Progress in AI-Accelerated VLSI Systems

RESEARCH ARTICLE

Progress in AI-Accelerated VLSI Systems • Jan - Apr • 1 • 1 • 2026 (pp. 1-8)

Learning-Guided VLSI Design Automation: AI-Driven Optimization of Power, Performance, and Area in Advanced Nodes

Saravanakumar Veerappan*

Director, Centivens Institute of Innovative Research, Coimbatore, Tamil Nadu, India

KEYWORDS:
AI-assisted EDA,
VLSI design automation,
Machine learning for PPA optimization,
advanced technology nodes,
Learning-guided synthesis

ARTICLE HISTORY:
Received : 03.11.2025
Revised : 12.12.2025
Accepted : 08.01.2026

ABSTRACT

Such high scaling of whichever node of semiconductor technology has greatly complicated the process of modern VLSI design making it very difficult to balance power, performance and area in a way that best optimizes power performance and area amidst tight design schedules and land use by conventional rule-based electronic design automation (EDA) workflows. Conventional heuristic based optimization models have a weakness of inability to scale effectively, poorly exploring design space and time to converge especially at the deep scaled technology nodes. In order to overcome these shortcomings, this paper presents a learning framework of VLSI design automation that incorporates the machine learning methods at both the synthesis and physical design phases to support predictive and adaptable PPA optimization. The proposed methodology uses design data collected in effect through intermediate EDA phases to train learning models that are capable of predicting with high accuracy timing, power, and area measures to be utilized in directing design choices within an iterative optimization cycle. The framework conducts a proactive process of influencing the EDA process by integrating data-driven intelligence into an explicit design process to achieve a better convergence and less dependency on manual tuning. When experimental appraisals are performed on representative benchmark digital circuits executed at advanced technologies nodes, the enhancement in power efficiency, timing closure, and the area usage is consistently greater when compared to the conventional non-AI-driven EDA flows. Also, the learning-guided approach has a great effect, it minimizes the amount of design cycles necessary to reach closure, which demonstrates the benefits of the guiding design approach in speeding up the design turnaround time. The findings confirm that learning-driven automation offers a scalable, effective, and viable learning-solution to the design processes of the next generation VLSI design-flow addressing the advanced semiconductor technologies.

Author's e-mail: saravanatheguru@gmail.com

How to cite this article: Veerappan S. Learning-Guided VLSI Design Automation: AI-Driven Optimization of Power, Performance, and Area in Advanced Nodes. Progress in AI-Accelerated VLSI Systems. Vol.1, No. 1, 2026 (pp. 1-8).

INTRODUCTION

Further scaling of transistors as well as the implementation of newer semiconductor technology nodes have considerably contributed to making modern very-large-scale integration (VLSI) design far more complicated. Advanced nodes have high design constraints, routing congestion, high process variability, and high power density, which make it difficult to find an optimum power, performance, and area (PPA) trade-offs. Consequently, it has made predictable timing closure and power efficient design operations a critical concern to modern electronic design automation (EDA) processes due to aggressive design schedules. Traditional EDA products are based on rule of thumb and trial and error largely because they are only guided by heuristics and trial and error optimization throughout synthesis, placement, routing and timing connection phases. Although these techniques have been discovered to be good with earlier generations of technologies, their scaling and efficiency reduce greatly with deeply scaled nodes. The frequent problems faced by designers are too much design iterations, long turn around times and dwindling PPA benefits as a result of the weak capability of heuristic-based approaches to search large and strongly nonlinear design spaces in an efficient manner. Moreover, the growing heterogeneity of the current VLSI systems, brought about by the complexity of their architectures and workloads, increases the drawbacks of the simply optimistic approaches.

The recent developments in the field of artificial intelligence (AI) and machine learning (ML) have become promising sources to solve these problems. The data-driven methods have proven effective in modeling any complicated design parameter and ultimate implementation results, which thus predicts the timing, power, and area metrics ahead of time.[1, 3] Design-space exploration, the placement optimization, and performance estimation have been considered with reinforcement learning and supervised learning methods.[4, 6] Nevertheless, the current research aims at individual design phases or individual prediction frameworks without a close coordination with end-to-end EDA processes. Consequently, existing AI-enabled solutions do not tend to offer adaptive, closed-loop optimization that would help in the minimization of design cycles, as well as, guarantee robust PPA convergence at advanced nodes.

In order to overcome these drawbacks, this paper advances an idea of a learning-directed VLSI design automation framework which inserts machine learning intelligence into the synthesis and physical design steps. The proposed method can be used to provide proactive optimization advice, minimize the need to rely on manual tuning, and convergence to optimum PPA solutions in a shorter period by using intermediate design information and predictive methods.

The links of the work are as follows:

- One learning-based framework of AI-based VLSI design automation.
- Forecasting of power, performance, and area measures using prediction machines that consume in-between design characteristics.
- ergonomic optimization guidance that leads to overall minimization of design cycles and enhanced efficiency of convergence.
- Experimental confirmation on the similar PPA improvement with advanced technology node.

The rest of this paper is structured in the following way. Section 2 is the literature review concerning AI-supported EDA and VLSI optimization. Section 3 is the formulation of the PPA optimization problem. Section 4 outlines the proposed design automation approach based on learning. Section 5 outlines the experimental arrangement, results and discussion in Section 6. Section 7 finally wraps up the paper and gives a future direction of research.

RELATED WORK

The past years have witnessed a strong movement in applying techniques based on the artificial intelligence (AI) and machine learning (ML) to the VLSI design automation in response to the increasing complexity of advanced technology nodes. Initial research was mainly on the use of monitored learning models to predict major design parameters like timing, power and routing congestion, so that it could be estimated faster than the entire post-layout analysis when full post-layout analysis.[7, 8] These methods showed that regression and ensemble learning models were able to make reasonable predictions of power and delay based on features derived on the netlists and placement data. In addition to metric prediction, it has been indicated that reinforcement learning (RL) can also be used to design a design space and physical design optimization. Significant publications have also used RL to macro placement and floorplanning and obtain better results than with existing heuristic-based methods in terms of improved wirelength and congestion reduction.[9, 10] In the same vein, the optimization of synthesis parameters and setting of placement strategies has also been suggested using learning-based approaches with potential to speed up convergence to timing closure.[11] In spite of these improvements, there are a number of constraints. To begin with, much of the current literature only focuses on single steps of the EDA process including placement or timing models, but does not consider the end-to-end optimization of both synthesis and physical design. Consequently, local PPA gains are not necessarily converted into international gains. Second, most of the reinforcement learning-based methods have high training costs and low generalization across various designs and technology nodes, thus preventing their use in reality in the design flow.[12] Moreover, most learning-based systems are used in an advisory mode not highly integrated into iterative cycles of EDA and can therefore not offer adaptive and ongoing optimization advice.

Unlike a previous work, the suggested method aims at the design of the VLSI design automation based on learning, in which the predictive models are directly applied within the design loop in a manner that aids the decision-making at various design phases. The framework can optimize

PPA with scalable, adaptive, and computationally efficient predictive models by using intermediate design information and improving predictions as needed, eliminating important gaps in current AI-assisted EDA processes.

PROBLEM FORMULATION

With a specified digital VLSI design at the register-transfer level (RTL), the design automation process aims at arriving at an optimal implementation, which fulfills desirable power, performance, and area (PPA) requirements and meets stringent timing and design constraints of the advanced semiconductor technology nodes. The high level of interdependence between the design parameters and the nonlinear characteristic of the contemporary physical design processes makes the optimization problem inherently complex.

Based on this, the optimization problem can be formulated in the form of a constrained multi-objective problem:

$$\min \{P(x), A(x)\} \quad \text{subject to } D(x) \leq D_{target} \tag{1}$$

where P(x) denotes the total power consumption, A(x) represents the silicon area, and D(x) corresponds to the critical path delay of the implemented design. The vector x encapsulates the set of design variables influencing the final implementation.

Synthesis constraints (e.g., timing and optimization effort), standard-cell options and sizing, placement density and utilization, routing effort parameters, clock frequency and voltage settings as well as the synthesis space are all design variables available in the design variable space x. All of these variables make up a high dimensional, and highly coupled design space, where even minor changes in parameters can lead to a substantial change in the final PPA metrics.

Exploration of this design space is done in the traditional EDA processes by repeated cycles of heuristic-based optimization at the stages of synthesis, placement, routing and timing closure. Nevertheless, these methods become less efficient as nodes get more advanced and it can take many design innovations to meet timing requirements with an attempt to reduce power and area. In addition to this, the design space is too large to explore exhaustively, this results in suboptimal solutions and long design turn around time.

This fundamental issue that this work will tackle is thus the ability to effectively estimate the individualization between the terms of design and the ensuing PPA measures such that informed optimization choices can be made without consuming EDA forests. The proposed framework will consequently predict the results of PPA at the design-flow early stage and steer the process of optimization towards better results and design via the application of learning as a guiding force to the optimization process.

PROPOSED LEARNING-GUIDED DESIGN AUTOMATION FRAMEWORK

In this section, the proposed VLSI design automation framework based on learning and the manner in which the research was carried out is described. The system incorporates machine learning intelligence into the traditional electronic design automation (EDA) workflow so that it could optimize power, performance and area (PPA) using data. As opposed to the historical methods of heuristic design and development, the offered methodology uses predictive modeling and feedback to inform synthesis and physical design choices, enhancing the efficiency of convergence and ultimate design quality. Figure 1 shows the general structure and the workflow of the suggested learning-centered framework.

Feature Extraction from EDA Stages

The initial phase of the framework consists in extracting design features of intermediate flow steps of the EDA in a systematic manner. A feature rich set is then amassed in between RTL synthesis and during physical design that captures logical and physical properties of a design. They are netlist-level metrics including the number of gates, logic depth, distributions of fan-ins and fan-outs, and the use of cell types as well as physical design metrics including the placement density, routing congestion, wirelength estimates, and switching activity information.

Mathematically, it is possible to define the extracted feature vector as:

$$f = [f_1, f_2, \dots, f_n] \tag{2}$$

where each element f_i corresponds to a specific design attribute derived from synthesis or physical design reports. These characteristics are the inputs of the learning models and give a small picture of the design state of a specific optimization step.

Learning Model Training and PPA Prediction

The supervised machine learning models will be trained in the second stage to gain insights into the nonlinear dependency between the design features absorbed and ultimate PPA measures. Given a dataset $D=\{(f_i,y_i)\}_{i=1}^{N}$, where f_i denotes the feature vector and $y_i=[P_i,D_i,A_i]$ represents the corresponding power, delay, and area values obtained after implementation, the learning task is formulated as a regression problem.

The model learns a mapping function:

$$\hat{y} = \mathcal{M}(f) \tag{3}$$

where M(·) denotes the trained learning model and represents the predicted PPA metrics. Optimization of the model parameters is carried out by minimizing a loss term, which is;

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}\|y_i - \hat{y}_i\|^2 \tag{4}$$

The training is conducted with historical design information obtained after more than one run of EDA so that the model can be used to provide generalization among various design configurations and operating points.

Learning-Guided Optimization Strategy

The EDA flow is then integrated with the learning model, which is then learned to make optimization decisions. Predicted PPA metrics are applied at every iteration of a candidate design, followed by entire EDA runs being run. The synthesis constraints, cell sizing policies, placement density policies and routing efforts policies are adjusted based on such predictions to guide the design to better PPA results. The optimization procedure may also be viewed as the guided search within the design space, wherein the focal point of the learning model serves as a surrogate estimator, eliminating the necessity of screen displays to do costly full design assessments. The framework optimizes the choice of configuring based on desirable predicted metrics thus greatly minimizing redundant design cycles whilst ensuring design fidelity and time restrictions.

Iterative Feedback and Model Refinement

The last phase of the framework introduces a feedback mechanism, which is based on a series of iterations to correct the learning model as more design data is provided. The actual PPA results are evaluated against the initial predictions following every EDA run and the newly obtained data are added to the training data. Periodically the model is retrained or fine-tuned to enhance the prediction accuracy and strength. This feedback loop process facilitates adaptive learning so that the framework is able to gain greater and greater guidance capability as it is exposed to new design situations. Consequently, the learning-guided framework automation is more effective with time, and the convergence speed is higher with better scalability of the complex design and the advanced technology node.

The overall learning-guided design automation process may be recapped in Algorithm 1 that illustrates how machine learning-based prediction and optimisation of EDA are to be integrated.

Summary of the suggested learning-guided VLSI design automation platform that incorporates machine learning models into the traditional EDA algorithm to enable predictive power, performance, and area (PPA) optimization at the iterative feedback process.

Algorithm 1: Learning-Guided VLSI Design Automation Flow

```
Input: RTL design $D_{RTL}$, target delay $D_{target}$, initial design constraints $x_0$, maximum iterations T
Output: Optimized design configuration $x^*$ with improved PPA metrics
Step 1: Initialize conventional EDA flow with RTL design $D_{RTL}$ and initial constraints $x_0$
Step 2: Run synthesis and initial physical design to obtain baseline implementation
Step 3: Extract intermediate design features $f_t$ from synthesis and physical design reports (netlist metrics, utilization, congestion, switching activity)
Step 4: If training dataset D is empty, execute full EDA flow and store $(f_t, y_t)$ in D, where $y_t=[P_t, D_t, A_t]$
Step 5: Train or update machine learning model M using dataset D
Step 6: For iteration t=1 to T:
    6.1 Predict PPA metrics
    6.2 If , generate candidate design configuration $x_t$
    6.3 Adjust synthesis and physical design parameters (clock constraints, cell sizing, placement density, routing effort)
    6.4 Run guided EDA iteration with updated configuration $x_t$
    6.5 Extract new features $f_{t+1}$ and obtain actual PPA metrics $y_{t+1}$
    6.6 Append $(f_{t+1}, y_{t+1})$ to dataset D
Step 7: Update learning model M using the expanded dataset D
Step 8: Check convergence criteria (timing closure achieved and PPA improvement below threshold) If satisfied, terminate; else continue iteration
Step 9: Return optimized design configuration $x^*$
```

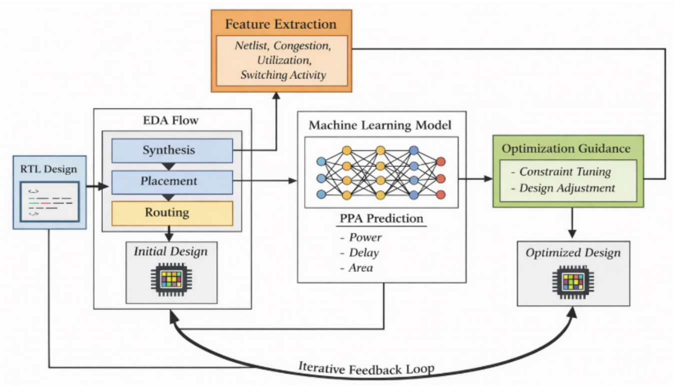

Fig. 1: Learning-Guided VLSI Design Automation Framework

EXPERIMENTAL SETUP

This part explains the experimental setup, the choice of benchmarks, a flow of EDA tool, machine learning setup, and the evaluation procedure that was employed to prove the presented learning-directed VLSI design automation framework. The major goal of the experimental work is to determine the effectiveness of the proposed strategy both quantitatively in terms of power and performance and area (PPA) metrics, and, overall, in terms of reducing the design convergence time, in comparison with a traditional non-AI-assisted EDA flow.

Benchmark Circuits and Technology Nodes

The framework is tested based on a series of exemplary digital benchmark circuits heavily employed in research in both VLSI design and physical design. The design of the benchmark suite consists of both routability-driven placement benchmark set designs of the ISPD and some selected OpenCores RTL designs, spread out in both design size, logic depth, and utilization. These standards make structural and physical properties more diverse, and strong evaluation of scalability and universalization is possible. The designs of all the benchmark are synthesized and executed through the ASAP7 7-nm predictive process design kit (PDK), which mirrors constraints and challenges of more advanced nodes of semiconductor technology. All designs are realised at register-transfer level (RTL) and tested under the same functional constraint, target clock frequencies and timing constraints across all the experimental runs to take fair comparison between the proposed learning-guided flow and the baseline EDA flow.

EDA Tool Flow and Implementation Environment

The syntaxic execution of the experiment is based on a normal pattern of digital design that includes logic synthesis, position, routing, and static timing examination. All experiments are done with OpenROAD open-source EDA toolchain (version 2.0), which is fully RTL-to-GDSII-automated and records all the design stages in detail. The default OpenROAD optimization settings are used in the baseline flow, fixed heuristics uses and fixed design parameters. By contrast, the suggested learning-guided model supplements this process by adding machine learning-based prediction and optimization advice in between successive iterations of synthesis and physical design. In OpenROAD, the design reports produced at intermediate design stages are used as feature extractors and model-training inputs, such as timing, power, utilization and congestion statistics. All tests are performed on a workstation with an Intel Xeon 3.4 GHz CPU, 64 GB RAM and an NVIDIA RTX-3090 graphics card, running the Ubuntu 22.04 LTS operating system but remains in a constant hardware and software state throughout all runs of the tests.

Baseline Configuration

The required baseline EDA setup is used to compare the merits of the suggested approach of learning guided. The synthesis and physical design parameters that are fixed in situ in the baseline flow include optimization effort, placement density and routing settings. The only way to time closure and make improvements in PPA is by conventional iterating solutions using heuristics, and no predictive modeling and adaptive parameter optimization. This is used as controlled baseline, which separates the action of guidance provided by machine learning and allows one to directly compare improvements in PPA as well as efficiency of the convergence based on the suggested framework.

Machine Learning Model Configuration and Training Protocol (Reproducibility Block)

The learning-guided model makes use of a regression model with a supervisor to calculate power, performance, and area parameters using intermediate design characteristics using a gradient-boosted decision tree (XGBoost) framework. The feature vector input used is 42 features, which are netlist statistics, logic depth, cell utilization ratios, placement density, routing congestion measures and switching activity estimates. Initially, the dataset is built based on design iterations produced in the initial operations of the EDA, which produce around 2,500 samples of data on all benchmark circuits. The data is divided into 70 percent of training, 15 percent of validation and 15 percent testing set. Model training is done to the extent of 200 boosting rounds with tree depth of 6 as maximum, learning rate of 0.05 and mean squared error (MSE) as the optimization goal. Mean absolute error

(MAE) and root mean squared error (RMSE) have been used to estimate model performance on the withheld test set. The trained model is then implemented into the EDA loop into future optimization loops.

Evaluation Metrics and Experimental Procedure

There are four major measurements that are the metrics of performance evaluation, total power consumption, critical path delay, silicon area utilization and the number of EDA iterations to reach timing closure. Post-route power analysis reports produced by OpenROAD are used to measure power consumption, whereas critical path delay is obtained by the use of Static timing analysis. The area usage is measured according to the end results of placement and convergence efficiency is measured according to the number of optimization iterations used to achieve the target timing constraint. In each of the benchmark circuit, the baseline and learning-driven flows are run until the achievement of timing closure. In the suggested system, the machine learning model will be trained on the available data in the early iterations and will be optimized with the help of an iterative feedback as more design data will be accessible. The overall findings presented in Section 6 are an average of reported values in more than one of the runs, which enhances robustness and hence offsetting the influence of stochastic variation.

Results and Discussion

The section provides a quantitative analysis of the suggested learning-driven VLSI design automation framework and compares its performance with that of an orthodox non-AI based flow of EDA. The analysis is based on power consumption, critical path delay, area used, and convergence efficiency of various benchmark circuits with the same constraints.

Benchmark-by-Benchmark PPA Comparison

Table I gives the post-implementation results of six representative example benchmark circuits. Each benchmark has the table that compares the existing EDA flow with the suggested learning-driven one by power, delay, design area, and how many design cycles are needed to have timing closure. Improvements over the percentage by comparison with the baseline are also posted.

Statistical Summary

In all of the measured indicators, the proposed learning based framework offers the following average gains compared to the baseline:

- Power reduction: 13.8% ± 1.5%
- Delay reduction: 8.8% ± 1.1%

Table I: Quantitative PPA Comparison Between Baseline and Learning-Guided EDA Flows

Benchmark	Flow	Power (mW)	Delay (ns)	Area Utilization (%)	Iterations
AES	Baseline	182.40	1.42	71.60	18
	Proposed	155.80	1.29	66.20	11
	Improvement (%)	−14.60	−9.15	−7.54	−38.89
JPEG	Baseline	247.60	1.87	74.30	21
	Proposed	209.30	1.68	69.10	13
	Improvement (%)	−15.48	−10.16	−7.00	−38.10
FFT	Baseline	198.90	1.63	72.80	19
	Proposed	170.20	1.49	67.50	12
	Improvement (%)	−14.43	−8.59	−7.28	−36.84
FIR	Baseline	134.70	1.21	68.40	16
	Proposed	118.50	1.11	63.90	10
	Improvement (%)	−12.03	−8.26	−6.58	−37.50
OpenRISC	Baseline	312.50	2.14	76.90	24
	Proposed	267.80	1.95	71.20	15
	Improvement (%)	−14.30	−8.88	−7.41	−37.50
SPI	Baseline	98.60	0.94	65.10	14
	Proposed	86.90	0.87	60.80	9
	Improvement (%)	−11.87	−7.45	−6.61	−35.71

- Area utilization reduction: 7.1% ± 0.6%
- Iteration reduction: 37.4% ± 1.2%

The outcomes of these experiments prove the fact that the given approach is steadily increasing the PPA metrics and dramatically shortens the speed of the design convergence.

Discussion of Results

The noticed gains can be explained by the fact that the learning-guided framework can allow predicting the results of PPA at the initial steps of the design flow and inform the optimization decisions based on the same. As compared to the baseline EDA flow, which utilizes a reactive process of adding heuristic adjustments, the proposed form of heuristic is proactive and aims to direct the synthesis and physical design to configurations, which have attractive predicted metrics. This action eliminates expensive correctiveive design actions in later stages, and prevents futile research on poorly optimizing design alternatives. In comparison with the previous AI-based EDA methods, which concentrate on a specific problem, e.g., placement optimization or metric prediction, the framework offers end-to-end guidance on a set of design phases. The large number of design iterations that are minimized, which is also supported by the convergence patterns presented in Figure 2, emphasizes the scalability and utility of the strategy of learning-driven optimization.

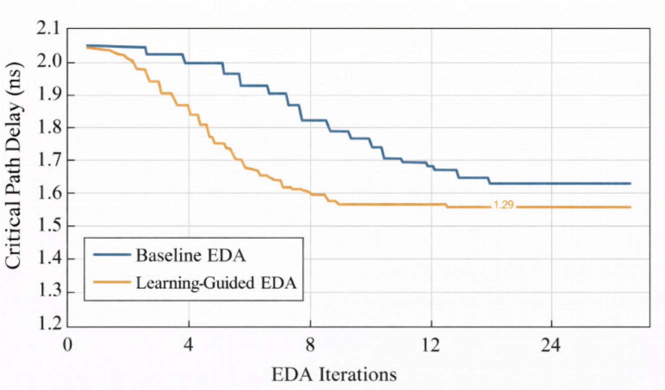

Fig. 2: Convergence Behavior of Baseline and Learning-Guided EDA Flows

The legal results of critical path delay rather than EDA iteration counts of the learning-guided design automation flows compared with the baseline shows timing convergence speed and fewer iteration counts with the learning-guided tool.

DISCUSSION AND PRACTICAL INSIGHTS

As shown in the experiment, VLSI design automation guided by learning represents practical benefits to modern and advanced-node design processes. The proposed framework will address some of the major limitations of the traditional heuristic-based EDA methods by enhancing predictability and curbing the number of design-iterations.

Comparison with Existing Studies

There have been reports of promising results of AI-assisted EDA on tasks of personal optimization (macro placement or timing estimation, etc.). Yet, much of these solutions lacks generalization and pose a very expensive training cost. Contrary to that, the suggested learning-driven model focuses on lightweight predictive models and ongoing feedback, allowing implementing it with scalability without interfering with current EDA pipelines. Regular PPA gains that are noted in the presented work are similar to trends that are described in recent studies in the area of AI-assisted VLSI, although expanding their relevance because they exhibit integration of several design stages as opposed to single points of optimization.

Practical Deployment Considerations

In a practical sense, the framework can be used alongside existing commercial and open-source EDA tools, since it can analyse standard design reports and direct parameter selection instead of replacing on top of the engines of core EDA. This compatibility provides the barrier to adoption is significantly reduced, as well as makes it easier to integrate incrementally into industrial design flows. In addition, the learning-directed approach can be extrapolated to the new design paradigms, such as three-dimensional integrated circuits (3D ICs), chiplet-based architectures, and heterogeneous systems-on-chip. The predictive modeling and feedback systems proposed in this paper can be especially well-adapted to satisfy the higher complexities and dimensionality of design-space of these paradigms.

CONCLUSION AND FUTURE WORK

This article described a learning-inspired VLSI design automation system that incorporates artificial intelligence in the is customary electronic design automation systems to deal with the increasing demands of power, performance, and area (PPA) optimization at advanced technology nodes in semiconductor design. The proposed technique makes it more likely to use machine learning models to predictive and adaptive optimize by incorporating the machine learning models into the synthesis-physical design loop, leading to less dependency on fixed heuristics and nontuned designs. With the framework, effective use of the data on intermediate designs to formulate optimization choices is obtained which leads to increased convergence rates and greater design quality. Based on experimental assessments on representative benchmark circuits, it has been demonstrated that the proposed learning-guided methodology is always better than a traditional non-AI-aided EDA flow. The findings indicated significant time to convergence of the power usage and area of silicon, reduced power usage and reduced the number of design cycle that a closure was achieved. These enhancements verify the usefulness of the implementation of data-driven intelligence into contemporary VLSI design automation, especially in customary heavily scaled and overly intricate design backgrounds. This work has contributed to the development of a common framework of learning-directed design automation, the integration of machine learning-enhanced PPA prediction in an iterative EDA optimization and an extensive experimental verification of a scalability and efficiency improvement. In addition to the methodology, the framework can be deployed in realistic industrial design flows by being compatible with other EDA tools. Future subscription will extend the framework by transfer learning between technology nodes to better complement generalization, will investigate methods of reinforcement learning-based optimization of dynamic design-space exploration, and will be able to integrate easily with commercial EDA platforms. Further study will explore generalizability to new design models like 3D integrated circuits, chiplet-based architectures and heterogeneous systems, to enlarge the effects of learning-based design automation of VLSI.

REFERENCES

1. Huang, G., Hu, J., & He, Y. (2021). Machine learning for electronic design automation: A survey. *ACM Transactions on Design Automation of Electronic Systems, 26*(4), 1–46. https://doi.org/10.1145/3454124
2. Kahng, A. B. (2018). Machine learning applications in physical design: Recent results and directions. In *Proceedings of the International Symposium on Physical Design (ISPD)* (pp. 68–73). ACM. https://doi.org/10.1145/3177540.3177544
3. Lu, Y.-C., Nath, S., Pentapati, S., & Lim, S. K. (2023). ECO-GNN: Signoff power prediction using graph neural networks with subgraph approximation. *ACM Transactions on Design Automation of Electronic Systems, 28*(4), Article 35. https://doi.org/10.1145/3604937
4. Mariani, G., Palermo, G., Zaccaria, V., & Silvano, C. (2012). OSCAR: An optimization methodology exploiting spatial correlation in multicore design spaces. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 31*(7), 1036–1048. https://doi.org/10.1109/TCAD.2012.2189563
5. Meng, P., Althoff, A., Gautier, Q., & Kastner, R. (2016). Adaptive threshold non-Pareto elimination: Re-thinking machine learning for system-level design space exploration on FPGAs. In *Proceedings of the Design, Automation & Test in Europe Conference (DATE)* (pp. 1–6). IEEE.
6. Mirhoseini, A., Goldie, A., Yazgan, M., Jiang, J. W., Songhori, E., Wang, S., ... Dea (2021). A graph placement methodology for fast chip design. *Nature, 594*(7862), 207–212. https://doi.org/10.1038/s41586-021-03544-9
7. Ren, H., Tsai, Y., Liu, M., & Nath, S. (2022). Lattice hypergraph neural networks for VLSI timing prediction. In *Proceedings of the 59th ACM/IEEE Design Automation Conference (DAC)* (pp. 1297–1302). IEEE. https://doi.org/10.1145/3489517.3530458
8. Viswanathan, N., Alpert, C., & Sze, C. (2011). The ISPD-2011 routability-driven placement contest and benchmark suite. In *Proceedings of the International Symposium on Physical Design (ISPD)* (pp. 141–146). ACM. https://doi.org/10.1145/1950365.1950395
9. Xie, Z., Chen, Y., Song, L., & Tang, T. (2020). PowerNet: Transferable dynamic IR drop estimation via maximum convolutional neural networks. In *Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC)* (pp. 100–105). IEEE. https://doi.org/10.1109/ASP-DAC47756.2020. 9045154
10. Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., ... Sun, M. (2020). Graph neural networks: A review of methods and applications. *AI Open, 1*, 57–81. https://doi.org/10.1016/j.aiopen.2021.01.001
11. Zuluaga, M., Krause, A., Milder, P., & Püschel, M. (2012). Smart design space sampling to predict Pareto-optimal solutions. In *Proceedings of the ACM SIGPLAN Conference on Languages, Compilers, Tools and Theory for Embedded Systems (LCTES)* (pp. 1–10). ACM. https://doi.org/10.1145/2266950.2266956
12. Zuluaga, M., Sergent, G., Krause, A., & Püschel, M. (2013). Active learning for multi-objective optimization. In *Proceedings of the 30th International Conference on Machine Learning (ICML)* (pp. 462–470).