algorithms," in *PRICAI 2000 Topics in Artificial Intelligence*, vol. 1886, R. Mizoguchi and J. Slaney, Eds., in Lecture Notes in Computer Science, vol. 1886. , Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 208–218. doi: 10.1007/3-540-44533-1_24.

[96] V. Mehta, S. Bawa, and J. Singh, "Analytical review of clustering techniques and proximity measures," *Artif Intell Rev*, vol. 53, no. 8, pp. 5995–6023, Dec. 2020, doi: 10.1007/s10462-020-09840-7.

[97] L. A. Rasyid and S. Andayani, "Review on Clustering Algorithms Based on Data Type: Towards the Method for Data Combined of Numeric-Fuzzy Linguistics," *J. Phys.: Conf. Ser.*, vol. 1097, p. 012082, Sept. 2018, doi: 10.1088/1742-6596/1097/1/012082.

[98] W. Fan, "Systematic data selection to mine concept-drifting data streams," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, Seattle WA USA: ACM, Aug. 2004, pp. 128–137. doi: 10.1145/1014052.1014069.

[99] S. Velliangiri, S. Alagumuthukrishnan, and S. I. Thankumar Joseph, "A Review of Dimensionality Reduction Techniques for Efficient Computation," *Procedia Computer Science*, vol. 165, pp. 104–111, 2019, doi: 10.1016/j.procs.2020.01.079.

[100] L.-J. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," *Mach Learn*, vol. 8, no. 3–4, pp. 293–321, May 1992, doi: 10.1007/BF00992699.

[101] I. H. Sarker, "Machine Learning: Algorithms, Real-World Applications and Research Directions," *SN COMPUT. SCI.*, vol. 2, no. 3, p. 160, May 2021, doi: 10.1007/s42979-021-00592-x.

[102] R. G. Brereton and G. R. Lloyd, "Support Vector Machines for classification and regression," *Analyst*, vol. 135, no. 2, pp. 230–267, 2010, doi: 10.1039/B918972F.

[103] M. Hamza and D. Larocque, "An empirical comparison of ensemble methods based on classification trees," *Journal of Statistical Computation and Simulation*, vol. 75, no. 8, pp. 629–643, Aug. 2005, doi: 10.1080/00949650410001729472.

[104] G. I. Webb and Z. Zheng, "Multistrategy ensemble learning: reducing error by combining ensemble learning techniques," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 8, pp. 980–991, Aug. 2004, doi: 10.1109/TKDE.2004.29.

[105]K. M. Kumar and A. R. M. Reddy, "An efficient k-means clustering filtering algorithm using density based initial cluster centers," *Information Sciences*, vol. 418–419, pp. 286–301, Dec. 2017, doi: 10.1016/j.ins.2017.07.036.

[106]N. Salem and S. Hussein, "Data dimensional reduction and principal components analysis," *Procedia Computer Science*, vol. 163, pp. 292–299, 2019, doi: 10.1016/j.procs.2019.12.111.

[107]S. Tan and M. L. Mayrovouniotis, "Reducing data dimensionality through optimizing neural network inputs," *AIChE Journal*, vol. 41, no. 6, pp. 1471–1480, June 1995, doi: 10.1002/aic.690410612.

[108]S. E. Li, "Deep Reinforcement Learning," in *Reinforcement Learning for Sequential Decision and Optimal Control*, Singapore: Springer Nature Singapore, 2023, pp. 365–402. doi: 10.1007/978-981-19-7784-8_10.

[109]R. Abdalla, W. Hollstein, C. P. Carvajal, and P. Jaeger, "Actor-critic reinforcement learning leads decision-making in energy systems optimization—steam injection optimization," *Neural Comput & Applic*, vol. 35, no. 22, pp. 16633–16647, Aug. 2023, doi: 10.1007/s00521-023-08537-6.

[110]Y. Uhlmann, M. Brunner, L. Bramlage, J. Scheible, and C. Curio, "Procedural- and Reinforcement-Learning-Based Automation Methods for Analog Integrated Circuit Sizing in the Electrical Design Space," *Electronics*, vol. 12, no. 2, p. 302, Jan. 2023, doi: 10.3390/electronics12020302.

[111]C. Hua, "Learning From Graph-Structured Data: Addressing Design Issues and Exploring Practical Applications in Graph Representation Learning," 2024, *arXiv*. doi: 10.48550/ARXIV.2411.07269.

[112]Z. Shangguan, L. Lin, W. Wu, and B. Xu, "Neural Process for Black-Box Model Optimization Under Bayesian Framework," 2021, *arXiv*. doi: 10.48550/ARXIV.2104.02487.

[113]Y.-P. Huang, Y.-T. Chang, S.-L. Hsieh, and F. E. Sandnes, "An adaptive knowledge evolution strategy for finding near-optimal solutions of specific problems," *Expert Systems with Applications*, vol. 38, no. 4, pp. 3806–3818, Apr. 2011, doi: 10.1016/j.eswa.2010.09.041.

[114]J. Hu, H. Wu, B. Zhong, and R. Xiao, "Swarm intelligence-based optimisation algorithms: an overview and future research issues," *IJAAC*, vol. 14, no. 5/6, p. 656, 2020, doi: 10.1504/IJAAC.2020.110077.