Survey of Machine Learning for Electronic Design Automation

Kevin Immanuel Gubbi
kgubbi@ucdavis.edu
University of California, Davis
Davis, CA, USA

Sayed Arash Beheshti-Shirazi
sbehesht@masonlive.gmu.edu
George Mason University
Fairfax, VA, USA

Tyler Sheaves
tsheaves@ucdavis.edu
University of California, Davis
Davis, CA, USA

Soheil Salehi
ssalehi@ucdavis.edu
University of California, Davis
Davis, CA, USA

Sai Manoj PD
spudukot@gmu.edu
George Mason University
Fairfax, VA, USA

Setareh Rafatirad
srafatirad@ucdavis.edu
University of California, Davis
Davis, CA, USA

Avesta Sasan
asasan@ucdavis.edu
University of California, Davis
Davis, CA, USA

Houman Homayoun
hhomayoun@ucdavis.edu
University of California, Davis
Davis, CA, USA

ABSTRACT

An increase in demand for semiconductor ICs, recent advancements in machine learning, and the slowing down of Moore's law have all contributed to the increased interest in using Machine Learning (ML) to enhance Electronic Design Automation (EDA) and Computer-Aided Design (CAD) tools and processes. This paper provides a comprehensive survey of available EDA and CAD tools, methods, processes, and techniques for Integrated Circuits (ICs) that use machine learning algorithms. The ML-based EDA/CAD tools are classified based on the IC design steps. They are utilized in Synthesis, Physical Design (Floorplanning, Placement, Clock Tree Synthesis, Routing), IR drop analysis, Static Timing Analysis (STA), Design for Test (DFT), Power Delivery Network analysis, and Sign-off. State-of-the-art ML-based VLSI-CAD tools, current trends, and future perspectives of ML in VLSI-CAD are also discussed.

CCS CONCEPTS

• **Hardware → Design databases for EDA**.

KEYWORDS

Electronic Design Automation (EDA), Computer-Aided Design, Machine learning, ASIC design flow

ACM Reference Format:
Kevin Immanuel Gubbi, Sayed Arash Beheshti-Shirazi, Tyler Sheaves, Soheil Salehi, Sai Manoj PD, Setareh Rafatirad, Avesta Sasan, and Houman Homayoun. 2022. Survey of Machine Learning for Electronic Design Automation. In *Proceedings of the Great Lakes Symposium on VLSI 2022 (GLSVLSI '22), June 6–8, 2022, Irvine, CA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3526241.3530834

GLSVLSI '22, June 6–8, 2022, Irvine, CA, USA

ACM ISBN 978-1-4503-9322-5/22/06…$15.00
https://doi.org/10.1145/3526241.3530834

1 INTRODUCTION

The Electronic Design and Automation (EDA) tools for Integrated Circuits (ICs) are crucial enablers for the semiconductor industry. Advances in EDA tools have enabled the integration of increasing number of transistors on a single semiconductor die. With groundbreaking innovations in IC design and integration, some chips have up to 1.2 trillion transistors [41]. The Application Specific Integrated Circuit (ASIC) design flow is complex and time consuming. A typical IC goes through several stages in the design process. Chip architects work together to define specifications for the IC that is to be designed. Once the constraints are set, the behavioural Register Transfer Language (RTL) code is completed, and the desired foundry is selected, the design is sent through the logic synthesis process, after which the gate-level netlist is obtained. The netlist is then sent to the Physical Design stages where the floorplan for the design is developed. This includes the design of the Power Delivery Network (PDN) and macro placement. Then the placement of standard cells and other logic blocks is done. Once the placement is finalized, the clock tree is built. This step is known as the Clock Tree Synthesis (CTS). Routing the signals is the next process in the design flow. Before the final netlist is sent to the foundry for fabrication, it will go through the sign-off process where the netlist is rigorously checked for design rule violations, layout vs. schematic violations, and timing violations. If there are violations pending to be fixed, Engineering Change Order (ECO) are introduced to fix the violations with minimal changes to the netlist. Static Timing Analysis (STA) is done every time a netlist is altered to make sure there are no new violations. Although this is a brief summary of the design process, it is evident that the IC design flow is complex and time consuming. This gets worse with increase in the design size. This challenge provides an opportunity to explore the benefits of Machine Learning (ML) for optimum design automation given the complexity of the IC design flow and the data availability. Moreover, ML methods have been used for various applications like healthcare, banking, military, scientific computing, automobiles, and consumer electronics. ML methods are classified based on the source of data that governs the learning mechanism. Herein, we have broadly classified the ML methods into four categories: (1) unsupervised

learning, (2) supervised learning, (3) semi-supervised, and (4) reinforcement learning. When only the input data is available and labels are required to be generated in order to distinguish between the input samples, the learning method is known as unsupervised learning. In supervised learning, the input data and corresponding output data samples are available to the algorithm with the structure or labels. In semi-supervised learning, only some parts of the input data samples have corresponding output sample pairs.

The Reinforcement Learning (RL) solutions are independent of data samples and are based on the interaction of an agent with an environment. The agent exploits and explores opportunities to achieve new Pareto-frontier optimal spaces beyond the optimization space covered by heuristic algorithms. A significant volume of data is used during the training and learning process. Additionally, the information that we employ in the process should be of impartial consistency and high quality, which might need the generation of more data, and as a result, more time, resources, and power are required for a better quality of results. The other drawback is that dependable resources are necessary when ML algorithms show errors and complexity. It is vital to ensure that the algorithms utilized, produce the desired output. To achieve this outcome, an accurate learning algorithm with high performance is needed. The selection and availability of such a precise algorithm is also a challenge. ML solutions still require significant improvements in algorithms and the software that performs the analysis. Moreover, due to the large volume of data, error susceptibility is high, which needs to be addressed. Drawbacks related to specific ML algorithms such as nonlinearity, sampling errors, overfitting, noisy datasets, incomprehensible datasets, low performance, complex and expensive computation, and insufficient runtime memory, should also be considered. The ever-increasing need for IC chips for various applications and the need to speed up the design and manufacturing process is an important task and needs collaborative effort. ML has proved to be an excellent tool to increase the efficiency and productivity of the IC design and manufacturing processes. This manuscript aims to provide a comprehensive survey of the landscape of ML-assisted EDA tools. A taxonomy of ML methods is shown in Figure 1.

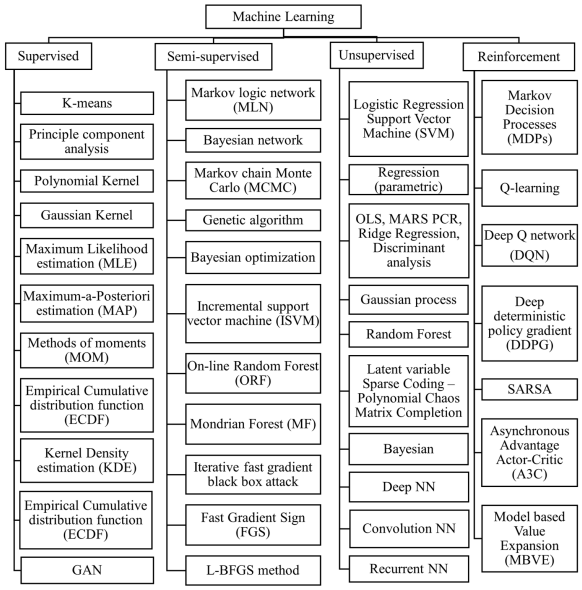

Figure 1: Classification of Machine Learning methods

2 MACHINE LEARNING FOR VLSI-CAD

Over the last few decades, there has been substantial imporvements in the semiconductors industry, mainly due to the technological progress in IC design and development to improve performance and reduce area, power, and cost. These technological advancements have allowed the integration of nearly billions of transistors on a single silicon die, commonly categorized as a Very-Large-Scale Integration (VLSI). The VLSI design flow involves several steps. ML based EDA tools used have been discussed below.

2.1 Synthesis

Synthesis is a fundamental process in the IC design flow as it enables designers to realize a logical circuit into a physical layout. All parameters, including area, timing, and power, can be reported and checked by the design team beforehand. Necessary changes can be made before the actual fabrication process, thus saving both time and cost. Several industry-standard tools such as Design Compiler from Synopsys, RTL compiler from Cadence, and other open-source tools, have been used to convert logical netlist into physical layout for fabrication and validation. This Section discusses recent works utilizing ML to advance the Computer-Aided Design (CAD) process. Authors in [11] cast logic optimization as a deterministic Markov Decision Process (MDP). They take advantage of recent advances in deep RL to build a system that learns how to navigate this process. Their design has several desirable properties. It is autonomous because it learns automatically and does not require human intervention. It generalizes to large functions after training on small examples. Additionally, it intrinsically supports both single- and multi-output tasks without the need to handle special cases. In [8], an extensive collection of C-to-FPGA results is built from a set of High-Level Synthesis (HLS) applications that are diverse and realistic to estimate post-implementation metrics. These features and data are then leveraged to train and compare a range of ML models to effectively and efficiently close the accuracy gap. The authors in [19] present some experimental results for application-specific many-core system design optimization and dynamic power management to demonstrate the efficacy of these methods over traditional EDA approaches.

In [14], a novel RL-based methodology is proposed that navigates the optimization space without human intervention. The training of an Advantage Actor-Critic (A2C) agent that seeks to minimize area subject to a timing constraint is demonstrated. Furthermore, the authors show that designs can be optimized autonomously without a human intervention using the proposed methodology. Authors in [31] propose LSOracle, which is a novel automated mixed logic synthesis framework that is the first to exploit state-of-the-art And-Inverter Graph (AIG) and Majority-Inverter Graph (MIG) logic optimizers. The proposed method relies on a Deep Neural Network (DNN) to automatically decide which optimizer should handle different portions of the circuit. To do so, LSOracle applies k-way partitioning to split a Directed Acyclic Graph (DAG) into multiple partitions and chooses the best-fit optimizer. In [47], the authors present an autonomous framework that artificially produces design-specific synthesis flows without human guidance and baseline flows,

using a Convolutional Neural Network (CNN). The demonstrations are made by successfully designing logic synthesis flows of three large-scaled designs. The Authors in [20] propose a transfer learning approach that reuses the knowledge obtained from previously explored design spaces in exploring a new target design space. The authors develop a novel neural network model for mixed-sharing multi-domain transfer learning. In [42] an end-to-end framework called IRONMAN is proposed. The main goal is to enable a flexible and automated Design Space Exploration (DSE), which can provide optimized solutions under user-specified constraints or Pareto trade-offs among different objectives, such as resource types, area, and latency. The IRONMAN framework consists of three main components: GPP (a graph-neural-network-based performance predictor), RLMD (an RL-based DSE engine that explores the optimized resource allocation strategy), and CT (a code transformer that assists RLMD and GPP by extracting data flow graphs from original HLS C/C++). Authors in [10] have presented MLSBench, a collection of around 5000 synthesizable designs written in C and C++. They provide a methodology to generate designs with variations of a design, which creates a potential for creating new designs and enlarging the database in the future. This is followed by analysis and validation that the generated designs are different. The authors in [9] propose MAFIA, a tool to compile ML inference on small form-factor FPGAs for IoT applications. MAFIA provides native support for linear algebra operations and can express a variety of ML algorithms, including state-of-the-art models. In [6] the authors develop hls4ml, which is an open-source hardware-software codesign flow. This is used to interpret and translate ML algorithms for ASIC and FPGA implementations. The paper introduces readers to the essential features of hls4ml, which includes network optimization techniques like pruning and quantization-aware training, which can be integrated into the device implementations.

2.2 Physical Design

Once again, attributing to the rise in the need for semiconductor ICs, generating a layout from a netlist is essential in the IC design process. Physical design is converting a logical netlist or RTL into a physical layout. Most fabrication processes require design houses to use certain design libraries specific to their fabrication process. Generating these design layouts from the design netlist and other design files is complex and time-consuming. Authors in [4] review available opportunities for ML with a focus on IC physical implementation. They give examples like (1) removing unnecessary design and modeling margins through correlation mechanisms, (2) achieving faster design convergence through predictors of low downstream outcomes that comprehend both tools and design instances, and (3) corollaries such as optimizing the usage of design resources licenses and available schedule. Some open challenges for ML in IC physical design are also discussed herein.

Floor planning and Placement: Chip placement and floor planning are two important processes in the IC design flow. Finding the optimal floor plan and placement of a design is considered one of the most time-consuming and complex processes. Modern IC designs have numerous smaller IPs, macros, and modules that require multiple iterations of placement to figure out the optimal position for each instance. Most ICs do not have the most optimal placement simply because of the high number of placement permutations possible, even for small designs. This is exacerbated by larger designs, and a solution to find the best placement while using reasonable resources is a challenge. Multiple research teams have been invested in finding solutions to improve the placement. In [16], the authors first train a model to predict the number of Design Rule Check (DRC) violations for the current macro placement. DRCs are design rules that make sure the netlist layout is compliant with the foundry-specific tape-out requirements. To generate macro placements with fewer DRCs, authors in [16] use the predictions obtained from their trained ML model. This is used as the evaluation function in the simulated annealing process. While this work represents an interesting direction, the results shared are based on netlists with less than six macros, which are not realistic compared to modern IC netlists. Moreover, their approach does not include any optimization during the place and route steps. Due to the optimization, the placement and routing can change dramatically, and the actual DRC will change accordingly, invalidating the model prediction. In addition, although adhering to the DRC criteria is necessary, the primary objective of macro placement is to optimize for wire length and timing like Worst Negative Slack (WNS) and Total Negative Slack (TNS), power, and area, and their work does not consider these metrics.

Recently, authors in [29] have presented a learning-based approach for chip placement, and unlike other prior methods, their approach has the ability to learn from past experience and progressively improve over time. As the model is trained over a greater number of chip blocks, it becomes better at generating optimized placements for previously unseen chip macros, modules, and blocks. The authors take chip placement as an RL problem and train an agent to place the nodes of a chip netlist onto a chip layout. Representation learning is used in the supervised task of predicting placement quality in order to enhance their RL policy. The authors were able to enable feature-rich embeddings of the input netlists by designing a neural network architecture that could accurately predict reward across the wide variety of input netlists and their placements. This architecture is used as the encoder of their RL policy and value networks to enable transfer learning. Mirhoseini and colleagues, in [28], use an RL-based graph placement method. As shown in Fig. 2, an RL agent is used to place macros one after the other, and once all macros are placed, the standard cell placement is done using a force-directed method. The method learns from past experiences, which in turn improves the speed and quality of producing solutions for new instances of the problem.

Clock Tree Synthesis (CTS): CTS is a physical design step in implementing the clock network. Historically, the EDA tools were designed to build balanced clock trees by minimizing the clock skew, giving each register-to-register timing path equal time. The problem with the zero skew clock tree is that all registers launched simultaneously, resulting in a surge of demanded current from the battery on the active edge of the clock. In this section, recent works utilizing ML to advance the CTS process are discussed. In [1] the author presents a fully automated RL-based solution for reducing the peak current. The agent modifies the clock arrival times for each of the registers to maximize the distribution of clock arrivals. Using RL allows the agent to explore optimization opportunities beyond the heuristic algorithm. The work in [39] employs a genetic algorithm to optimize the clock-skew by limiting the maximum number of

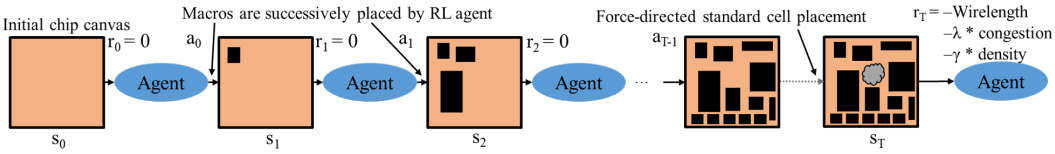

Figure 2: Overview of RL-based chip floorplanning method and training scheme in [28] (reproduced from [28])

clock drivers introduced and utilizes clustering techniques. In [26] the authors utilize a mixed technique to achieve CTS optimization. The paper employs a Generative Adversarial Network (GAN) augmented by RL. It is worth noting that the traditional GAN includes a generator and discriminator. In this paper, RL uses a pre-trained regression model as a supervisor of the generator. The work in [21] estimates the clock tree elements such as how many buffers to be used or the wire-loads utilizing Artificial Neural Network (ANN). During CTS, the proposed technique uses ANN to determine the number of buffers to be added or removed to achieve the designated target clock skew. The result maximizes input transition times for clock buffers and sinks. The work presented by [34] suggests a two-tier hybrid approach to optimization. The paper discusses the employment of supervised ML techniques such as the Support Vector Machine (SVM) algorithm to estimate clock buffer and wire sizing. The report focused on providing an alternative to the expensive circuit-level simulations and reducing clock skew without significantly increasing power dissipation. The paper [30] utilizes a Convolutions Neural Network (CNN) augmented and enhanced by K-Means clustering and Linear Programming optimization to estimate different parameters of CTS. The paper focuses on decreasing the power consumption of the clock network by reducing the clock sinks along various data paths.

Routing: Routing has been a critical and complex challenge for IC designers. Due to the huge number of routing possibilities for each design, the need to optimize the EDA tools and routing algorithm is paramount especially with larger and more complex designs. ML has been used to improve routing quality and time. Authors in [45] provide insights into learning-free placement and routing approaches and then provide a detailed review of recent advances in ML for routing and placement. The proposed method in [38] uses a deep learning-based congestion estimation algorithm to improve routing quality. Their routing algorithm extracts appropriate three dimensional features from already placed netlists. The authors also propose a congestion estimator that produces a heatmap to serve as a guide for initial pattern during global routing phase. In [22], a deep RL method is proposed to solve the global routing problem in a simulated environment and an RL agent is used to produce an optimal policy for routing.

2.3 IR drop

The on-chip power delivery network (PDN) is a vital part of any chip, as it determines the quality and reliability of the fabricated IC. Ideally, a grid that is dense and compact is desired. However, a sparse PDN leaves more room for a clock, signal, and Engineering Change Order (ECO) routing. Most complex designs need multiple iterations of PDN design before finalizing the final PDN layout. Authors in [13] extract relevant SOC floorplan and PDN features using superposition and partitioning techniques. An ML model is then used to predict the updated static IR drop for each power node by a series of SOC floorplan alterations. This is done without the need to run a golden IR drop tool. The manuscript also shows significant improvement over an industry-leading golden IR drop sign-off tool with negligible error rates. Similarly, authors in [4] present a design flow to generate a PDN with negligible overhead for standard cell routing while still meeting the IR drop and EM constraints for a given placement. The ML model used in [4] predicts the total wire length of the global route associated with a given PDN configuration to speed up the search process. Calculating the IR drop after each ECO, authors in [7] use timing, power, and physical features collected before ECO to predict the IR drop of a design after ECO. Regional models for cell instances near IR drop violations are built to improve prediction accuracy and training time. Results in [7] show that IR drop prediction for a design with 100,000 cell instances can be predicted within 2 minutes. In [24], the authors propose an ML technique to build IR drop prediction models based on circuits before ECO revision. Once the ECO revision is done, these prediction models are reused to predict the IR drop of the ECO-revised circuit. The work in [43] provides a review of the process in IR drop estimation techniques that use ML algorithms. Authors in [44] propose PowerNet, a CNN-based dynamic IR drop estimation technique that can handle both vector-based and vectorless IR analysis. The CNN model used in PowerNet is general and transferable to different designs. The authors in [23] propose an automatic flow to alter IR drop violations by ECO, which provides cell movement and downsize solutions. An ML algorithm is used to predict IR drop in order to prevent over-fixing. A novel multi-round bipartite matching is used to optimize the resources used during the ECO flow. MAVIREC [5] is a tool that uses ML techniques like three dimension convolutions and regression-like layers to suggest a larger subset of worst-case test patterns in order to improve test coverage and accurately predict the IR drop. Another method to predict IR drop of an IC layout is presented in [15] where XGBoost, an ML technique, is used to make dynamic IR drop predictions, which can be applied to vector-based and vector-less IR drop analysis, simultaneously. In [15], the authors use a correlation coefficient to characterize the symmetry of predicted data and golden data.

2.4 Static Timing Analysis

Static Timing Analysis (STA) is a process that takes several iterations. Each iteration may take several hours for larger designs. STA is a technique to check if a design satisfies the timing rules required for the end product to function correctly. The input to an STA tool is the routed netlist, clock definitions (or clock frequency), and external environment definitions. The STA validates whether the design could operate at the rated clock frequency without any timing violations. Some of the basic timing violations are setup violation and hold violation. Almost always, the initial layout will have multiple timing violations, which the STA tool resolves by an iterative process of buffer insertion, signal and clock-tree re-routing, and layout alterations. Accurate timing closure is an important step in the IC design flow. STA will be done multiple times after each

design alteration to check for potential timing violations that may have been inducted into the design.

In [2], as shown in Figure 3, the authors first observe that aging can be thought of as a type of correlated dynamic On-Chip Variations (OCV) and identify the problem introduced by such OCV. In particular, they take the Negative Bias Temperature Instability (NBTI) as an example dynamic OCV mechanism and then propose a Learning-based STA (LSTA) library to predict the timing of gates by capturing the correlation between our designed predictors. A linear regressor, support vector regression, and a non-linear method, random forest, were used to create the prediction model in their experiment. Authors in [35] discuss the timing closure problem explaining the root cause of its difficulty. They also explain traditional techniques that address the timing closure problem. Furthermore, new challenges that appear at advanced process nodes are highlighted, and solutions to these problems are discussed.

Authors in [17] propose a ML model based on bigrams of path stages to predict expensive Path Base Analysis (PBA) results from relatively inexpensive Graph Base Analysis (GBA) results. They identify electrical and structural features of the circuit that affect PBA-GBA divergence with respect to endpoint arrival times and use GBA and PBA analysis of a given test case along with artificially generated timing paths. A pyramid framework is proposed in [27] that estimates the optimal resource and performance utilization of a High Level Synthesis (HLS) design using ML. For this purpose, they first create a database of C-to-FPGA results from a diverse set of benchmarks. To find the achievable maximum clock frequency, Minerva, which is an automated hardware optimization tool, is used. Minerva determines the close-to-optimal settings of tools using STA along with a heuristic algorithm and targets either optimal throughput or throughput-to-area. A multivariate linear regression based data-driven approach is investigated by authors in [18] to predict the timing analysis results at observed corners. The authors use a simple backward step-wise selection strategy to choose which corners to observe and which to predict.

In [32], the impact of Multiple-Input Switching (MIS) is modeled by deriving a corrective measure that should be applied to the conventional Single-Input Switching (SIS) delay under different conditions. They call this corrective measure MIS-SIS Difference (MSD). In this work, they have evaluated polynomial regressions, support vector regression, and ANNs to model MSD. Additionally, they integrate the ANN-based MSD model into existing timing libraries and employ them in carrying out MIS-aware timing analysis. In [36], they present novel Deep Neural Network (DNN) based operations which can accurately approximate the signal arrival-time's distributions with linear-time complexity. The various DNN architectures have been used to implement both the maximum and the convolution operations using proper training dataset. An ML-based automatic timing closure solution for relative timed circuits is presented in [37]. The ML implementation is expected to speed up the process by learning from the features during each iteration, minimizing the overall run-time to timing close a design. In [33], the authors use deep learning non-linear autoencoders to compress voltage and current waveforms and then compare them with the singular component analysis approach. In [25], they propose an ML-based approach to estimate pin-to-pin delays for RTL combinational circuits. To gain accuracy, they combine slew and delay estimation.

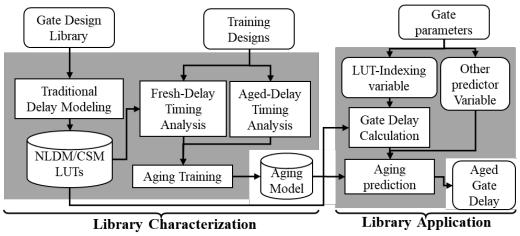

Figure 3: Overview of library characterization and application processes with learning-based training in [2] (reproduced from [2])

To that end, a training set is built using features of components generated by a model-driven hardware generator framework. Ground truth labels for delays, slews, and their inter-dependencies are extracted using open-source tools for logic synthesis and STA. In [12], they propose a stage-based delay model based on an ML technique with a customized loss function to rapidly generate predicted PBA timing results from the pessimistic GBA timing report considering the asymmetric loss. The model could also enable the designers to identify the false violation path in the GBA report within less time, to reduce the margin in the post-route optimization phase. Authors in [40] provide a latency analysis of the inference path of their proposed hardware design which is intended for the learning datapath of the Tsetlin machine algorithm. They use a combination of asynchronous design techniques like petri nets, signal transition graphs, bundled data, and dual-rail method, in order to generate a low energy hardware, which can be used for pervasive Artificial Intelligence (AI) applications. In [3], a learning-based timing prediction framework is proposed to predict path delays across wide voltage regions by Light Gradient Boosting Machine (LightGBM) with data augmentation strategies including Conditional Generative Adversarial Networks (CTGAN) and Synthetic Minority Oversampling Technique for Regression (SMOTER), which generate realistic synthetic data of circuit delays to improve prediction precision and reduce data sampling effort. In [46], an efficient and accurate pre-routing path delay prediction framework is proposed by using a transformer and residual model. Timing and physical information at the placement stage is extracted as sequence features while the residual path delay is modeled to calibrate the mismatch between the pre- and post-routing path delays.

3 DISCUSSION AND CONCLUSION

The use of ML methods for CAD has become an active area of research. Due to the ever increasing complexity and scale of variables in an IC design process, there is an increasing need for efficient ML-assisted EDA tools. Moreover, with the advent of emerging hardware security threats, there is a growing need for EDA tools to incorporate security countermeasures and mitigation techniques into the IC design flow. Although there has been significant progress in the development of tools and methods for hardware security, the need for efficient, easy to integrate, and scalable EDA tools is growing. With growing threats like IC counterfeiting, overproduction, reverse engineering, hardware trojan insertion, and side-channel attacks, the need to implement security mitigations and countermeasures into the IC design process is one to be addressed. In summary, this paper provides an insight on the advancements of using ML algorithms for EDA. The survey is categorized by the different IC design flow stages and the state-of-the-art ML-assisted

methods are discussed under the relevant sections. Finally, future perspectives of ML in EDA are given, and opportunities in hardware security automation are discussed.

ACKNOWLEDGMENT

This work was supported in part by the National Science Foundation through Computing Research Association for CIFellows #2030859.

REFERENCES

[1] S. A. Beheshti-Shirazi, A. Vakil, S. Manoj, et al. 2021. A Reinforced Learning Solution for Clock Skew Engineering to Reduce Peak Current and IR Drop. In *Proceedings of the 2021 on Great Lakes Symposium on VLSI*. 181–187.

[2] S. Bian, M. Hiromoto, M. Shintani, et al. 2017. LSTA: Learning-based static timing analysis for high-dimensional correlated on-chip variations. In *2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC)*. IEEE, 1–6.

[3] P. Cao, W. Bao, K. Wang, et al. 2021. A Timing Prediction Framework for Wide Voltage Design with Data Augmentation Strategy. In *Proceedings of the 26th Asia and South Pacific Design Automation Conference*. 291–296.

[4] W.-H. Chang, C.-H. Lin, S.-P. Mu, et al. 2017. Generating Routing-Driven Power Distribution Networks With Machine-Learning Technique. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 36, 8 (2017), 1237–1250.

[5] V. A. Chhabria, Y. Zhang, H. Ren, et al. 2021. MAVIREC: ML-Aided Vectored IR-Drop Estimation and Classification. In *2021 Design, Automation Test in Europe Conference Exhibition (DATE)*. 1825–1828.

[6] F. Fahim, B. Hawks, C. Herwig, et al. 2021. hls4ml: An open-source codesign workflow to empower scientific low-power machine learning devices. *arXiv preprint arXiv:2103.05579* (2021).

[7] Y.-C. Fang, H.-Y. Lin, M.-Y. Su, et al. 2018. Machine-Learning-Based Dynamic IR Drop Prediction for ECO. In *Proceedings of the International Conference on Computer-Aided Design (ICCAD '18)*. Association for Computing Machinery, New York, NY, USA, Article 17, 7 pages.

[8] M. Ferianc, H. Fan, R. S. W. Chu, et al. 2020. Improving Performance Estimation for FPGA-Based Accelerators for Convolutional Neural Networks. In *Applied Reconfigurable Computing. Architectures, Tools, and Applications*, Fernando Rincón, Jesús Barba, Hayden K. H. So, Pedro Diniz, and Julián Caba (Eds.). Springer International Publishing, Cham, 3–13.

[9] N. P. Ghanathe, V. Seshadri, R. Sharma, et al. 2021. MAFIA: Machine Learning Acceleration on FPGAs for IoT Applications. In *2021 31st International Conference on Field-Programmable Logic and Applications (FPL)*. 347–354.

[10] P. Goswami, M. Shahshahani, and D. Bhatia. 2020. MLSBench: A Synthesizable Dataset of HLS Designs to Support ML Based Design Flows. In *Proceedings of the 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA '20)*. Association for Computing Machinery, New York, NY, USA, 312.

[11] W. Haaswijk, E. Collins, B. Seguin, et al. 2018. Deep Learning for Logic Optimization Algorithms. In *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*. 1–4.

[12] A. Han, Z. Zhao, C. Feng, et al. 2021. Stage-based Path Delay Prediction with Customized Machine Learning Technique. In *Proceedings of the 2021 5th International Conference on Electronic Information Technology and Computer Engineering*. 926–933.

[13] C.-T. Ho and A. B. Kahng. 2019. IncPIRD: Fast Learning-Based Prediction of Incremental IR Drop. In *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 1–8.

[14] A. Hosny, S. Hashemi, M. Shalan, et al. 2020. DRiLLS: Deep Reinforcement Learning for Logic Synthesis. In *2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC)*. 581–586.

[15] P. Huang, C. Ma, and Z. Wu. 2021. Fast Dynamic IR-Drop Prediction Using Machine Learning in Bulk FinFET Technologies. *Symmetry* 13, 10 (2021).

[16] Y.-H. Huang, Z. Xie, G.-Q. Fang, et al. 2019. Routability-Driven Macro Placement with Embedded CNN-Based Prediction Model. In *2019 Design, Automation Test in Europe Conference Exhibition (DATE)*. 180–185.

[17] A. B. Kahng, U. Mallappa, and L. Saul. 2018. Using machine learning to predict path-based slack from graph-based timing analysis. In *ICCD*. IEEE, 603–612.

[18] A. B. Kahng, U. Mallappa, L. Saul, et al. 2019. " Unobserved Corner" Prediction: Reducing Timing Analysis Effort for Faster Design Convergence in Advanced-Node Design. In *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 168–173.

[19] R. G. Kim, J. R. Doppa, and P. P. Pande. 2018. Machine Learning for Design Space Exploration and Optimization of Manycore Systems. In *Proceedings of the International Conference on Computer-Aided Design (ICCAD '18)*. Association for Computing Machinery, New York, NY, USA, Article 48, 6 pages.

[20] J. Kwon and L. P. Carloni. 2020. Transfer Learning for Design-Space Exploration with High-Level Synthesis. In *2020 ACM/IEEE 2nd Workshop on Machine Learning for CAD (MLCAD)*. 163–168.

[21] Y. Kwon, J. Jung, I. Han, et al. 2018. Transient Clock Power Estimation of Pre-CTS Netlist. In *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*. 1–4.

[22] H. Liao, W. Zhang, X. Dong, et al. 2019. A Deep Reinforcement Learning Approach for Global Routing. *ArXiv* abs/1906.08809 (2019).

[23] H.-Y. Lin, Y.-C. Fang, S.-T. Liu, et al. 2020. Automatic IR-Drop ECO Using Machine Learning. In *2020 IEEE International Test Conference in Asia (ITC-Asia)*. 7–12.

[24] S.-Y. Lin, Y.-C. Fang, Y.-C. Li, et al. 2018. IR drop prediction of ECO-revised circuits using machine learning. In *2018 IEEE 36th VLSI Test Symposium (VTS)*. 1–6.

[25] D. S. Lopera, L. Servadei, V. P. Kasi, et al. 2021. RTL Delay Prediction Using Neural Networks. In *2021 IEEE Nordic Circuits and Systems Conference (NorCAS)*. IEEE, 1–7.

[26] Y.-C. Lu, J. Lee, A. Agnesina, et al. 2019. GAN-CTS: A generative adversarial framework for clock tree prediction and optimization. In *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 1–8.

[27] H. M. Makrani, F. Farahmand, H. Sayadi, et al. 2019. Pyramid: Machine learning framework to estimate the optimal timing and resource usage of a high-level synthesis design. In *2019 29th International Conference on Field Programmable Logic and Applications (FPL)*. IEEE, 397–403.

[28] A. Mirhoseini, A. Goldie, M. Yazgan, et al. 2021. A graph placement methodology for fast chip design. *Nature* 594, 7862 (2021), 207–212.

[29] A. Mirhoseini, A. Goldie, M. Yazgan, et al. 2020. Chip Placement with Deep Reinforcement Learning. *ArXiv* abs/2004.10746 (2020).

[30] S. Nagaria and S. Deb. 2020. Designing of an Optimization Technique for the Prediction of CTS Outcomes using Neural Network. In *2020 IEEE International Symposium on Smart Electronic Systems (iSES) (Formerly iNiS)*. 312–315.

[31] W. L. Neto, M. Austin, S. Temple, et al. 2019. LSOracle: a Logic Synthesis Framework Driven by Artificial Intelligence: Invited Paper. In *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 1–6.

[32] O. S. Ram and S. Saurabh. 2020. Modeling multiple-input switching in timing analysis using machine learning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 40, 4 (2020), 723–734.

[33] W. Raslan and Y. Ismail. [n. d.]. Deep Learning Autoencoder-based Compression for Current Source Model Waveforms. In *2021 28th IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*. IEEE, 1–6.

[34] R. Samanta, J. Hu, and P. Li. 2010. Discrete Buffer and Wire Sizing for Link-Based Non-Tree Clock Networks. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 18, 7 (2010), 1025–1035.

[35] S. Saurabh, H. Shah, and S. Singh. 2018. Timing closure problem: Review of challenges at advanced process nodes and solutions. *IETE Technical Review* (2018).

[36] M. A. Savari and H. Jahanirad. 2020. NN-SSTA: A deep neural network approach for statistical static timing analysis. *Expert Systems with Applications* 149 (2020), 113309.

[37] T. Sharma, S. Kolluru, and K. S. Stevens. 2020. Learning Based Timing Closure on Relative Timed Design. In *IFIP/IEEE International Conference on Very Large Scale Integration-System on a Chip*. Springer, 133–148.

[38] M. Su, H. Ding, S. Weng, et al. 2022. High-Correlation 3D Routability Estimation for Congestion-guided Global Routing. In *2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC)*. 580–585.

[39] P. Vuillod, L. Benini, A. Bogliolo, et al. 1996. Clock-skew optimization for peak current reduction. In *Proceedings of 1996 International Symposium on Low Power Electronics and Design*. IEEE, 265–270.

[40] A. Wheeldon, A. Yakovlev, and R. Shafik. 2021. Self-timed Reinforcement Learning using Tsetlin Machine. In *2021 27th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*. IEEE, 40–47.

[41] A. Wlliams. 2019. LARGEST CHIP EVER HOLDS 1.2 TRILLION TRANSISTORS. Retrieved April 5, 2022 from https://hackaday.com/2019/08/21/largest-chip-ever-holds-1-2-trillion-transistors/

[42] N. Wu, Y. Xie, and C. Hao. 2021. IRONMAN: GNN-assisted Design Space Exploration in High-Level Synthesis via Reinforcement Learning. *Proceedings of the 2021 on Great Lakes Symposium on VLSI* (2021).

[43] Z. Xie, H. Li, X. Xu, et al. 2020. Fast IR Drop Estimation with Machine Learning. In *Proceedings of the 39th International Conference on Computer-Aided Design (ICCAD '20)*. Association for Computing Machinery, New York, NY, USA, Article 13, 8 pages.

[44] Z. Xie, H. Ren, B. Khailany, et al. 2020. PowerNet: Transferable Dynamic IR Drop Estimation via Maximum Convolutional Neural Network. In *2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC)*. 13–18.

[45] J. Yan, X. Lyu, R. Cheng, et al. 2022. Towards Machine Learning for Placement and Routing in Chip Design: a Methodological Overview. *ArXiv* abs/2202.13564 (2022).

[46] T. Yang, G. He, and P. Cao. 2022. Pre-Routing Path Delay Estimation Based on Transformer and Residual Framework. In *2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC)*. 184–189.

[47] C. Yu, H. Xiao, and G. De Micheli. 2018. Developing Synthesis Flows without Human Knowledge. In *Proceedings of the 55th Annual Design Automation Conference (DAC '18)*. Association for Computing Machinery, New York, NY, USA, Article 50, 6 pages.