A PVT-aware Surrogate Modeling Framework for Digital, Analog, and Mixed-Signal VLSI Circuits

A thesis submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy
in
Electronics and Communication Engineering

by

Deepthi Amuru
20163038
deepthi.amuru@research.iiit.ac.in

Advisor: Dr. Zia Abbas

International Institute of Information Technology Hyderabad
500 032, India

June 2024

International Institute of Information Technology Hyderabad
Hyderabad, India

CERTIFICATE

This is to certify that work presented in this thesis proposal titled ***A PVT-aware Surrogate Modeling Framework for Digital, Analog, and Mixed-Signal VLSI Circuits*** by *Deepthi Amuru* has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Advisor: Dr. Zia Abbas

Abstract

The invention of Complementary Metal–Oxide–Semiconductor (CMOS) transistors marked a revolutionary shift in the field of electronics, ushering in the era of semiconductor devices within the Integrated Circuit (IC) industry. Since then, CMOS technology has dominated the realm of microelectronics. The key to advancing ICs lies in transistor scaling, which boosts transistor density, switching speed, and operational frequency, enabling the creation of higher-performing electronic circuits. However, the aggressive down-scaling of CMOS technology has posed challenges for device engineers while opening up new opportunities. As transistor dimensions decrease, the complexity of the semiconductor process increases. As we approach atomic scales, simple scaling reaches its limits. Despite their minute size, devices can encounter various performance issues, including increased leakage, reduced gain, and increased sensitivity to manufacturing process variations. The substantial rise in process variations significantly impacts circuit operation, resulting in variable performance even in transistors of identical size. This, in turn, affects the propagation delay of the circuit, which behaves as a stochastic random variable, making timing-closure techniques more complex and exerting a substantial influence on chip yield. FinFETs, which have superseded CMOS in the nanoscale IC designs, also face performance deviations due to process variations despite demonstrating good resistance to Short Channel Effects (SCE). Surging process variations in the nanometer regime significantly contribute to the degradation of parametric yield.

Under such scenarios, the Process, Voltage, and Temperature (PVT) aware performance estimation of VLSI circuits through traditional Electronic Computer Aided Design (E-CAD) tools is a complex endeavor. These tools often exhibit intricacies, heavily relying on specific circuit architectures and licensing agreements, while demanding significant simulation times proportional to the design complexity. Moreover, conventional tools tend to adhere rigidly to standardized processes and workflows, potentially limiting their efficacy and stifling innovation by confining designers to predetermined methodologies. Furthermore, the traditional approaches employed for such tasks frequently involve manual intervention, introducing time-sensitive and resource-intensive procedures that can contribute to delays in the time-to-market. Additionally, upon receiving simulation results, designers may face challenges in comprehending the underlying functionalities, including identifying the root causes of issues and implementing necessary fixes. This can result in additional time consumption and impede the overall design iteration process.

The objective of this research is to develop a rapid and efficient surrogate modeling framework to precisely predict PVT-aware circuit performance within Very Large Scale Integration (VLSI) circuits,

with a focus on overcoming the limitations of conventional modeling approaches. The framework is designed to be versatile, platform-independent, and capable of modeling a wide range of digital, analog, mixed-signal, and RF circuit applications. To accomplish these objectives, we integrate Artificial Intelligence (AI) and Machine Learning (ML) based surrogate models into the framework to monitor circuit performance under the influence of PVT variations. The framework encompasses several stages, including PVT-aware simulation data generation, the development of a bank of AI/ML models employing supervised learning representing each circuit under consideration, and a high-level intelligent entity responsible for identifying the most effective model. To demonstrate the methodology, our framework incorporates 22 statistically aware standard digital logic circuits and various application-specific datasets for analog circuits, considering the impacts of design, process, and environmental variations across multiple technology nodes. Digital circuit modeling spans high-performance CMOS technology nodes, namely 45nm, 32nm, 22nm, and 16nm, and FinFET technology nodes, such as 16nm, 10nm, and 7nm. Analog, mixed-signal, and RF circuit modeling encompasses designs ranging from 180nm to 65nm and 28nm technology nodes.

Moreover, the framework utilizes an automated methodology for estimating the performances of complex digital circuits, grounded in statistical variations within standard cells and facilitated by the most accurate AI/ML model. This methodology is versatile and applicable to a broad spectrum of digital circuits, eliminating the need for generating labor-intensive simulations tailored to individual circuits. It offers a streamlined solution that can be readily adapted across various circuit designs, saving considerable time and resources typically invested in customized simulation configurations for each circuit type. Additionally, we expand the PVT-aware surrogate modeling approach to optimize transistor sizing for improved yield, achieving a reduction of up to 64.6% in Power-Delay Product (PDP) of ISCAS-74X and ISCAS-85 benchmark circuits.

Furthermore, the research delves into the computational hurdles entailed in generating a significant volume of simulation samples for each targeted technology node, while considering diverse sources of variation. Consequently, it introduces the concept of transfer learning to circuit modeling, entailing a comprehensive exploration of relationships within CMOS/FinFET technologies across various technology nodes. The objective is to furnish precise predictions of PVT-aware circuit performance at advanced process nodes, thereby diminishing the considerable data requirements typically associated with such modeling endeavors. This is accomplished by capitalizing on knowledge accumulated during circuit modeling from one technology node to another. Additionally, the research concentrates on extrapolating PVT-aware performance onto forthcoming/future nodes by leveraging insights obtained from established nodes. It is noteworthy that this framework adopts a black-box approach, facilitating the incorporation of any number of PVT variations in any desired technology node with minimal computational complexity. This adaptive methodology not only enhances efficiency but also facilitates the flexibility to accommodate diverse circuit designs and technological advancements.

In addition, this research integrates rapid sensitivity analysis methods alongside a systematic approach to pinpointing predominant parameters, leveraging extensive statistical datasets encompassing

both digital and analog elements. The framework adeptly constructs models for analog and mixed-signal circuits with precision using the dominant parameters. These developed analog models present practical solutions for implementing self-adapting microelectronic circuits via appropriate digital control logic, thereby alleviating performance discrepancies stemming from PVT variations. These models not only enhance the adaptability and efficiency of circuit designs but also propel the microelectronics domain forward by facilitating dynamic responses to fluctuating operational environments, ultimately fostering innovation and advancement.

Contents

List of Figures

List of Tables

Abbreviations

ABC	Artificial Bee Colony
ADC	Analog-to-digital converter
ADE	Analog Design Environment
AI	Artificial Intelligence
AMS	Analog and Mixed-signal
ANN	Artificial Neural Network
ASIC	Application Specific Integrated Circuit
BGR	Bandgap Reference
BI	Bayesian Inference
BO	Bayesian Optimization
BSIM	Berkeley Short-channel IGFET Model
BSIM-CMG	Berkeley Short-channel IGFET Model – Common Multi-Gate
BTBT	Band-to-Band Tunneling
CART	Classification and Regression Trees
CCD	Complex Cell Delay
CCL	Complex Cell Leakage
CMOS	Complementary Metal–Oxide–Semiconductor
CUT	Circuit Under Test
DAG	Directed Acyclic Graph
DCA	Dominant Component Analysis
DCM	Design Circuits under Modeling
DIBL	Drain-induced Barrier Lowering
DNN	Dense Neural Networks
DT	Decision Tree
DUT	Design Under Test
E-CAD	Electronic Computer Aided Design
EDA	Electronic Design Automation
EHO	Elephant Herd Optimization
ELC	Cadence Encounter Library Characterizer

ER	ElasticNet Regression
ET	Extra Tree Regression
FDSOI	Fully Depleted Silicon On Insulator
FSL	Few Shot Learning
GA	Genetic Algorithm
GBDT	Gradient Boosting Decision Trees
GBM	Gradient Boosting Machine
GBRT	Gradient Boosted Regression Trees
GPR	Gaussian Process Regression
HHO	Harris Hawk Optimization
HP	High Performance
IC	Integrated Circuit
ICA	Independent Component Analysis
IG	Information Gain
KNN	K-Nearest Neighbors
LAR	Lasso Regression
LDO	Low Drop-out Regulator
LER	Line Edge Roughness
LGBM	Light Gradient Boosting Machine
LNA	Low-Noise Amplifier
LP	Low Power
LR	Linear Regression
LUT	Look-Up Table
MAE	Mean Absolute Error
MaPE	Mean average Percentage Error
MC	Monte Carlo
MGFET	Multi-gate Field-Effect Transistor
ML	Machine Learning
MLP	Multi-Layer Perceptron
MOSFET	Metal-Oxide Semiconductor Field Effect Transistor
MPR	Multivariate Polynomial Regression
MSE	Mean Squared Error
nFET	n-channel FinFET
nMOS	n-channel MOSFET
NN	Neural Network
PDK	Process Design Kit
PDP	Power-Delay Product

PDSOI	Partially Depleted Silicon On Insulator
pFET	p-channel FinFET
PLL	Phase Locked Loop
PMON	Process Monitor
pMOS	p-channel MOSFET
PR	Polynomial Regression
PSO	Particle Swarm Optimization
PTM	predictive technology models
PVT	Process, Voltage, and Temperature
RDF	Random Dopant Fluctuation
ResNN	Residual Neural Networks
RF	Random Forest
RL	Reinforcement Learning
RR	Ridge Regression
RSM	Response Surface Modeling
s-DoE	Statistical Design of Experiment
SA	Sensitivity Analysis
SCE	Short Channel Effects
SOC	Silicon On Chip
SOI	Silicon On Insulator
SSTA	Statistical Static Timing Analysis
SVM	Support Vector Machines
TCAD	Technology Computer-aided Design
TL	Transfer Learning
TxDr	Transmitter Driver
U-DNN	Unified Deep-Learning Neural Network
URV	Uniform Random Variations
VLSI	Very Large Scale Integration
XGBM	Xtreme Gradient Boosting Machine
ZSL	Zero Shot Learning

Symbols

V_{th}	Threshold voltage
V_{DD}	Supply voltage
I_{DS}	Drain-source current
I_{sub}	Sub-threshold leakage current
V_{ds}	Drain to source voltage
V_{gs}	Gate to source voltage
L	Channel length
W	Channel width
t_{ox}	Oxide thickness
C_{ox}	Gate oxide capacitance
t_{pLH}	Low-to-high propagation delay
t_{pHL}	High-to-low propagation delay
I_{Leak}	Leakage current
P_{Leak}	Leakage power
$P_{Leak-Stat}$	Statistical variation in leakage power
C_L	Load capacitance
t_s	Input slew time
T_{op}	Operating temperature
G_{Delay}	Propagation delay
$G_{Delay-Stat}$	Statistical variation in propagation delay
R^2	Coefficient of determination
r	Pearson's correlation coefficient
$\%\mu_E$	Mean error percentage
$\%\sigma_E$	Standard deviation error percentage

List of Patents

[P1] Khanh M. Le, Koushik De, **Deepthi Amuru**, Zia Abbas, **"AI-Driven Self Adaptive Microelectronic Circuits"**, US Patent no. 11416664, 2022, https://patents.google.com/patent/US11416664B2/en ***[Granted]***

[P2] Khanh M. Le, Koushik De, **Deepthi Amuru**, Zia Abbas, **"Design and Fabrication Methods of Runtime Self-tuning Analog Integrated Circuits using Machine Learning"**, US Patent application no. US20230153597A1, 2022, https://patents.google.com/patent/US20230153597A1/en ***[Published]***

[P3] **Deepthi Amuru**, Zia Abbas **"System and Method To Estimate Circuit Performances At Future Nodes Through Transfer Learning"**, Indian Patent application no. 202241071268, 2022 ***[Published]***

[P4] Zia Abbas, Khanh M. Le, Koushik De, **Deepthi Amuru "Self-Adapting Analog Circuit Design Method and System"**, 2022 ***[US patent Filed]***

[P5] **Deepthi Amuru**, Zia Abbas, Khanh Le **"Method and System of Creating a Unified Deep Learning Neural Network For Analog and Mixed-Signal Circuit Characterization"**, US Patent application no. US 63661535, 2024 ***[US patent Filed]***

List of Publications

[P1] **Deepthi Amuru**, R. Vechalapu, Zia Abbas, **"Transfer Learning enabled Modeling Paradigm with Bayesian Optimization for Statistical Circuit Performance Evaluation"**, in *ACM Transactions on Design Automation of Electronic Systems*. (Impact factor: 1.4) ***[Accepted]***

[P2] **Deepthi Amuru**, R. Goswami, M. Akhtar, Zia Abbas, **"OptiMo: Machine Learning Enabled Rapid Optimization Engine for Digital VLSI Circuits"**, in *IEEE Transactions on Very Large Scale Integration Systems*. (Impact factor: 2.312) ***[Revised version submitted]***

[P3] **Deepthi Amuru**, Amir Ahmad, Zia Abbas, **"A Combination of Ensembles Approach to Improve the Performance of Ensemble of Regression Models Approach for One-Class Classification"**, in *Applied Soft Computing Journal*, ScienceDirect. (Impact factor: 8.7) ***[Revised version submitted]***

[P4] N. Raghavendra, **Deepthi Amuru**, and Zia Abbas, **"MetaCirc: A Meta-learning Approach for Statistical Leakage Estimation Improvement in Digital Circuits"**, in proceedings of *2024 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2024

[P5] **Deepthi Amuru**, Zia Abbas, **"AI-Assisted Circuit Design and Modeling"**, chapter in book titled *AI-Enabled Electronic Circuit and System Design*, 2023, Quality Electronic Design, Springer Nature. ***[Accepted]***

[P6] **Deepthi Amuru**, Andeel Zahra, H. Vudumula, P. Cherupally, S. Gurram, Amir Ahmad, Zia Abbas, **"AI/ML Algorithms and Applications in VLSI Design and Technology"**, in *Integration, Elsevier*, Volume 93, 2023, 102048, ISSN 0167-9260. https://doi.org/10.1016/j.vlsi.2023.06.002. (Impact factor: 1.9)

[P7] K. Agarwal, A. Jain, **D. Amuru** and Z. Abbas, **"Fast and efficient ResNN and Genetic optimization for PVT aware performance enhancement in digital circuits"**, in proceedings of *2022 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, 2022, pp. 1-4, doi: 10.1109/VLSI-DAT54769.2022.9768067. [***Best paper nominee***]

[P8] **D. Amuru**, M. S. Ahmed and Z. Abbas, **"An Efficient Gradient Boosting Approach for PVT Aware Estimation of Leakage Power and Propagation Delay in CMOS/FinFET Digital Cells"**,

in proceedings of *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020, 2020, pp. 1-5, doi: 10.1109/ISCAS45731.2020.9180600.

[P9] **Amuru D.**, Zahra A., Abbas Z., **"Statistical Variation Aware Leakage and Total Power Estimation of 16 nm VLSI Digital Circuits Based on Regression Models"**, in *VLSI Design and Test. VDAT 2019*, 2019, Communications in Computer and Information Science, vol 1066. Springer, Singapore, https://doi.org/10.1007/978-981-32-9767-8_47.

[P10] **D. Amuru**, A. Zahra, V. Karakuram and Z. Abbas, **"Design of Approximate Full Adders for Error Resilient Applications"**, in proceedings of *2023 International Conference on Computer and Applications (ICCA)*, Cairo, Egypt, 2023, pp. 1-6, doi: 10.1109/ICCA59364.2023.10401821.

[P11] M. S. Ahmed, **D. Amuru** and Z. Abbas, **"ATM: Approximate Toom-Cook Multiplication for Speech Processing Applications"**, in proceedings of *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020, pp. 1-5, doi: 10.1109/ISCAS45731.2020.9181069.

Chapter 1

Introduction

The semiconductor industry, driven by the relentless pursuit of meeting the demands of the microelectronic industry, has achieved innumerable inconceivable milestones. The urging need for IoT-enabled smart devices is increasing daily, which demands higher data rate transmission, higher storage, retention, and faster data access from cloud databases [1]. The pace of the CMOS scaling is impelling on par with the needs of the electronic industry, keeping Moore's law intact. Transistor scaling, complex design methodologies, and sophisticated fabrication techniques have paved the way for developing promising system designs. As a result, microchips have become smaller yet more potent, capable of handling intricate data processing and communication tasks. Nevertheless, transistor scaling poses a multitude of problems in the nanometer regime. The increased chip density has increased susceptibility to process variations, particularly in deeply scaled technology nodes. The physical variations in the fabrication process lead to electrical performance deviations, inducing randomness in the circuit behavior that manifests to the system level, causing significant deviations in performance from the intended design specifications. Moreover, operating voltage fluctuations, ambient temperature variations during chip operation, and the complex dependency of process on temperature and voltage variability create challenges in accurately modeling the circuit performances in the sub-nanometer regime.

ICs are anticipated to operate continuously under diverse ambient conditions, electrical setups, and customer environments. The performance of an IC, including measures like latency, throughput, power dissipation, and other aspects, is influenced by changes in deviations in the semiconductor process (P), fluctuations in supply voltage (V), and operating temperatures (T). This parameter variability has emerged as a critical concern in advanced nodes, precipitating functional and electrical discrepancies within IC circuits. Consequently, the leakage current escalates, the logical characteristics of the device deteriorate, and the performance yield of logic circuits is adversely affected.

1.1 Challenges due to Transistor Scaling

Maintaining continuous scaling targets proves challenging, as each technological advancement brings forth new hurdles. These obstacles encompass SCE stemming from reduced channel length, gate leakage due to quantum mechanical tunneling, velocity saturation, interconnect delays due to RC coupling,

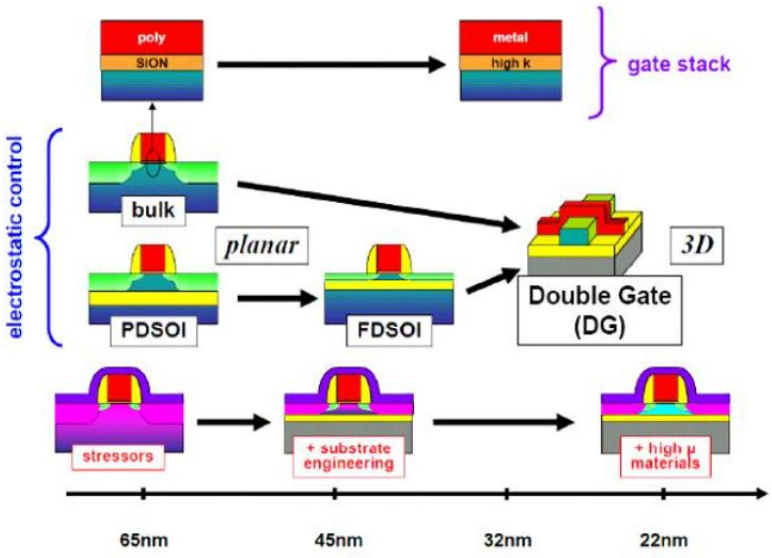


Figure 1.1: Structure and technology innovation for MOSFETs [4]

and amplified process variations leading to lithography difficulties. These factors impose practical constraints on downsizing the dimensions of bulk Metal-Oxide Semiconductor Field Effect Transistor (MOSFET)s below 20nm. Chief concerns for CMOS devices lie in the high sub-threshold and gate dielectric leakage currents. The Voltage-Doping Transformation model gauges the impact of reducing device parameters like gate length or drain voltage on electrical characteristics. While augmenting the doping concentration in the channel region can mitigate the effects of SCE to a certain degree, excessively high doping levels hinder proper device operation [2]. Gate dielectric scaling provides some relief from SCEs, yet further reduction results in significant static leakage current and heightened power consumption even in the device's off state.

Silicon On Insulator (SOI) devices were introduced as an alternative to bulk MOSFETs, presenting enhanced electrical characteristics [3]. There are two categories of SOI devices: Partially Depleted Silicon On Insulator (PDSOI) and Fully Depleted Silicon On Insulator (FDSOI). PDSOI devices are simpler to manufacture, exhibit reduced junction capacitance, are less susceptible to soft errors, and demonstrate improved operational speed. However, they grapple with an unconventional 'floating body' effect, which leads to hysteresis and instabilities. Conversely, FDSOI devices, characterized by a thinner silicon thickness resulting in complete channel depletion, offer superior mitigation of SCEs. Nevertheless, the efficacy of SCEs in FDSOI, compared to bulk MOSFETs, hinges on various factors such as silicon film thickness, buried oxide thickness, and doping concentrations. While FDSOI circumvents the floating-body effect, it gives rise to heightened junction capacitance and body effect.

To push advancements beyond the sub-20nm regime, more efficient device configurations with smaller gate lengths are imperative. Multi-gate Field-Effect Transistor (MGFET)s, notably FinFETs, have emerged as viable alternatives to planar bulk/SOI CMOS devices at technology nodes below 20nm [5]. FinFETs have demonstrated superior electrostatic integrity, effectively mitigating perfor-

mance degradation issues brought about by SCEs like threshold voltage roll-off, Drain-induced Barrier Lowering (DIBL), and subthreshold leakage. This has improved performance metrics, including a higher I_{ON}/I_{OFF} ratio, lower output conductance, greater gain, enhanced overall performance, and reduced variation due to lower doping concentrations. FinFETs have substantiated their effectiveness as replacements for bulk CMOS devices, offering enhanced gate control, reduced sub-threshold and gate dielectric leakage currents, and providing advantages in speed and transistor sizing, especially in sub-threshold regions [6]. They offer a wider channel width within a compact footprint, making them well-suited for driving large capacitive loads with lengthy interconnects. Double gate FinFETs essentially function as planar SOI MOSFETs with a thin buried oxide. Threshold voltage, V_{th} tuning can compensate for variability in IC manufacturing, whether from chip to chip or within the same chip from circuit to circuit. This proves to be an effective method for managing power consumption and enhancing IC speed. Fig 1.1 illustrates the structure of MOSFETs and technological advancements over the past 60 years.

FinFETs also demonstrate superior resilience to process variations when compared to complementary MOSFETs. This is attributed to the absence of channel dopants, which minimizes the impact of Random Dopant Fluctuation (RDF). However, it's important to note that process variations can still influence the performance of FinFETs [7]. Notably, Line Edge Roughness (LER) is a prominent variation source. Unlike planar bulk MOSFETs, controlling the resulting V_{th} variation in FinFETs is more challenging [8]. The fluctuations in V_{th} subsequently lead to variances in crucial transistor performance parameters, including leakage current, static power, dynamic power, and propagation delay. In certain cases, this can cause the performance to deviate significantly from its specified values. Therefore, it is important to meticulously assess the impact of design, process, and environmental parameter variations on the device during the IC design phase. This enables the implementation of suitable countermeasures to compensate for these variations effectively.

1.2 Challenges in Traditional Simulation Tools

Given these circumstances, prioritizing variation-aware design, testing, and validating ICs is crucial for ensuring reliable system designs. The effectiveness of VLSI circuit design and performance evaluation hinges on the capabilities of E-CAD tools. These tools have undergone continuous development and enhancement to accommodate the increasing complexity of designs across all stages of the VLSI design process. Simulation tools forecast higher-level performance based on lower-level design descriptions, while synthesis tools generate optimal lower-level design solutions to meet higher-level requirements. However, current Electronic Design Automation (EDA)/E-CAD tools have notable limitations. They are largely manual, time-consuming, and resource-intensive, lacking comprehensive knowledge-sharing and reuse capabilities. While these methodologies are widely used, they are rigid and typically follow fixed methodologies for conducting simulations which may not always be the most efficient approach for all design scenarios. While standardization can promote consistency and interoperability, it also lim-

its flexibility and innovation by constraining designers to predefined practices and approaches. Further, these tools heavily lean on the expertise of individual design engineers, presenting challenges in meeting stringent time-to-market constraints and production cost targets. The extensive simulation times of these tools when analyzing circuit statistical variability further impact the chip's turnaround time, which scales up proportionally with transistor down-scaling and circuit complexity [9]. Embracing state-of-the-art design methodologies and intelligent analysis tools is imperative to address these issues. These advancements assist in estimating the range of variability that could potentially lead to circuit malfunctions and chip failures. This, in turn, enables the implementation of appropriate countermeasures to enhance performance and bolster chip yield.

1.3 Need for Intelligent Modeling Framework

VLSI CAD tools necessitate a paradigm shift towards novel technologies and methodologies, departing from conventional event-driven or time-domain simulations. This shift is crucial for accurately modeling the complex behavior of digital, analog, and mixed-signal circuits across current and future technology nodes. A primary area of focus involves advancing methods for modeling and forecasting outputs based on diverse inputs at various levels of the VLSI CAD hierarchy [10]. Additionally, there is a significant emphasis on developing density estimation techniques to effectively capture variations and uncertainties in VLSI systems stemming from PVT fluctuations. Given the immense volume of data traversing billions of devices on a chip, there exists an opportunity to harness this information for analyzing input-output relationships within and across different hierarchical levels. Integration of simulation tools with design automation and optimization frameworks empowers designers to automate repetitive tasks, expedite exploration of design alternatives, and fine-tune design parameters to achieve performance objectives. This transition towards automation and optimization reduces dependency on rigid, manual processes, fostering heightened innovation and adaptability in VLSI design practices. Furthermore, the incorporation of AI/ML methodologies [11], [12] into E-CAD tools holds promise for automating diverse processes, leading to swift convergence in identifying design solutions. Consequently, this reduces the prolonged simulation times associated with traditional approaches while enhancing overall time-to-market and manufacturing yield, thereby elevating accuracy levels in VLSI design endeavors.

1.4 Objective and Scope of the proposal

Our primary objective is to create and refine a surrogate modeling framework capable of accommodating variations in process, voltage, and temperature to estimate the performance of VLSI circuits in CMOS/FinFET technologies. This modeling framework should emphasize both speed and computational efficiency while delivering performance predictions that are highly accurate compared to

traditional simulation tools. It must be versatile enough to accommodate different circuit designs and technology nodes, consistently delivering reliable results across all scenarios.

- Study and analyze the influence of PVT variations on circuit performance in CMOS/FinFET nodes, gaining insight into their ramifications across different technology nodes.
- Understand the limitations associated with conventional circuit simulation tools.
- Formulate a methodology to comprehensively capture changes in circuit performances resulting from variations in process parameters and operating conditions.
- Conduct an extensive sensitivity analysis to identify and extract the dominant variational parameters exerting the most significant influence on circuit performances.
- Design and construct a robust PVT-aware surrogate modeling framework tailored for estimating the performance of digital/analog/mixed-signal/RF circuits under PVT variations. This framework should demonstrate adaptability, enabling its broad application across various designs and technology nodes.
- Design and develop an automated methodology for estimating the performance of complex digital circuits using pre-characterized surrogate standard cell models without necessitating additional simulations.
- Apply the surrogate modeling approach to construct a multi-objective optimization engine to identify the optimal transistor sizes to enhance chip yield.
- Devise a strategy to reduce the data dependency of the surrogate models and propose a suitable modeling approach.
- Formulate surrogate modeling framework for implementing self-adaptive microelectronic circuits.

The scope of the research is spread across digital, analog, and mixed-signal design implementations. It encompasses a diverse array of technology nodes, encompassing CMOS and FinFET transistors. The designed surrogate models should exhibit versatility, re-usability, and applicability across a wide spectrum of practical applications.

1.5 Contributions

The following are our technical contributions as part of the thesis:

- A methodology to comprehensively capture the variations due to the design, process, and operating conditions causing performance deviations in digital and analog circuits.

- Development of statistically aware digital simulation datasets for 22 standard digital logic cells accounting for the impacts of design, process, and environmental variations on key circuit performance metrics, namely power and propagation delay. This comprehensive dataset generation will cover CMOS high-performance technology nodes, including 45nm, 32nm, 22nm, and 16nm, and FinFET technology nodes of 16nm, 10nm, and 7nm. A total of 308 digital cell datasets have been meticulously generated, considering PVT-aware factors, ensuring their adaptability for various applications.

- Development of variation aware datasets for analog circuits capturing their performance deviations across different corners.

- Development of a rapid and efficient approach to perform sensitivity analysis on performance simulations, considering variations, to identify and extract the most influential process parameters.

- Design and development of a fast, resilient, and platform-independent machine learning-based surrogate modeling framework for PVT-informed standard cell characterization across various CMOS (45nm, 32nm, 22nm, and 16nm nodes) and FinFET (16nm, 10nm, and 7nm nodes) technologies, aimed at forecasting propagation delay and leakage power. This adaptable framework extends its capabilities to encompass analog/mixed-signal and RF circuit modeling, accommodating PVT and design parameter fluctuations.

- Development of an automated approach to estimate leakage and delay in complex digital circuits, while accounting for PVT variations. This approach will employ pre-characterized ML surrogate models of standard digital cells, eliminating the need for additional simulations.

- Development of a fast and efficient multi-objective optimization engine for finding the optimal transistor sizing across a wide range of design, process, and environmental variations, enhancing the performance and yield of digital circuits.

- Development of Transfer-Learning technique for surrogate modeling of circuit performance, minimizing the requirement for extensive training data by capturing the correlation between design and PVT variations across different technology nodes for a circuit under test. This method enables the estimation of the PVT-aware behavior of the circuit in upcoming or future nodes without relying on training data.

- Design and development of Unified Deep-Learning Neural Network (U-DNN) architecture for Analog and Mixed-signal (AMS) circuit modeling, that promotes a versatile and adaptable network capable of generalizing across various designs.

- Development of a surrogate modeling framework for analog/mixed-signal circuits, enabling the implementation of self-adaptive microelectronic circuits.

1.6 Organization of the Thesis

The thesis is structured as follows: Chapter 2 offers an extensive review of the literature on PVT-aware modeling in VLSI circuit designs. Chapter 3 explores process variability and modeling, covering simulation setup and performance modeling of both digital and analog circuits. In Chapter 4, the design and workflow of the PVT-aware Surrogate Modeling framework methodology are elucidated, including a comprehensive analysis of ML algorithm design for circuit modeling, selection of optimal trained models, results, and discussions. This chapter encompasses surrogate model development for digital, analog, and mixed-signal circuits, and introduces the U-DNN design for analog circuit modeling. Chapter 5 outlines the methodology for identifying the primary parameters influencing circuit specifications. The transfer learning approach to reduce training data requirements of AI/ML algorithms is discussed in Chapter 6, emphasizing knowledge transfer across successive technology nodes and performance estimation at future/upcoming technology nodes. Chapters 7 and 8 showcase the applications of digital and analog circuit surrogate models, respectively, towards PVT-aware complex circuit estimation, the design of OptiMo: a statistically informed transistor sizing and optimization engine, and the development of AI-driven self-adapting microelectronic circuits. Finally, Chapter 9 presents a conclusion and outlines future scope.

Chapter 2

Literature Survey

Simulation plays a crucial role in characterizing IC devices. With technology shrinking to the nanometer scale, the complexity of evaluating circuit performance through simulations is increasing due to rising process and environmental variabilities. Detecting discrepancies in functional and electrical performance early in the design phase can significantly enhance IC yield, a parameter heavily dependent on the capabilities of simulation tools [2, 13, 14]. Over time, a variety of physics-based and empirical models have been employed for transistor device modeling. Designers utilize these models to forecast the electrical behavior of transistors (such as I-V and C-V characteristics) and other devices across different operational conditions. Physics-based models like the Berkeley Short-channel IGFET Model (BSIM) and Berkeley Short-channel IGFET Model – Common Multi-Gate (BSIM-CMG) [15] are known for their accuracy and generality, albeit requiring the solution of intricate nonlinear equations. Empirical models, such as Look-Up Table (LUT) based models, rely on extensive SPICE simulations and may encounter convergence challenges for larger circuits. These models are commonly integrated into Process Design Kit (PDK) for IC design purposes. Nevertheless, as transistors scale down to nanometer dimensions, addressing non-idealities becomes increasingly complex, making it challenging to derive closed-form solutions for device models. Furthermore, developing accurate physics-based models for each rapidly evolving next-generation device or technology is a complex and time-consuming task that demands a high level of expertise.

A promising frontier in VLSI design involves the integration of AI/ML algorithms into E-CAD tools for circuit modeling. This innovative approach holds great potential to significantly enhance the precision and effectiveness of circuit modeling, particularly for advanced nanometer technologies, while also reducing computational demands. AI-assisted models prioritize data-driven modeling methodologies, eliminating the necessity for extensive analysis of the device's physical attributes and facilitating the rapid development of accurate surrogate device models. These models exhibit the capability to forecast the electrical behaviors of devices across diverse design and process parameter variations with remarkable precision and efficiency, even in scenarios involving complex devices and operational settings. Such advancements contribute to expediting turn-around times and strive to generate dependable models that substantially augment computational efficiency. Consequently, this facilitates the swift integration of AI/ML techniques into emerging devices, enabling convenient and prompt assessments of

transistor transfer characteristics and underlying impacts, even before a comprehensive understanding of the underlying physics is established.

2.1 Surrogate models for Digital Circuit Modeling

Over the past twenty years, diverse methodologies have emerged within the literature to navigate variability analysis concerning digital circuit performance. Their primary aim is to precisely gauge the influence of PVT variations on circuit behavior. An approach for evaluating statistical parametric yield was introduced [16] to measure the comprehensive parametric yield of MOSFET circuits. Alvarez et al. and Young et al. introduced a statistical design analysis leveraging Response Surface Modeling (RSM) for computer-aided VLSI device design [17, 18]. These proposed frameworks have been effectively employed in optimizing BiCMOS transistor design. Khan et al. [19] recommended the adoption of Multivariate Polynomial Regression (MPR) to estimate early voltage and MOSFET characteristics during saturation. They utilized a curve-fitting technique employing the least-squares method in MPR to streamline the complexity within BSIM3 and BSIM4 equations [15], leading to a more realistic computation of MOSFET characteristics.

In subsequent advancements, RSM modeling gained popularity for assessing the impact of process variations on circuit design. Mutlu et al. conducted an extensive investigation into RSMs to evaluate their effectiveness in analyzing the effects of process variations on circuit design [20]. Basu et al. [21] established a repository of statistical intra-gate variation-tolerant cells by developing RSM-based gate-delay models with reduced dimensions. These optimized standard cells proved instrumental in chip-level optimization efforts, to achieve critical path timing. In the studies by [22] and [23], RSM learning models were formulated using a blend of Statistical Design of Experiment (s-DoE) and an automated selection algorithm for Statistical Static Timing Analysis (SSTA) of gate-level library-cell characterization in VLSI circuits. These models incorporated threshold voltage (V_{th}) and current gain (β) as model parameters for compact transistor model characterization of power, delay, and output transitions. The RSM and linear sensitivity methodologies proposed in [22] markedly enhanced analysis speed by one and two orders of magnitude, respectively, compared to Monte Carlo (MC) simulations, albeit with a marginal sacrifice in accuracy, up to 2% and 7%. In [23], the s-DoE approach exhibited an error of 0.22% at the tails of the 3σ distribution, contrasting with the 10-fold error observed with sensitivity analysis conducted by the Cadence Encounter Library Characterizer (ELC).

Miranda et al. [24] introduced a variation-aware s-DoE approach for predicting the parametric yield of static random access memory circuits amidst process variability. Their methodology exhibited an accuracy of approximately two orders of magnitude superior to sensitivity analysis in tail responses under 3σ process variations, while requiring CPU time 10–100 times less than that of MC simulations. The case studies outlined in their article demonstrated the advantage of s-DoE in selecting the region of interest in the distribution, leading to improved accuracy while reducing the number of simulations. Similarly, Chaudhuri et al. [25] devised precise RSM-based analytical leakage models tailored for 22nm

shorted-gate and independent-gate FinFETs, taking into account process variations. Employing a central composite rotatable design, they estimated leakage current in FinFET standard cells. Their findings closely aligned with quasi-MC simulations conducted in Technology Computer-aided Design (TCAD) utilizing 2D cross-sections.

There has been longstanding interest in uncovering potential patterns within simulated data of VLSI designs and leveraging this data across multiple phases of circuit design. Towards this objective, Cao et al. [26] introduced a robust method employing table-lookup techniques to estimate leakage power and switching energy at the gate-level for all possible states. This approach integrated Bayesian Inference (BI) principles and Neural Network (NN)s. Their model utilized pattern recognition, categorizing potential states based on average power consumption values utilizing NNs. The fundamental idea is to leverage statistical insights gleaned from a circuit's existing SPICE power data to establish the correlation between state-transition patterns and power consumption values within the circuit. This correlated pattern data is then utilized to predict power consumption for both observed and unforeseen state transitions across the entire spectrum of circuit state transitions. The estimation errors generated by NNs consistently adhere to normal distributions, demonstrating significantly reduced variations compared to benchmark curves. Furthermore, the estimation error diminishes with increasing numbers of clusters and complexity of NNs, given appropriate feature extraction. Additionally, the time required for training and validating NNs is minimal compared to the computational time needed for generating statistical distributions using the SPICE environment.

Over time, polynomial regression has emerged as a significant analytical modeling technique. In their work, [27] introduced a statistical leakage estimation method employing polynomial regression. Through experimentation on the MCNC benchmark [28], it was demonstrated that this approach is five times more efficient than Wilkinson's method [29], while maintaining mean estimation accuracy and experiencing only approximately 1% loss in standard deviation. Moshrefi et al. [30] introduced an accurate and cost-effective Burr distribution function for delay estimation. This method takes into account threshold voltages within $\pm 10\%$ of the mean across samples generated at the 90, 45, and 22 nm technology nodes. Statistical data from MATLAB were employed in SPICE simulations to capture delay variations, with the relationship between threshold voltage and delay variations expressed as a fourth-order polynomial equation. Beyond mean and variance, maximum likelihood was considered as the third parameter, forming a three-parameter probability density function. This proposed Burr distribution offers an additional degree of freedom compared to the normal distribution [31], resulting in a lower error distribution.

Kim et al. [32] devised a methodology to estimate propagation delay in digital circuits by transforming any logic gate into an equivalent inverter, obviating the need for preliminary simulation. Although this model entails a nominal 5% deviation from SPICE results. Kahng et al. [33] introduced a hybrid surrogate model amalgamating Artificial Neural Network (ANN) and Support Vector Machines (SVM)) to predict the incremental delay attributed to signal integrity-aware path delay in a 28nm FDSOI technology. They demonstrated a maximum error of less than 10 ps. Janakiraman et al. [34] proposed an

efficient ANN model for analyzing voltage- and temperature-aware statistical aspects of leakage power, employing transistor-level stack models for leakage estimation. This model showcased a remarkable 100-fold enhancement in runtime while maintaining errors in the mean and standard deviation of MC statistical leakage estimations below 1% and 2%, respectively. Its complexity is comparable to existing linear and quadratic models, with computational complexity rated as O(N). Xingsheng Wang et al. [7] examined the statistical variability and reliability of FinFETs, scrutinizing various design and process parameters, albeit without delving into their influence on leakage and delay. Garg et al. devised SVM-based macro models to characterize transistor stacks of CMOS gates, yielding an average runtime reduction of 17 times compared to SPICE computations for leakage power estimation [35]. Helms et al. [36] proposed RT-level leakage macro models for gate-level leakage estimation, reporting discrepancies of 2.1% (for 16nm Low Power (LP)) and 6.8% (for 65nm bulk) against SPICE simulations. Due to complexity considerations, the paper limited process variations to 10 among 90000 training samples. A robust power estimation method for CMOS VLSI circuits utilizing Random Forest (RF) was presented in [37], surpassing NNs in accuracy, as evidenced by its agreement with ISCAS'89 Benchmark circuits [38].

The primary challenge in PVT-aware circuit modeling for advanced technology nodes lies in integrating diverse sources of variation for analysis while minimizing computational overhead. It is imperative to generate numerous simulation samples to effectively capture high-dimensional variation spaces through ML algorithms. However, the generation of a substantial volume of training samples poses significant computational burdens, particularly for complex modern digital circuits. Traditional methods of performance modeling have fallen short in addressing this core issue.

A few studies have endeavored to confront this challenge. For instance, Yu et al. [39] introduced an analytical timing model employing a BI framework to estimate delay and slew time under process variations. This approach showcased a 15x acceleration compared to conventional look-up table methods. It entailed constructing a novel library with sparse training samples tailored to the target technology node while leveraging insights from timing model coefficients of previous technology nodes. In the domain of AMS circuits, approaches to performance modeling based on BI that span multiple corners were proposed [40, 41]. These methods drew upon simulation data from early stages across different corners to facilitate performance modeling during the post-layout phase, thus mitigating the necessity for extensive simulation samples. However, it's noteworthy that in the context of digital circuit variability analysis, a statistical approach typically takes precedence over corner-based methodologies [42]. Additionally, it is important to acknowledge that BI may incur substantial computational costs as process parameters scale. In an alternate methodology, the power-delay characteristics were investigated across multiple CMOS and FinFET standard cells utilizing Polynomial regression and predictive technology models (PTM) [43] for technology nodes from 180nm to 7nm [44]. Scaling coefficients were computed to assess the effects of scaling across diverse technology nodes. However, this investigation was carried out under typical operating conditions, without incorporating variations in PVT.

2.2 Surrogate models for Analog/Mixed-Signal/RF Circuit Modeling

The unique challenges in designing analog circuits arise from their susceptibility to process variations, temperature shifts, and various environmental influences. Conventional design techniques typically require manual adjustments, extensive post-production fine-tuning, and significant design safety margins to address these uncertainties. Nevertheless, conventional design methods are proving insufficient and ineffective with technology advancing to nanoscale dimensions and a growing need for enhanced performance. Furthermore, leading Application Specific Integrated Circuit (ASIC) manufacturers frequently encounter errors stemming from altered, inaccurate, or incomplete specifications.

Leveraging AI/ML, self-adaptive microelectronic circuits possess the capability to autonomously optimize their operational parameters and configurations, ensuring effective performance across varying conditions. offers numerous advantages, including heightened robustness, real-time adaptability, and improved energy efficiency. Rapid and effective prototyping of analog/mixed-signal circuits plays a crucial role in the development of self-adjusting circuits. Past studies have introduced NN-based methodologies for analog circuit design, aiming to capture intricate circuit behaviors and furnish precise models to address the inherently complex, nonlinear characteristics of analog circuit performance. The utilization of NNs in automating analog circuit design has been suggested by previous research [45–47]. Notably, [48] advocates for the adoption of ANNs for analog circuit design automation. Furthermore, it introduces a novel methodology employing the g_m/I_d technique for data acquisition, to expedite the analysis process. Experimental results, conducted on a common source amplifier and a two-stage single-ended Opamp, demonstrate that the ANN model utilizing the g_m/I_d technique offers superior accuracy even with a reduced dataset, thereby streamlining the data collection process in terms of both time and effort. Other research efforts on analog circuit modeling, design sizing, and optimization are documented in [49–54]. These methodologies employ optimization algorithms instead of conventional simulators to expedite the delivery of design solutions that are both efficient and accurate. An instance of an ANN-based technology-independent circuit sizing model, as discussed in [55], demonstrated up to 90% accuracy in predicting transistor sizes in a $0.18\mu m$ technology while satisfying multiple circuit specifications. Notably, the ANN was trained using simulations from various technologies, including $1.5\mu m$, $0.5\mu m$, $0.35\mu m$, and $0.25\mu m$, with the $0.18\mu m$ technology being unfamiliar to it. For a more comprehensive examination of the application and efficacy of ANNs in automating analog design sizing, [56], provides detailed insights. To mitigate the data dependency of ML models and enhance circuit yield, cluster-based approaches can be utilized, as proposed by [57].

In recent times, Reinforcement Learning (RL) techniques have been extensively utilized in enhancing analog circuits, showcasing adaptability and refinement through continuous learning and optimization based on feedback. Incorporating domain-specific expertise from previous experiences into the RL algorithm workflow aims to explore and improve designs. An example of such an approach is AutoCkt, a machine learning optimization framework trained using deep reinforcement learning. AutoCkt efficiently identifies post-layout circuit parameters for a given target specification and comprehensively explores the design space using a sparse subsampling technique [58]. Results show that AutoCkt achieves

convergence and meets target specifications 40 times faster than conventional genetic algorithms or the current state-of-the-art approach, particularly in considering layout parasitic effects.

Bayesian techniques have gained recognition in electronic IC design for addressing various challenges, particularly amidst unpredictable process variations. These challenges encompass circuit performance modeling, yield/failure rate estimation, and circuit optimization. For instance, in the paper [59], a Bayesian Optimization (BO) method is introduced for analog circuit synthesis using neural networks. It integrates Gaussian Process Regression (GPR) with a NN trained initially with a randomly generated set. Model ensembles enhance prediction accuracy, while an acquisition function identifies new design points for simulation and inclusion in the training set. This iterative process continues until convergence or the simulation budget is exhausted. Experimental results indicate superior performance compared to current algorithms, requiring fewer simulations. Moreover, the GPR model with NN scales linearly with the number of observed data points (N), making it suitable for applications with numerous iterations. Further insights into Bayesian methods in VLSI design are provided by [60].

Self-healing circuit designs autonomously compensate for various circuit variations, including static, quasi-static, and dynamic changes [61]. They incorporate sensors to monitor these variations and tuning knobs to adjust performance. A core self-healing algorithm determines the appropriate knob settings based on measured conditions. ML techniques are employed [62, 63] to relate sensor values to circuit performance, enabling performance prediction solely based on sensor measurements. Developing resource-efficient on-chip adaptive algorithms is challenging, requiring maintaining adaptation quality without significant area cost increase, increasing adaptation speed without substantial power consumption rise, and enabling on-chip characterization without additional power and area expenses.

2.3 Conclusion

In conclusion, the demand for an expedited, intelligent statistical process variability modeling approach, adept at accommodating diverse designs and technology nodes, is underscored in current literature. Such an approach must skillfully integrate varied sets of PVT variations, considering their correlations across technology nodes, while minimizing reliance on extensive training data. However, recent advancements reveal a gap in comprehensive modeling efforts, particularly in integrating digital circuit parameters and extending this modeling to analog circuits. In light of this imperative, our proposed solution presents a swift and effective surrogate modeling framework tailored to address the needs of various circuit applications, spanning digital and AMS circuit domains. By offering a holistic approach that bridges the gap between digital and analog realms, our framework holds promise for accelerating the design process and enhancing overall circuit performance in increasingly complex semiconductor ecosystems.

Chapter 3

Process Variability and Modeling

Dynamic changes occur within the chip's periphery during manufacturing. These alterations are influenced by various factors such as temperature, pressure, non-uniform dopant concentrations, and impurity diffusion, collectively leading to process variations. Typically, the dies located at the center of fabrication tend to adhere more closely to the intended process values than those situated at the periphery, which may deviate from the standard. These deviations in process values ultimately contribute to inconsistencies in the fabrication process, resulting in deviations from the expected circuit performance specifications.

Primary sources of process variation comprise:

- Fluctuations in UV light wavelength during lithography
- Defects in the manufacturing process
- Variances in oxide thickness
- Discrepancies in metal thickness

Consequently, significant repercussions of process variation encompass:

- Variability in oxide thickness
- Fluctuations in dopant concentration and mobility
- Variations in resistance and capacitance
- Irregularities in transistor dimensions, including channel length
- Variability in threshold voltage
- Alterations in source-to-body voltage

Variations in the lithographic process lead to fluctuations in sheet resistance, resulting in variations in channel dimensions and transistor characteristics, such as threshold voltage. These variations translate to reliability issues at the circuit level, differing across transistors within a die, between dice on a wafer,

or even among different wafers. As a result, circuit performance deviates due to disparities in current and latency.

Variations in manufacturing processes, combined with operating and environmental conditions, significantly influence the performance characteristics of circuits. These factors contribute to the uncertainties in how a chip operates under different conditions and throughout its lifespan, encompassing variables such as temperature, voltage supply, longevity, and usage wear. The voltage supply across a chip is inconsistent and is contingent upon the types of circuits present and their placements. Key sources of voltage variations arise from factors like IR drop caused by resistance in supply lines, voltage fluctuations due to parasitic inductance coupled with resistance and capacitance, and variations stemming from voltage regulation mechanisms. At sub-nanometer scales, circuit performance experiences nonlinear effects due to fluctuations in temperature. Temperature variations occur across the chip during operation, primarily due to power dissipation. These lead to temperature shifts impacting transistors' mobility, with mobility decreasing as temperatures rise, leading to longer propagation delays. Furthermore, higher temperatures correspond to a decrease in threshold voltage, resulting in increased current flow and improved delay performance. This phenomenon is heavily influenced by the cell's power supply, threshold voltage, load, and input characteristics. Generally, the impact of temperature on transistor mobility tends to be the predominant factor.

Moreover, the process itself is affected by both supply voltage and temperature, further complicating the combined effects of PVT variations, which are complex and non-linear. Consequently, there is a need for the development of a comprehensive modeling strategy to depict these effects accurately.

3.1 PVT Variations in Nanometer Technology Nodes and Challenges

As we advance towards advanced technology nodes, the significance of random variations in the fabrication process is growing. While process variability primarily affected analog designs in the recent past, it has now emerged as a major concern for digital engineers too.

Parametric variations can be broadly categorized into design parameter variations, process variations, and environmental variations (see Fig. 3.1). Design parameter variations, alternatively referred to as device geometry variations, stand as a primary source of variability. Lateral dimensions such as channel length and width are subject to influences such as variations in gate-oxide thickness, lithographic proximity effects, and the effects of channel length. These variations directly impact the output performance characteristics of the device. Process variations can be broadly classified into two primary types: global and local variations. Global variations, also known as Die-to-die or Inter-die variations, such as variances in oxide thickness and dopant concentrations, affect all transistors within a die uniformly. Conversely, local variations, referred to as Within-die or Intra-die variations, are also known as mismatch or random uncorrelated variations. These variations affect each transistor on the die differently. Local variations can be further divided into spatially correlated and random variations. Spatially correlated variations exhibit similar characteristics for devices in close proximity within the die, as opposed

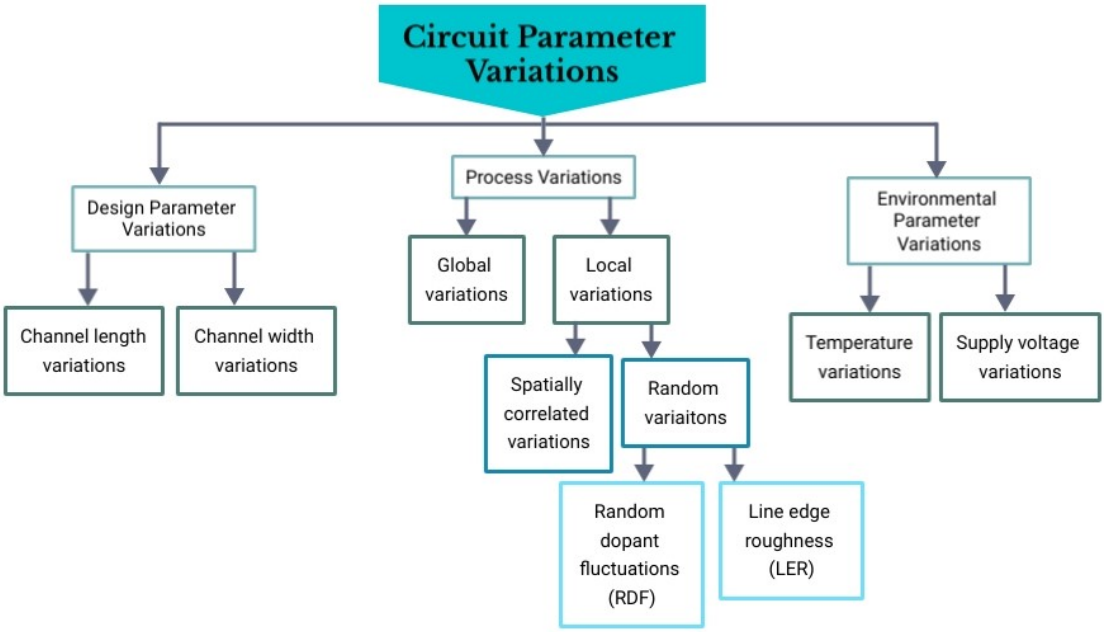


Figure 3.1: Types of circuit parameter variations

to those located farther apart. Random or independent variations demonstrate statistically independent behavior from other device variations. Notable random variations include RDF and LER, which play significant roles in deep sub-micron technology.

Fluctuations in the statistical distribution of random atoms give rise to Random Dopant Fluctuations RDF, causing variations in the threshold voltage of a transistor. LER refers to localized variations along the width of the polysilicon gate. Factors contributing to LER include fluctuations in incoming photon counts and the molecular composition of the resist. LER directly influences the standard deviation of the threshold voltage. Variations in the V_{th} due to physical inconsistencies significantly impact circuit performance. These variations result in changes in leakage currents, propagation delay, dynamic, and overall power dissipation in digital circuits. In analog circuits, they introduce variations in circuit specifications such as output gain, phase margin, slew rate, and other parameters.

The increasing physical variations and their complexity emphasize the necessity for early-stage validations in VLSI circuit design. Assessing the impact of random process variations, alongside device and environmental parameter fluctuations, is pivotal during the design phase. This necessitates the accumulation of significant volumes of data. While conventional VLSI CAD tools such as SPICE and PrimeTime can be utilized for PVT-aware data generation, direct analysis using these tools can prove time-intensive [9]. Hence, we advocate for the adoption of PVT-aware ML models as proxies for swift and accurate analysis of VLSI circuits.

To establish a precise surrogate modeling framework encompassing multiple technology nodes, it is crucial to initially identify the principal performance metrics affected and the primary sources of variations responsible for these effects. Subsequently, generating appropriate PVT-aware circuit per-

formance data for various nodes becomes imperative. A major concern in diverse digital applications is the divergence in power and delay induced by PVT variations. Consequently, our focus lies in the PVT-aware modeling of power and delay in digital circuits designed across both CMOS and FinFET technology nodes. Acknowledging the significant impact of PVT variations on the output specifications of analog and mixed circuits, we also develop models that accommodate these discrepancies based on the design requisites across different applications.

3.2 Digital Circuit Performance Measures

The growing demand for portable devices and wireless communication necessitates fast data processing while minimizing power consumption. This highlights the crucial significance of power and delay as primary considerations in the design of digital circuits. As ICs continue to decrease in size, the power dissipation levels also increase. This necessitates larger cooling solutions, which in turn affect the overall size of the chip. Moreover, the impact of process variation on power and delay becomes more pronounced in nanometer nodes, presenting a significant challenge. Given these circumstances, accurately predicting the variations in path delay and power consumption induced by PVT in digital circuits is crucial. It requires a comprehensive framework for PVT-aware power and delay estimation that spans both CMOS and FinFET nodes to effectively address the impact of shrinking transistor dimensions. With this objective, we propose the implementation of PVT-aware surrogate modeling for power and delay in digital circuits.

Our approach involves a deep exploration of the intricacies among process, voltage, and temperature dependencies on circuit delay and power, particularly focusing on their interconnections. This entails a detailed examination of their fundamental associations with delay and power through basic equations, followed by the derivation of PVT parameters for modeling, as elaborated below.

3.2.1 Power

Power dissipation is the rate at which energy is drawn from the power source V_{DD} and transformed into heat. Equ. 3.1 outlines the primary power components in CMOS/FinFET circuits.

$$P = P_{dynamic} + P_{shortcircuit} + P_{static} \tag{3.1}$$

where P is the total power, $P_{dynamic}$ is the dynamic component of power with N switching activities, $P_{shortcircuit}$ is the short circuit power and P_{static} is the static power dissipation.

$$P_{dynamic} = \sum_{i=1}^{N} \alpha C_L {V_{DD}}^2 f \tag{3.2}$$

C_L is the load capacitance, f is the operating clock frequency, and α is the node transition activity factor. The dynamic power dissipation (equ. 3.2) was the dominant factor compared with other components of

power dissipation in digital CMOS circuits for technologies up to $0.18\mu m$, which is about 90% of total circuit dissipation [64].

$$P_{shortcircuit} = K(V_{DD} - V_{th})^3.\tau.f \tag{3.3}$$

where K is a constant that depends on the size of the transistor and the technology parameters, V_{th} is the threshold voltage, τ is the rise or fall time of the input signal, and f is the clock frequency.

$P_{shortcircuit}$ arises from the direct-path short-circuit current that conducts electricity directly from the power supply to the ground when both the n-channel MOSFET (nMOS)/n-channel FinFET (nFET) and p-channel MOSFET (pMOS)/p-channel FinFET (pFET) transistor networks are active simultaneously (equ. 3.3). This short-circuit current only flows for a short interval of time. It constitutes less than 20% of the overall dynamic switching power consumption if the sizing of the nMOS/nFET and pMOS/pFET transistors is such that it balances the rise and fall signal slopes at input and output nodes [65]. Short-circuit power and dynamic switching power are interconnected with the charging and discharging of output capacitances, which cause transitions at the gate terminals. Consequently, they are collectively referred to as dynamic power.

Static power dissipation concerns the current flow when the gate terminals remain unchanged (equation 3.4). In an ideal scenario, CMOS circuits should exhibit no static power dissipation in a stable state. Nevertheless, practical systems often experience degraded voltage levels supplying CMOS gates. As a result, a current flows from the power supply to ground nodes, leading to static current I_{Stat}.

$$P_{static} = \sum_{j=1}^{M} (I_{Stat}) V_{DD} \tag{3.4}$$

Here, P_{static} arises from the static current I_{Stat}, with M representing the number of nodes. In earlier technology nodes, the magnitude of P_{static} was typically minimal and often considered negligible. However, as devices have undergone scaling over time to enhance density and performance, the consequent rise in leakage current has considerably escalated static dissipation. Consequently, it has evolved into a significant component of power dissipation in CMOS circuits. Beyond the 22nm threshold, the dissipation of leakage power in digital circuits equals that of dynamic power dissipation.

The scaling of supply voltage across various technology nodes, coupled with the corresponding decrease in V_{th}, reduction in oxide thickness, and higher substrate doping concentration, has resulted in increased tunneling currents. These currents lead to significant leakage through the drain- and source-to-substrate junctions, particularly under high reversed biasing conditions. Subthreshold leakage (I_{sub}), gate oxide leakage, and reverse-bias pn-junction leakage (Band-to-Band Tunneling (BTBT)) are the primary contributors to this leakage current. Gate-induced drain leakage and punch-through current also represent other leakage current mechanisms. I_{sub} occurs between the drain and source when the transistor operates in the weak inversion region, indicating that the gate voltage is lower than the V_{th}.

I_{sub} is mainly driven by diffusion current and demonstrates exponential dependence on the gate-to-source and threshold voltage.

Considering the BSIM MOS transistor model [15], I_{sub} for a MOSFET device can be expressed as equ. 3.5.

$$I_{sub} = I_0 e^{\frac{V_{gs}-V_{th}}{nV_T}} \left[1 - e^{-\frac{V_{ds}}{V_T}}\right] \tag{3.5}$$

where, $I_0 = \frac{W\mu_0 C_{ox} V_T^2 e^{1.8}}{L}$, $V_T = \frac{KT}{q}$ is the thermal voltage, V_{th} is the threshold voltage, V_{ds} and V_{gs} are the drain-to-source and gate-to-source voltages respectively. The effective transistor width and length are W and L, respectively. C_{ox} is the gate oxide capacitance, μ_0 is the carrier mobility and n is the subthreshold swing coefficient.

In devices featuring short channels, the extension of the depletion regions from the source and drain into the channel region significantly influences the field and potential profile within. As a consequence of the reduced channel length, a noticeable decline in the transistor's threshold voltage, termed as V_{th} roll-off, occurs, accompanied by an elevation in drain-induced barrier lowering. These SCEs collectively contribute to the emergence of a substantial I_{sub} in short-channel devices.

To mitigate the challenges posed by SCEs, a strategic reduction in the oxide layer's thickness becomes imperative with each successive leap in technology generation. However, this aggressive scaling of oxide thickness introduces a new hurdle in the form of heightened electric fields, consequently engendering a notable direct-tunneling current across the transistor's gate insulator. This intriguing phenomenon, dubbed gate oxide tunneling current, involves the migration of electrons (or holes) from the bulk and source/drain overlap region, surmounting the gate oxide's potential barrier into the gate (or vice versa) (equ. 3.6).

Gate leakage can be modeled as:

$$I_{Gate} = W \cdot L \cdot A \left(\frac{V_{DD}}{t_{ox}}\right)^2 \exp\left(\frac{-B\left(1 - \left(1 - \frac{V_{DD}}{\phi_{ox}}\right)^{3/2}\right)}{\frac{V_{DD}}{t_{ox}}}\right) \tag{3.6}$$

where W and L are the effective transistor width and length, respectively, $A = q^3/16\pi^2 h\phi_{ox}$, $B = 4\pi\sqrt{2m_{ox}}\phi_{ox}^{3/2}/3hq$, m_{ox} is the effective mass of the tunneling particle, f_{ox} is the tunneling barrier height, t_{ox} is the oxide thickness, h is $1/2p$ times Planck's constant and q is the electron charge.

BTBT is the phenomenon in which electrons transition from the valence band of p-type material to the conduction band of n-type material when a high electric field (greater than 10^6 V/cm) is applied across the reverse-biased pn junction. This tunneling current can be mathematically represented as follows (equ. 3.7).

$$I_{BTBT} = A\alpha \frac{EV}{E_g^{1/2}} \exp\left(-\beta \frac{E_g^{3/2}}{E}\right) \tag{3.7}$$

where A is the total area of the junction, α and β are constants, E_g is the bandgap voltage, and V is the applied forward-bias voltage. The electric field along the junction at a reverse bias of V is as follows.

$$E = \sqrt{\frac{2eN_aN_d\left(V+\phi_0\right)}{\varepsilon_{si}\left(N_a+N_d\right)}} \tag{3.8}$$

High doping concentrations in deep sub-micron devices and sudden changes in doping profiles result in a notable BTBT current flowing through the drain-well junction.

SCEs induce changes in the effective mobilities of electrons and holes within short-channel devices, consequently impacting the transistor's threshold voltage and current-voltage (I-V) characteristics. The empirical formula representing effective mobility (μ_{eff}) is articulated as follows:

$$\mu_{eff} = \frac{A}{1+\left(\frac{E_{\text{norm}}}{B}\right)} \tag{3.9}$$

where A is 670 $cm2/V{\cdot}s$ for electrons and 160 $cm2/V{\cdot}s$ for holes, B is 6.6 x 10^5 V/cm for electrons and 7 x10^5 V/cm for holes, and the factor E_{norm} can be calculated as follows.

$$E_{\text{norm}} = \frac{V_{gs}+V_{th}}{6t_{ox}} \tag{3.10}$$

Further, the threshold voltage for an nMOS transistor can be defined as follows:

$$V_{th} = V_{FB} + 2\Phi_b + \frac{\Phi_{dep}}{C_{ox}} \tag{3.11}$$

where, V_{FB} is the flat band voltage due to oxide charge and work function difference given by $V_{FB} = \Phi_{GS} - \frac{Q_{ss}}{C_{ox}}$; Φ_b is the bulk potential given as $\Phi_b = \frac{KT}{q} ln \frac{N_A}{n_i}$, K is the Boltzmann constant, T is the temperature and q is the electron charge; Φ_{dep} is the depletion charge per unit area given as $\Phi_{dep} = qN_AX_{dep}\alpha\frac{1}{\sqrt{N_A}}$; C_{ox} is the gate oxide capacitance given as $C_{ox} = \frac{\epsilon_{ox}}{t_{ox}}$.

The drain current I_{DSsat} is highly influenced by temperature, primarily because the threshold voltage and mobility exhibit variations with temperature. Examine the nMOS current equation,

$$I_{DSsat} = \frac{1}{2}\mu_n\left(\frac{\varepsilon_{ox}}{t_{ox}}\right)\left(\frac{W_n}{L_n}\right)\left(V_{GS}-V_{Tn}\right)^2 \tag{3.12}$$

The dependence of V_{Tn} (nMOS V_{th}) with temperature can be expressed as follows,

$$\frac{dV_{Tn}}{dT} = -\frac{1}{T}\left(\frac{E_g}{2e} - \left|\Phi_b\right|\right)\left(2+\frac{\gamma}{\sqrt{2\left|\Phi_b\right|}}\right) \tag{3.13}$$

If $\mid \Phi_b < \frac{E_g}{2e}$, the threshold voltage decreases as the temperature rises, while it increases with temperature under other conditions. The mobility also exhibits temperature dependence and can be described

by,

$$\mu(T) = \mu(T_0)\frac{T_0}{T}^{1.5} \tag{3.14}$$

Therefore, the reduction of mobility with temperature can be expressed as,

$$\frac{d\mu(T)}{dT} = -1.5\mu\left(T_0\right)\left(\frac{T_0}{T^{2.5}}^{1.5}\right) \tag{3.15}$$

As temperature increases, both V_{th} and mobility undergo a decrease. According to the current equation, the decline in V_{th} leads to an augmentation in I_{DSat}, while the reduction in mobility yields a decrease in drain current. In instances of low V_{gs}, alterations in V_{Tn} wield a dominant influence, causing the drain current to escalate with temperature elevation. Conversely, at higher V_{gs} levels, mobility takes precedence, resulting in a drain current decline with rising temperature. At specific values of V_{gs}, these effects balance each other, yielding no net change in drain current with temperature. Within the sub-threshold voltage region, the drain current rises with increasing temperatures due to the behavior of the bipolar junction transistor. However, in the above-threshold region, the drain current decreases with temperature ascent owing to the behavior of the MOSFET.

For long-channel devices,

- nMOS transistor

$$\begin{aligned}
&V_{Tn} = V_{T0n} + \gamma\left(\sqrt{2\left|\Phi_{fp}\right| + V_{SB}} - \sqrt{2\left|\Phi_{fp}\right|}\right)\\
&I_{DS} = \tfrac{\mu_n C_{ox}}{2}\left(\tfrac{W_n}{L_n}\right)\left[2\left(V_{GS} - V_{Tn}\right)V_{DS} - V_{DS}^2\right] \quad \text{for } V_{GS} \geq V_{Tn} \text{ and } V_{DS} < V_{GS} - V_{Tn}\\
&I_{DSsat} = \tfrac{\mu_n C_{ox}}{2}\left(\tfrac{W_n}{L_n}\right)\left(V_{GS} - V_{Tn}\right)^2\left(1 + \lambda V_{DS}\right) \quad \text{for } V_{GS} \geq V_{Tn} \text{ and } V_{DS} \geq V_{GS} - V_{Tn}
\end{aligned} \tag{3.16}$$

- pMOS transistor

$$\begin{aligned}
&\left|V_{Tp}\right| = \left|V_{T0p}\right| + \gamma\left(\sqrt{2\left|\Phi_{fn}\right| + \left|V_{SB}\right|} - \sqrt{2\left|\Phi_{fn}\right|}\right)\\
&I_{DS} = \frac{\mu_p C_{ox}}{2}\left(\frac{W_p}{L_p}\right)\left[2\left(V_{SG} - \left|V_{Tp}\right|\right)V_{SD} - V_{SD}^2\right] \text{ for } V_{SG} \geq \left|V_{Tp}\right| \text{ and } V_{SD} < V_{SG} - \left|V_{Tp}\right|\Big)\\
&I_{DSsat} = \frac{\mu_p C_{ox}}{2}\left(\frac{W_p}{L_p}\right)\left(V_{SG} - \left|V_{Tp}\right|\right)^2\left(1 + \lambda V_{SD}\right) \text{ for } V_{SG} \geq \left|V_{Tp}\right| \text{ and } V_{SD} \geq V_{SG} - \left|V_{Tp}\right|\Big)
\end{aligned} \tag{3.17}$$

For short-channel devices,

- nMOS transistor

$$
\begin{aligned}
&V_{DSsat} = \frac{(V_{GS} - V_{Tn})\, E_{satn} L_n}{(V_{GS} - V_{Tn}) + E_{satn} L_n} \\
&I_{DS} = \frac{\mu_{effn} C_{ox}}{2} \left(\frac{W_n}{L_n}\right) \left[2\,(V_{GS} - V_{Tn})\, V_{DS} - V_{DS}^2\right] \frac{1}{1 + V_{DS}/E_{satn} L_n} \\
&\text{for } V_{GS} \geq V_{Tn} \text{ and } V_{DS} < V_{DSsat} \\
&I_{DSsat} = W_n v_{sat} C_{ox}\,(V_{GS} - V_{Tn} - V_{DSsat})\,(1 + \lambda V_{DS}) \qquad \text{for } V_{GS} \geq V_{Tn} \text{ and } V_{DS} \geq V_{DSsat}
\end{aligned}
\tag{3.18}
$$

- pMOS transistor

$$
\begin{aligned}
&V_{SD\,\text{sat}} = \frac{(V_{SG} - |V_{Tp}|)\, E_{\text{satp}}\, L_p}{(V_{SG} - |V_{Tp}|) + E_{\text{satp}}\, L_p} \\
&I_{DS} = \frac{\mu_{effn} C_{ox}}{2} \left(\frac{W_p}{L_p}\right) \left[2\,(V_{SG} - |V_{Tp}|)\, V_{SD} - V_{SD}^2\right] \frac{1}{1 + V_{SD}/E_{\text{satp}}\, L_p} \\
&\text{for } V_{SG} \geq |V_{Tp}| \text{ and } V_{SD} < V_{SD\,\text{sat}} \\
&I_{DSsat} = W_p v_{sat} C_{ox}\,(V_{SG} - |V_{Tp}| - V_{SD\,\text{sat}})\,(1 + \lambda V_{SD}) \qquad \text{for } V_{SG} \geq |V_{Tp}| \text{ and } V_{SD} \geq V_{SD}
\end{aligned}
\tag{3.19}
$$

The saturation electric field $E_{sat} = \frac{2v_{sat}}{\mu_{eff}}$, where v_{sat} is the saturation velocity of an electron or a hole in an electric field.

3.2.2 Propagation Delay

The propagation delay signifies the time required for a logic circuit to process its input signal and produce a stable signal at its output. Typically, it is defined as the duration between 50% of the maximum input change and 50% of the maximum output change. Since the output signal can transition either from high to low or low to high, two propagation delays, labeled as t_{pHL} and t_{pLH}, are designated accordingly. These propagation delays, t_{pHL} and t_{pLH}, can be articulated in relation to two output voltage levels: the low-level output voltage (V_{OL}) and the high-level output voltage (V_{OH}).

- t_{pLH} is defined as the duration it takes for the output voltage to transition from V_{OL} to $V_{50\%}$ of $(V_{OH} - V_{OL})$ in response to $V_{50\%}$ of $(V_{OH} - V_{OL})$ of the input signal.

- t_{pHL} refers to the time needed for the output voltage to shift from V_{OH} to $V_{50\%}$ of $(V_{OH} - V_{OL})$ in response to $V_{50\%}$ of $(V_{OH} - V_{OL})$ of the input signal.

The propagation delay is the time the transistor takes to charge or discharge the load capacitance.

$$t_{pHL} = \frac{C_L \triangle V_{HL}}{I_{HL}(avg)} = \frac{C_L(V_{OH} - V_{50\%})}{I_{HL}(avg)} \tag{3.20}$$

$$t_{pLH} = \frac{C_L \triangle V_{LH}}{I_{LH}(avg)} = \frac{C_L(V_{50\%} - V_{LH})}{I_{LH}(avg)} \tag{3.21}$$

where the average current $I_{HL(avg)}$ and $I_{LH(avg)}$ can be calculated, respectively, as follows.

$$I_{HL(avg)} = \frac{1}{2}[I_{DS}(V_{in} = V_{OH}, V_{out} = V_{OH}) + I_{DS}(V_{in} = V_{OH}, V_{out} = V_{50\%})] \tag{3.22}$$

$$I_{LH(avg)} = \frac{1}{2}[I_{DS}(V_{in} = V_{OL}, V_{out} = V_{OL}) + I_{DS}(V_{in} = V_{OL}, V_{out} = V_{50\%})] \tag{3.23}$$

The input signal is assumed to be a step function with negligible rise and fall times. Depending on its operational region, the I_{DS} follows either a linear or saturation profile. In the discharging phase, the nMOS transistor is activated to discharge the charge previously stored on the load capacitor, directing it to the ground. Consequently, the output voltage (V_{out}) will gradually decrease from its initial value (V_{OH}) towards V_{OL}. The charge and discharge phases of a CMOS inverter are illustrated in Fig. 3.2.

Initially, the nMOS transistor operates in its saturation region and then transitions into its linear region as the V_{out} falls below V_{OH}, reaching approximately V_{T0n}. Conversely, the charging phase mirrors the discharging phase, with the nMOS and pMOS transistors reversed roles. During the charging phase, the pMOS transistor begins in its saturation region due to the low output voltage (V_{OL}), which is typically lower than the absolute value of the pMOS threshold voltage (V_{T0p}). This condition persists until the output voltage rises to $V_{OL} + V_{T0p}$, at this point, the pMOS transistor transitions into its linear operational region and remains there for the remainder of the relevant charging phase.

3.2.3 Power-Delay Product

The primary trade-off in most VLSI designs revolves around power consumption and delay. The PDP is utilized to assess this trade-off. It quantifies the energy needed for a gate to execute a specific operation, such as transitioning from a low to a high state or vice versa. To define the PDP more precisely, let's assume that the gate is switched at its highest possible rate of $f_{max} = \frac{1}{2}t_{pd}$; where t_{pd} represents the propagation delay of the gate. In this context, the PDP can be formulated as follows.

$$PDP = P_{avg}.t_{pd} = C_L {V_{DD}}^2 f.\frac{1}{2f} = \frac{1}{2}C_L {V_{DD}}^2 \tag{3.24}$$

3.3 Validation of PVT Variability on Circuit Performances

Tables 3.1 and 3.2 showcase the compilation of PVT parameters across both CMOS and FinFET technologies as delineated in the proposed framework (chapter 4). The corresponding variations across

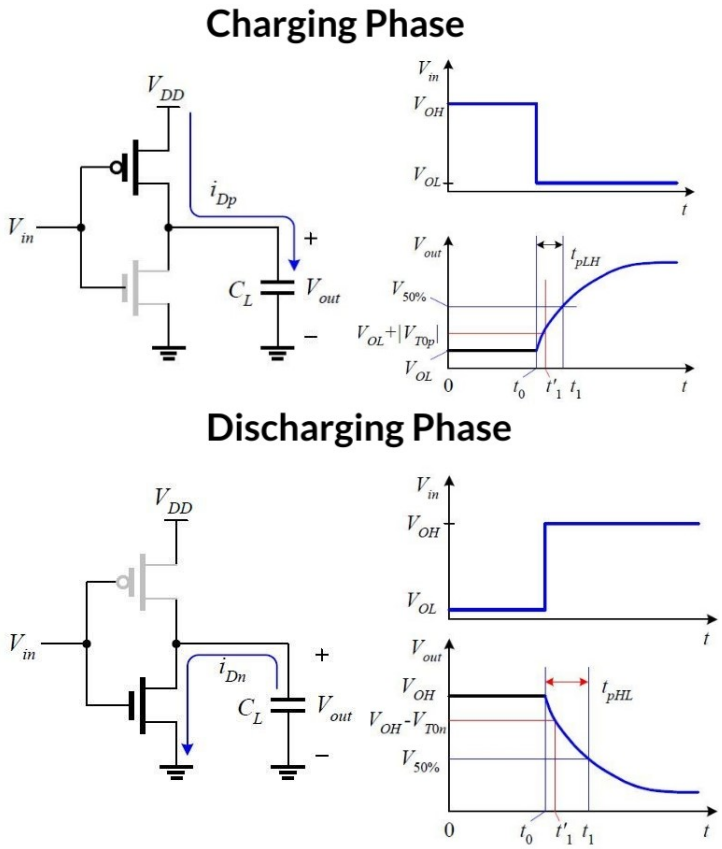


Figure 3.2: Charging and discharging phases of a typical CMOS inverter with timing diagrams [66]

different nodes are detailed in tables 3.4 and 3.5. SPICE MC simulations were utilized to validate the relationships in fundamental transistor equations (section 3.2). Our analysis of the temperature's impact on power and delay is depicted in Fig. 3.3, with the supply voltage held constant. There is an exponential surge in leakage and power with rising temperature, which is particularly notable in FinFET technology compared to CMOS (Fig. 3.3 (a), (c), (d), (f)). This effect is compounded by the increase in supply voltage as depicted in these plots. We note that delay decreases non-linearly with temperature in FinFET technologies while it increases in CMOS technologies (Fig. 3.3 (b), (e)). While keeping the temperature constant, we varied the supply voltage and observed a non-linear relationship between supply voltage, power, and delay. The observations are plotted in Fig. 3.4. At elevated temperatures, this effect becomes more pronounced. Additionally, we can discern an inverse correlation between leakage and delay as they vary with temperature and supply voltage.

Load capacitance strongly influences propagation delay and power. Both delay and power demonstrate a linear increase with rising capacitive load, as illustrated in Fig. 3.5(a) for a full adder logic cell. Input slew also plays a crucial role in determining delay. In both FinFET and CMOS technologies, delay increases with increasing input slew, although the effect is more pronounced in FinFET and relatively insignificant in CMOS (Fig. 3.5(b)).

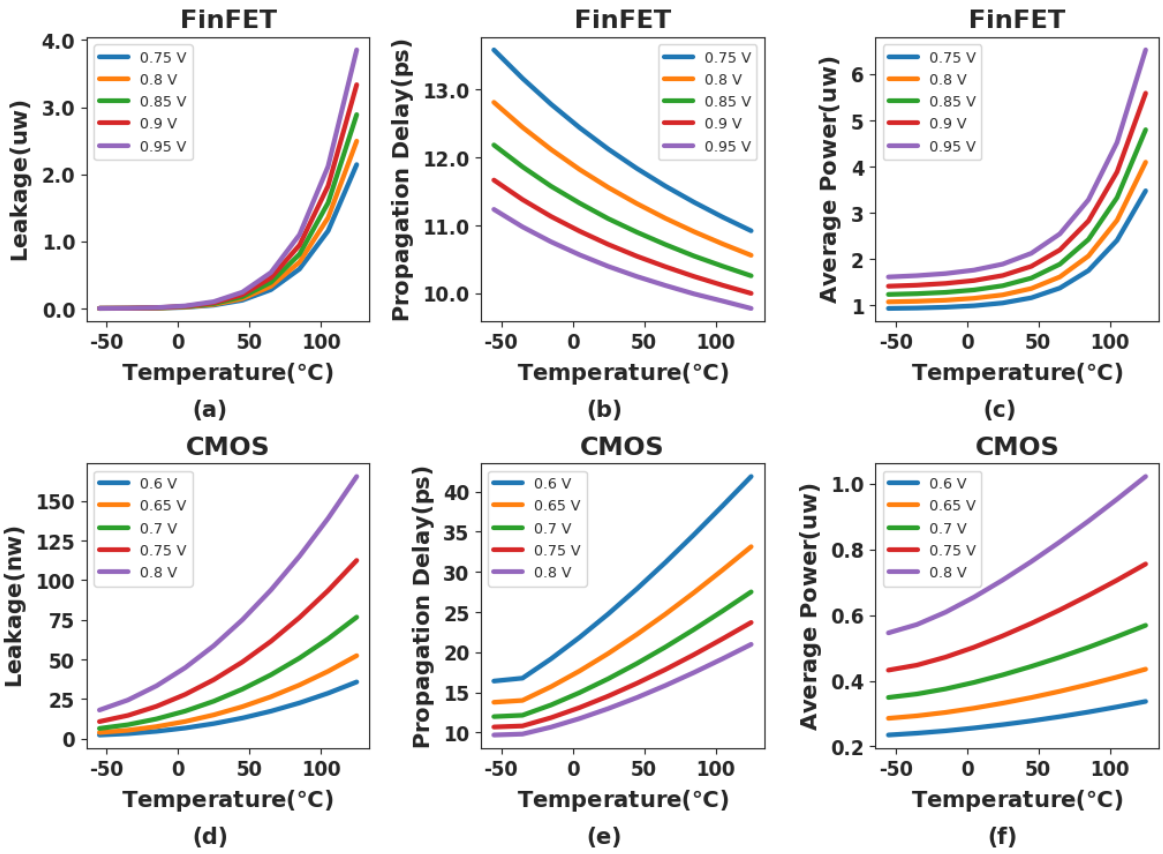


Figure 3.3: Temperature impact on 16nm CMOS/FinFET power and delay

Furthermore, we conducted simulations to visualize the influence of each process parameter on both leakage and delay. The influence of PVT over leakage and delay in 16nm FinFET and CMOS are plotted in Figs. 3.6, 3.7, 3.8, 3.10 with full adder standard cell. The resulting plots exhibit distributions as outlined in Tables 3.5 and 3.4 across 100 simulations. We validated these derived relationships from the simulations and the fundamental leakage and delay equations (3.5 to 3.7, 3.20 and 3.21).

In summary, our observations reveal that FinET leakage experiences an exponential rise with temperature and fin thickness while decreasing exponentially with the physical oxide thickness of nFET. Leakage demonstrates a linear increase with supply voltage, equivalent gate dielectric thickness, fin height, conduction band density, and S/D doping concentration while decreasing with channel doping concentration and intrinsic carrier concentration. Remarkably, we noted minimal fluctuations in leakage concerning *Nbody*, *Ni0sub*, and *Nsd* and it remains constant with *Toxpp*.

The propagation delay in FinFETs displays a linear decrease with temperature, supply voltage, equivalent gate dielectric thickness of pFET, physical oxide thickness, nFET fin height, thickness, and conduction band density. Conversely, it exhibits a linear increase with nFET gate dielectric thickness, pFET fin height and thickness, and S/D doping concentration. Notably, intrinsic carrier and channel doping concentrations do not exert any discernible influence on the delay.

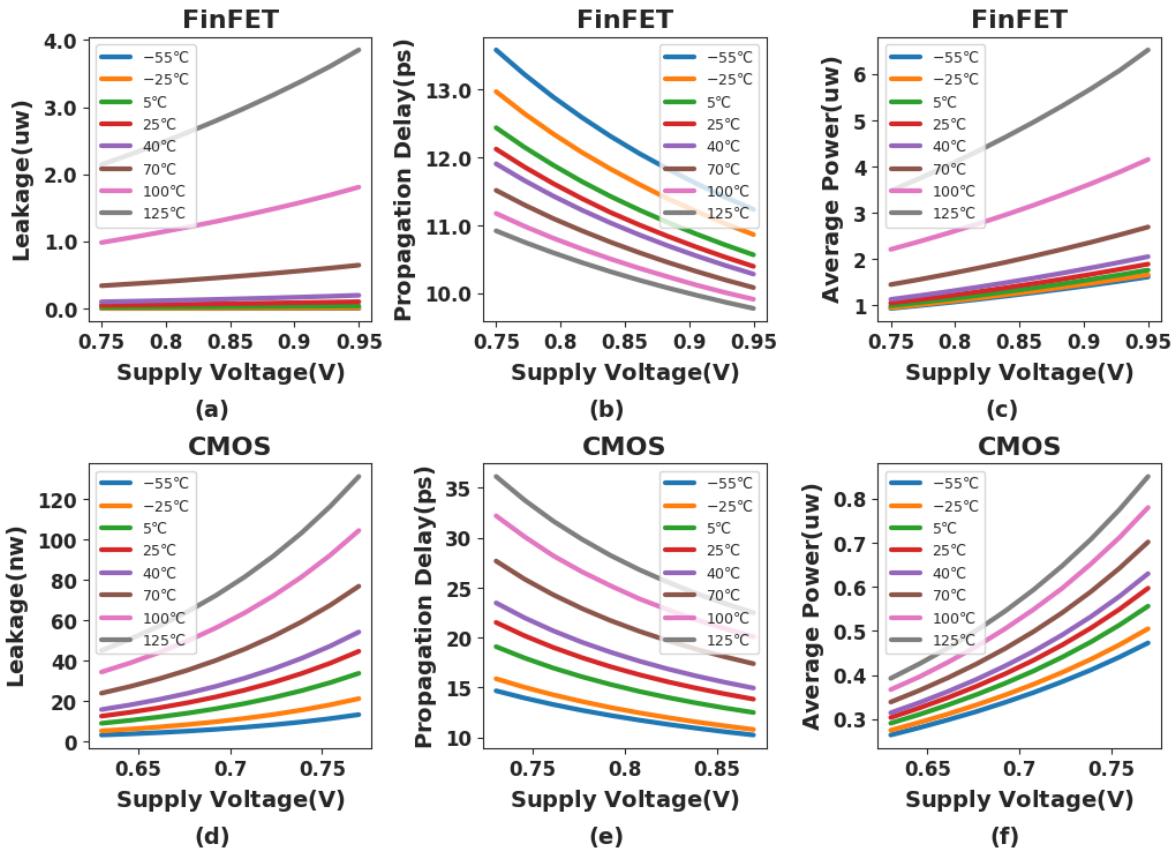
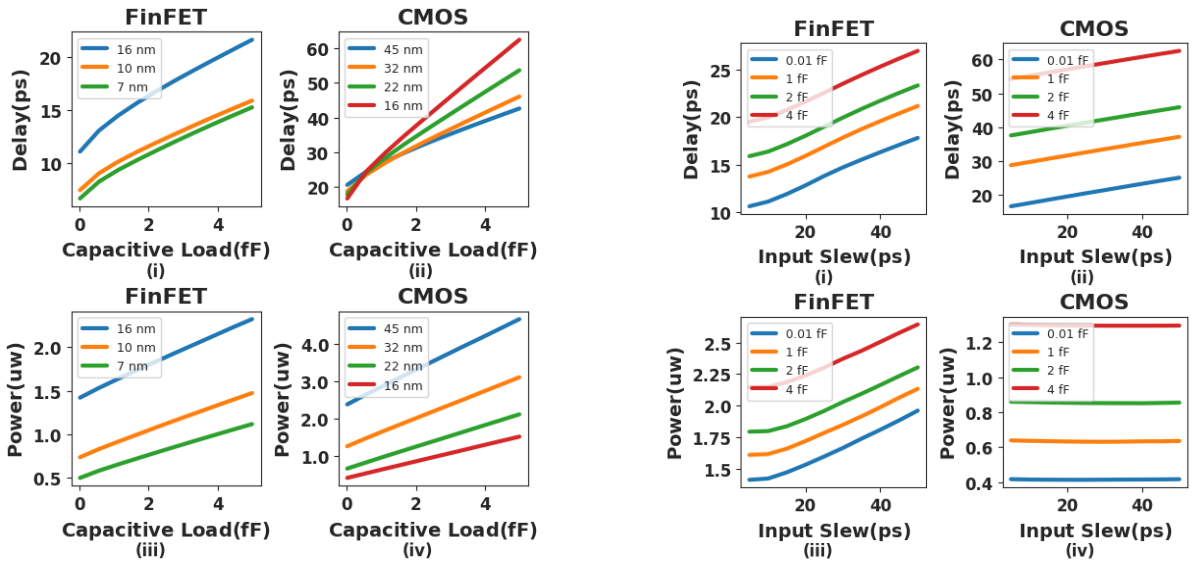


Figure 3.4: Supply voltage impact on 16nm CMOS/FinFET power and delay

Even slight variations in PVT conditions can yield notable disparities in circuit behavior, especially concerning leakage and delay. Illustrated in Fig. 3.9, we showcase the impact of PVT variations, measured at 3σ deviations from nominal values, on leakage and delay using a standard 2-input AND FinFET cell. $\mu_{Max.dev}$ in Fig. 3.9 is the maximum deviation in leakage/delay from the mean value (at nominal PVT). The findings underscore a significant influence, with leakage displaying an average deviation of approximately 150nW from the nominal value across FinFET technology nodes of 16nm, 10nm, and 7nm. Similarly, the delay demonstrates an average statistical deviation of nearly 13ps across these nodes. This highlights how even a seemingly simple AND cell can induce substantial deviations. Such effects are magnified across all cells within an IC resulting in considerable deviations from specified performance levels. Conducting PVT-aware performance analysis in the early stages of circuit design enables the implementation of effective compensation techniques and unlocks avenues for enhancing overall metrics.



(a) (b)

Figure 3.5: Impact of load capacitance and slew time on CMOS/FinFET power and delay across different technology nodes

3.4 Simulation Setup and Performance Modeling of Digital Circuits

We have generated crucial training data for PVT-aware surrogate modeling,, leveraging PTM representations of both BSIM4 (CMOS) and BSIM-CMG (FinFETs) models [67], [68]. By scrutinizing the electrical equations governing power and delay, we identified the process variables responsible for introducing performance discrepancies in digital circuits. These variations were validated through simulations and subsequently incorporated into our modeling process. Notably, fluctuations in supply voltage, temperature, mobility, threshold voltage, and oxide thickness significantly affect a digital circuit's power and delay characteristics. Moreover, the threshold voltage is intricately tied to oxide thickness, substrate doping concentration, and junction depth. Additionally, factors such as load capacitance, parasitic capacitances, and input slew time influence propagation delay, while dynamic power depends on the switching factor, operating frequency, and process parameters. To model the primary performances in sub-micro and nano nodes, namely leakage power and propagation delay, we comprehensively considered all process variables impacting power and delay. These variables are detailed in Tables 3.1 and 3.2.

Corner case analysis is the prevailing method for conducting reliability assessments to ascertain how a device performs under varying PVT conditions. Each unique combination is referred to as a 'design corner.'

- The worst-power or highest-speed corner combines the lowest temperature with the highest operating voltage. This corner utilizes fast-n/fast-p parameters and defines the lowest permissible

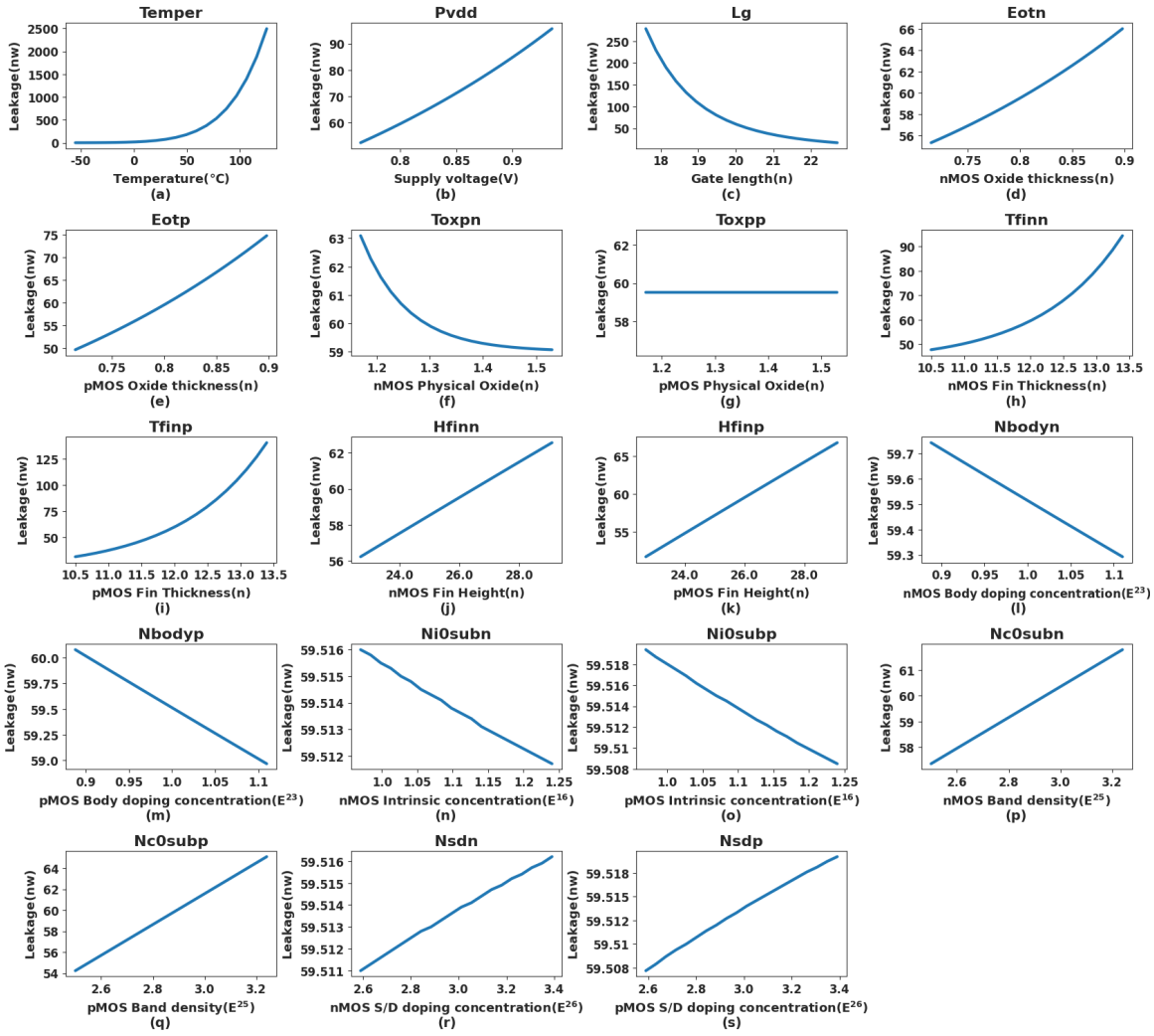


Figure 3.6: PVT parameter's impact on leakage power in 16nm FinFET technology

operating temperature and the maximum allowable supply voltage. Its objective is to evaluate power dissipation and assess the hold time constraints of the circuit.

- The best-power or worst-speed corner, on the other hand, combines the highest temperature with the lowest operating voltage. This corner employs slow-n/slow-p parameters and establishes the highest permissible operating temperature and the minimum allowable supply voltage. It is employed to evaluate circuit speed and examine setup time constraints.

The behavior of digital circuits has become increasingly nonlinear and unpredictable due to the multitude of process variations affecting them, particularly at advanced nodes. This renders corner case analysis less effective [42]. For instance, a 7nm technology node involves over 100 process parameters. Additionally, the combined impact of various process parameters, along with temperature and voltage fluctuations, significantly impacts circuit behavior compared to nodes before 45nm. Therefore, vali-

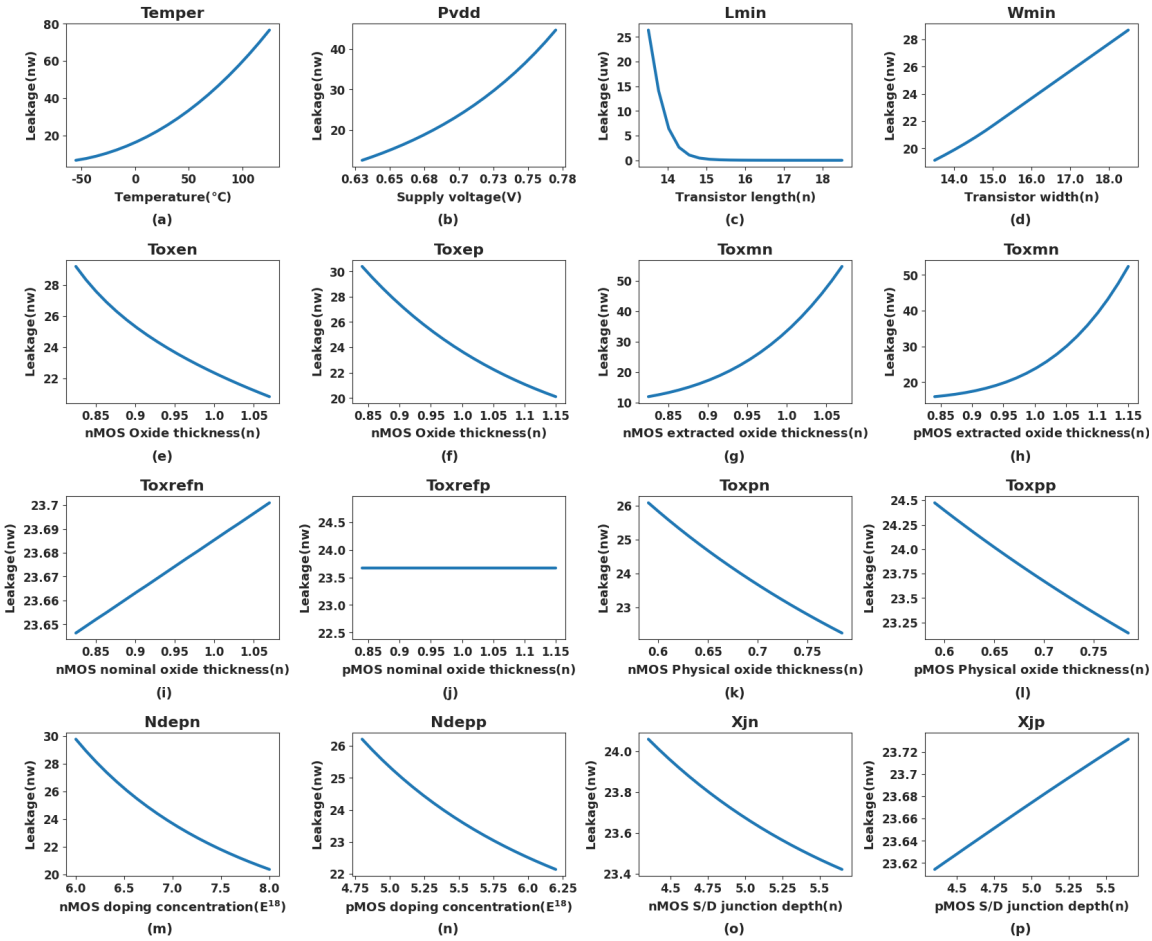


Figure 3.7: PVT parameter's impact on leakage power in 16nm CMOS technology

dating the design across the entire spectrum of variations is imperative. To address this challenge, we employ a statistical modeling approach, intermittently introducing PVT variations at each node under consideration. The list of process parameters considered for modeling, along with their respective variations, is tabulated in Tables 3.1, 3.2, 3.4, and 3.5. We conducted modeling of leakage and power in digital circuits across CMOS 45nm, 32nm, 22nm, and 16nm nodes, as well as FinFET 16nm, 10nm, and 7nm high-performance technology nodes. However, the design of the surrogate framework is akin to a black box model, facilitating easy extension to any technology node of interest.

The process variations are represented as a normal distribution with 3σ variations around the mean, simulating the statistical variability in manufacturing processes. Since environmental uncertainty is inherently random, it is modeled as uniform random variations. The statistical variations in the process can be represented by an arbitrary function $\overrightarrow{P}_{stat} = G(\overrightarrow{P})$, where $\overrightarrow{P}$ represents $\{P_1, P_2, ..., P_K\}$ with a set of random variations that has Gaussian distributions $N(\mu_{P_i}, {\sigma_{P_i}}^2)$ (say for i_{th} process parameter) and K represents the number of process parameters of interest. The uniform random distributions of

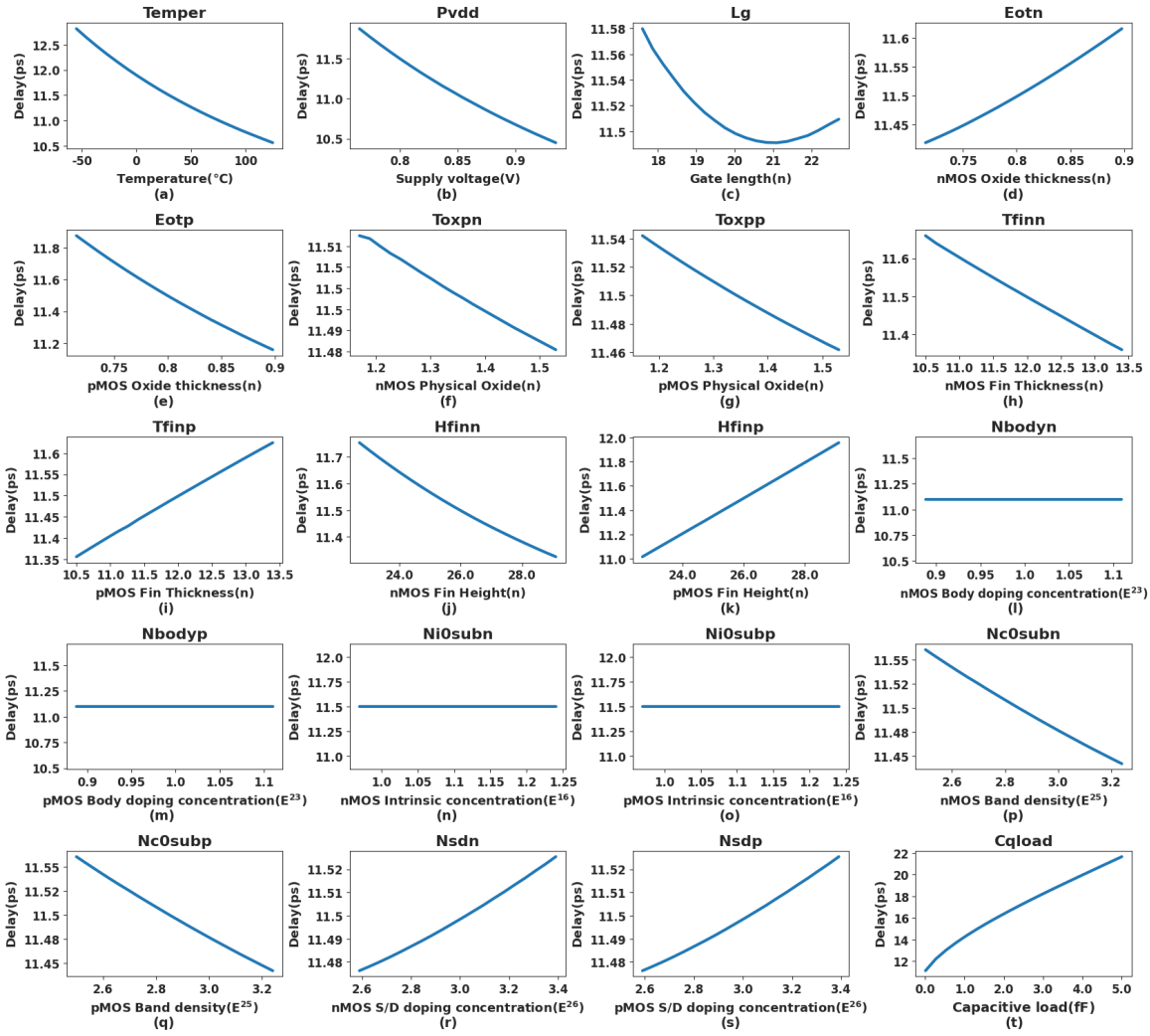


Figure 3.8: PVT parameter's impact on propagation delay in 16nm FinFET technology

operating temperature, supply voltage, and capacitive load can be represented as $T_{op} = f(T)$, $V_{dd} = f(V)$, and $C_{var} = f(C_{Load})$.

We are prototyping the PVT-aware circuit behavior using gate-level surrogate modeling. The PVT variations are introduced uniformly to each transistor in a standard logic cell, and the performance deviations are measured and recorded through SPICE MC simulations. Such data with PVT-aware standard cell simulations is employed to build a surrogate modeling framework.

3.4.1 Leakage Modeling

Leakage power is given by

$$P_{Leak} = \sum_{i=1}^{N} (I_{Leak}) \cdot V_{dd} \tag{3.25}$$

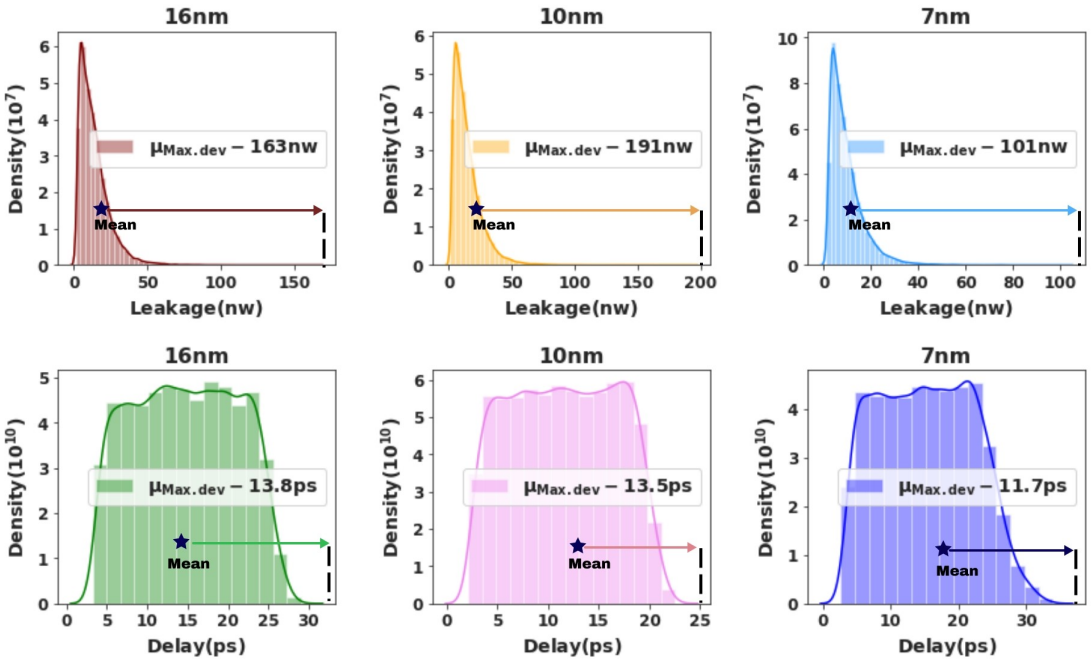


Figure 3.9: Effect on leakage and delay due to 3σ PVT variations

where P_{Leak} represents leakage power due to the I_{Leak}(leakage current) at each node of the logic cell, with N being the number of nodes. Then, the statistical variation in leakage power is given by

$$P_{Leak-Stat} = P_{Leak}|_{\{\Delta\overrightarrow{P}_{stat},\Delta T_{op},\Delta V_{dd}\}} \tag{3.26}$$

where $P_{Leak-Stat}$ is the statistical variation in leakage power induced by the normalized deviations in PVT from the nominal values represented as $\Delta\overrightarrow{P}_{stat}$, ΔV_{dd} and ΔT_{op} respectively.

3.4.2 Delay Modeling

In addition to statistical deviations in PVT, the propagation delay of a logic gate, denoted as G_{Delay}, is influenced by factors such as the input slew time t_s and equivalent load capacitance C_L. Estimated C_L is considered as explained earlier. However, for our analysis, we assumed a consistent slew time of 10ps in this work. Nonetheless, we noticed a linear correlation between delay and input slew for a specific C_L, providing a straightforward means to extend the analysis through interpolation. The delay dependencies on input slew and load capacitance are plotted in Fig. 3.5 (b).

$$G_{Delay} = f(C_L, V_{DD}, t_s) \tag{3.27}$$

We generate uniform random distributions of C_L within the prescribed range of load combinations, encompassing the minimum load (inverter) and maximum load (Full adder). These values typically

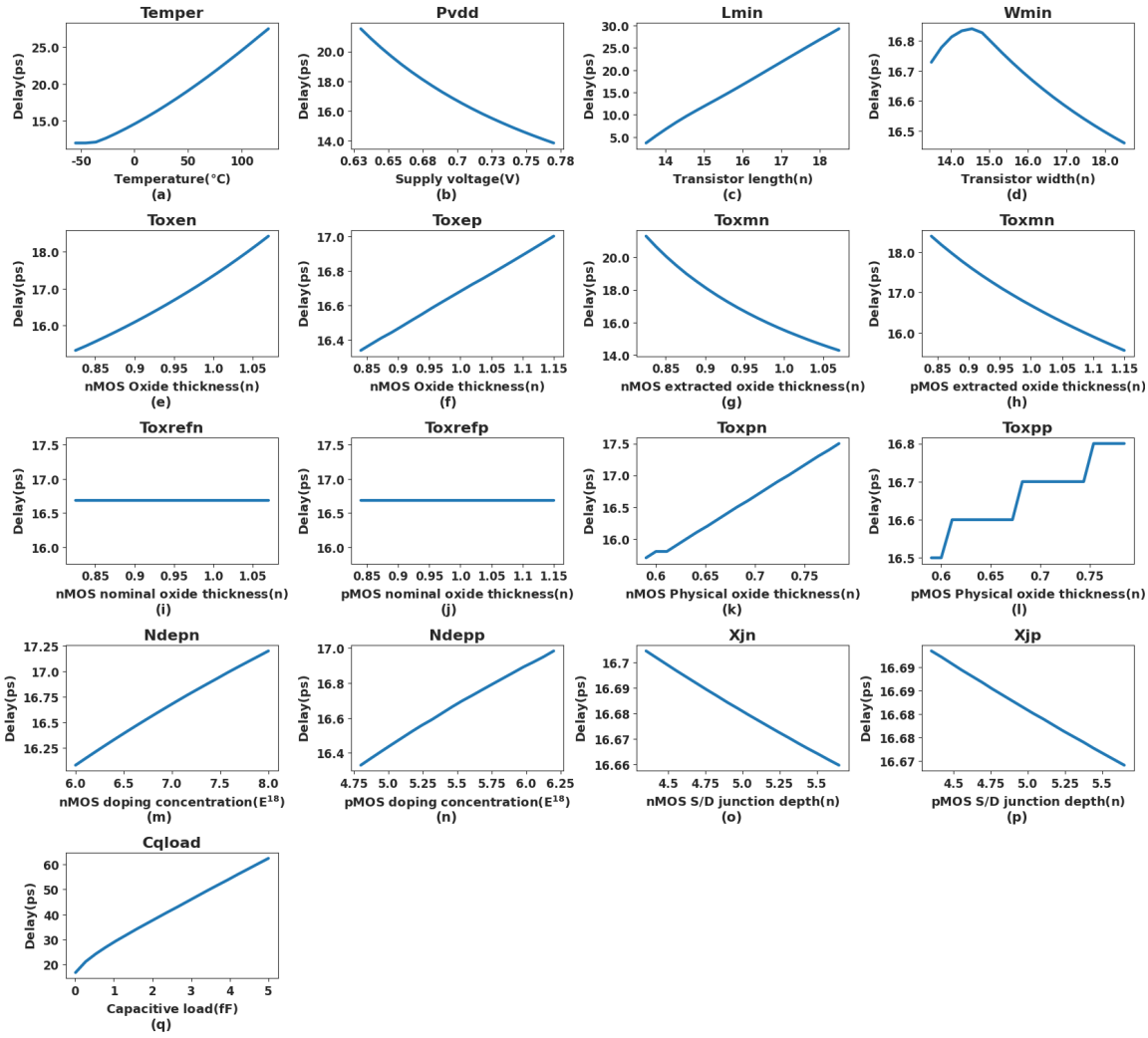


Figure 3.10: PVT parameter's impact on propagation delay in 16nm CMOS technology

range from 0.01 to 5 fF (femto farads), and they are employed for delay estimation in conjunction with PVT considerations. $G_{Delay-Stat}$, the statistical delay of each standard cell acting as driver given by

$$G_{Delay-Stat} = G_{Delay}|_{\{\Delta\overrightarrow{P}_{stat},\Delta T_{op},\Delta V_{dd}\}} \tag{3.28}$$

We derived a range of capacitive load values, ranging from the minimum load represented by an inverter to the maximum load represented by a full adder. Within this range, uniform random distributions were computed and integrated into delay estimations, accounting for PVT considerations. A comprehensive explanation of load calculation is provided in section 3.4.3. Subsequently, we delve into the formulation of equivalent capacitance estimation.

Simulation data sets for leakage and delay are generated through SPICE MC DC and transient analysis, respectively.

Table 3.1: CMOS Process, Voltage, Temperature parameters considered

Process parameter	Description	Type of parameter
L	Channel length	Design parameters
W	Channel width	
Toxe	Electrical gate equivalent oxide thickness	Process parameters
Toxm	Tox at which parameters are extracted	
Toxref	Nominal gate oxide thickness for the gate dielectric tunneling	
Toxp	Physical gate equivalent oxide thickness	
Ndep	Channel doping concentration at the depletion edge for the zero-body bias	
Xj	Source/drain junction depth	
Temper	Temperature	Environmental parameters
Pvdd	Supply voltage	

3.4.3 Equivalent Capacitance Estimation

The intrinsic capacitance of a cell undergoes variations due to input voltage transitions and interactions between the driver and the load cell. These factors collectively contribute to the total input capacitance. An accurate estimation of overall equivalent capacitance values for different combinations of driver gates is crucial for precise delay estimations of complex cells based on standard cells. In our research, we have compiled a comprehensive database containing estimated equivalent load capacitance values for every possible combination of standard library cells at the specified technology node. We determined a range of capacitive load values by employing an inverter as the load for the minimum load condition and a full adder for the maximum load scenario. The fundamental concept involves approximating a load capacitance value by correlating the time needed to drive the cell and the time to charge or discharge a known capacitance. This procedure is illustrated in Fig. 3.11, where we estimate the propagation delay for a driver-driven cell combination in one configuration. In an alternative setup, we replace the driven cell with a variable capacitor and estimate the time required for the capacitor's charge/discharge cycle. The capacitive load at which both delays closely align is approximated as the input load presented to any preceding cell(s) in the circuit. This iterative process is applied to all standard cells acting as the load under nominal PVT conditions to establish the capacitance variation range from minimum to maximum.

Table 3.2: FinFET Process, Voltage, Temperature parameters considered

Process parameters	Description	Type of parameter
Lg	Gate length	Design parameters
Tfin	Fin thickness	
Hfin	Fin height	
Eot	SiO2 equivalent gate dielectric thickness (including inversion layer thickness)	Process parameters
Toxp	Physical oxide thickness	
Nbody	Channel (body) doping concentration	
Ni0sub	Intrinsic carrier concentration of channel at 300.15K	
Nc0sub	Conduction band density of states at 300.15K	
Nsd	Source/Drain doping concentration	
Temper	Temperature	Environmental parameters
Pvdd	Supply voltage	

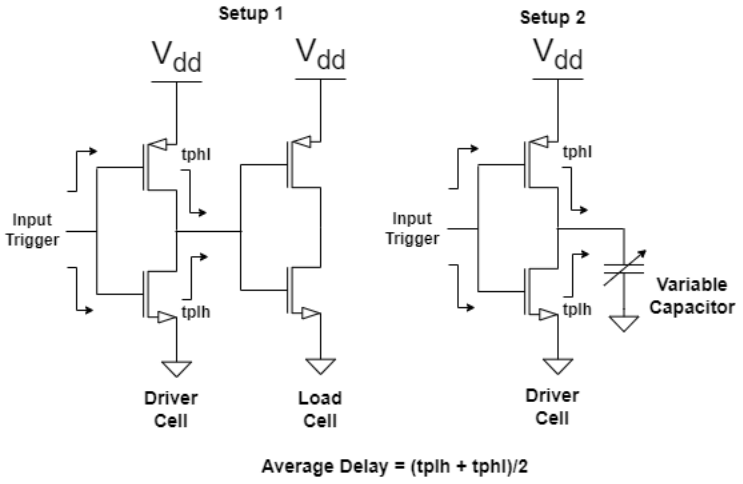


Figure 3.11: Equivalent load capacitance computation setup

Our investigation revealed that the Inverter cell, comprising only two transistors, serves as the minimum load, while a full adder cell, typically incorporating 28 transistors, represents the maximum load among the standard cells considered in our study. Therefore, we determined the limits of capacitive load variation by considering inverter-inverter and inverter-full adder as minimum and maximum driver-load

combinations. We further extended the maximum C_L variation value by considering a fanout of 4 over the resultant inverter-full adder value and maintaining a small margin. Importantly, we observed that the node where the load gate under examination is being driven, along with the type of other input nodes, exerts a significant influence. These factors include internal and parasitic capacitances, which vary based on the active transistor stacks for a given combination in a driver-driven standard cell [69].

Introducing the development of the equivalent capacitance matrix at the standard cell level, we estimated C_L for all possible combinations of driver-driven cells using the aforementioned methodology and then computed their average for the final value. The C_L values for 13 standard cells in our setup for each technology node are tabulated in Table 3.3. It is essential to emphasize that when estimating propagation delay in the circuit's critical path, one must consider a linear multiple of the capacitive load value to accommodate any fanout considerations.

3.5 Analog Circuit Modeling and Simulation Setup

The intricacies and complexities abound in analog circuits, primarily due to the critical reliance on transistor dimensions and passive components. These circuits are notably susceptible to the unpredictable random variations of fabrication processes, temperature variations, and bias conditions. The manifestation of PVT variations can precipitate substantial performance degradation, sometimes rendering the circuit non-functional. The strong dependency on the intuition, skills, and expertise of designers, with a dearth of formalization, constricts the dissemination and reuse of knowledge in this domain. Moreover, attempting to engineer solutions for complex circuits through conventional mathematical modeling, with its inherent inability to foresee all scenarios and temporal changes, often leads to extended design cycles, time-to-market, and heightened production costs. In response to these challenges, our research introduces automated circuit modeling, a paradigm shift that significantly accelerates the design and development process while simultaneously enhancing performance.

Our core concept revolves around collecting input and output data points for the Circuit Under Test (CUT), enabling the creation of a surrogate model that reliably replicates its statistical behavior. This surrogate model, in turn, facilitates the rapid generation of solutions within shorter timeframes. To establish our training datasets, we employ a meticulous approach, subjecting the CUT to thorough biasing and simulating design and process parameters randomly across a predefined space, spanning different corners. The ultimate objective is the formulation of an optimized multi-corner robust model with a wide range of PVT and design parameter variations. This robust model(s) is employed for self-adaptation of the microelectronic circuits.

Fig. 3.12 outlines the training data generation for analog circuit modeling. For a given analog circuit under examination, a simulation dataset is generated to capture the variations in M output specifications, denoted as Y_M, where $Y_M \in \{Y_1, Y_2, ..., Y_m\}$. These variations occur due to changes in PVT, as well as diverse design parameters varied across the design space represented by X_K, where $X_K \in \{X_1, X_2, ..., X_K\}$. For example, the value of the reference current, Iref (output) as a function of vari-

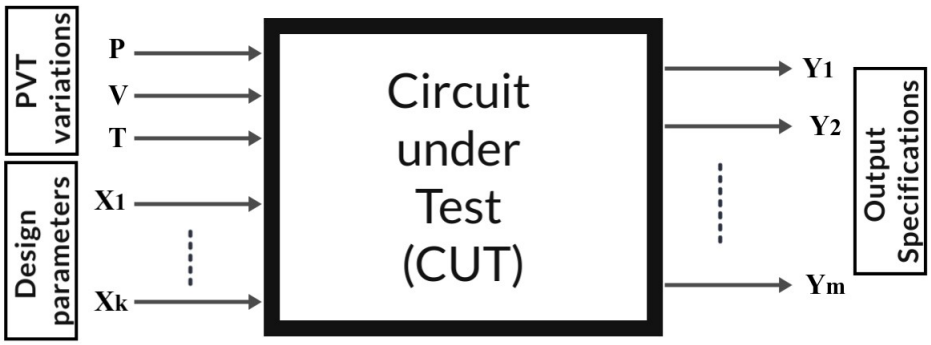


Figure 3.12: Training data generation of analog circuit under test

ables P, V and T is calculated for each PVT combination in the current reference design. At first, the schematic of the analog CUT is designed using Cadence Analog Design Environment (ADE). This is succeeded by setting up biasing configurations, scrutinizing operating point margins, aligning the design to typical conditions, and subsequently generating its netlist from Spectre. The simulation dataset is created, considering the circuit netlist and the PDK as inputs and sweeping the PVT and design parameters across the corners. The PDK serves as a comprehensive database provided by each silicon foundry, tailored for a specific process technology. It encompasses all essential information and files indispensable for the physical design of integrated circuits in that particular process. More precisely, the design system employed in this context extracts the SPICE model parameters of the devices from the PDK, enabling simulations to be executed.

3.6 Conclusion

In conclusion, digital and analog circuits are fraught with intricacies and complexities, largely stemming from their heavy reliance on transistor dimensions and passive components. These circuits are particularly vulnerable to the unpredictable variations inherent in fabrication processes, temperature fluctuations, and bias conditions. The emergence of PVT variations can lead to significant performance degradation, at times rendering the circuit non-functional. The heavy dependence on designers' intuition, skills, and expertise, coupled with a lack of formalization, hampers the dissemination and reuse of knowledge in this domain. Conventional mathematical modeling struggles to address all scenarios and temporal changes, resulting in prolonged design cycles, delayed time-to-market, and increased production costs. In response, our research introduces automated circuit modeling, representing a paradigm shift that accelerates the design and development process while simultaneously enhancing performance. Central to our approach is the creation of surrogate models based on thorough data collection from the CUT, which replicates its statistical behavior reliably. These surrogate models enable rapid solution generation within shorter time frames. Through meticulous training dataset generation, incorporating varied PVT and design parameter variations, we aim to develop optimized multi-corner robust models. The outlined process for training data generation, leveraging simulation datasets through PTM and

Table 3.3: Equivalent load capacitance estimations of digital standard cells for CMOS high-performance technology nodes 45nm, 32nm, 22nm, and 16nm and FinFET nodes 16nm, 10nm, and 7nm

Standard Cell/ Technology node	**CMOS**				**FinFET**		
	45nm	**32nm**	**22nm**	**16nm**	**16nm**	**10nm**	**7nm**
INVERTER	0.126	0.073	0.041	0.026	0.215	0.142	0.112
2-input NAND	0.193	0.117	0.068	0.044	0.278	0.183	0.140
2-input AND	0.193	0.117	0.068	0.044	0.278	0.183	0.140
2-input NOR	0.213	0.126	0.073	0.047	0.364	0.236	0.177
2-input OR	0.213	0.126	0.073	0.047	0.364	0.235	0.177
2-input XOR	0.425	0.252	0.146	0.093	0.616	0.398	0.319
2-input XNOR	0.643	0.361	0.201	0.12	0.892	0.584	0.456
3-input NAND	0.25	0.155	0.092	0.061	0.353	0.229	0.174
3-input AND	0.25	0.155	0.092	0.061	0.353	0.229	0.174
3-input NOR	0.301	0.18	0.105	0.068	0.512	0.329	0.263
2x1 MULTIPLEXER	0.443	0.321	0.211	0.135	0.467	0.307	0.245
FULL ADDER	1.367	0.834	0.492	0.276	2.006	1.279	0.987
AND-OR INVERT12	0.301	0.18	0.105	0.068	0.418	0.274	0.208

PDKs, highlights the systematic approach employed to navigate the complexities of digital and analog circuit modeling and design.

Table 3.4: CMOS Process, Voltage and Temperature Variations considered across 45nm to 16nm high-performance technology nodes

Process parameter	Device	45nm Technology node			32nm Technology node			22nm Technology node			16nm Technology node		
		Lower dev.	Nominal	Higher dev.	Lower dev.	Nominal	Higher dev.	Lower dev.	Nominal	Higher dev.	Lower dev.	Nominal	Higher dev.
L (nm)	both	39	45	51	26	32	36	19	22	25	14	16	18
W (nm)	both	39	45	51	26	32	36	19	22	25	14	16	18
Toxe (E-9)	nMOS	1.1	1.25	1.4	0.98	1.15	1.3	0.92	1.05	1.2	0.825	0.95	1.07
	pMOS	1.12	1.3	1.46	1.05	1.2	1.35	0.96	1.1	1.25	0.85	1	1.14
Toxm (E-9)	nMOS	1.1	1.25	1.4	0.98	1.15	1.3	0.92	1.05	1.2	0.825	0.95	1.07
	pMOS	1.12	1.3	1.46	1.05	1.2	1.35	0.96	1.1	1.25	0.85	1	1.14
Toxref (E-9)	nMOS	1.1	1.25	1.4	0.98	1.15	1.3	0.92	1.05	1.2	0.825	0.95	1.07
	pMOS	1.12	1.3	1.46	1.05	1.2	1.35	0.96	1.1	1.25	0.85	1	1.14
Toxp (E-9)	nMOS	0.885	1	1.12	0.78	0.9	1	0.705	0.8	0.91	0.6	0.7	0.78
	pMOS	0.885	1	1.12	0.78	0.9	1	0.705	0.8	0.91	0.6	0.7	0.78
Ndep (E+18)	nMOS	2.74	3.24	3.65	3.56	4.12	4.64	4.83	5.5	6.25	6.03	7	8.09
	pMOS	2.15	2.44	2.77	2.7	3.07	3.49	3.7	4.4	5	4.82	5.5	6.19
Xj (E-8)	nMOS	1.22	1.4	1.6	0.86	1	1.12	0.62	0.72	0.8	0.437	0.5	0.565
	pMOS	1.22	1.4	1.6	0.86	1	1.12	0.62	0.72	0.8	0.437	0.5	0.565
Temper (°C)	both	-55	25	125	-55	25	125	-55	25	125	-55	25	125
Pvdd (V)	both	0.9	1	1.1	0.81	0.9	0.99	0.72	0.8	0.88	0.63	0.7	0.77

Table 3.5: FinFET Process, Voltage and Temperature Variations considered across 45nm to 16nm high-performance technology nodes

Process parameter	Device	16nm Technology node			10nm Technology node			7nm Technology node		
		Lower dev.	Nominal	Higher dev.	Lower dev.	Nominal	Higher dev.	Lower dev.	Nominal	Higher dev.
Lg (nm)	both	17.6	20	22.7	12.2	14	15.7	9.51	11	12.4
Tfin (nm)	nFET	10.5	12	13.4	7.03	8	9.01	5.63	6.5	7.4
	pFET	10.5	12	13.4	7.03	8	9.01	5.63	6.5	7.4
Hfin (nm)	nFET	22.7	26	29.1	17.9	21	23.6	15.7	18	20.2
	pFET	22.7	26	29.1	17.9	21	23.6	15.7	18	20.2
Eot (nm)	nFET	0.715	0.8	0.898	0.597	0.68	0.767	0.538	0.62	0.695
	pFET	0.715	0.8	0.898	0.597	0.68	0.767	0.538	0.62	0.695
Toxp (nm)	nFET	1.17	1.35	1.53	1.05	1.2	1.35	1.01	1.15	1.29
	pFET	1.17	1.35	1.53	1.05	1.2	1.35	1.01	1.15	1.29
Nbody (E+22)	nFET	8.88	10	11.1	2.18	2.5	2.85	0.88	1	1.12
	pFET	8.88	10	11.1	2.18	2.5	2.85	0.88	1	1.12
Ni0sub (E+15)	nFET	9.7	11	12.4	9.7	11	12.4	9.7	11	12.4
	pFET	9.7	11	12.4	9.7	11	12.4	9.7	11	12.4
Ni0sub (E+25)	nFET	2.5	2.86	3.24	2.5	2.86	3.24	2.5	2.86	3.24
	pFET	2.5	2.86	3.24	2.5	2.86	3.24	2.5	2.86	3.24
Nsd (E+26)	nFET	2.59	3	3.39	2.59	3	3.39	2.59	3	3.39
	pFET	2.59	3	3.39	2.59	3	3.39	2.59	3	3.39
Temper (°C)	both	-55	25	125	-55	25	125	-55	25	125
Pvdd (V)	both	0.72	0.8	0.88	0.675	0.75	0.825	0.675	0.75	0.825

Chapter 4

PVT-aware Surrogate Modeling of VLSI Circuits

In the realm of nanometer-scale technology, it is crucial to accurately and efficiently predict the performance of CMOS/FinFET devices in the presence of random variations in process, temperature, and supply voltage before IC fabrication. Typically, methods like MC simulations from traditional tools such as SPICE and Prime Time estimate leakage and delay properties [69] of digital and AMS circuits. However, while providing precise results, SPICE MC simulations, come at the expense of extensive computational time [32]. Given the current landscape of high-computing chips housing billions of transistors, these prolonged simulation times are impractical in meeting chip turn-around deadlines, rendering them unsuitable for assessing complex devices. One potential solution to circumvent lengthy simulation runs is the development of surrogate models that approximate circuit performance based on statistical parameters. These surrogate models can substitute the simulator in large MC simulations, significantly reducing computational overhead.

We have designed and developed an efficient surrogate modeling framework for predicting circuit performances under diverse PVT conditions across various digital/analog/mixed-signal design applications. Machine Learning algorithms power this framework, ensuring computational efficiency. Digital circuit modeling encompasses the assessment of propagation delay and leakage power, both pivotal performance metrics for digital circuits across a wide range of variations. Furthermore, the framework relies on pre-characterized gate-level models for complex cell delay and leakage estimations in contrast to the traditional circuit-specific time and resource-consuming simulations. This eliminates the need for additional simulations.

We developed customized models for key components such as current references, Low Drop-out Regulator (LDO), oscillators, Transmitter Driver (TxDr), Bandgap Reference (BGR), Phase Locked Loop (PLL), Low-Noise Amplifier (LNA), and mixers. These models are designed to accommodate variations in output specifications across diverse design configurations and under varying PVT conditions. These components constitute the fundamental blocks of any complex analog circuit. Consequently, the methodology centered around these standardized analog models can be seamlessly extended to more intricate circuit implementations. The results obtained from the surrogate model(s) are subjected to comprehensive validation against precise simulations carried out using conventional tools.

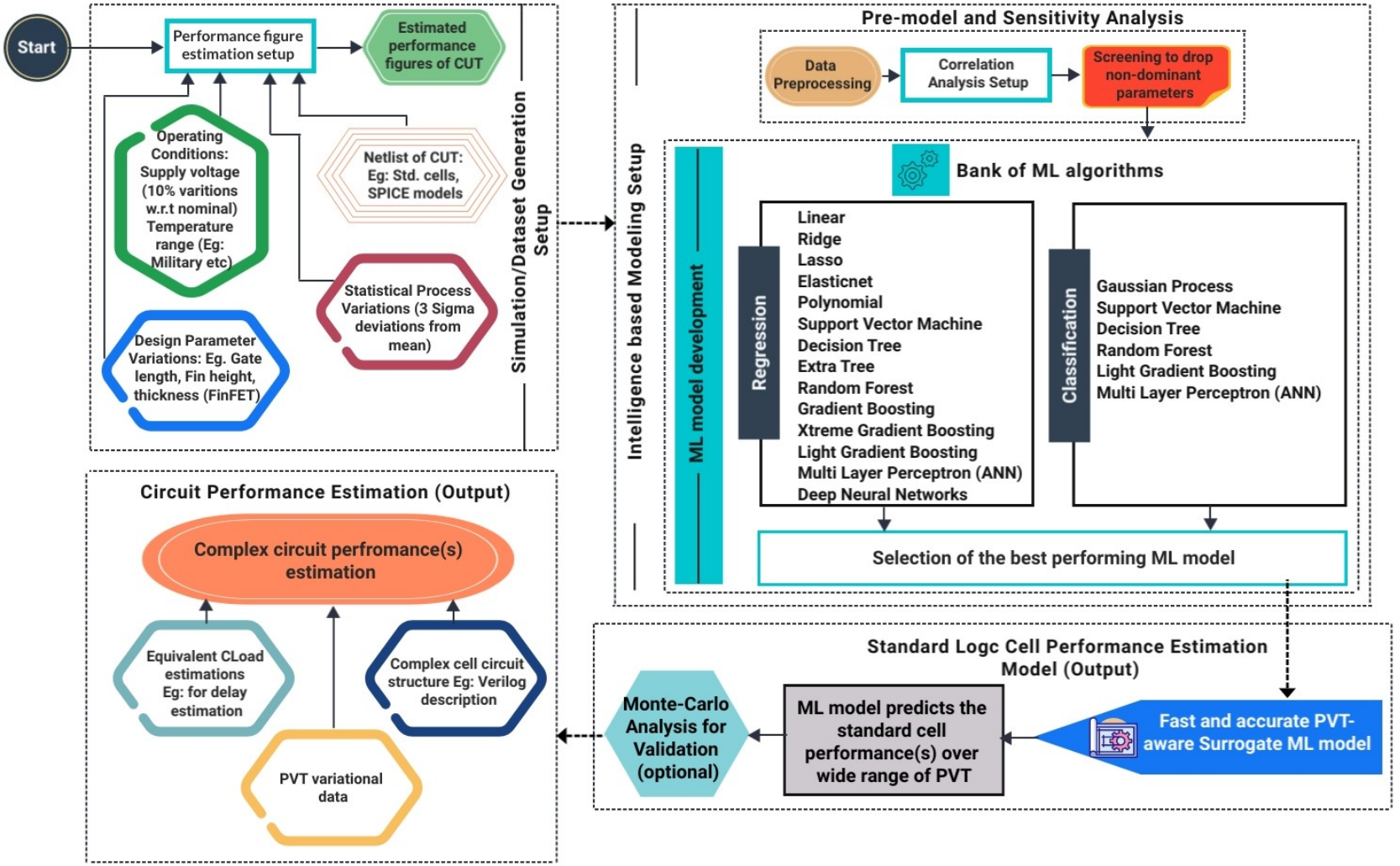


Figure 4.1: Design of the PVT-aware intelligent surrogate modeling framework for circuit performance estimation

4.1 Design of Surrogate Modeling Framework

Our designed surrogate framework utilizes an intelligent configuration comprising a bank of supervised learning algorithms adept at precisely capturing PVT-informed circuit characteristics. The subsequent sections explore the development, automation, and evaluation of this framework. As depicted in Fig. 4.1, the surrogate modeling framework is designed to automate the generation of circuit performance simulations under various operating conditions, design parameters, and process parameters. This entails the utilization of tailored netlists and model files specific to the CUT, typically representing a standard cell of interest.

- **Simulation Setup and Performance Estimation:** Simulation setup and performance estimation process are as detailed in Chapter 3. Standard cell performance estimations are generated for various PVT scenarios, serving as input datasets for subsequent modeling.

- **Data Preprocessing and Sensitivity Analysis:** An Intelligent-based modeling setup incorporates data preprocessing steps, including identifying and filtering erroneous entries and outlier detection. Furthermore, a sensitivity analysis is conducted to estimate correlation coefficients and identify non-contributing/non-dominant input variables (PVT), which are subsequently excluded from the model development process. This optimization enhances modeling speed and efficiency.

- **Supervised Learning Algorithms Bank:** The framework employs a diverse array of supervised learning algorithms tailored for both regression and classification applications. Regression-based modeling is utilized in statistical analysis setups, such as digital circuit performance estimation, while classification algorithms are employed in scenarios involving corner case analysis, such as analog circuit modeling. Dominant parameters obtained from the sensitivity analysis guide the model development process.

- **Model training and selection:** PVT variations serve as input data, while circuit performances (e.g., leakage power, propagation delay) are the target variables. Multiple ML algorithms (from the ML bank) are trained on PVT-aware simulations to generate a set of ML models. The choice of the most suitable model is determined through careful analysis of a composite performance metric.

- **Model Validation and Verification:** The resultant ML model's accuracy is verified through SPICE MC simulations, if necessary. However, it is worth noting that if the ML model meets the specified complex performance metric, additional MC simulations are deemed optional. However, they authenticate the generalization capabilities of the designed model by providing additional unseen test cases.

- **Application in Circuit Performance Estimation:** The surrogate ML models, representing statistically aware standard cell performances, facilitate their performance estimation as such and/or

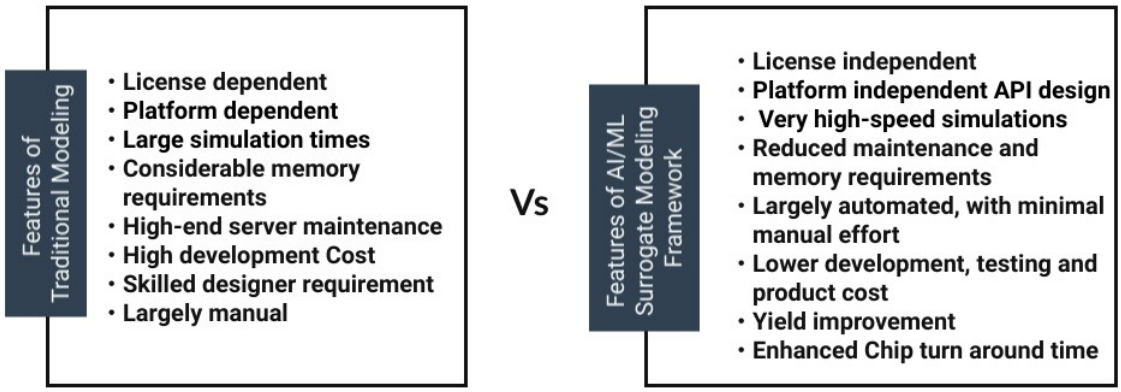


Figure 4.2: Features of AI/ML surrogate modeling framework Vs traditional modeling paradigm

can be extended to complex circuit performance estimation. For complex digital circuit estimation, given a Verilog description of the circuit structure, surrogate standard cell ML models, capacitive load estimation in case of delay estimation, automated complex cell performance estimations (e.g., leakage/delay in digital circuits) can be obtained over a wide range of PVT scenarios, bypassing the need for further simulations.

Henceforth, the AI/ML surrogate modeling framework leverages a comprehensive setup of supervised ML algorithms to model PVT-aware behavior accurately. The features of the surrogate modeling framework compared to the traditional modeling are highlighted in Fig. 4.2.

4.2 Machine Learning Algorithms for Circuit Modeling

Harnessing the automated learning capabilities offered by AI/ML algorithms in E-CAD tools has the potential to boost chip performance, streamline design processes, and ultimately result in a more efficient turnaround. The application of statistical learning techniques enables the analysis of inherent patterns in circuit simulations, allowing for the resolution and design of solutions for a multitude of problems and input-output relationships that conventionally pose challenges and demand significant time investments.

Supervised learning algorithms prove highly effective in delineating input-output correlations within intricate circuit designs and implementations through the analysis of simulation data [70]. Furthermore, ML algorithms can facilitate the construction of surrogate models when provided with simulation data incorporating output responses across diverse operating conditions and design parameter variations. These models offer significantly accelerated estimations, surpassing the speed of traditional simulators by several orders of magnitude. Early detection of functional and electrical performance discrepancies in the design cycle can notably enhance IC yield, a metric contingent upon the capabilities of simulation tools. The intricate interplay of inputs and outputs among components, processes, and diverse levels of abstraction within each level can be systematically explored using AI/ML algorithms, leveraging

insights gleaned from a spectrum of simulations and analyses. Integrating AI and ML in VLSI design streamlines the comprehension and processing of data across various levels of abstraction, thereby economizing time and effort.

Our approach employs regression algorithms to meticulously analyze and model statistical data, with a particular focus on digital/AMS standard cell modeling, as depicted in Fig. 4.1. Additionally, we explore classification techniques for corner-case analysis and to facilitate self-adaptation, as discussed in section 4.4. In instances where it proves beneficial, we apply a regression algorithm to the data classified under specific corner cases of analog circuits. This approach lays the groundwork for developing robust Classification and Regression Trees (CART) models. We incorporated 14 ML algorithms in our framework for regression tasks. They are Linear Regression (LR), Ridge Regression (RR), Lasso Regression (LAR), ElasticNet Regression (ER), Polynomial Regression (PR)2 (degree 2) and PR3 (degree 3), SVM, Decision Tree (DT), RF, Extra Tree Regression (ET), ensembles approach [71] such as Gradient Boosting Machine (GBM), Xtreme Gradient Boosting Machine (XGBM) [72], Light Gradient Boosting Machine (LGBM) [73], ANN (Multi-Layer Perceptron (MLP)) and Dense Neural Networks (DNN) [74]. We employed Logistic Regression, DT, SVMs, K-Nearest Neighbors (KNN), and LGBM for classification [12].

4.3 PVT-aware Digital Circuit Surrogate Modeling

In our research, we employ regression analysis to investigate the correlation between output variables, such as leakage/delay in digital cells, and a set of input parameters, specifically PVT variables. Our main aim is to develop a model capable of accurately predicting variations in output based on observed fluctuations in PVT conditions. To achieve this, we have conducted extensive experimentation involving various regression algorithms, resulting in a comprehensive ensemble tailored specifically for PVT-aware digital circuit performance, as detailed in Fig. 4.1. This ensemble comprises 14 algorithms, resulting in 15 ML models (we employed polynomial regression of degree 2 and 3 making them 15 models) for each standard cell across every technology node under investigation. Each algorithm within our ensemble offers unique advantages and operates within its performance constraints, primarily influenced by the selection of hyperparameters. To identify a universally applicable set of hyperparameters for each algorithm, compatible with all CMOS and FinFET digital standard cells across all technology nodes, we conducted a rigorous hyperparameter search process. This process involved a combination of manual experimentation, grid search, and BO [75, 76] to pinpoint optimal hyperparameters, including the number of iterations for each algorithm. Grid search is a widely used technique for optimizing hyperparameters in ML models [77]. On the other hand, BO is a sophisticated technique that employs probabilistic models to guide the search process toward promising regions of the hyperparameter space efficiently, particularly in complex and noisy data scenarios [78].

The procedural steps for modeling digital standard cells, as outlined in our investigation, are presented in Algorithm 1. It encompasses the sequential development of surrogate models employing all

Table 4.1: The final set of hyper-parameters of regression algorithms post-fine-tuning

Algorithm	Hyper-parameters after fine-tuning
LR	Default parameters, , solver='auto'
RR	alpha = 2, solver='auto'
LAR	alpha = 0.0001, , solver='auto'
ER	alpha = 0.001, , solver='auto'
PR	Degree = 2 or 3, solver = 'auto'
SVM	Optimized via grid search using 'rbf' kernel, C = 1000, gamma = 0.01, epsilon = 0.1
DT	Optimized via grid search with max_depth=3, max_features='auto', max_leaf_nodes=5, min_samples_leaf=3, min_weight_fraction_leaf=0.1, splitter='best'
ET	Optimized via BO with n_estimators=300, min_samples_split=2, min_samples_leaf=1, bootstrap=True, n_jobs=8, random_state=5
RF	Optimized via BO with n_estimators = 1000, min_samples_split=2, max_depth = 7, random_state = 18
GBM	Optimized via gradient descent with learning_rate=0.1, n_estimators=500, subsample=1.0, criterion='friedman_mse', min_samples_split=2, min_samples_leaf=1, max_depth=3, alpha=0.9, max_leaf_nodes=5, bagging_fraction=0.1, tol=0.0001
XGBM	Optimized via 'dart' booster with learning_rate=0.1, max_depth=4, n_estimators=500, n_jobs=20, num_parallel_tree=5, predictor='auto', random_state=5, reg_alpha=0.01
LGBM	Optimized via 'dart' with learning_rate=0.12, max_depth=5, num_leaves = 50, bagging_fraction= 1, feature_fraction= 0.6, subsample = 0.8, bagging_freq = 50, num_boost_round = 3000, early_stopping_rounds = 200
ANN (MLP)	Optimized via BO, the network has 4 hidden layers with 200x150x100x50 neurons, alpha = 0.01, max_iter = 500, activation = 'relu', Solver = 'ADAM'
DNN	Optimized via BO, the network has 4 hidden layers with neurons varying from 32 to 256 for each standard cell, learning rate between 0.001 to 0.0001, activation = 'relu', Solver = 'ADAM', epochs = 200, validation_split = 0.12, early stopping with patience = 15

the specified regression algorithms, alongside the necessary selection and fine-tuning of hyperparameters. The final set of hyper-parameters of regression algorithms post-fine-tuning are tabulated in Table 4.1. These choices aim to strike a balance between model complexity and generalization, effectively capturing the nonlinear relationship between input features and the target variable.

Our observations indicate that ET, LGBM, ANN, and DNN consistently outperform other algorithms in estimating the performance of digital circuits across various technology nodes, even with smaller datasets. Polynomial regression also demonstrates reliable performance in some cases, although selecting a polynomial degree requires careful consideration based on the number of variables and their distributions within the analog/digital CUT. Our analysis revealed that polynomial regressions of degrees 2 and 3 yielded satisfactory modeling results for the majority of the designs under examination. However, we extended our experimentation to degree 5 for predictions involving digital cell power modeling to achieve more precise estimations, albeit with the trade-off of increased training time [79].

Within our framework, we have developed an intelligent modeling setup with the goal of pinpointing the key inputs that impact circuit performance. Subsequently, we employ ML algorithms trained on these inputs, selecting the most effective model for each experiment. The process of identifying dominant parameters entails estimating statistical correlations within the simulation dataset, as elaborated in Chapter 5.

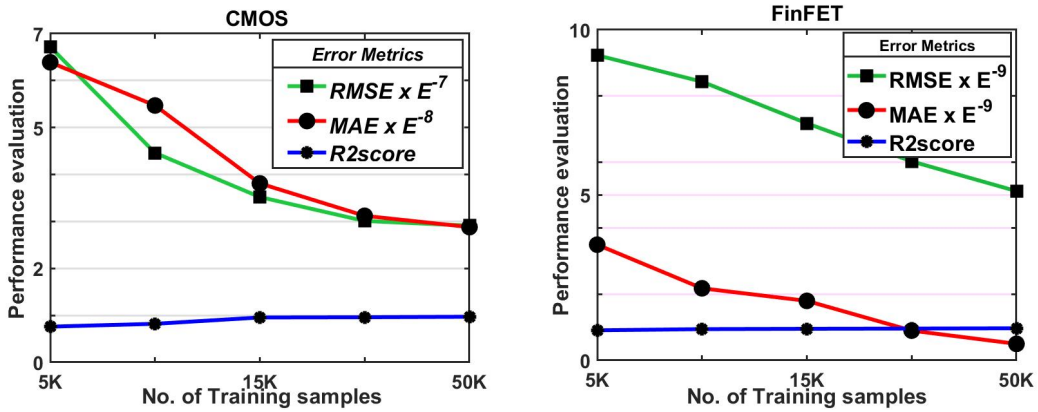
Ensuring precise estimations of complex cells necessitates thorough evaluation of the best-performing surrogate model for each standard cell, given their pivotal role in predictions. Accurate predictions at the standard cell level are indispensable for achieving precise estimations at the complex cell level. During our regression algorithm training, we noted that some models excelled with training samples but faltered with new, unseen PVT scenarios. Consequently, we've devised a comprehensive set of testing criteria (highlighted in subsection 4.3.1.2) to identify the most effective machine learning model with the optimal balance of training accuracy and generalization ability.

4.3.1 Results and Discussion

In this section, we scrutinize and deliberate on the outcomes of the regression models in our surrogate framework for estimating circuit performance. Across all experiments, we divided the datasets into training and testing sets in an 80:20 ratio, reserving 20% of the training data for validation purposes. We conducted experiments using varying sizes of simulation data to determine the optimal training set at the source node. Our goal was to select a sufficient quantity of labeled data at the source node, as this significantly impacts the effectiveness of knowledge transfer to other nodes. Notably, our experimentation revealed that the optimal performance for leakage/delay estimation is attained with a training dataset comprising 15,000 samples. The empirical findings of the LGBM regressor on 16nm CMOS and FinFET concerning varied training sample sets (demonstrated on full adder standard cell), as illustrated in Fig. 4.3, demonstrate that the most accurate modeling occurs when utilizing 15,000 samples as the training data. Fig. 4.4 displays the computational time needed by SPICE and our model (LGBM regressor) for modeling the full adder standard cell in 16nm CMOS and FinFET technologies. ML modeling

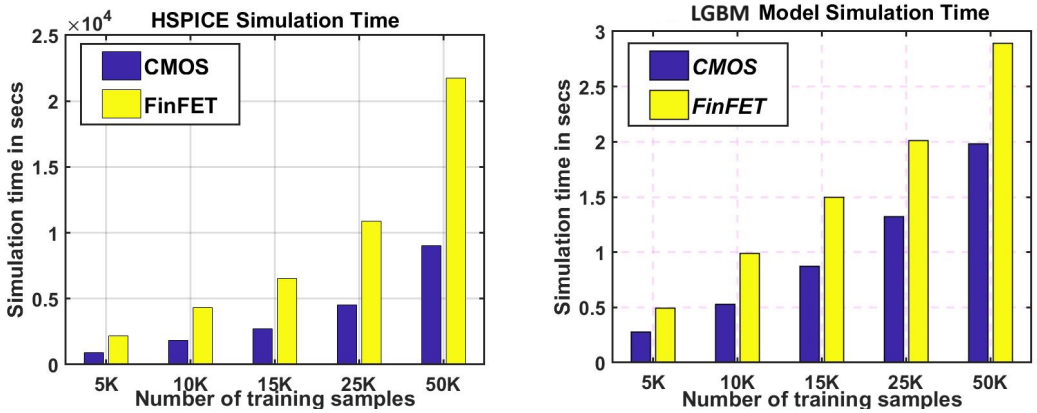
Algorithm 1 Statistical Digital Standard Cell Performance Modeling

while $i < N$ **do** ▷ N is the number of samples
  while $j < K$ **do** ▷ K is the number of PVT parameters
    $X_p \leftarrow$ Gaussian($\pm 10\%$ variations at 3σ of j_{th} Process parameter) ▷ Gaussian process
  end while
  $X_r \leftarrow$ Random(Temperature, $\pm 10\%$ variations of Supply voltage) ▷ Random distributions
  $X_l \leftarrow$ Random(C_{Load}) ▷ C_{Load} is the capacitive load for delay estimation
end while
Combine the PVT variations $X = \sum_{i=1}^{N}\{X_r, X_t, X_{p,1}, X_{p,2}, ...X_{p,k}\}$ with standard cell netlist.
Call SPICE tool()
for $i < N$ **do**
  $Y \leftarrow$ target (X) ▷ Say, target is leakage or delay estimation
end for
Training data $(X, Y) = \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\}$
Extract feature set $X = \{X_r, X_t, X_p\}$
Define target variable $Y\{target\}$
for $j < K$ **do**
  Calculate Pearson coefficient, r for all process
  Perform Sensitivity analysis
end for
Determine effective feature set $X_e = \{X_{re}, X_{te}, X_{pe}\}$ ▷ Dominant parameter extraction
Call bank of ML algorithms to train over the effective dataset (X_e, Y)
Calculate the defined ML metrics *RMSE, MAE,* R^2Score
if $R^2Score < 0.99$ & $\mu_E > 1\%$ & $\sigma_E > 1\%$ **then**
  Hyper-parameter fine-tuning
  Retrain the respective ML model
  Until $R^2Score > 0.99$ & $\mu_E < 1\%$ & $\sigma_E < 1\%$
end if
return trained *ML models*
Call each ML model over (unseen test cases$X_U = \sum_{i=1}^{M}\{X_r, X_t, X_p\}$)
return $Y_{U,Predicted}$, estimated target (leakage/delay) $\forall X_U$
Estimate $Y_{U,Actual} \forall X_U$ calling SPICE tool
Determine $\%Error(Y_{U,Predicted}, Y_{U,Actual})$
if $\%Error > 2\%$ **then**
  Perform grid-search on the respective ML model parameters
  repeat
    Repeat steps 21 to 32
  until $\%Error < 2\%$
end if
save the finalized *ML models*



(a) (b)

Figure 4.3: Impact of the number of training samples on delay model's (LGBM) accuracy (a) 16nm CMOS (b) 16nm FinFET



(a) (b)

Figure 4.4: Runtime comparison of LGBM delay model Vs SPICE simulations

achieves an outstanding speed-up of 10^4 times compared to SPICE MC simulations. It is worth noting that our standard cell approach involves a one-time effort in model computation, whereas traditional SPICE modeling requires repeated simulations across various PVT conditions for each circuit under modeling.

Our goal is to develop surrogate models that match the performance of traditional simulation tools across all levels. To achieve this, we conducted thorough validations by comparing their predictions with those of a standard simulation tool, namely SPICE, at each level. The development and training of ML models were carried out using Python 3.9, leveraging libraries such as Scikit-Learn [77], Keras, PyTorch, TensorFlow [80], NumPy, and Pandas [81, 82]. These tasks were performed on a system equipped with an i5 core CPU and 8GB of RAM. SPICE simulations were executed on a 12th Gen Intel(R) Core(TM) i9 processor with 32GB RAM.

4.3.1.1 Surrogate Modeling Metrics

When assessing the regression models, we utilized different metrics, including the R^2Score or R^2, Mean Squared Error (MSE), Mean Absolute Error (MAE), $\%\mu_E$ (mean error percentage) and $\%\sigma_E$ (standard deviation error percentage) [12]. These metrics were computed based on a 20% split of test samples from the simulation dataset, which were isolated during the training process.

The R^2 statistic gauges how much of the dependent variable's variance can be explained by the independent variables, acting as an indicator of model fit (equ. 4.1). This metric ranges from 0 to 1, with higher values indicating a better fit, and a value of 1 indicating a perfect match. It provides valuable insights into how well the model captures the variation in the data, facilitating the evaluation and comparison of different regression models. MSE measures the average squared difference between predicted and actual values, with an emphasis on larger errors (equ. 4.2). A lower MSE signifies a better fit of the model to the data. MAE offers a straightforward and interpretable measure of model accuracy in the original units of the data (equ. 4.3), and it is less sensitive to outliers compared to MSE. Both MSE and MAE are crucial for evaluating and refining regression models to ensure accurate predictions. We examined them for exceptionally low scores, below E^{-9} in magnitude.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{4.1}$$

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{4.2}$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{4.3}$$

In addition to the standard regression metrics mentioned earlier, we employed $\%\mu_E$ and $\%\sigma_E$ as evaluation criteria to measure the trained model's ability to predict statistical output distributions across various PVT conditions. These criteria are defined as follows:

$$\%\mu_E = \frac{\mu_{Predicted} - \mu_{Actual}}{\mu_{Actual}} * 100 \tag{4.4}$$

$$\%\sigma_E = \frac{\sigma_{Predicted} - \sigma_{Actual}}{\sigma_{Actual}} * 100 \tag{4.5}$$

Here, 'Predicted' refers to the predictions of the regression model for leakage/delay, while 'Actual' pertains to the corresponding SPICE values.

We deliberately included another evaluation criterion, Mean average Percentage Error (MaPE), in each experiment, which constitutes 500 unseen samples (referred to as N) prediction and comparison with SPICE simulations at each node (equ. 4.6). This thorough analysis assesses the models' generalized capacity to apply acquired knowledge effectively. The close correspondence between predictions

Table 4.2: The performance metrics of CMOS high-performance ML Delay models from 45nm and 32nm technology nodes (averaged across the training of all standard cells)

ML metrics/ model	45nm					32nm				
	$\mathbf{R^2}$	**MSE**	**MAE**	$\mu_{\mathbf{E}}$	$\sigma_{\mathbf{E}}$	$\mathbf{R^2}$	**MSE**	**MAE**	$\mu_{\mathbf{E}}$	$\sigma_{\mathbf{E}}$
LR	0.927	3.40E-19	3.01E-09	0.175	4.059	0.912	4.90E-17	1.37E-09	0.610	5.923
RR	0.927	5.45E-18	1.36E-09	0.175	4.059	0.912	4.97E-17	8.39E-10	0.610	5.924
LAR	0.927	1.10E-18	1.02E-08	0.175	4.060	0.912	1.64E-16	1.67E-09	0.610	5.925
ER	0.927	6.79E-17	1.36E-08	0.175	4.202	0.912	2.78E-16	1.02E-09	0.610	6.062
PR2	0.997	1.32E-17	3.40E-09	0.020	0.237	0.995	2.04E-16	2.46E-09	0.080	0.892
PR3	1	1.48E-17	1.67E-09	0.003	0.013	1	3.55E-16	1.51E-09	0.012	0.057
SVR	0.997	1.70E-16	1.43E-06	0.159	1.570	0.996	1.08E-15	2.41E-09	0.036	0.501
DT	0.744	6.19E-10	1.33E-05	0.200	13.508	0.725	1.28E-09	1.70E-05	0.641	16.394
RF	0.986	7.27E-17	3.22E-08	0.123	2.923	0.987	5.02E-16	1.68E-09	0.051	3.629
ET	0.963	1.24E-16	7.26E-09	0.008	4.785	0.961	1.55E-16	1.08E-09	0.222	5.820
GBM	0.995	6.90E-17	3.29E-09	0.051	1.056	0.994	5.28E-16	2.54E-09	0.064	1.494
XGB	0.997	5.87E-17	9.44E-09	0.015	0.875	0.996	1.61E-16	1.53E-09	0.003	1.022
LGBM	0.996	9.40E-17	7.61E-10	0.056	1.333	0.996	1.34E-16	4.54E-09	0.104	1.421
MLP	1	2.82E-17	8.84E-09	0.061	0.044	1	9.53E-16	4.25E-09	0.062	0.113
DNN	0.999	2.27E-17	3.45E-09	0.115	0.243	1	2.09E-16	9.23E-09	0.087	0.056

and actual simulations highlights the models' robust capability.

$$MaPE = \overline{\sum_{i=1}^{N} \left| \frac{Predicted - Actual}{Actual} \right|} * 100 \tag{4.6}$$

4.3.1.2 Comprehensive Evaluation Criteria

To thoroughly evaluate the performance of the trained ML models, we formulated a comprehensive set of evaluation criteria. This set includes R^2, $\%\mu_E$, $\%\sigma_E$ metrics for the test split data, as well as MaPE for unseen PVT samples. Before undergoing additional testing with MaPE on 500 unseen test samples to confirm its generalization capability, the model must meet predefined criteria in terms of R^2, $\%\mu_E$, $\%\sigma_E$ on the test split data. The comprehensive evaluation criteria are as follows: $R^2 \leq 0.99$, $\%\mu_E, \%\sigma_E \leq 1\%$, and $MaPE \leq 2\%$ for all standard cells across the CMOS/FinFET technology nodes in each experiment. The best-performing model for a given standard cell under test is determined based on its adherence to our composite test criteria. This selected model is then designated as the final choice using an intelligent routine within our framework. These finalized models, established at each technology node, are applicable in tasks such as estimating complex circuit performance and within optimization engines. Further details are provided in Chapter 7.

Table 4.3: The performance metrics of CMOS high-performance ML Delay models from 22nm and 16nm technology nodes (averaged across the training of all standard cells)

ML metrics/ model	22nm					16nm				
	$\mathbf{R^2}$	**MSE**	**MAE**	$\mu_{\mathbf{E}}$	$\sigma_{\mathbf{E}}$	$\mathbf{R^2}$	**MSE**	**MAE**	$\mu_{\mathbf{E}}$	$\sigma_{\mathbf{E}}$
LR	0.907	1.01E-17	1.39E-09	0.080	5.609	0.89	6.44E-17	1.17E-08	0.248	3.573
RR	0.907	5.77E-18	1.49E-09	0.080	5.610	0.89	5.45E-17	3.48E-09	0.248	3.574
LAR	0.907	8.52E-17	1.96E-09	0.080	5.611	0.89	1.78E-16	8.15E-09	0.248	3.575
ER	0.907	2.26E-16	2.20E-09	0.078	5.747	0.89	2.95E-16	5.81E-09	0.246	3.715
PR2	0.994	8.43E-17	2.09E-09	0.014	0.351	0.992	2.43E-16	1.99E-08	0.241	0.228
PR3	1	1.90E-16	2.24E-09	0.006	0.029	0.999	3.99E-16	1.15E-08	0.004	0.018
SVR	0.995	3.08E-16	5.33E-09	0.167	1.060	0.994	1.22E-15	2.76E-08	0.071	0.811
DT	0.713	2.04E-09	2.07E-05	0.237	14.441	0.706	1.34E-09	1.10E-05	1.056	16.437
RF	0.987	3.80E-16	2.10E-09	0.015	3.326	0.987	5.08E-16	8.99E-09	0.080	3.343
ET	0.958	1.49E-16	1.78E-09	0.396	5.377	0.959	1.61E-16	5.40E-09	0.457	5.755
GBM	0.994	2.68E-16	2.25E-09	0.003	1.342	0.993	5.73E-16	2.03E-08	0.073	1.149
XGB	0.996	1.13E-16	1.94E-09	0.019	0.947	0.996	1.88E-16	1.13E-08	0.013	0.864
LGBM	0.996	3.58E-16	5.42E-09	0.153	1.372	0.996	1.50E-15	2.73E-08	0.103	1.145
MLP	1	3.00E-16	6.10E-09	0.051	0.049	0.999	1.20E-15	6.66E-08	0.032	0.100
DNN	1	8.02E-17	2.95E-09	0.039	0.036	1	2.40E-16	1.51E-08	0.175	0.020

We have developed a total of 15 ML models for 22 digital standard cells. These cells encompass the commonly used digital logic gates in various circuit implementations. Our framework is designed to accommodate any number of such gates along similar lines easily.

The digital standard cells are categorized as follows:

- **One input gates**→ buffer (BUF), inverter (NOT)
- **Two input gates**→ nand (NAND2), nor (NOR2), and (AND2), or (OR2), ex-or (XOR2), ex-nor (XNOR2)
- **Three input gates**→ nand (NAND3), nor (NOR3), and (AND3), or (OR3), 2x1 multiplexer (MUX21), full adder (FA), and-or-invert (AOI12)
- **Four input gates**→ nand (NAND4), nor (NOR4), and (AND4), or (OR4), and-or-invert (AOI22, AOI31, AOI112)

Tables 4.2, 4.3 present the training metrics for CMOS delay spanning from the 45nm to 16nm technology nodes. The training outcomes for FinFET leakage and delay across the 16nm to 7nm technology nodes are presented in tables 4.4, 4.5. These results were derived by averaging the metrics across all the standard cells. From these training outcomes, we identified four models (ET, LGBM, MLP(ANN), and

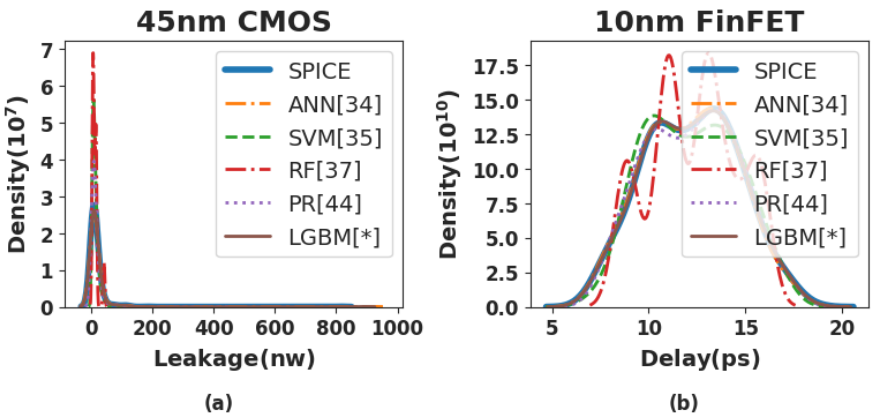


Figure 4.5: Comparison of prediction results of our (*) best performing (LGBM) model with State-of-the-art works on Full adder standard cell over 500 random PVT conditions (a) Leakage (45nm CMOS) (b) Delay (10nm FinFET)

DNN) that demonstrated equally strong performance in estimating both leakage and delay. Tables 4.6, 4.7, 4.8, 4.9, 4.10, 4.11 present the metrics of our test criteria along with the MaPE for 500 unseen test samples (MPE_U) across all standard cells, focusing on FinFET leakage and delay estimations from the 16nm to 7nm technology nodes. The best-performing model, the final model, is denoted in bold for each standard cell. In FinFET delay training and testing, LGBM consistently exhibits the highest performance. On the other hand, when it comes to leakage estimation, DNN takes the lead. Nevertheless, in a few instances with specific standard cells, MLP and LGBM occasionally surpass DNN. ET seldom outperforms LGBM, MLP, and DNN, particularly when predicting unseen test cases.

4.3.2 Comparison with the State-of-the-art Works

In Fig. 4.5, we compare the state-of-the-art surrogate models with those developed within our framework. The surrogate ML models proposed in previous works [34, 35, 37, 44] were trained using the Full adder dataset, with each work employing specific numbers of training samples and hyperparameters. While [34, 35, 37] focused on leakage estimation, [44] concentrated on power and delay estimation. However, we trained our model to address both leakage and delay for a comprehensive comparison.

From the observations in Fig. 4.5, it becomes apparent that the LGBM model, representing the best performance from our framework, accurately predicts both delay and leakage, aligning closely with SPICE MC simulations. Conversely, the SVM [35] and RF [37] models exhibit significant deviations from actual values, while PR [44] shows errors mainly at the distribution peaks. The ANN model [34] agrees well with the LGBM model. It's crucial to note that [44] proposed modeling delay and power without considering process variations, a significant factor affecting circuit performance in an IC.

Regarding runtime, our LGBM-based model offers a 10,000-fold acceleration compared to SPICE, whereas [34] and [35] report speed-ups of 100x and 17x, respectively, over SPICE. A detailed comparison of literature works with our modeling framework is provided in Table 6.2.

Our framework provides precise PVT-aware surrogate models with substantial computational acceleration across various technology nodes. It adeptly models both power and delay with equal proficiency across CMOS and FinFET technologies. Additionally, it excels in sensitivity analysis, efficiently identifying dominant parameters, as discussed in Chapter 5, ensuring high accuracy while minimizing training time.

Table 4.4: The performance metrics of FinFET high-performance ML Leakage models from 16nm, 10nm, and 7nm technology nodes (averaged across the training of all standard cells)

ML metrics/ model	16nm					10nm					7nm				
	$\mathbf{R^2}$	**MSE**	**MAE**	$\mu_{\mathbf{E}}$	$\sigma_{\mathbf{E}}$	$\mathbf{R^2}$	**MSE**	**MAE**	$\mu_{\mathbf{E}}$	$\sigma_{\mathbf{E}}$	$\mathbf{R^2}$	**MSE**	**MAE**	$\mu_{\mathbf{E}}$	$\sigma_{\mathbf{E}}$
LR	0.459	4.59E-11	5.26E-07	26.744	38.769	0.452	2.94E-12	2.04E-07	23.910	40.731	0.415	5.64E-11	7.66E-07	27.493	46.544
RR	0.459	4.59E-11	5.26E-07	26.744	38.769	0.452	2.94E-12	2.04E-07	23.910	40.731	0.415	5.64E-11	7.66E-07	27.493	46.544
LAR	0.459	4.59E-11	5.26E-07	26.758	38.804	0.452	2.94E-12	2.04E-07	23.924	40.763	0.415	5.64E-11	7.66E-07	27.504	46.572
ER	0.459	4.59E-11	5.26E-07	26.842	39.001	0.452	2.94E-12	2.04E-07	24.000	40.945	0.415	5.64E-11	7.66E-07	27.563	46.727
PR2	0.578	2.54E-11	3.66E-07	9.751	4.220	0.833	1.85E-12	1.51E-07	10.958	18.037	0.726	4.94E-11	6.85E-07	17.825	27.102
PR3	0.892	1.64E-12	7.63E-08	1.416	1.188	0.986	8.98E-14	3.04E-08	0.589	1.283	0.951	8.89E-12	2.47E-07	5.115	8.892
DT	0.291	4.49E-11	5.28E-07	28.075	55.318	0.338	2.96E-12	2.04E-07	21.898	51.423	0.269	5.37E-11	7.02E-07	27.915	57.438
ET	0.970	2.50E-15	4.94E-09	2.106	7.179	0.971	3.25E-15	5.87E-09	1.544	6.480	0.966	5.10E-15	7.42E-09	2.009	8.146
GBM	0.813	1.25E-11	2.29E-07	7.054	6.771	0.909	1.18E-12	1.11E-07	7.438	13.253	0.864	2.16E-11	4.29E-07	11.138	19.064
XGB	0.935	2.18E-12	8.64E-08	1.845	0.300	0.907	2.51E-12	1.30E-07	0.103	2.824	0.965	1.79E-12	1.15E-07	1.249	3.174
LGBM	0.994	7.09E-14	1.86E-08	0.070	0.010	0.992	3.27E-14	1.63E-08	0.069	0.510	0.994	9.82E-14	2.72E-08	0.094	0.444
MLP	0.998	8.19E-13	6.24E-08	0.753	0.865	0.998	2.90E-15	5.98E-09	0.018	0.688	0.999	9.43E-14	2.17E-08	0.014	0.500
DNN	0.998	7.10E-13	6.06E-08	1.111	1.028	0.999	5.85E-15	8.07E-09	0.146	0.580	0.999	1.70E-14	1.37E-08	0.235	0.233

Table 4.5: The performance metrics of FinFET high-performance ML Delay models from 16nm, 10nm, and 7nm technology nodes (averaged across the training of all standard cells)

ML metrics/ model	16nm					10nm					7nm				
	$\mathbf{R^2}$	**MSE**	**MAE**	$\mu_{\mathbf{E}}$	$\sigma_{\mathbf{E}}$	$\mathbf{R^2}$	**MSE**	**MAE**	$\mu_{\mathbf{E}}$	$\sigma_{\mathbf{E}}$	$\mathbf{R^2}$	**MSE**	**MAE**	$\mu_{\mathbf{E}}$	$\sigma_{\mathbf{E}}$
LR	0.977	1.34E-04	1.34E-04	0.016	0.731	0.989	3.10E-25	3.76E-13	0.118	0.401	0.995	1.48E-25	2.71E-13	0.006	0.482
RR	0.991	3.22E-25	3.80E-13	0.016	0.749	0.989	3.10E-25	3.76E-13	0.118	0.402	0.995	1.48E-25	2.71E-13	0.006	0.483
LAR	0.991	3.22E-25	3.80E-13	0.016	0.749	0.989	3.10E-25	3.76E-13	0.118	0.403	0.995	1.48E-25	2.71E-13	0.006	0.483
ER	0.991	3.23E-25	3.80E-13	0.016	0.863	0.989	3.10E-25	3.75E-13	0.119	0.525	0.995	1.48E-25	2.71E-13	0.003	0.595
PR2	1.000	1.40E-26	8.04E-14	0.007	0.090	0.999	1.56E-26	7.92E-14	0.022	0.098	1.000	6.06E-27	5.08E-14	0.006	0.029
PR3	1.000	3.68E-27	4.05E-14	0.006	0.036	0.999	6.58E-27	4.12E-14	0.005	0.015	1.000	2.90E-27	3.48E-14	0.005	0.003
SVR	0.998	7.80E-26	2.21E-13	0.082	2.286	0.997	6.46E-26	2.01E-13	0.152	2.496	0.998	5.91E-26	1.93E-13	0.000	1.756
DT	0.897	3.67E-24	1.48E-12	0.275	4.767	0.881	3.24E-24	1.38E-12	0.351	6.446	0.917	2.45E-24	1.21E-12	0.035	3.953
RF	0.998	8.08E-26	1.91E-13	0.027	0.590	0.997	7.07E-26	1.76E-13	0.005	0.637	0.999	3.43E-26	1.27E-13	0.038	0.215
ET	0.993	2.55E-25	3.61E-13	0.033	0.962	0.992	2.22E-25	3.37E-13	0.045	1.340	0.997	9.08E-26	2.14E-13	0.086	0.431
GBM	0.999	2.61E-26	1.20E-13	0.000	0.168	0.999	2.60E-26	1.16E-13	0.015	0.186	1.000	1.10E-26	7.99E-14	0.002	0.055
XGB	0.999	1.84E-26	9.87E-14	0.002	0.198	0.999	1.88E-26	9.42E-14	0.009	0.208	1.000	5.53E-27	5.52E-14	0.002	0.040
LGBM	0.999	2.42E-26	1.07E-13	0.021	0.225	0.999	2.24E-26	9.81E-14	0.007	0.249	1.000	7.55E-27	6.12E-14	0.005	0.054
MLP	0.999	2.04E-26	9.77E-14	0.042	0.115	0.999	1.45E-26	8.11E-14	0.052	0.137	1.000	8.42E-27	6.79E-14	0.018	0.083
DNN	1.000	6.24E-27	5.55E-14	0.033	0.028	0.999	8.11E-27	5.44E-14	0.064	0.018	1.000	3.65E-27	4.39E-14	0.064	0.053

Table 4.6: Performance of the best-performing ML models of 16nm FinFET Leakage for all standard cells

ML models/ Standard cells	ET				LGBM				MLP				DNN			
	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$
BUF	0.989	1.538	5.847	2.746	0.996	0.209	0.434	1.191	0.999	0.555	0.511	0.874	**1.000**	**0.128**	**0.566**	**0.449**
NOT	0.994	1.336	3.690	2.714	0.997	0.198	0.534	1.512	**1.000**	**0.227**	**0.309**	**1.043**	0.999	1.133	1.994	1.161
NAND2	0.991	4.441	9.907	3.032	0.984	0.100	1.015	4.341	0.999	1.341	1.352	1.353	**1.000**	**0.418**	**0.216**	**1.086**
NOR2	0.995	1.557	3.402	2.600	0.996	0.194	0.101	2.111	0.999	1.458	2.023	1.757	**1.000**	**0.010**	**0.192**	**1.006**
AND2	0.984	5.421	20.847	5.502	0.994	0.047	0.523	2.547	**0.998**	**0.635**	**1.854**	**0.751**	0.997	1.393	3.076	1.047
OR2	0.991	1.324	5.702	2.275	0.998	0.049	0.005	1.234	1.000	0.564	0.925	0.631	**1.000**	**0.233**	**0.681**	**0.437**
XOR2	0.991	1.458	6.133	2.481	0.996	0.124	0.611	1.525	1.000	0.523	0.327	1.138	**0.999**	**1.263**	**2.138**	**1.095**
XNOR2	0.991	1.431	5.843	3.094	0.997	0.088	0.238	1.362	1.000	0.645	0.180	1.294	**1.000**	**0.564**	**0.355**	**1.063**
NAND3	0.993	3.255	7.990	2.316	0.993	0.014	0.385	3.343	0.999	1.455	1.334	1.853	**0.999**	**1.365**	**1.288**	**1.112**
NOR3	0.996	1.532	3.314	2.324	0.988	0.263	0.823	3.006	**0.999**	**0.314**	**0.547**	**1.687**	0.998	2.663	3.467	1.108
AND3	0.981	4.461	20.466	5.111	0.994	0.030	0.216	2.726	0.999	0.009	0.226	0.662	**1.000**	**0.237**	**0.601**	**0.571**
OR3	0.994	1.335	6.318	1.600	0.996	0.031	0.221	1.862	0.998	0.455	1.500	0.650	**0.999**	**0.481**	**1.131**	**0.463**
MUX21	**0.999**	**0.787**	**1.078**	**1.740**	0.976	0.110	0.398	6.710	0.999	1.675	1.686	1.792	0.999	2.570	3.018	1.256
FA	0.995	1.421	8.175	2.236	0.998	0.008	0.055	1.416	**0.999**	**0.561**	**0.363**	**0.762**	0.999	0.395	1.620	0.512
AOI12	0.993	1.628	5.933	1.852	0.999	0.024	0.142	1.335	**0.999**	**0.385**	**0.131**	**0.734**	0.999	0.871	1.204	0.701
NAND4	0.994	2.855	7.361	2.174	0.991	0.509	0.681	3.119	1.000	0.479	0.899	0.956	**1.000**	**0.488**	**0.712**	**0.735**
NOR4	**1.000**	**0.024**	**0.035**	**2.022**	0.998	0.023	0.087	2.561	0.981	12.600	12.657	3.898	0.983	12.217	12.159	3.359
AND4	0.998	0.987	1.078	2.397	0.996	0.970	0.979	6.070	**0.997**	**1.622**	**1.686**	**1.792**	0.998	2.981	3.018	1.634
OR4	0.995	1.313	8.175	1.343	0.993	0.723	0.566	1.646	**0.991**	**0.873**	**0.363**	**0.762**	0.998	0.384	1.620	0.569
AOI22	0.992	1.513	5.933	1.725	0.998	0.232	0.722	1.336	**0.991**	**0.335**	**0.131**	**0.734**	0.993	0.756	1.204	0.374
AOI31	0.991	1.947	7.361	2.721	0.998	0.908	0.547	3.875	0.995	0.918	0.899	0.956	**1.000**	**0.711**	**0.712**	**0.460**
AOI122	0.998	0.294	0.035	2.425	**0.994**	**0.323**	**0.818**	**2.058**	0.986	9.180	12.657	3.898	0.993	10.117	8.163	3.407

Table 4.7: Performance of the best-performing ML models of 10nm FinFET Leakage for all standard cells

ML metrics/ model	ET				LGBM				MLP				DNN			
	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$
BUF	0.945	2.598	12.409	2.745	0.995	0.378	1.510	1.353	**1.000**	**0.270**	**0.191**	**0.455**	1.000	0.204	0.226	0.594
NOT	0.905	2.267	12.412	1.816	0.954	0.507	5.495	1.210	**0.990**	**0.218**	**1.040**	**0.950**	0.993	0.601	2.726	1.125
NAND2	0.988	0.964	3.631	0.850	0.994	0.047	0.010	3.753	**0.998**	**0.699**	**0.213**	**1.230**	0.998	2.326	3.835	1.987
NOR2	0.998	0.513	1.605	0.563	**0.998**	**0.055**	**0.379**	**1.075**	1.000	1.026	1.481	0.743	1.000	1.444	1.450	1.316
AND2	0.980	1.953	8.238	1.933	0.995	0.066	0.464	2.697	0.998	0.894	2.223	0.912	**1.000**	**0.379**	**1.024**	**0.447**
OR2	0.995	0.556	2.714	0.748	**0.999**	**0.038**	**0.069**	**1.001**	1.000	0.610	0.295	0.753	1.000	0.088	0.160	0.493
XOR2	0.992	0.680	2.955	0.816	0.998	0.064	0.384	1.240	0.999	0.713	1.605	0.917	**1.000**	**0.626**	**1.136**	**0.807**
XNOR2	0.992	0.750	3.243	1.130	0.999	0.073	0.385	1.265	0.999	0.354	1.003	0.625	**1.000**	**0.180**	**0.123**	**0.451**
NAND3	0.954	2.875	8.357	2.306	0.991	0.051	0.801	3.190	0.997	1.865	1.449	2.699	**0.997**	**0.291**	**1.726**	**0.709**
NOR3	0.990	1.365	3.721	1.118	0.983	0.136	1.592	1.919	0.999	2.006	2.475	0.800	**1.000**	**0.739**	**1.039**	**0.511**
AND3	0.893	4.120	20.425	3.603	0.991	0.059	0.572	2.781	0.999	0.076	1.008	0.588	**1.000**	**0.070**	**0.045**	**0.359**
OR3	0.985	1.070	4.823	1.562	0.997	0.117	0.399	1.693	**1.000**	**0.137**	**0.182**	**0.425**	1.000	0.472	0.377	0.562
MUX21	**0.999**	**0.326**	**0.415**	**2.253**	0.987	0.079	0.696	4.736	0.999	2.875	2.588	2.283	0.998	3.966	4.163	1.682
FA	0.971	1.338	6.784	2.394	0.998	0.022	0.293	1.332	0.999	0.233	1.223	0.515	**1.000**	**0.047**	**0.513**	**0.404**
AOI12	0.981	1.787	5.472	2.120	0.998	0.024	0.047	1.494	**1.000**	**0.051**	**0.152**	**0.729**	0.997	1.728	4.454	2.233
NAND4	0.999	0.750	3.253	2.019	0.995	1.946	2.591	1.200	0.972	1.650	1.593	1.969	**0.972**	**0.917**	**1.466**	**1.282**
NOR4	0.942	3.048	7.710	1.336	0.991	1.894	1.608	2.822	0.996	1.516	1.548	1.000	**0.996**	**0.932**	**1.392**	**0.908**
AND4	0.991	2.649	2.208	0.815	0.992	1.725	0.986	1.930	0.999	0.793	1.088	0.887	**0.961**	**0.690**	**0.508**	**0.518**
OR4	0.928	4.673	10.786	2.603	0.987	0.955	0.964	3.629	**0.992**	**0.163**	**0.581**	**1.251**	0.988	0.757	1.387	1.623
AOI22	0.949	1.003	2.229	1.686	**0.989**	**0.315**	**0.932**	**1.186**	0.991	1.839	2.895	1.827	0.980	2.676	2.635	1.899
AOI31	0.989	0.526	0.725	3.525	**0.990**	**0.990**	**0.675**	**1.524**	0.990	0.966	1.305	1.147	0.998	0.748	1.529	0.944
AOI122	0.962	2.751	5.781	3.385	0.994	0.959	0.960	2.976	**0.996**	**0.534**	**0.871**	**0.905**	0.974	2.275	3.380	2.818

Table 4.8: Performance of the best-performing ML models of 7nm FinFET Leakage for all standard cells

ML models/ Standard cells	ET				LGBM				MLP				DNN			
	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$
BUF	0.960	2.301	9.701	2.690	0.992	0.502	2.379	1.171	**1.000**	**0.088**	**0.804**	**0.621**	1.000	0.242	1.434	0.862
NOT	0.981	1.349	5.702	1.457	**0.996**	**0.027**	**0.620**	**1.029**	0.998	2.291	3.804	1.564	1.000	1.120	1.624	0.810
NAND2	0.972	3.863	9.098	2.553	0.980	0.019	0.613	4.283	**0.998**	**0.115**	**0.884**	**0.935**	0.998	1.235	1.869	1.176
NOR2	0.983	1.731	5.838	1.097	**0.998**	**0.194**	**0.352**	**0.968**	0.998	1.916	2.895	1.313	0.999	1.396	2.060	0.692
AND2	0.907	4.445	20.151	4.484	0.990	0.112	0.575	2.801	0.999	0.954	1.460	0.972	**1.000**	**0.064**	**0.212**	**0.451**
OR2	0.967	1.642	8.347	2.202	0.998	0.035	0.239	1.135	**1.000**	**0.011**	**0.484**	**0.764**	0.999	1.109	2.386	0.853
XOR2	0.970	1.706	7.229	1.793	0.996	0.126	1.021	1.246	0.999	0.028	0.563	0.604	**0.999**	**0.245**	**0.777**	**0.579**
XNOR2	0.964	1.779	8.114	2.753	0.997	0.036	0.447	1.335	**0.999**	**0.239**	**0.040**	**0.660**	1.000	0.502	0.282	0.748
NAND3	0.975	3.512	8.499	2.727	0.994	0.111	0.001	2.881	**0.999**	**0.822**	**1.618**	**0.919**	0.993	2.561	7.281	3.660
NOR3	0.984	1.628	5.492	0.800	0.985	0.009	1.231	1.243	**0.996**	**1.352**	**0.352**	**0.794**	0.999	1.690	2.630	0.665
AND3	0.921	3.241	16.824	3.274	0.993	0.041	0.042	2.701	**0.999**	**0.906**	**0.872**	**1.092**	0.999	1.350	1.719	1.336
OR3	0.978	1.385	6.713	1.683	0.996	0.002	0.288	1.539	**0.999**	**0.186**	**0.970**	**0.589**	1.000	0.534	1.236	0.490
MUX21	**0.999**	**0.194**	**0.315**	**2.092**	0.991	0.072	0.220	3.930	0.999	1.976	1.830	2.107	0.998	3.287	3.530	1.821
FA	0.969	1.429	7.453	2.371	0.998	0.002	0.077	1.281	**0.999**	**0.338**	**0.584**	**0.691**	1.000	0.459	1.210	0.458
AOI12	0.962	1.861	7.414	2.215	**0.999**	**0.063**	**0.418**	**1.340**	0.996	1.917	2.375	1.388	0.999	1.098	2.120	0.728
NAND4	0.958	3.018	9.590	2.469	0.994	0.445	2.258	2.533	**0.998**	**0.399**	**0.276**	**0.674**	0.999	0.548	1.318	0.529
NOR4	**1.000**	**0.020**	**0.028**	**0.709**	0.999	0.099	0.175	1.429	0.997	2.164	2.031	1.411	1.000	0.986	1.010	0.553
AND4	0.947	2.070	11.918	2.153	0.990	0.242	1.815	3.107	**0.999**	**0.663**	**0.897**	**0.717**	0.999	0.902	0.158	0.962
OR4	1.000	0.021	0.027	1.312	0.999	0.045	0.185	1.518	0.999	0.887	0.820	1.008	**1.000**	**0.944**	**0.941**	**0.600**
AOI22	0.962	1.796	8.161	2.286	0.997	0.011	0.230	2.936	0.999	0.521	0.010	0.747	**0.999**	**0.095**	**0.194**	**0.542**
AOI31	0.908	3.818	16.492	3.784	0.996	0.220	0.933	2.393	0.997	0.699	2.419	0.681	**0.999**	**0.665**	**0.665**	**0.611**
AOI122	0.977	1.389	6.107	1.770	0.998	0.017	0.185	1.734	**0.999**	**0.012**	**0.958**	**0.677**	0.999	0.981	1.556	0.697

Table 4.9: Performance of the best-performing ML models of 16nm FinFET Delay for all the standard cells

ML model/ Standard cells	ET				LGBM				MLP				DNN			
	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$
BUF	0.998	0.007	0.558	0.434	**0.999**	**0.006**	**0.218**	**0.192**	0.999	0.395	0.285	0.7	1	0.245	0.028	0.335
NOT	0.998	0.033	0.456	0.664	**0.999**	**0.014**	**0.195**	**0.259**	1	0.465	0.411	0.828	1	0.241	0.004	0.438
NAND2	0.997	0.022	0.695	0.785	**0.999**	**0.018**	**0.254**	**0.285**	1	0.013	0.681	0.557	1	0.189	0.552	0.591
NOR2	0.998	0.023	0.447	0.615	**0.999**	**0.021**	**0.162**	**0.299**	1	0.197	0.583	0.73	1	0.363	1.592	1.157
AND2	0.998	0.029	0.441	0.513	**0.999**	**0.015**	**0.188**	**0.196**	1	0.093	0.573	0.527	1	0.109	0.561	0.334
OR2	0.998	0.023	0.619	0.811	**0.999**	**0.013**	**0.252**	**0.3**	1	0.343	0.331	0.778	1	0.004	0.262	0.506
XOR2	0.998	0.025	0.308	0.625	**0.999**	**0.002**	**0.076**	**0.23**	0.999	0.69	0.519	1.032	1	0.139	0.462	0.416
XNOR2	0.998	0.035	0.429	0.57	**0.999**	**0.016**	**0.214**	**0.211**	1	0.178	0.084	0.5	1	0.255	0.648	0.598
NAND3	0.997	0.017	0.841	0.772	**0.999**	**0.013**	**0.311**	**0.246**	1	0.095	0.615	0.464	1	0.349	0.072	0.438
NOR3	0.998	0.027	0.456	0.54	**0.999**	**0.023**	**0.189**	**0.175**	1	0.524	0.152	0.685	1	0.097	0.025	0.205
AND3	0.996	0.147	1.327	0.638	**0.998**	**0.14**	**0.46**	**0.246**	0.993	0.066	0.916	0.591	0.999	0.122	0.623	0.536
OR3	0.998	0.02	0.54	0.41	**0.999**	**0.003**	**0.229**	**0.176**	1	0.033	0.052	0.363	1	0.182	0.247	0.334
MUX21	0.998	0.002	0.523	0.426	**0.999**	**0.003**	**0.219**	**0.169**	0.999	0.275	1.364	0.495	1	0.179	0.307	0.28
FA	0.998	0.014	0.53	0.45	**1**	**0**	**0.227**	**0.183**	1	0.127	0.098	0.415	1	0.331	0.492	0.511
AOI12	0.998	0.014	0.597	0.571	**0.999**	**0.009**	**0.254**	**0.229**	1	0.288	0.09	0.604	1	0.159	0.188	0.285
NAND4	0.996	0.032	0.846	0.774	**1**	**0.002**	**0.254**	**0.195**	1	0.212	0.115	0.575	1	0.219	0.792	0.521
NOR4	0.997	0.002	0.699	0.707	**0.999**	**0.019**	**0.368**	**0.232**	1	0.198	0.551	0.609	1	0.209	0.292	0.735
AND4	0.997	0.033	0.548	0.558	**0.999**	**0.002**	**0.249**	**0.182**	1	0.192	0.356	0.554	1	0.006	0.301	0.323
OR4	0.997	0.018	0.594	0.59	**0.999**	**0.001**	**0.195**	**0.209**	1	0.274	0.447	0.549	1	0.009	0.539	0.276
AOI22	0.998	0.025	0.436	0.681	**0.999**	**0.002**	**0.208**	**0.231**	0.999	0.227	0.289	0.538	1	0.437	0.877	0.448
AOI31	0.998	0.002	0.508	0.446	**0.999**	**0.016**	**0.172**	**0.195**	1	0.019	0.508	0.754	1	0.342	0.499	0.221
AOI122	0.998	0.024	0.472	0.701	**0.999**	**0.021**	**0.185**	**0.294**	1	0.337	0.479	0.556	1	0.288	0.817	0.416

Table 4.10: Performance of the best-performing ML models of 10nm FinFET Delay for all standard cells

ML models/ Standard cells	ET				LGBM				MLP				DNN			
	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$
BUF	0.998	0.019	0.592	0.568	**0.999**	**0.011**	**0.206**	**0.233**	1	0.014	0.119	0.479	1	0.261	0.611	0.311
NOT	0.998	0.009	0.606	0.857	**0.999**	**0.036**	**0.277**	**0.302**	1	0.446	0.835	1.186	1	0.17	0.469	0.626
NAND2	0.997	0.034	0.711	0.838	**0.999**	**0.05**	**0.242**	**0.296**	1	0.126	0.128	0.648	1	0.282	0.034	0.464
NOR2	0.998	0.01	0.698	0.78	**0.999**	**0.008**	**0.182**	**0.307**	1	0.123	0.326	0.551	1	0.262	0.478	0.586
AND2	0.998	0.011	0.631	0.664	**0.999**	**0.02**	**0.249**	**0.266**	1	0.239	0.419	0.616	1	0.097	0.461	0.447
OR2	0.997	0.042	0.671	0.866	**0.999**	**0.043**	**0.222**	**0.33**	1	0.296	0.202	0.782	1	0.325	0.078	0.699
XOR2	0.998	0.016	0.609	0.758	**0.999**	**0.005**	**0.166**	**0.266**	1	0.472	0.228	0.78	1	0.069	0.402	0.37
XNOR2	0.998	0.017	0.641	0.697	**0.999**	**0.023**	**0.273**	**0.291**	1	0.33	0.322	0.678	1	0.045	0.337	0.398
NAND3	0.997	0.041	0.804	0.841	**0.999**	**0.05**	**0.295**	**0.298**	1	0.128	0.157	0.514	1	0.472	0.428	0.59
NOR3	0.998	0.011	0.648	0.642	**0.999**	**0.018**	**0.267**	**0.259**	1	0.636	0.188	0.882	1	0.512	0.149	0.602
AND3	0.994	0.083	0.918	0.861	**0.995**	**0.049**	**0.429**	**0.906**	0.994	0.632	0.993	1.258	0.996	0.101	1.618	1.284
OR3	0.998	0.011	0.627	0.53	**0.999**	**0.01**	**0.22**	**0.232**	1	0.475	0.232	0.573	1	0.026	0.697	0.331
MUX21	0.998	0.004	0.594	0.413	**0.999**	**0**	**0.227**	**0.174**	1	0.068	0.038	0.303	1	0.056	0.49	0.231
FA	0.998	0.006	0.562	0.509	**0.999**	**0.007**	**0.21**	**0.209**	1	0.134	0.117	0.504	1	0.398	0.167	0.507
AOI12	0.998	0.02	0.673	0.681	**0.999**	**0.025**	**0.21**	**0.26**	1	0.47	0.167	0.851	1	0.144	0.201	0.404
NAND4	0.998	0.03	0.758	0.667	**1**	**0.04**	**0.262**	**0.254**	1	0.757	0.229	0.583	1	0.052	0.262	0.572
NOR4	0.997	0.04	0.647	0.886	**1**	**0.05**	**0.376**	**0.263**	1	0.831	0.338	0.756	1	0.041	0.454	0.623
AND4	0.998	0.022	0.668	0.611	**1**	**0.011**	**0.268**	**0.242**	0.999	0.376	0.532	0.631	1	0.192	0.275	0.336
OR4	0.999	0.019	0.732	0.587	**1**	**0.021**	**0.325**	**0.247**	0.999	0.231	0.376	0.696	1	0.207	0.823	0.624
AOI22	0.997	0.012	0.632	0.675	**0.999**	**0.002**	**0.261**	**0.239**	0.999	0.362	0.531	0.518	1	0.165	0.577	0.563
AOI31	0.998	0.002	0.576	0.539	**0.999**	**0.013**	**0.247**	**0.178**	0.999	0.121	0.158	0.596	1	0.245	0.398	0.282
AOI122	0.997	0.022	0.659	0.654	**0.999**	**0.023**	**0.249**	**0.227**	0.999	0.072	0.314	0.446	1	0.514	0.515	0.376

Table 4.11: Performance of the best-performing ML models of 7nm FinFET Delay for all the standard cells

ML model/ Standard cells	ET				LGBM				MLP				DNN			
	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$	$\mathbf{R^2}$	$\mu_\mathbf{E}$	$\sigma_\mathbf{E}$	$\mathbf{MPE_U}$
BUF	0.999	0.031	0.189	0.402	**1**	**0.004**	**0.084**	**0.136**	1	0.258	0.261	0.342	1	0.258	0.261	0.342
NOT	0.999	0.048	0.266	0.542	**1**	**0.011**	**0.021**	**0.207**	1	0.159	0.534	0.587	1	0.159	0.534	0.587
NAND2	0.999	0.051	0.163	0.581	**1**	**0.004**	**0.089**	**0.187**	1	0.069	0.252	0.522	1	0.159	0.47	0.458
NOR2	0.999	0.058	0.124	0.572	**1**	**0.001**	**0.078**	**0.193**	1	0.371	0.827	0.862	1	0.131	0.479	0.414
AND2	0.999	0.04	0.225	0.418	**1**	**0.007**	**0.029**	**0.157**	1	0.297	0.534	0.735	1	0.432	1.152	0.451
OR2	0.999	0.054	0.155	0.602	**1**	**0.005**	**0.077**	**0.216**	0.999	0.409	0.38	0.789	1	0.035	0.611	0.621
XOR2	0.999	0.058	0.128	0.54	**1**	**0.01**	**0.064**	**0.182**	1	0.076	0.854	0.714	1	0.033	0.316	0.301
XNOR2	0.999	0.044	0.247	0.446	**1**	**0.013**	**0.017**	**0.17**	1	0.04	0.034	0.653	1	0.245	0.189	0.451
NAND3	0.998	0.041	0.155	0.55	**1**	**0.007**	**0.059**	**0.191**	1	0.028	0.655	0.565	1	0.427	1.257	0.566
NOR3	0.999	0.046	0.276	0.479	**1**	**0.005**	**0.04**	**0.17**	1	0.233	0.052	0.567	1	0.388	0.036	0.524
AND3	0.999	0.032	0.259	0.392	**1**	**0.013**	**0.077**	**0.18**	1	0.065	0.181	0.519	1	0.044	0.625	0.293
OR3	0.999	0.033	0.189	0.392	**1**	**0.003**	**0.075**	**0.131**	1	0.01	0.063	0.365	1	0.259	0.61	0.327
MUX21	0.999	0.023	0.219	0.377	**1**	**0**	**0.087**	**0.13**	1	0.271	0.316	0.398	1	0.043	0.406	0.245
FA	0.999	0.032	0.172	0.411	**1**	**0.003**	**0.066**	**0.142**	1	0.054	0.428	0.56	1	0.217	0.084	0.311
AOI12	0.999	0.034	0.184	0.551	**1**	**0.005**	**0.071**	**0.2**	1	0.022	0.255	0.729	1	0.222	1.328	0.813
NAND4	0.997	0.031	0.146	0.647	**1**	**0.003**	**0.125**	**0.175**	1	0.019	0.857	0.755	1	0.397	0.972	0.621
NOR4	0.997	0.028	0.299	0.787	**1**	**0.002**	**0.036**	**0.321**	1	0.087	0.457	0.792	1	0.413	0.352	0.436
AND4	0.998	0.023	0.343	0.514	**1**	**0.002**	**0.042**	**0.129**	1	0.923	0.623	0.645	1	0.045	0.131	0.311
OR4	0.999	0.048	0.246	0.329	**1**	**0.003**	**0.091**	**0.211**	1	0.178	0.457	0.495	1	0.126	0.592	0.175
AOI22	0.998	0.045	0.643	0.431	**1**	**0.002**	**0.08**	**0.214**	1	0.237	0.192	0.383	1	0.335	0.732	0.358
AOI31	0.999	0.032	0.539	0.496	**1**	**0.003**	**0.721**	**0.159**	1	0.029	0.108	0.574	1	0.432	0.323	0.323
AOI122	0.998	0.03	0.274	0.548	**1**	**0.001**	**0.851**	**0.148**	1	0.074	0.178	0.651	1	0.387	0.754	0.546

4.4 PVT-aware Analog Circuit Surrogate Modeling

Analog circuit design encounters distinctive hurdles stemming from process variations, temperature fluctuations, and environmental factors. Traditional approaches involve manual adjustments, extensive fine-tuning post-fabrication, and large design buffers to accommodate uncertainties. However, these methods lose efficacy as technology progresses towards smaller scales, with design errors increasingly causing functional issues. Furthermore, specification issues like changes, inaccuracies, or incompleteness are common among leading ASIC manufacturers. Surrogate machine-learning models offer a swift and efficient solution, modeling analog circuits under diverse conditions, addressing modern VLSI challenges through rapid prototyping, reliability assessment, functional verification, and fault detection, thereby enhancing energy efficiency and power conservation. Moreover, such modeling significantly reduces the need for calibration and time, paving the way for self-adaptive analog design and minimizing ASIC failures while improving IC yield.

By harnessing the power of AI/ML, self-adaptive analog circuits can autonomously adjust their operating parameters and configurations to optimize performance across varying conditions. This paradigm shift offers numerous benefits, including increased resilience, real-time adaptability, and heightened power efficiency. Details regarding the implementation of self-adaptive analog designs are outlined in Chapter 8.

In our analog circuit modeling strategy, regression algorithms were employed, as described in the discussion on digital modeling and classification techniques [83,84], circuit's operating mode by analyzing variational data across different corners. This analysis considers design, process, and environmental variables, aiming to categorize the circuit's operating corner based on real-time PVT data acquired randomly using onboard sensors. This enables the examination of necessary inputs for control circuitry, allowing it to self-adjust to typical specifications. Regression models are developed to forecast circuit outcomes, considering fluctuations in circuit-specific design characteristics alongside PVT variations.

4.4.1 Training and Building Analog Surrogate Models

Creating a reliable surrogate model relies on achieving PVT-aware statistical behavior across diverse corners of AMS circuits. This model streamlines the rapid generation of solutions within compressed time frames. Our approach involved leveraging a SPICE simulator on AMS circuits developed within the ADE to build training datasets, as detailed in Section 3.5. This entailed conducting DC, AC, and transient simulations to calculate multiple PVT-aware standard performance metrics across various corners over a wide temperature range from $-40°C$ to $125°C$ and $\pm 10\%$ from the operating voltage. Careful selection of each circuit's input parameter (PVT) range ensured proper functionality of all MOSFETs within their intended operational states.

Here, we present the results of modeling different AMS circuits using ML algorithms. The datasets were generated within the ADE environment of Cadence, utilizing designated PDKs for 180nm, 65nm,

and 28nm technologies. Each model was constructed by partitioning the datasets into 90% for training and 10% for testing.

The choice between a classification or regression algorithm depends on the complex interplay between circuit specifications and Process Monitor (PMON)s. These process variations, obtained from the PDK, are captured as frequency monitors known as PMONs, with ring oscillators employed to generate PMON frequencies. PMON frequency ranges typically vary from kHz to MHz, depending on the circuit's characteristics. However, circuits with PMON ranges that significantly overlap across different corners pose challenges in developing precise regression algorithms and adapting them accordingly. In such cases, classification algorithms are used to initially identify the corner corresponding to a given PVT condition. Subsequently, a LUT based approach is employed to generate digital control codes for circuit tuning. This approach facilitates accurate corner identification and enables efficient adjustment of circuit parameters to optimize performance under varying operating conditions.

4.4.1.1 Regression algorithms

We utilize regression algorithms to unravel the intricate relationships between inputs and outputs in analog VLSI circuits, crafting precise predictive models while tackling challenges related to variability and performance enhancement. Techniques like MPR and the application of gradient boosting regressors such as LGBM and XGBM are instrumental in calibrating AMS circuits. Through experimentation, we have determined that third to seventh-order polynomials yield the most accurate circuit models. Similar to digital circuit modeling, we employ a Gradient Descent optimizer with the L2 norm serving as the loss function. Learning rates for LGBM and XGBM fall within the range of 0.01 to 0.15, while the bagging fraction varies from 0.5 to 1. Additionally, parameters like maximum depth (2 to 5), number of leaves (10 to 100), and the utilization of 1000 to 2000 estimators are fine-tuned using the Gradient Boosted Regression Trees (GBRT) optimizer. The algorithms iterate until predefined stopping criteria are met, determined by the goodness of fit, with an R^2 surpassing 0.95 indicating accurate modeling. Furthermore, model validation is conducted by evaluating the MaPE, which is expected to remain below 1%.

4.4.1.2 Classification algorithms

Employing classification algorithms proves effective in modeling the behavior of AMS circuits across diverse corners. We utilize logistic regression, SVM, DT, LGBM, and KNN to classify the PVT-aware behavior of analog circuits at different corners. Additionally, identifying suitable corners amidst process variability assists in selecting appropriate digital codes from the control circuitry, which are then applied to the analog circuit to optimize their performance to typical levels. This systematic approach ensures the consistent performance of analog circuits under varying PVT conditions. The hyper-parameters utilized for the classification of analog circuits are detailed below.

- Multi-class classification is performed using logistic regression with a regularization parameter of 0.1 and the 'Lbfgs' solver until achieving the desired accuracy ($r2 > 0.95$) [85].
- The DT classifier utilizes the 'Gini' criterion for splitting, with a maximum depth ranging from 3 to 5, a minimum sample split of 2 to 5, and a maximum leaf node count of 10 to 25.
- The SVM classifier [86] is set with a regularization parameter (C) of 50 and gamma set to 1, influencing the decision boundary shape. Employing the radial basis function (rbf) kernel enables capturing nonlinear decision boundaries effectively in high-dimensional spaces. The chosen value of C emphasizes strong regularization to mitigate over-fitting risks.
- Utilizing scikit-learn, we deploy a KNN classifier with three neighbors and the ball tree algorithm. The parameter p is set to 2, representing the L2 norm. Additionally, we explore the number of jobs from 2 to 8 to optimize computational efficiency. The 'ball tree' algorithm aids in identifying nearest neighbors in high-dimensional spaces by organizing the training data into a tree structure.
- The LGBM classifier is configured with Gradient Boosting Decision Trees (GBDT) with a learning rate of 0.015, controlling the step size in weight updating during each boosting iteration to balance convergence speed and accuracy. With 1000 estimators, the classifier iteratively improves predictive capability by combining multiple weak learners. Furthermore, a maximum of 200 leaves per tree is allowed, balancing model complexity and generalization ability.

We utilized accuracy, recall, and F1 score metrics to assess the classification accuracy.

- **Accuracy Score:** This metric evaluates the proportion of correctly classified instances among all instances. It is given by,

$$\text{Accuracy Score} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \tag{4.7}$$

- **Precision Score:** Precision measures the accuracy of positive predictions by dividing the number of true positive predictions by the total number of positive class predictions. It indicates the relevance of selected instances.
- **F1 Score:** The F1 score represents the harmonic mean of precision and recall. It offers a balanced measure of precision and recall, particularly valuable when dealing with imbalanced classes. It is given by,

$$F1Score = \frac{2 * (precision * recall)}{precision + recall} \tag{4.8}$$

4.4.2 Results and Discussion

We constructed surrogate models for commonly utilized analog circuit macros, including the current reference generator, voltage reference generator, free-running oscillator, low-dropout regulator, transmission driver, phase-locked loop, band gap reference, low noise amplifier, and mixer. This section

Table 4.12: Training metrics of Current reference generator modeling

Training dataset with 45996 samples						
Regressor	R^2	**RMSE**	**MAE**	μ_E	σ_E	**Training time (Sec)**
LGBM	0.996	0.017	0.008	0.004	0.359	2.584
XGBM	0.993	0.021	0.010	0.006	0.251	0.262
Poly3	0.910	0.075	0.052	-0.007	3.319	1.768

elaborates on the outcomes of training these circuits. We utilized three regression algorithms — LGBM, XGBM, and polynomial regression ranging from degrees 3 to 7 to develop regression models and identify the best-performing algorithm based on R^2, $\%\mu_E$ and $\%\sigma_E$ values derived from the trained models. The inputs for modeling AMS circuits encompass process, supply voltage, and temperature variations, while the outputs consist of the corresponding specifications.

4.4.2.1 Current Reference Generator

We designed a current reference circuit as described in [87], engineered using a standard 180nm CMOS process. The core functional circuit is optimized for the specific requirements of the current reference, which depend on the characteristics and variations of the MOS transistors and resistor R across PVT conditions. The dataset includes four inputs representing variations in supply voltage (Supply(V)), temperature (Temp(C)), PMON frequencies (Freq1(Hz), and Freq2(Hz)), which influence the output current (Current μA). The statistical correlation among their distributions in the dataset is illustrated in Fig. 4.6. We employed three regression techniques for circuit modeling: LGBM, XGBM, and polynomial regression of degree 7. The training metrics are outlined in Table 4.12. The most effective algorithm is LGBM, achieving an R^2 value of 0.997. Additionally, MaPE and $\%\sigma_E$ between actual simulations from Cadence ADE and model predictions are both less than 1%, affirming the model's capability in accurately representing the CUT.

Histograms displaying the distributions of actual simulations and predictions made by the LGBM model for the 10% test subset of their corresponding training datasets are depicted in Fig. 4.7.

4.4.2.2 Low Drop-out Regulator (LDO)

We investigated another commonly utilized standard analog circuit, namely LDO [88] as illustrated in Fig. 4.8, developed in 65nm technology. The crucial performance criterion for an LDO is its ability to maintain a stable output voltage despite variations in load and line conditions. The input, denoted as V_{ref}, represents the reference voltage obtained from a low-power bandgap reference (BGR). The dataset includes variations in V_{ref}, temperature, supply voltage, process variations (represented through three process frequencies), and induced fluctuations in load current and opamp output. In this context, the

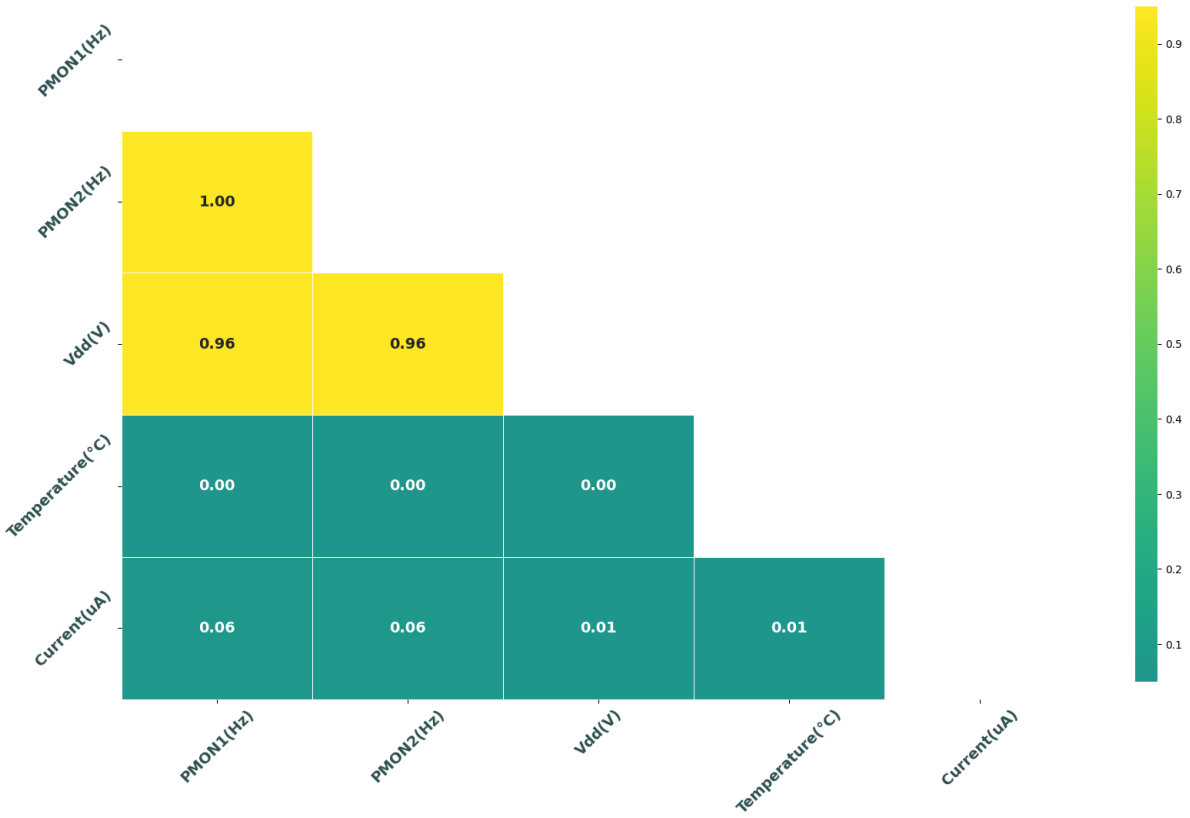


Figure 4.6: Correlation plot of current reference generator

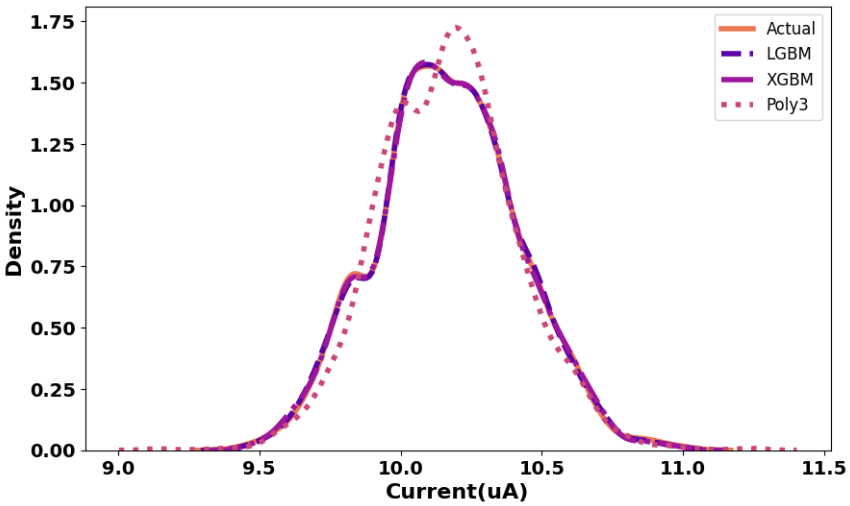


Figure 4.7: Comparison of the ML models' predicted output current Vs SPICE simulations of current reference generator

pass transistor assumes a significant role as the primary component, employed in self-adaptive design

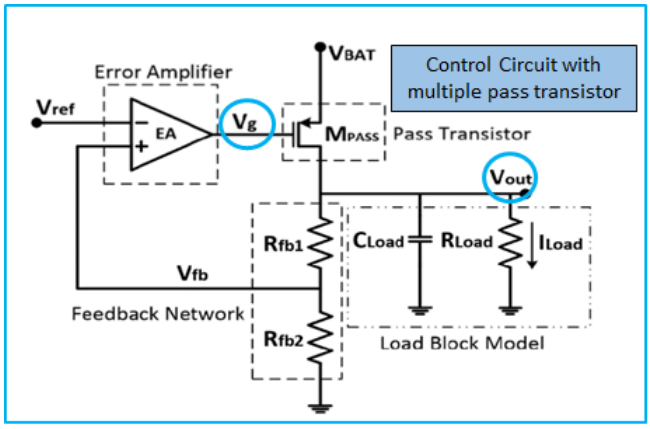


Figure 4.8: A low-dropout regulator circuit

for control code generation. Fig. 4.9 illustrates the statistical correlation among different parameters in the dataset, where PMON1, PMON2, and PMON3 represent process monitors.

Table 4.13 shows the training outcomes obtained with varying numbers of training samples. The metrics indicate satisfactory performance for all three algorithms. Histograms in Figs. 4.10 and 4.11 illustrate the distribution of machine learning predictions compared to actual simulations from Cadence ADE. Analysis of the regressor metrics distributions reveals that the LGBM model outperforms the others, emerging as the top-performing model.

4.4.2.3 A Free-running Clock Generator (Osc)

We examined a free-running RC oscillator [89] for our modeling purposes. This oscillator consists of two sub-blocks: a bias generator and an oscillator core, illustrated in Fig. 4.12. The biasing generator provides the necessary gate bias for the NMOS [M4] in conjunction with the oscillator core. While the biasing generator is designed to offer a supply-independent current reference in theory, there inevitably exists some influence from fluctuations in power supply and temperature. Therefore, the dataset for the oscillator covers a range of variations in biasing voltages, NMOS resistance bias, and PVT conditions. Process monitors PMON1 to PMON3 are included. The performance of the developed LGBM, XGBM, and MPR models in accurately predicting the oscillator frequency while considering all necessary design and PVT variations is depicted in Fig. 4.15. The scatter plot showcasing different PVT conditions is presented in Fig. 4.13. Remarkably, predictions from all three models closely align with the actual simulations. The associated metrics, comparing the predicted distributions for the test split, are tabulated in Table 4.14, where all models demonstrate $\%Error$ in mean, variance, and standard deviations $< 1\%$.

Table 4.13: Training metrics of Low dropout regulator modeling

5000 samples of training data							
Voltage(V)/ ML metrics	**Actual simulations**	**LGBM**		**XGB**		**PR**	
		Model predictions	$\%Error$	**Model predictions**	$\%Error$	**Model predictions**	$\%Error$
Mean	1.8532	1.8532	2.90E-07	1.8532	0.0004	1.8532	7.19E-14
Variance	0.0011	0.0011	0.1948	0.0010	0.6207	0.0011	0.1923
Standard deviation	1.15E-06	1.15E-06	0.0974	1.08E-06	3.1535	1.15E-06	0.0962
7500 samples of training data							
Mean	1.8530	1.85304	0.0000	1.8530	0.0003	1.8530	8.39E-13
Variance	0.0027	0.0027	0.1370	0.0027	2.9299	0.0027	0.1368
Standard deviation	7.27E-06	7.26E-06	0.0685	7.06E-06	1.4758	7.26E-06	0.0684
14000 samples of training data							
Mean	1.8525	1.8525	1.13E-07	1.8525	0.0001	1.8525	1.02E-06
Variance	0.0047	0.0047	0.0694	0.0047	0.6276	0.0047	0.0872
Standard deviation	2.23E-05	2.22E-05	0.0347	2.21E-05	0.3143	2.22E-05	0.0436

Table 4.14: Training metrics of Free running oscillator modeling

4750 samples of training data							
Clock frequency(Hz)/ ML metrics	**Actual simulations**	**LGBM**		**XGB**		**PR**	
		Model predictions	$\%Error$	**Model predictions**	$\%Error$	**Model predictions**	$\%Error$
Mean	45918.10	45918.10	0.00	45918.10	9.94E-06	45918.10	0.00
Variance	5071.31	5065.97	0.21	5065.97	0.21	5065.97	0.21
Standard deviation	2.57E+07	2.57E+07	0.11	2.57E+07	0.11	2.57E+07	0.11
8500 samples of training data							
Mean	48751.10	48751.20	2.10E-05	48751.10	-1.49E-05	48751.10	3.91E-05
Variance	6263.39	6259.73	0.12	6259.73	0.12	6259.72	0.12
Standard deviation	3.92E+07	3.92E+07	0.06	3.92E+07	0.06	3.92E+07	0.06

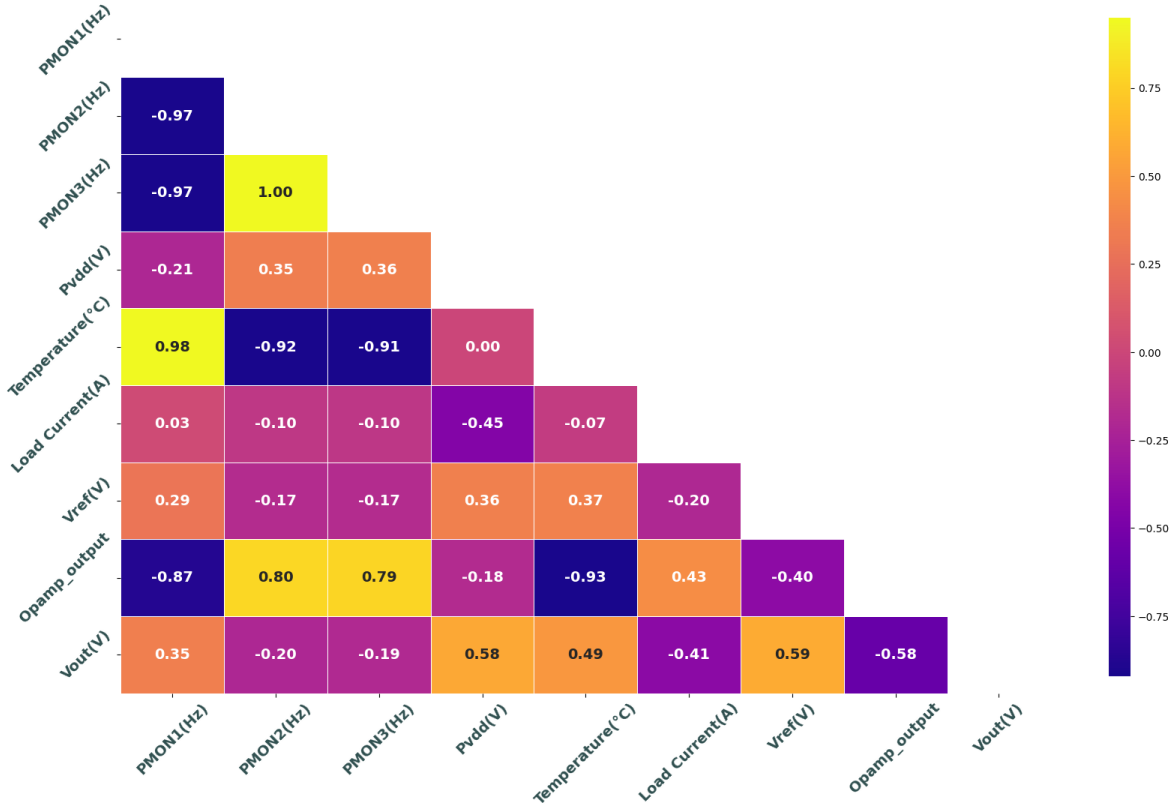


Figure 4.9: Correlation plot of LDO

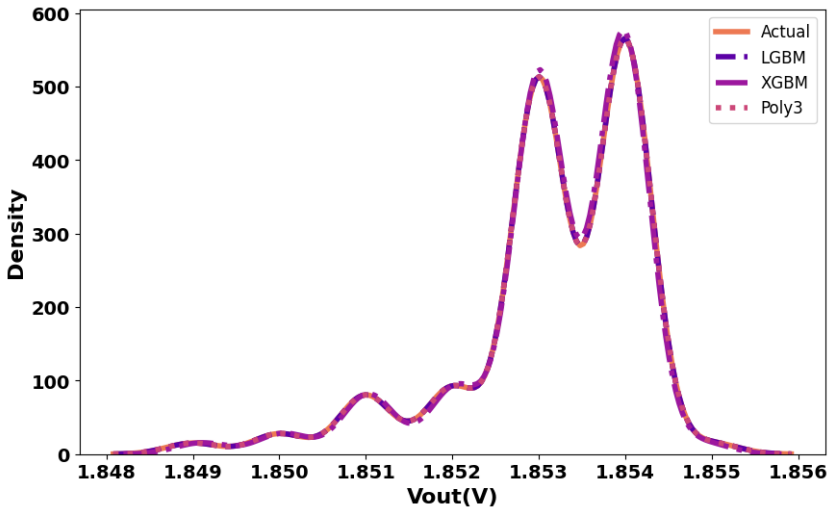


Figure 4.10: Comparison of the ML models' predicted output voltage Vs SPICE simulations of LDO with 5000 training samples

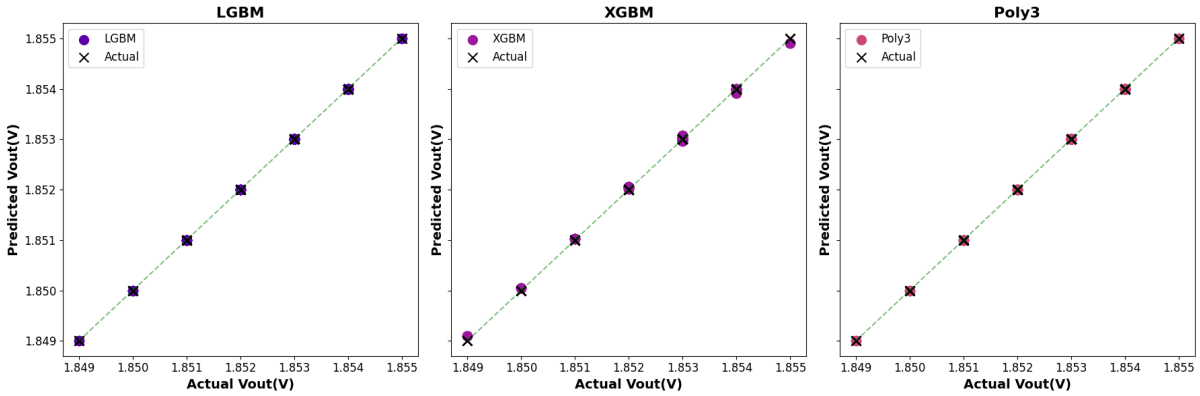


Figure 4.11: Scatter plot of SPICE simulations and ML models' predicted output voltage of LDO with 5000 training samples

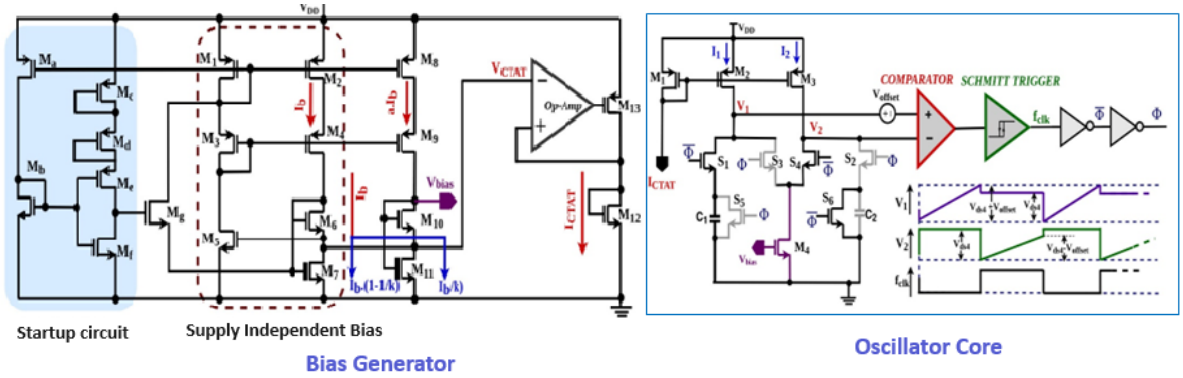


Figure 4.12: A free-running oscillator circuit

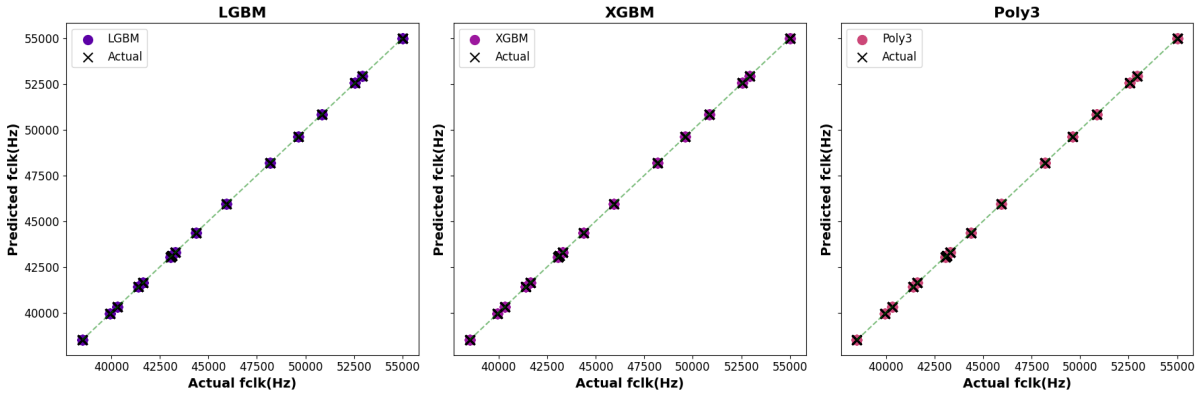


Figure 4.13: Scatter plot of SPICE (actual) and ML models' predicted clock frequency of oscillator with 4750 training samples

4.4.2.4 Transmitter Driver (TxDr)

We designed a differential voltage mode driver TxDr [90] operating at a data rate of 6 Gb/s (gigabits per second), engineered to propel high-speed differential signals with a voltage mode driver configu-

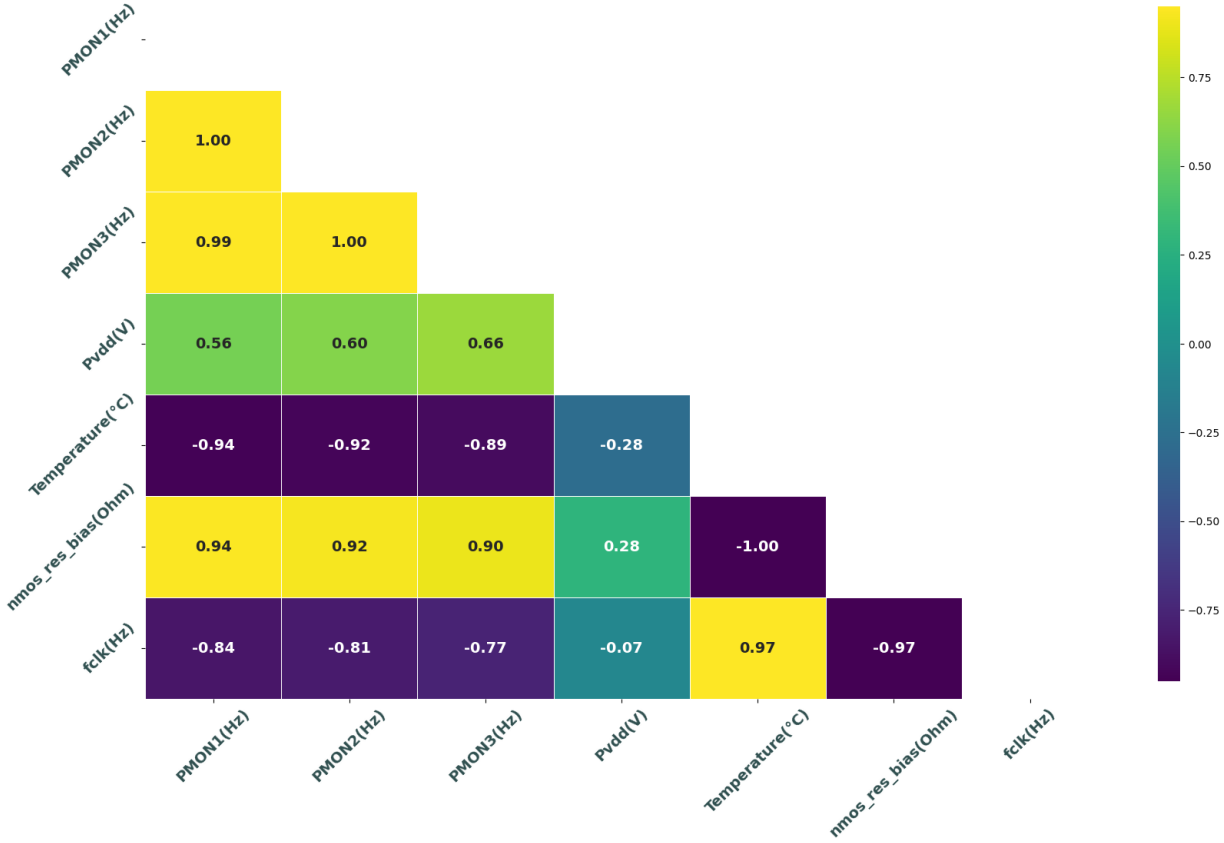


Figure 4.14: Correlation plot of oscillator

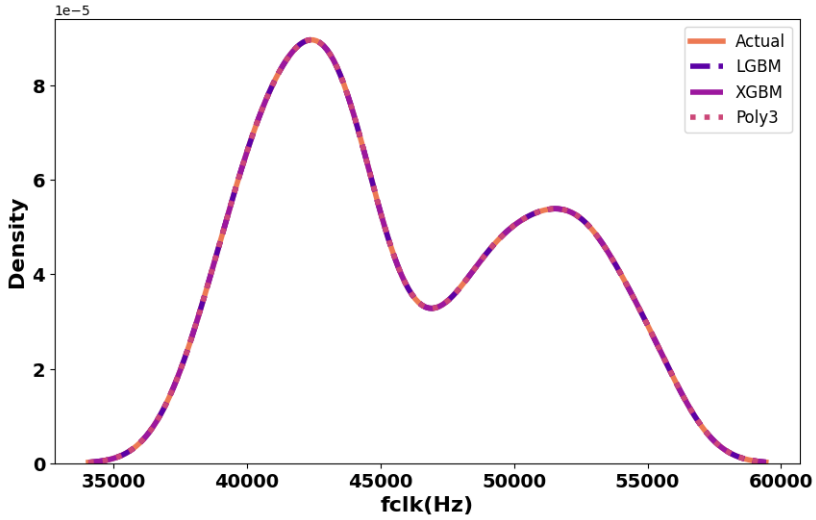
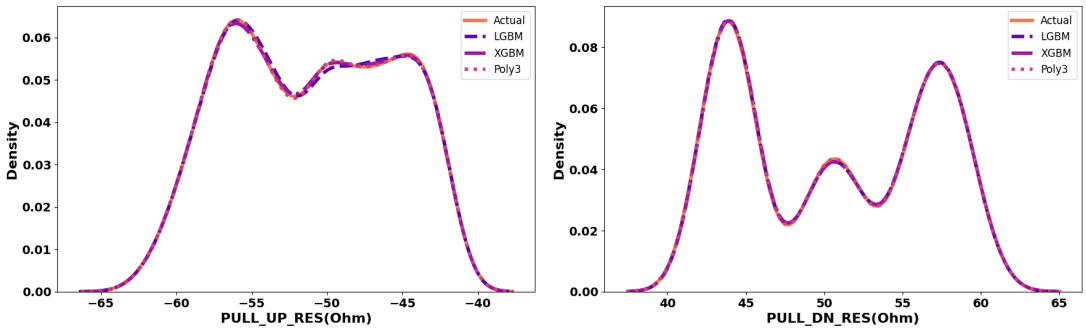


Figure 4.15: Comparison of ML model's predicted oscillator frequency Vs SPICE (actual) simulations



(a) (b)

Figure 4.16: Comparison of accuracy on TxDr outputs - SPICE (actual) and ML models' predictions (a) Pull up resistance (b) Pull down resistance

Table 4.15: Training metrics of Transmitter Driver modeling

Training dataset with 16473 samples						
Regressor	R^2	**RMSE**	**MAE**	μ_E	σ_E	**Training time (Sec)**
Pull up resistance modeling metrics						
LGBM	0.999	0.133	0.097	0.004	0.123	2.770
XGBM	0.999	0.163	0.097	0.006	0.154	0.662
Poly3	1.000	0.029	0.022	0.002	0.026	0.060
Pull down resistance modeling metrics						
LGBM	0.999	0.175	0.125	0.005	0.187	2.520
XGBM	0.999	0.234	0.120	0.003	0.222	0.870
Poly3	1.000	0.033	0.025	0.002	0.034	0.098

ration. This setup boasts reduced current consumption, occupies less area, and offers superior noise immunity compared to current mode drivers. Additionally, it features independent control of the pre-emphasis level. The capability to adjust pre-emphasis levels independently facilitates optimizing signal integrity and compensating for channel losses. To capture process variations in CMOS and resistors, we employed two process monitors ('PMON CMOS res' and 'PMON res'). Fig. 4.17 illustrates that these two PMONs are highly correlated, as are the pull-up and pull-down resistors. Such high correlation between the PMONs and resistors suggests that one of them suffices for modeling the impact of PVT. Consequently, we utilized variations in 'PMON CMOS res', supply, and temperature to model the pull-up and pull-down resistors separately, as depicted in Fig. 4.16. Both models proved accurate, leading to the utilization of pull-up resistance predictions for self-adaptation at a later stage. Table 4.15 illustrates that polynomial regression with a degree of 3 exhibits an advantage over gradient boosting models.

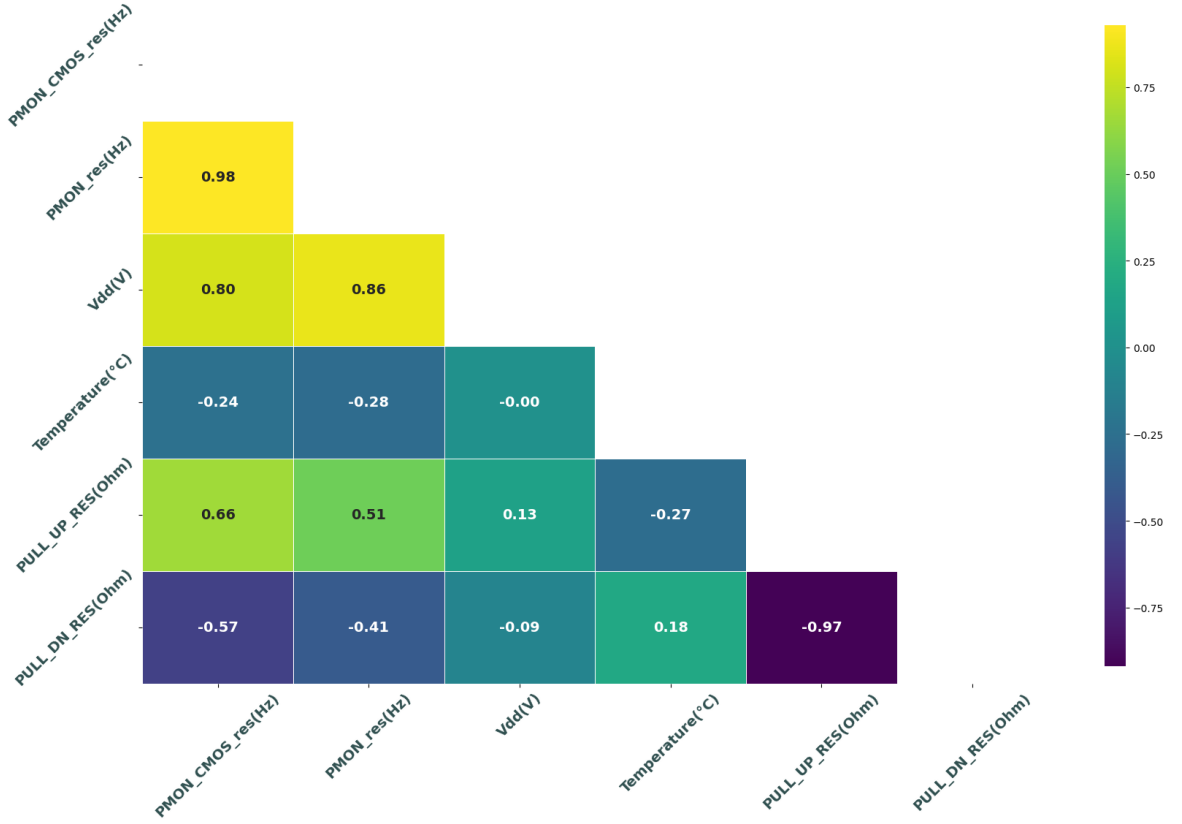


Figure 4.17: Correlation plot of TxDr

We have additionally created classification models to discern the operational corner of the TxDr concerning variations in PVT. The training samples across five corners - 'SS', 'FF', 'TT', 'SF', 'FS' are generated and applied as imput to the classifiers. We conducted experiments with five classification algorithms: Logistic regression, DT, SVM, KNN and LGBM with their accuracy scores post-training depicted in Fig. 4.18. The accuracy of SVM, KNN and LGBM is recorded as 100%. The prediction accuracy is also visualized thorough confusion matrix in Fig. 4.19. The predicted and actual corners are exactly matching for 330 unseen test cases.

4.4.2.5 Band-gap Reference (BGR)

We configure a compact self-adaptive BGR with sub-threshold biased MOS devices capable of operating at a reduced power supply voltage level as low as 0.6V while meeting precision and system-level integration requirements [91]. A significant challenge in BGR design is ensuring temperature stability, thereby guaranteeing that the reference voltage remains consistent across a broad range of temperatures. Tracking process variation is crucial in BGR design, a task that can be accomplished using the existing process monitor module of the Silicon On Chip (SOC).

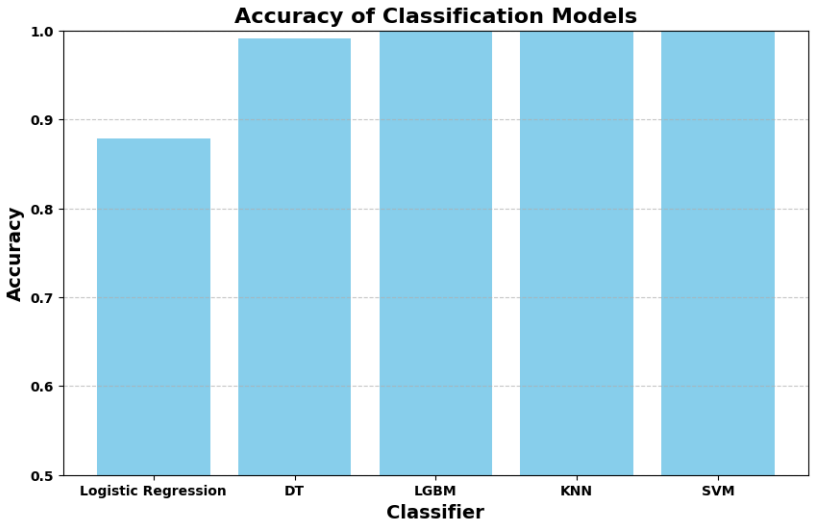


Figure 4.18: Classification models' accuracy scores on training TxDr

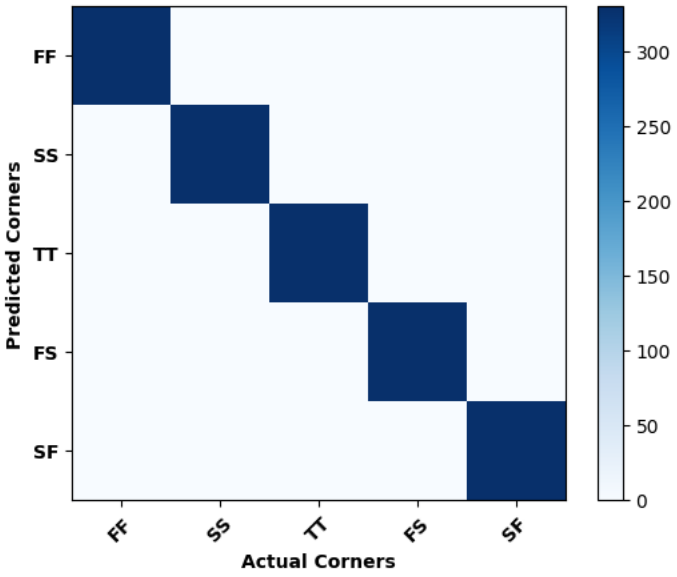


Figure 4.19: Confusion matrix of TxDr showing 100% accuracy

The dependency of the output voltage (Vbgr(V)) on the supply voltage (Vdd(V)) and the dominant PMON frequency ('PMON RING CMOS') is illustrated in Fig. 4.20. Temperature dependency is observed in internal parameters of the BGR, such as 'PTAT current VR' and 'Vinp VR(V)' of the voltage reference. 'Vinp VR(V)' and 'Pbias VR(V)' refer to the supply voltage and bias voltage of the voltage reference, respectively. 'VR ref tran(V)' signifies the transient behavior or response of the reference voltage (VR ref) generated by the BGR circuit, indicating how rapidly the reference voltage settles to its steady-state value when subjected to changes in operating or load conditions. 'PTAT current VR' represents the Proportional-To-Absolute-Temperature (PTAT) current generated by the BGR circuit, a

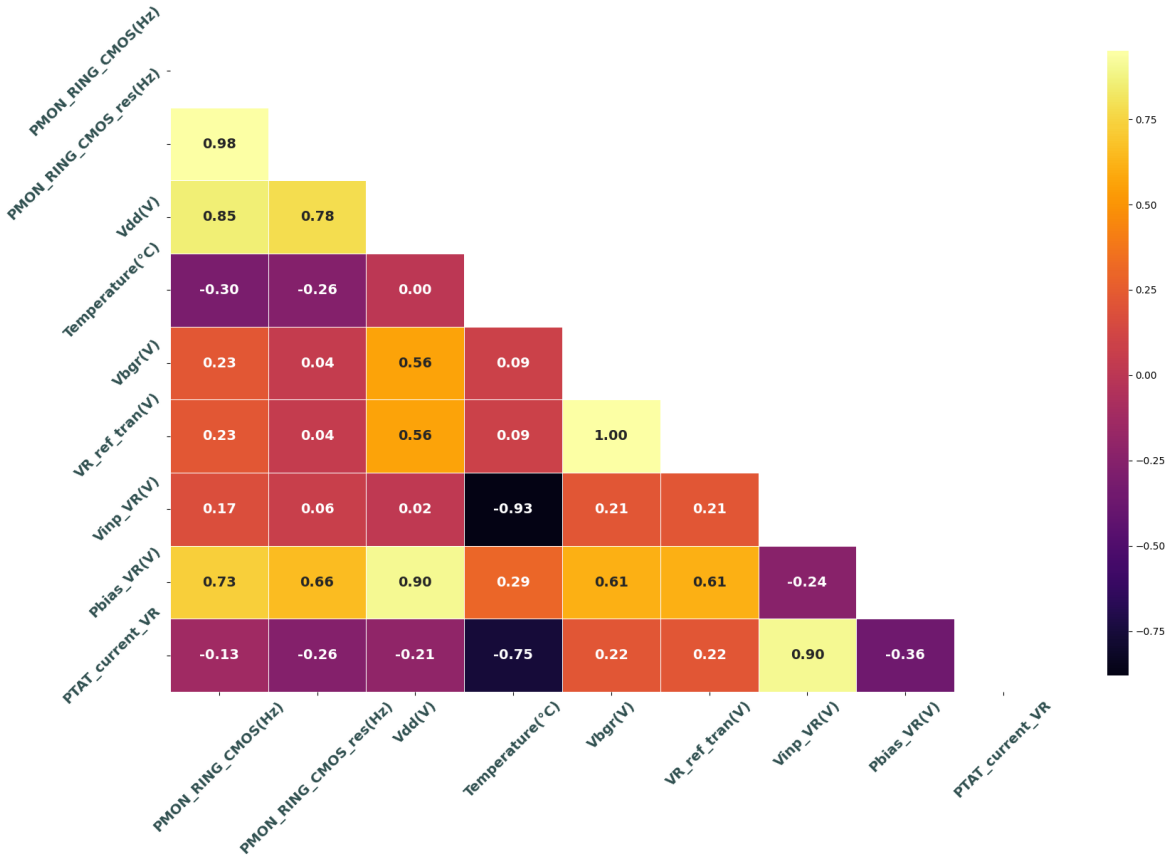


Figure 4.20: Correlation plot of bandgap reference

crucial element in band-gap reference circuits utilized to generate a temperature-dependent voltage that compensates for variations in the base-emitter voltage of bipolar junction transistors.

The modeling metrics and comparisons with actual simulations from ADE are presented in Table 4.16 and Figs. 4.21 and 4.22, respectively. A comprehensive analysis of the results indicates that LGBM comparatively outperforms other models.

Moreover, we developed classification models to identify the operational corner of the BGR concerning variations in PVT. The classifier accuracy scores post-training are depicted in Fig. 4.23, where SVM, KNN, and LGBM achieve an accuracy, recall and F1 score of 100%, surpassing the logistic regressor and DT.

4.4.2.6 Phase-Locked Loop (PLL)

The Phase-Locked Loop functions as a closed-loop system with negative feedback, designed to track the input signal. It serves the purpose of generating high-frequency signals from low voltage-to-frequency V_{th} signals while maintaining the same phase as the input signal. Stability is paramount in PLL operation due to its negative feedback mechanism. Variations in internal parameters induced

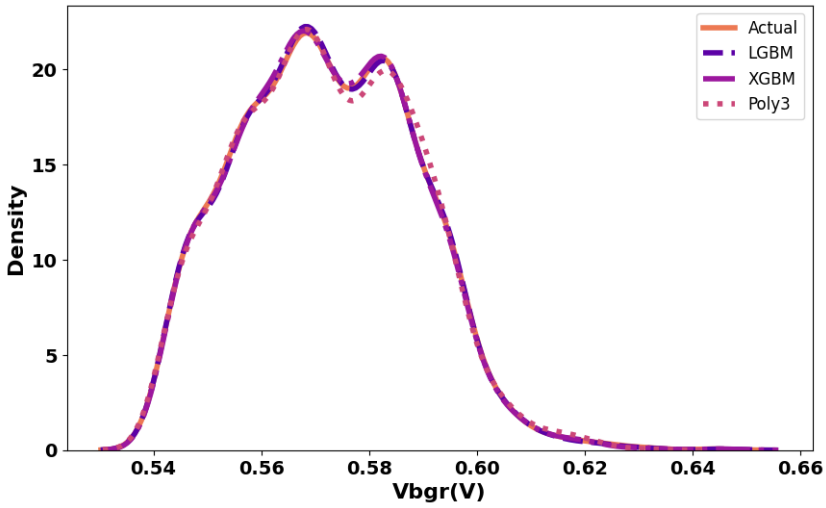


Figure 4.21: Comparison of ML model's predicted BGR output voltage w.r.t SPICE simulations

Table 4.16: Training metrics of Bandgap reference modeling

Training dataset with 18150 samples						
Regressor	R^2	**RMSE**	**MAE**	μ_E	σ_E	**Training time (Sec)**
LGBM	0.9987	0.0006	0.0004	0.0016	0.3973	3.3461
XGBM	0.9975	0.0008	0.0005	0.0048	0.6588	1.1725
Poly3	0.9873	0.0018	0.0010	0.0143	-0.2264	0.0668

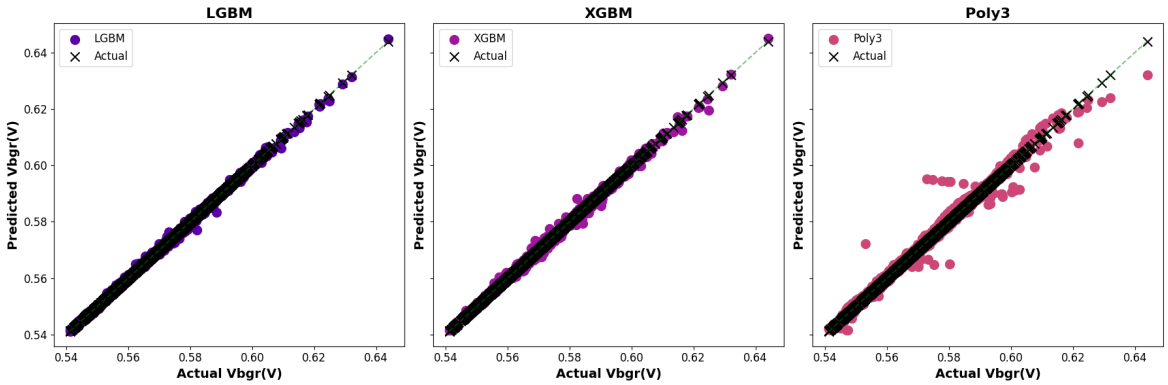


Figure 4.22: Scatter plot of SPICE and ML models' predicted output voltage of BGR

by PVT can lead to instability. Maintaining stability in the PLL loop becomes challenging as the Kvco (Voltage-Controlled Oscillator Gain) varies significantly, typically ranging from 2 to 3 times with re-

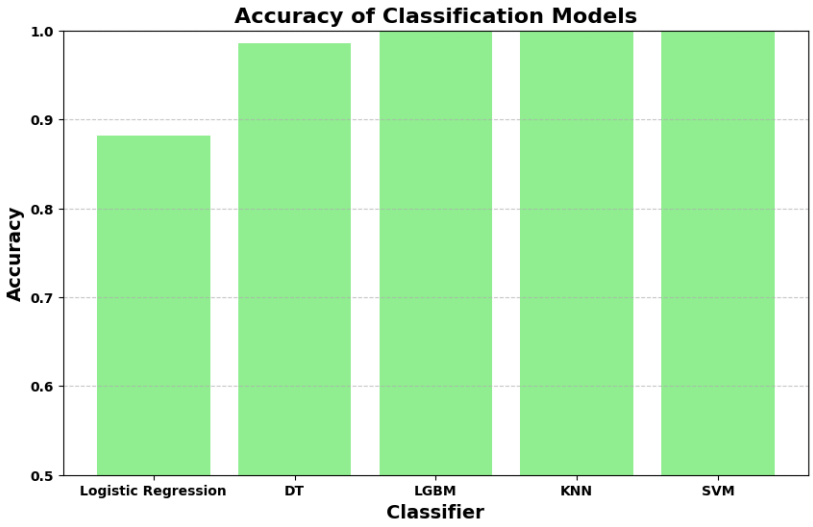
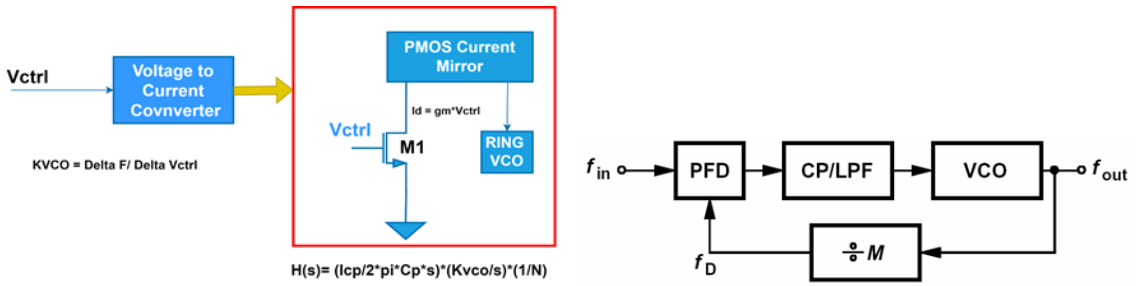


Figure 4.23: Classification accuracy scores on training BGR



(a) (b)

Figure 4.24: Phase-locked loop (a) Circuit diagram (b) Internal diagram

spect to PVT variations. Transistor M1 is responsible for generating the Id current.

$$I_d = \frac{1}{2} U_0 C_{ox} (Vctrl - V_{th})^2 \tag{4.9}$$

Open loop transfer function of type II 3^{rd} order PLL is given by,

$$H(s) = \frac{I_{cp}}{2\pi C_p s} \frac{Kvco}{s} \frac{1}{N} \tag{4.10}$$

where, I_{cp} denotes the charge pump current, C_p represents the low-pass filter capacitor, Kvco stands for the Voltage-Controlled Oscillator (VCO) gain, and N represents the divider ratio. Due to variations in V_{th} and mobility with temperature, Kvco undergoes significant fluctuations, leading to stability issues. To mitigate Kvco variation, we utilize a trained PVT-aware ML model. We determine the PMON frequencies, which capture process variations, using the ring oscillator structure illustrated in Fig. 4.25. Similar ring oscillators are employed for all circuits to capture the process variations.

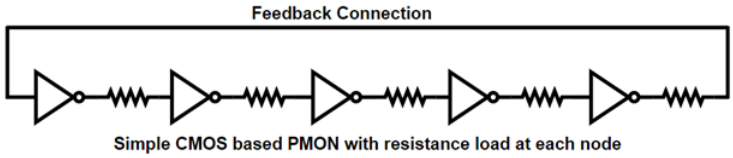


Figure 4.25: Ring Oscillator structure to derive PMON frequencies

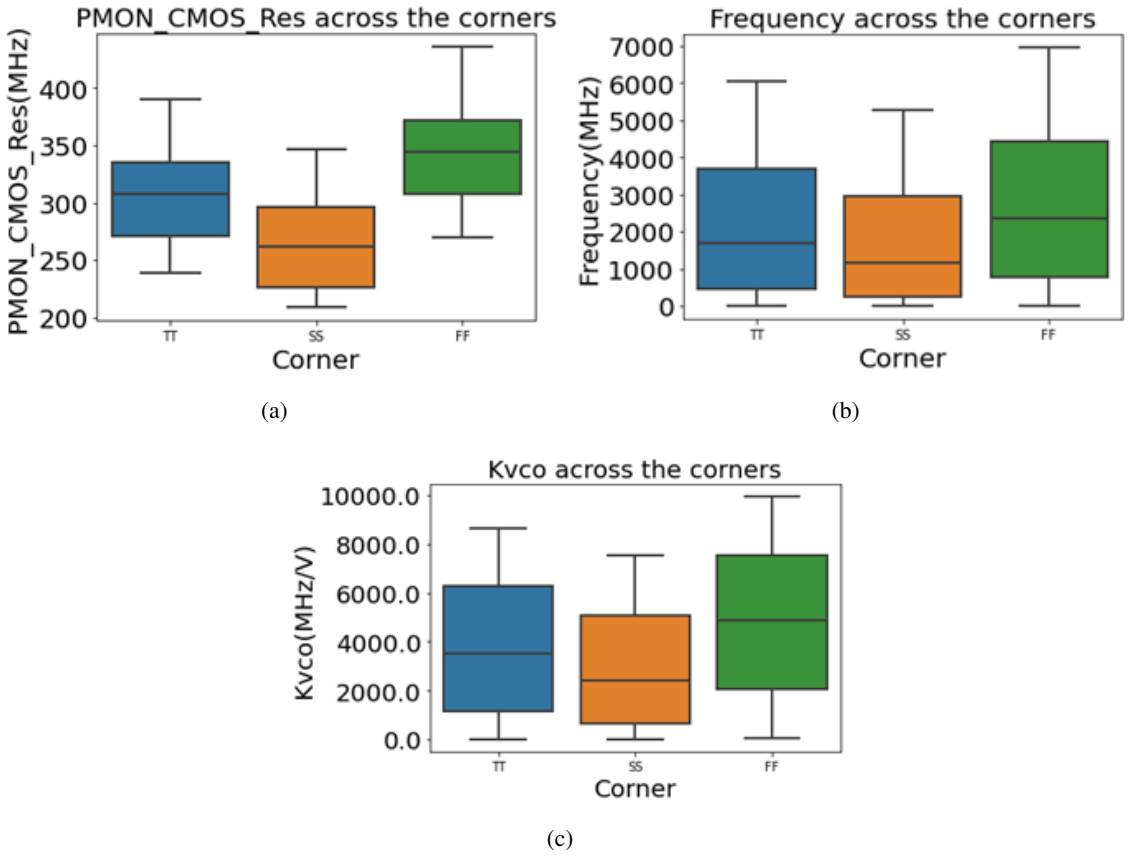


Figure 4.26: Distributions of PLL (a) PMON frequencies (b) Frequency (c) Kvco across different corners

We observed a significant overlap between the distributions of PLL frequency and Kvco across various corners, including PMON frequencies, as depicted in Fig. 4.26. Therefore, for our modeling, and particularly for self-adaptation, it becomes crucial to precisely identify the operational corner of the circuit under given PVT conditions. To achieve this, we employed classifiers such as Logistic Regression, DT, SVM, KNN and LGBM. All these algorithms accurately predict the PLL corners under PVT conditions, achieving a perfect accuracy score of 1.0 (100%). The accuracy scores of these classifiers are illustrated in Fig. 4.27.

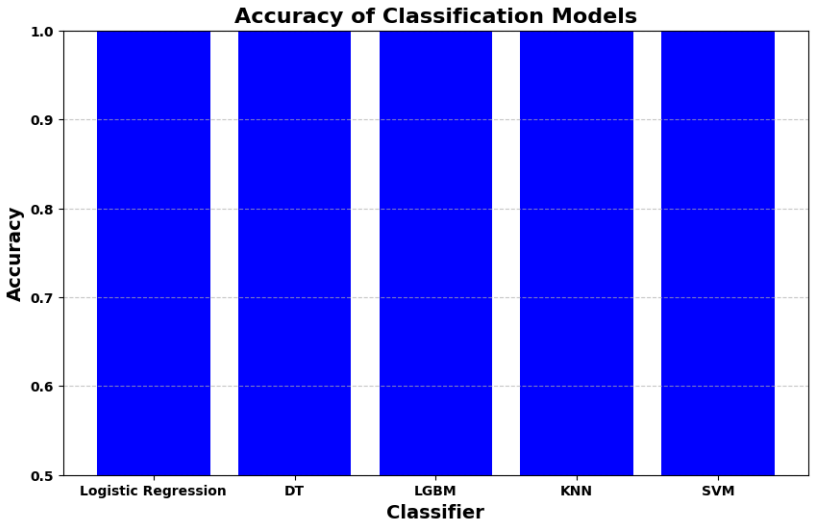


Figure 4.27: Classification accuracy scores on training PLL

4.4.2.7 Low-noise Amplifier (LNA)

Low-noise amplifiers play a vital role in front-end receivers, although their performance can be susceptible to temperature variations. In this case study, ML-based self-adaptive biasing techniques are utilized to dynamically adjust the LNA's biasing conditions based on real-time temperature measurements. The gain of LNAs is a critical specification directly impacting their performance across various applications. Its reliance on MOSFET characteristics and sensitivity to PVT variations emphasize the significance of calibration, ensuring reliable and consistent operation across diverse conditions and environments. The resistive feedback topology in LNA design plays a pivotal role in noise reduction. This configuration, incorporating resistors in the feedback loop, contributes to minimizing unwanted noise in the amplifier. The resistive feedback technique aids in achieving a balance between signal amplification and noise figure, ensuring that the amplifier introduces as little noise as possible to the incoming signal.

The addition of a second stage common-source amplifier is aimed at increasing gain and maintaining an output impedance of 50 Ohms. The LNA [92], as depicted in Fig. 4.28, is designed using a resistive feedback technique, replacing inductors with resistors to achieve a smaller die area, better linearity, and higher gain. The designed LNA achieves a gain greater than 14dB, a noise figure (NF) less than 3.21dB, and a linearity of -13dB for the required frequency of 1GHz.

The modeling metrics of the LNA are presented in Table 4.17. Both LGBM and XGBM demonstrate superior performance compared to MPR (degree 3), with an R^2 value close to 1.

4.4.2.8 High-Speed Low-noise Amplifier (HSLNA)

The high-speed LNA (Fig. 4.31) is designed in a 28nm technology node, boasting a gain of 18dB, a NF of 3dB, and a third-order intercept point (IIP3) of -15dB, functioning at 25GHz. The power dissipation of the design is 7.8mW. Both gain and NF are adjusted concurrently, ensuring that IIP3 exceeds

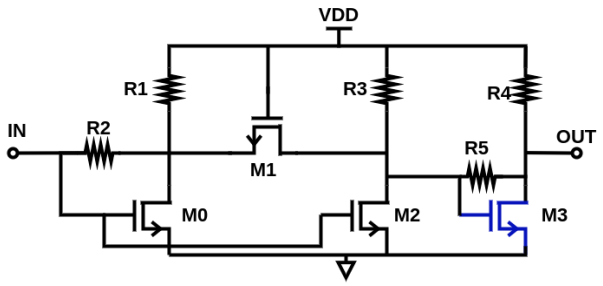


Figure 4.28: Low noise amplifier circuit

Table 4.17: Training metrics of Low noise amplifier modeling

Training dataset with 2000 samples						
Regressor	R^2	**RMSE**	**MAE**	μ_E	σ_E	**Training time (Sec)**
LGBM	0.9999	0.0172	0.0106	0.0003	0.1936	0.2225
XGBM	1.0000	0.0104	0.0031	0.0096	0.2950	0.2438
Poly3	0.9992	0.0492	0.0370	-0.0118	0.2156	0.0095

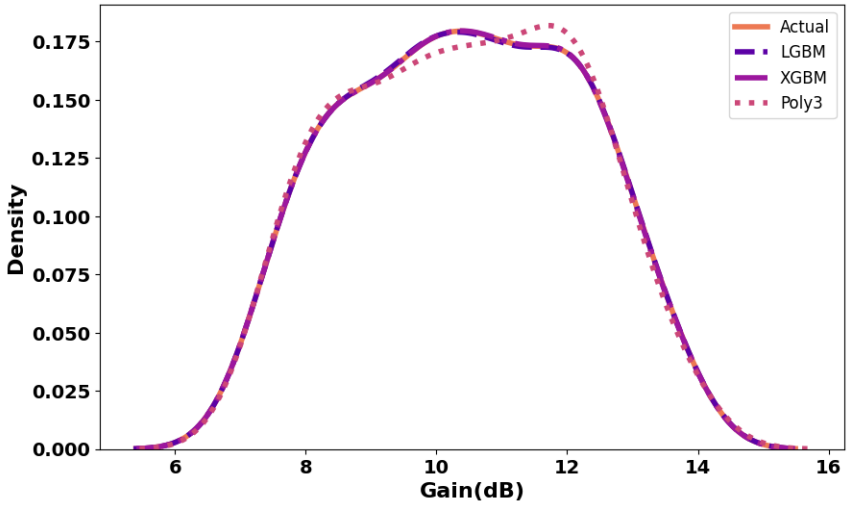


Figure 4.29: Comparison of ML models' predicted output gain of LNA Vs SPICE (actual) simulations

-20dB. The supply voltage is set at 0.9V. The common-source configuration with source degeneration is adopted in the design to enhance matching and impedance transformation capabilities. This configuration is advantageous for interfacing with subsequent stages in the RF signal chain, facilitating efficient signal transfer and minimizing losses. Regarding biasing, the performance of the LNA is significantly influenced by the bias current flowing through it. To achieve optimal performance across various con-

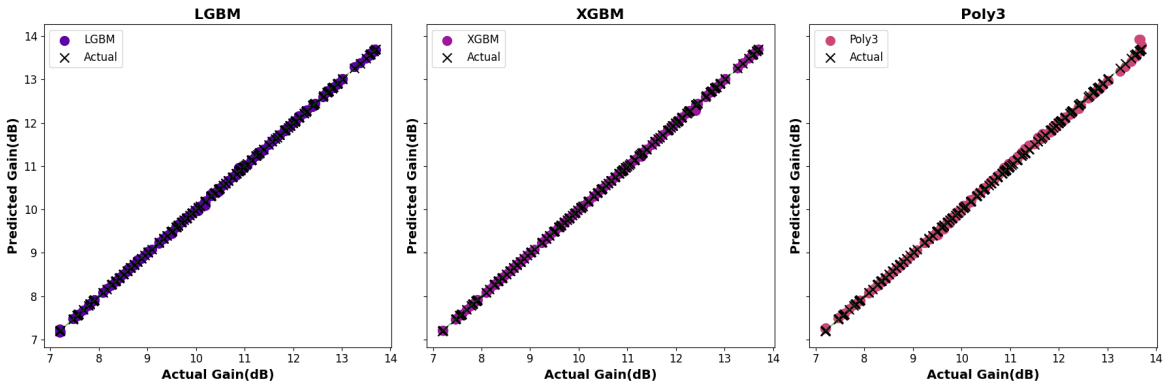


Figure 4.30: Scatter plot of SPICE and ML models' predicted output gain of LNA

ditions, a beta-multiplier circuit is employed to determine the bias current. Tuning the resistor of the beta-multiplier circuit is crucial to attain optimal results and ensure the desired performance of the LNA.

The heatmap (Fig. 4.32) illustrates a strong correlation between PMON and the output specifications, namely gain and noise figure. There is a notable correlation between supply voltage and temperature with the output specifications. Gain and noise margin exhibit a strong inverse linear relationship. During training, all three regression models developed consistently demonstrate performance in modeling gain and noise figure, with polynomial regression holding a slight advantage.

4.4.2.9 Mixer

The Mixer plays a critical role in radio frequency receivers as a frequency translator, commonly referred to as a heterodyne. Operating as a nonlinear circuit, it performs signal multiplication essential for generating both sum and difference frequencies, facilitating up and down conversion. With its versatility, the mixer finds applications in phase detection, modulation, frequency multiplication, and product detection, establishing itself as a fundamental component within the radio frequency receiver's front end. Parameters such as conversion gain, integrated noise margin, third-order input intercept point (IIP3), and input-referred 1dB compression point are crucial for evaluating its performance. The Mixer is designed in a 65nm technology node, boasting an IIP3 of -3.263 dBm and a P1dBm of -13.18 dBm at an input frequency of 900 MHz, following the design principles outlined in [93].

The training results and metrics of the mixer outputs are depicted in Fig. 4.34 and Table 4.18, respectively. As per modeling results, LGBM outperforms XGBM and MPR.

In all of these CUTs, the temperature range spans from $-40°C$ to $125°C$, with the supply voltage varying within $\pm 10\%$ from the nominal value and process variation extending up to 80% from the typical value. For analog circuit modeling, we explored various regression techniques and advocate the use of MPR and LGBM for efficiently calibrating analog circuits susceptible to PVT effects. Both algorithms accurately capture dependencies within analog circuits, with LGBM consistently demonstrating robust performance.

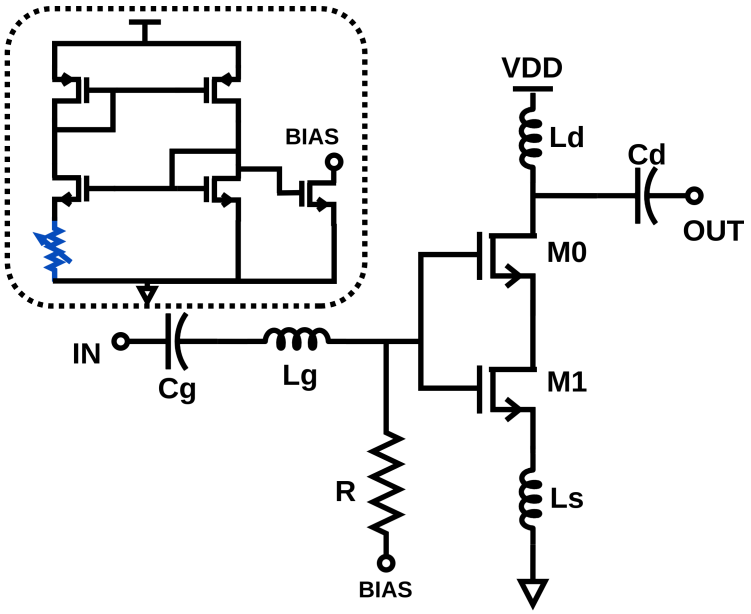


Figure 4.31: High-Speed low noise amplifier circuit

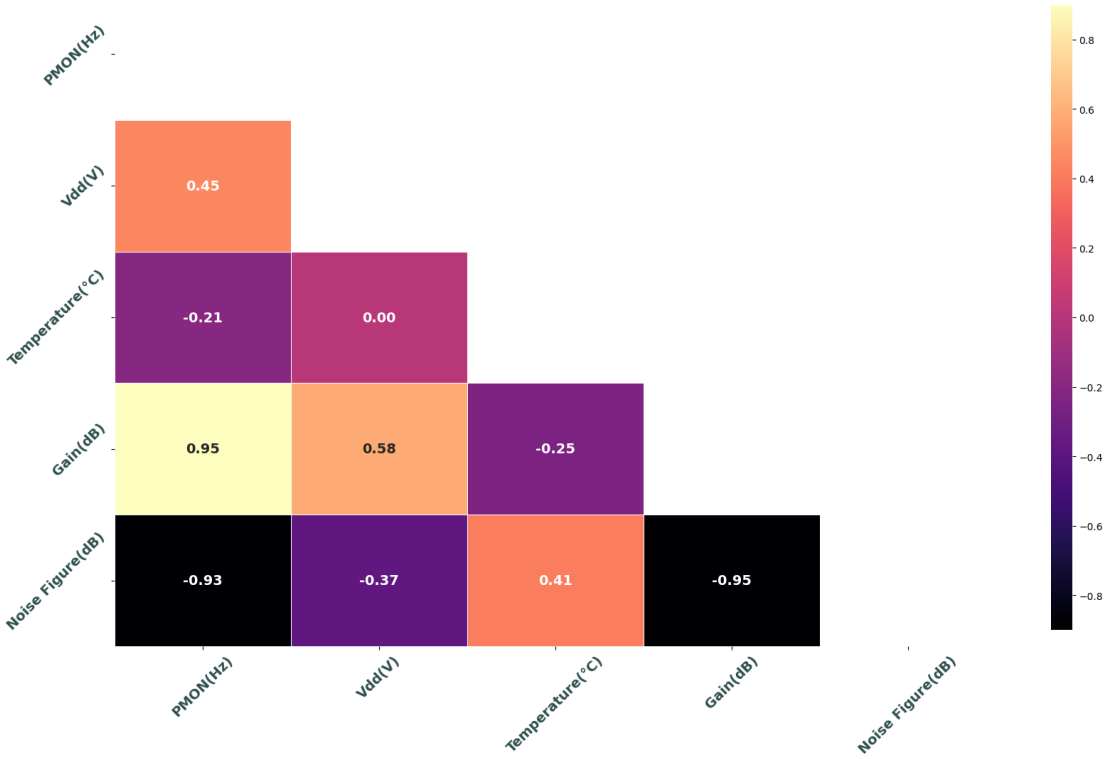
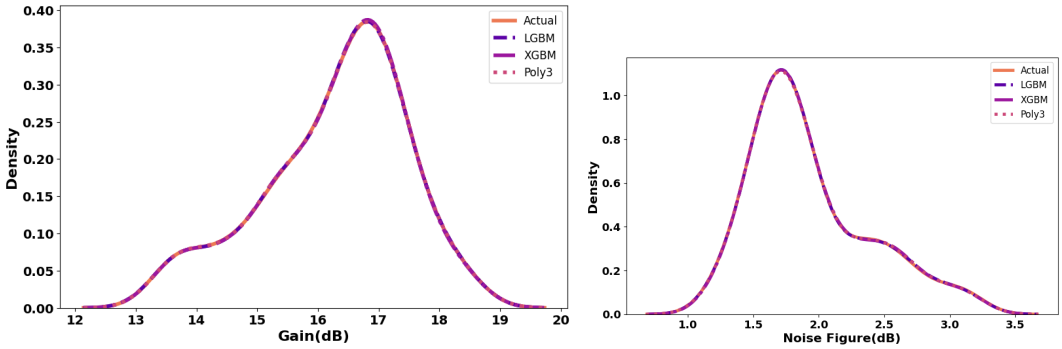


Figure 4.32: Correlation plot of high-speed LNA



(a) (b)

Figure 4.33: Distribution plot of high-speed LNA - Actual and ML predictions (a) Gain(dB) (b) Noise figure(dB)

Table 4.18: Training metrics of Mixer modeling

Training dataset with 2000 samples						
Regressor	R^2	**RMSE**	**MAE**	μ_E	σ_E	**Training time (Sec)**
Conversion Gain(dB)						
LGBM	0.9985	0.0630	0.0467	0.1558	0.6348	0.6058
XGBM	0.9886	0.1757	0.0558	0.1727	1.3296	0.2074
Poly3	0.9970	0.0906	0.0651	-0.0109	0.1758	0.0113
Noise Figure(dB)						
LGBM	0.9997	0.0153	0.0104	-0.0065	0.4462	0.5835
XGBM	0.9990	0.0280	0.0144	0.0151	0.2771	0.4319
Poly3	0.9993	0.0227	0.0161	0.0051	0.1459	0.0305
Input Referred 1dB Compression Point(dBm)						
LGBM	0.9989	0.0409	0.0252	0.0070	0.1003	0.5844
XGBM	0.9983	0.0494	0.0295	0.0113	0.1700	0.1933
Poly3	0.9924	0.1058	0.0754	-0.0505	-0.3076	0.0149
Input Referred IP3 Point(dBm)						
LGBM	0.9994	0.0319	0.0241	0.0203	0.2345	0.9329
XGBM	0.9988	0.0429	0.0284	0.1012	0.4600	0.5138
Poly3	0.9728	0.2071	0.1763	-0.0321	4.2215	0.0605

Table 4.19: Training metrics of High-speed low noise amplifier modeling

Training dataset with 2000 samples						
Regressor	R^2	**RMSE**	**MAE**	μ_E	σ_E	**Training time (Sec)**
Gain(dB)						
LGBM	0.9995	0.0259	0.0188	0.0043	0.2473	0.2737
XGBM	0.9995	0.0259	0.0214	0.0005	0.4272	0.6978
Poly3	0.9998	0.0143	0.0109	0.0056	0.2559	0.0221
Noise Figure(dB)						
LGBM	0.9996	0.0093	0.0071	-0.0140	0.3752	0.5601
XGBM	0.9991	0.0139	0.0108	-0.0814	0.6100	1.0943
Poly3	0.9998	0.0065	0.0050	-0.0047	0.2028	0.0200

Among classifiers, SVM and LGBM consistently perform well, achieving an accuracy score of 1.0 (100%).

4.4.3 Unified Deep-Learning Neural Network Architecture (U-DNN)

Machine Learning surrogate models have been increasingly replacing the traditional simulators due to their computational advantages. NNs have attracted significant attention as a prominent AI/ML modeling approach across various AMS applications [47]. They excel in capturing complex circuit behaviors and providing precise models for analog circuit performances' inherently high-dimensional, non-linear nature. This has led many researchers to investigate NN implementations for tasks such as analog circuit modeling, design sizing, and optimization [49–54]. The exploration has extended to NN-based transfer learning techniques, which enable reduced data requirements and the utilization of knowledge from pre-existing models [94,95]. However, the researchers have typically developed distinct NN architectures for modeling the output performances of each specific circuit under examination. This means that circuit-specific NN architectures have been the norm. While there have been instances of applying pre-trained NNs to various topologies of the same circuit [95], their suitability for entirely new and diverse applications remains uncertain, i.e, these surrogate models are typically tailored to specific circuits, requiring significant design efforts and lacking re-usability across different designs and topologies. Consequently, this necessitates a non-trivial design space exploration process to determine a suitable NN architecture for each new class of AMS circuits. Our unified neural network architecture addresses these challenges by offering versatility and computational efficiency. Our approach provides a streamlined and effective method for modeling a wide range of analog circuits susceptible to PVT variations. We conduct experimental trials on various analog circuits to demonstrate the applicability of our proposed model across different CMOS technology nodes (180nm, 65nm, and 28nm).

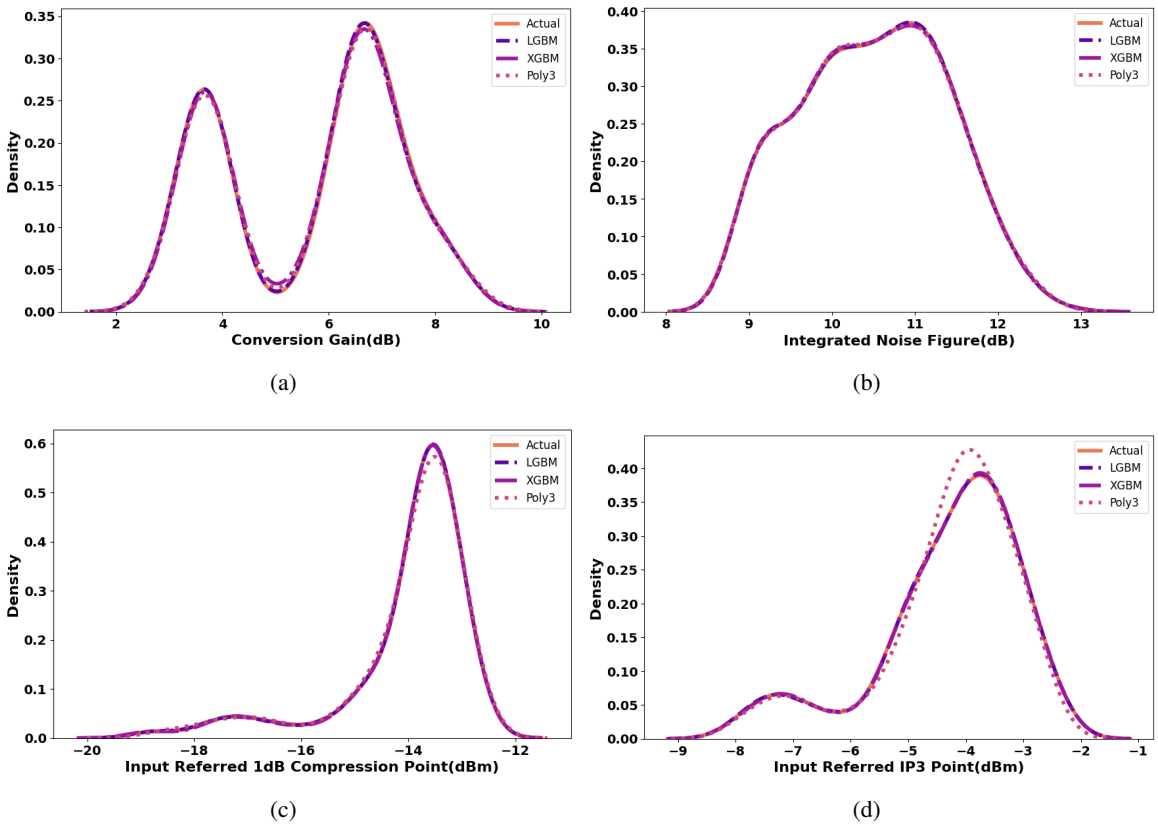


Figure 4.34: Performance comparison of SPICE (actual) and ML predictions (a) Conversion gain(dB) (b) Integrated noise figure(dB) (c) Input Referred 1dB Compression Point(dBm) (d) Input Referred IP3 Point(dBm)

4.4.3.1 Key Idea

The core concept originates from critically examining the surrogate model generation techniques in contemporary literature. These models exhibit limitations in their capacity to adapt the established NN architecture from one circuit to simulate another. Rather than leveraging existing NN structures, they require the construction of a surrogate model from scratch for each individual design. Consequently, this approach mandates an exhaustive exploration of hyperparameters across a vast design space to determine an optimal NN configuration for each application, consuming substantial time in parameter search before model training. Furthermore, if the desired level of accuracy is not achieved, this cycle necessitates repetition.

Driven by this significant drawback, we introduce the U-DNN approach, which addresses the need for versatility, re-usability, and generality in AMS circuit characterization. Proficient in capturing PVT-sensitive behavior across diverse circuits and technology nodes, the U-DNN simplifies the identification of optimal neural network architectures, streamlining characterization processes. This saves valuable time and reduces the effort required in model development. Thorough evaluations demonstrate its effi-

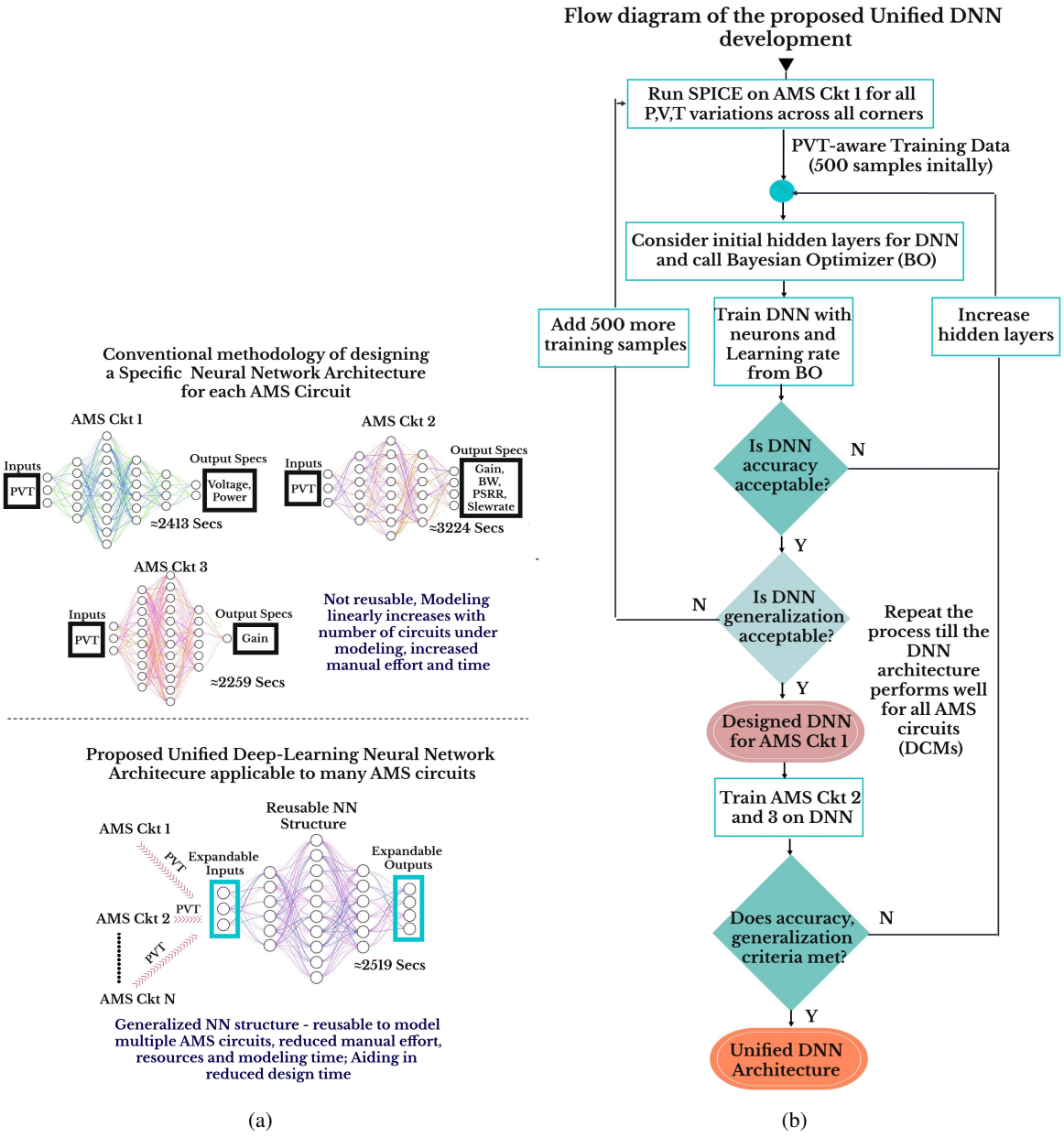


Figure 4.35: (a) Comparison of the conventional neural network designing with the Unified Deep-Learning Neural Network methodology (b) Workflow

cacy in modeling parametric variability, minimizing characterization costs while maintaining reliability. The designed U-DNN demonstrates remarkable performance even with a modest training dataset, typically starting with as few as 500 samples. Moreover, the U-DNN's seamless integration into existing frameworks ensures its broad applicability in analog circuit modeling without performance compromise.

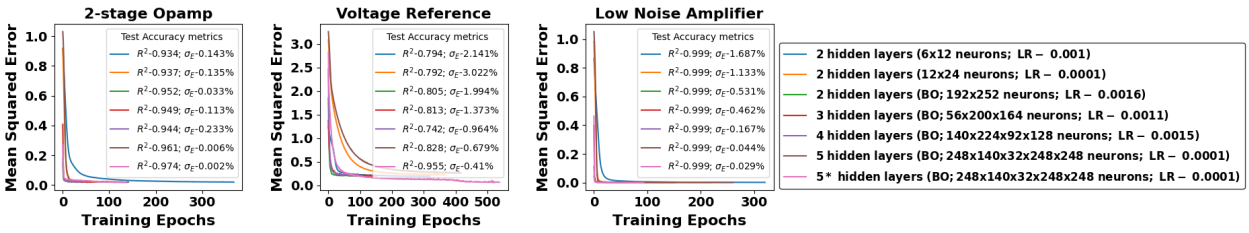


Figure 4.36: Training results of U-DNN model with different hidden layers

4.4.3.2 Unified Deep-Learning Neural Network Architecture Design

The traditional process of designing a NN typically involves stages such as preprocessing, model development, training, and validation. This cycle must be repeated for each AMS circuit. While these circuit-specific NNs can provide high accuracy for a particular circuit, they come at the expense of increased design time, resources, and expertise needed for their development and maintenance. This approach may not always be the most efficient or practical, especially in scenarios involving multiple circuits or evolving design requirements. This lack of reusability can lead to duplicated efforts in surrogate model development.

Our U-DNN architecture models multiple AMS circuits to address these challenges (Fig. 4.35(a)). This approach offers a more efficient, adaptable, and cost-effective method for surrogate modeling compared to designing separate NNs for each circuit. We specifically considered three AMS circuits (Design Circuits under Modeling (DCM)s), namely a two-stage Opamp [96], a voltage regulator [97], and a low noise amplifier [92], as the basis for designing our U-DNN (Table 4.20). These circuits operate in different regions and across two distinct CMOS technology nodes (180nm and 65nm), with different output specifications and unique relationships with PVT. Our objective is to create a deep neural network architecture (including the number of hidden layers, neuron count in each layer, learning rate, suitable optimizer, etc.) capable of effectively training these diverse variability-aware circuits, resulting in a generalized architecture capable of modeling a wide range of AMS circuits. The aim is to balance precision and generalization while utilizing minimal training data.

The U-DNN approach, as illustrated in Fig. 4.35(b), begins by creating PVT-aware statistical datasets for AMS circuits operating in different corners, each initially containing 500 training samples, 90% employed for training (within that 10% allocated for validation) and 10% reserved for testing. Hyper-parameter search starts with a NN structure featuring two hidden layers. However, manual trials reveal a lack of generalization across all configurations, as depicted in Fig. 4.36 (with neuron counts of 6x12 and 12x24), prompting the exploration of a novel architecture through optimization algorithms. ReLU and ADAM optimizers were employed in U-DNN while training the DCMs, with MSE for validation data and R^2 and $\%\sigma_E$ for test data. The best performance is observed with five hidden layers, with a learning rate of 0.0001. Training data consists of 500 samples in all the experiments except for * marked case, which includes 2000 samples. We employed BO to efficiently identify globally optimal

Table 4.20: The details of AMS circuits considered for U-DNN design and testing and U-DNN training results after modeling each circuit

AMS Circuit Details (PVT analysis at all corners (SS, TT, FF, SF, FS) across a wide temperature range $-40°C$ to $125°C$)					**U-DNN Training Results**			
Circuit	**Tech. Node (nm)**	**Supply Range (V)**	**Training Samples**	**Output Specifications**	$\mathbf{R^2}$	$\sigma_{\mathbf{E}}$	%MaPE	**U-DNN Training time (Secs)**
AMS Circuits for U-DNN Architecture Design (DMCs)								
Two Stage Opamp [96]	180	1 - 1.4	500	Gain(dB), Phase margin(°), GBW(Hz)	0.974	0.006	0.235	33.56
Voltage Reference* [97]	180	1 - 1.2	2000	Reference voltage (mV), Power(nW)	0.914	0.448	0.482	104.51
Low Noise Amplifier (LNA)	65	1 - 1.8	500	Gain(dB)	0.999	0.448	0.173	38.48
AMS Test Circuits for Validating U-DNN Architecture (DUTs)								
Mixer [93]	65	1.62 - 1.98	500	Gain(dB), Noise figure(dB), Compression point(dBm), IP3(dBm)	0.998	0.287	0.072	79.52
Current Reference*(CR)	180	0.8 - 2	2000	Ref. Current(nA), Power(nW), Temperature Coefficient($ppm/°C$)	0.954	0.381	1.248	96.36
Band-gap Reference*(BGR)* [91]	28	0.8 - 0.99	4000	Reference Voltage (V)	0.952	0.986	0.575	122.19
High speed LNA*(HSLNA)	28	0.81 - 0.99	500	Gain(dB), Noise figure(dB)	0.999	0.067	0.015	27.42
Tx Driver (TxDr) [90]	28	0.9 - 1.1	500	nMOS Res.(Ω) , pMOS Res.(Ω)	0.999	0.377	0.982	78.42

The circuits marked with * operate in sub-threshold saturation region and the rest in saturation region.

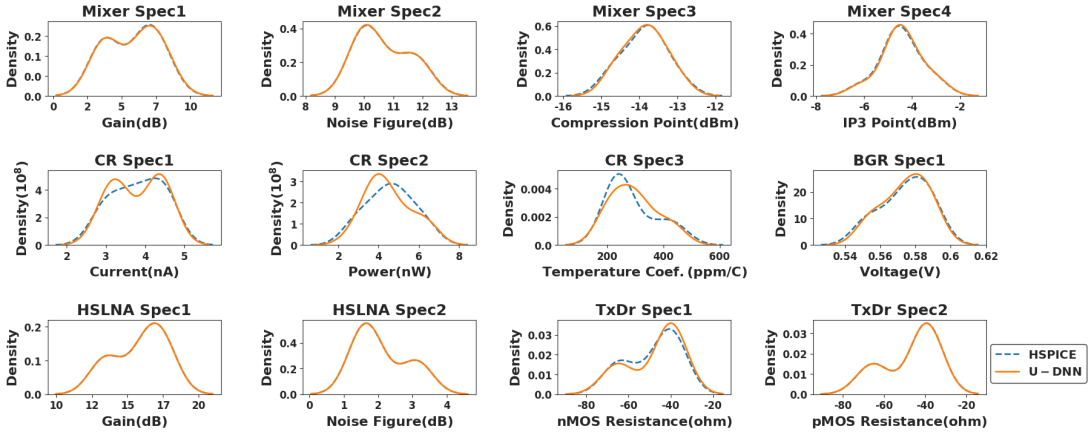


Figure 4.37: Trained U-DNN model predictions for the DUTs in comparison with SPICE values

hyperparameters, particularly in complex and noisy data scenarios [78]. BO was executed on each DCM for parameter search, ranging from 32 to 256 neurons per hidden layer and learning rates from 0.0001 to 0.1 with batch sizes of 32 and 64. Utilizing BO and varying hidden layers and training data size, we devised the U-DNN architecture with five hidden layers (248x140x32x248x248 neurons) and a learning rate of 0.0001. This network takes PVT as inputs and produces customizable outputs based on circuit requirements.

4.4.3.3 Results and Discussion

The U-DNN architecture aims to achieve adaptability to circuit variations, conserve resources, enhance generalization, and streamline model management and integration, leading to significant time savings in the overall design process. Experimental validation was conducted on five Design Under Test (DUT)s, including test circuits from the 28nm technology node that were not part of the training process.

The U-DNN demonstrated effectiveness in accurately modeling highly variable process-aware circuits at the CMOS 28nm node, showcasing its versatility and robustness. Evaluation metrics, including R^2, MSE, MaPE [12], along with $\%\mu_E$ and $\%\sigma_E$ (the percentage error between the mean and standard deviation of the predicted circuit performances and SPICE values) were used to assess performance. The finalized U-DNN architecture meets validation criteria set as $R^2 > 0.95$, $\%\mu_E$ $\%\sigma_E$, MaPE below 1%, confirming its effectiveness and suitability for our application. Training results and ML metrics are summarized in Table 4.20. On average, U-DNN's training time is 50 seconds for 500 simulation samples and 100 seconds for above 2000 samples.

The U-DNN's prediction results of all circuits' output specifications, for 20 unseen test cases after verifying the training and testing accuracy as per the set test criteria were compared against SPICE simulations and plotted in Fig. 4.37. Notably, all the DUTs report highly accurate training and validation performance, with a MaPE of less than $\leq 1\%$. Nonetheless, the current reference circuit reports the most significant discrepancy, reaching 3% in the prediction of the temperature coefficient. On average, across all of its output specifications, the MaPE results in 1.2%. The network architecture facilitates rapid convergence in the optimization process, particularly in instances of discrepancies, serving as an advantageous starting point for adjustments.

We employed Tensorflow and Keras in Python 3.9 to implement the U-DNN [98]. All the experiments were conducted on an i5 core CPU with 8GB of RAM.

The U-DNN architecture, developed through extensive exploration of circuit behavior under PVT variations across three distinct DCMs, offers versatility in effectively training a wide range of AMS circuits. This results in significant savings in computational time and resources. Comparison with conventional approaches reveals that while it takes 7,896 seconds using BO to choose deep NN architecture for three DCMs (Fig. 4.35(a)), our U-DNN architecture represents a one-time design effort applicable across diverse applications, streamlining the modeling process and reducing computational overhead. Moreover, we made efforts to minimize SPICE characterization costs. While a single SPICE simulation takes roughly 3 seconds, generating a set of 500 samples requires 1500 seconds. In contrast, leveraging the U-DNN approach entails approximately 2500 seconds for U-DNN architecture modeling (a one-time effort) and an average training time of 50-100 seconds for each circuit. Predictions for test inputs can be obtained in less than 5-10 seconds, showcasing the time-saving advantages of employing AI/ML surrogate modeling in various scenarios. In traditional analog circuit characterization, this cost escalates proportionally with the augmentation of MC simulations to meet design specifications.

4.5 Conclusion

This chapter outlines the training framework, procedures, and experimental results of PVT-aware digital and analog circuits. Regression algorithms achieve R^2 values of up to 0.999, indicating approximately 100% accuracy in training digital standard cells across multiple technology nodes of CMOS and FinFET technologies. AMS circuit characterization across various designs demonstrates consistent per-

formance, with an R^2 exceeding 0.95 and $\mu_E, \sigma_E < 1\%$ for the test split data, validating the accuracy of the trained regression models. Similarly, classifier algorithms such as SVM, KNN and LGBM achieve 100% accuracy in predicting the operating corners of BGR, TxDr and PLL for given PVT conditions.

Remarkably, the knowledge encapsulated in the U-DNN architecture, acquired from modeling diverse AMS circuits in CMOS 180nm and 65nm technologies, translates into accurate modeling of AMS circuits in CMOS 28nm. Our proposed architecture serves as a versatile platform for modeling various AMS circuits for different applications, significantly reducing the extensive design exploration time typically required to select appropriate NN structures.

Chapter 5

Sensitivity Analysis and Dominant Parameter Extraction

Designing advanced, complex circuits and systems entails the concurrent measurement and estimation of a multitude of parameters using fundamental circuit equations, a task that is known for its high level of complexity. Sensitivity Analysis (SA) is a valuable tool for discerning which parameters exert the most substantial influence on circuit performance. ML is gaining traction as a valuable tool for modeling and optimizing intricate digital and analog circuits. In high-dimensional complex systems, where conventional parameter estimation methods face challenges like computational complexity, uncertainty, or accuracy, ML techniques prove especially beneficial. Another approach involves modeling complex systems using continuous-time and spatial partial differential equations to capture the underlying spatio-temporal dynamics. However, this method becomes increasingly time-consuming and intricate with higher design complexity. Integrating machine learning in circuit sensitivity analysis enables engineers to gain deeper insights into the impact of parameters, streamline the design process, and achieve more robust and optimized circuit designs in a shorter time.

Dominant Component Analysis (DCA) is a ML technique for sensitivity analysis, dimensionality reduction, and feature extraction. It is particularly valuable when dealing with high-dimensional data, helping to identify and characterize the most significant components or features within a dataset. DCA specifically focuses on identifying and extracting the dominant components, making it especially valuable for applications where understanding the most influential factors is crucial. DCA assists in identifying the primary features that impact the circuit output specifications amidst shifting requirements or variations in PVT conditions.

5.1 Sensitivity Analysis using Feature Selection

Feature selection seeks to extract a concise subset of pertinent features from the initial set, eliminating irrelevant, duplicative, or noisy attributes based on feature redundancy and relevance. This process typically results in reduced over-fitting, improved learning performance, enhanced accuracy, reduced computational overhead, and greater model interpretability [99]. Irrelevant features lack a statistical relationship with other attributes and don't contribute to accuracy. Redundant features, while weakly relevant, can be replaced without affecting the target distribution. Strongly relevant features are crucial

for an optimal feature subset and cannot be removed without impacting the prediction accuracy. Relevance and redundancy are assessed differently: relevance is determined for individual features, while redundancy is evaluated in multivariate cases.

Further, a regression model that is accurately specified with the most effective set of features provides unbiased regression coefficients and precise predictions of the response variable. Conversely, an over-specified regression equation that includes the entire feature set can result in problems like inflated standard errors for the regression coefficients and increased computation time [100]. Moreover, the issue of multicollinearity arises when two or more independent variables in a regression are highly correlated, meaning they do not offer distinct or independent information to the regression analysis.

Feature selection strategies in supervised learning fall into three main categories: filter-based, wrapper-based, and embedded approaches.

- *Filter-Based Approach:* This method evaluates features based on their intrinsic attributes like correlation or statistical properties. It doesn't consider a specific learning algorithm's performance. It is computationally efficient and can be used with various learning algorithms. Examples include Information Gain (IG), Pearson coefficient, chi-square test, Fisher's score, and missing value ratio. Information gain and person coefficient are best suited for regression and the rest for classification.

- *Wrapper-Based Approach:* This approach involves methods like forward selection, backward selection, and recursive feature elimination, which are sequential processes. They assess the impact of each feature on model performance and iteratively add or eliminate features based on their contribution.

- *Embedded Approach:* This approach integrates feature selection with the learning algorithm. It includes techniques like regularization, which adds a penalty term to the loss function to prevent overfitting. Regularization methods like Lasso (L1) and Ridge (L2) can be combined with feature selection to emphasize the most relevant features.

Additionally, using ML algorithms such as RF, NNs with grid/random search, or optimization wrappers is an effective technique under the embedded category. However, this process can be time-consuming and provides a specific solution for each modeling algorithm.

In the context of dominant parameter analysis and selection for regression algorithms, we provide generalized solutions to extract dominant parameters by analyzing the circuit simulation data using statistical measures like Pearson coefficient and IG.

5.1.1 Pearson Coefficient

The strength of the relationship between two quantitative variables is assessed using a correlation coefficient, commonly known as Pearson's correlation coefficient, denoted by r (equation 5.1) [12]. This is the most prevalent method for gauging a linear correlation. The correlation coefficient is measured

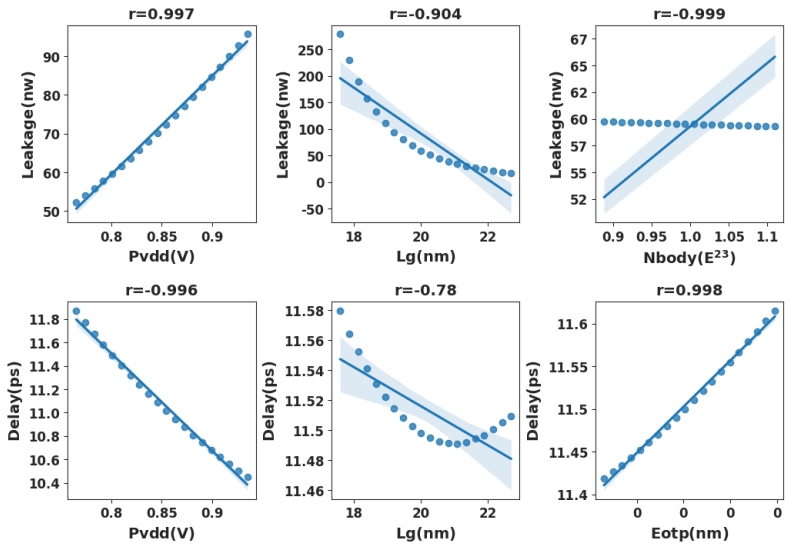


Figure 5.1: Correlation coefficients of FinFET full adder logic cell's leakage and delay

on a continuum ranging from +1 to 0 to –1. A perfect correlation between two variables is indicated by either +1 or -1. When one variable increases alongside the other, the correlation is positive; conversely, when one decreases while the other increases, it is negative. A complete absence of correlation is denoted by 0. r serves as a descriptive statistic that encapsulates the characteristics of a dataset and promptly assesses a significant relationship between two variables.

$$r = \frac{\sum_{i=1}^{n}(u_i - \bar{u})(v_i - \bar{v})}{\sqrt{\sum_{i=1}^{n}(u_i - \bar{u})^2 (v_i - \bar{v})^2}} \tag{5.1}$$

Here, n denotes the number of samples in the dataset, u_i and v_i represent individual sample points of two variables, and $\bar{u}$ and $\bar{v}$ denote the means of the sample variables, respectively. Fig 5.1illustrates a series of correlations within the FinFET leakage and delay of a full adder logic cell. The Pearson coefficient value, r is shown on top of each graph. A robust positive correlation is observed between supply and leakage, dielectric oxide thickness, and delay. Conversely, a negative correlation can be discerned between channel doping concentration and leakage. It's worth noting that both delay and leakage exhibit non-linear variations with respect to gate length. The Pearson coefficient of population, $r \approx 0$ for all the relationships, indicates a rejection of the null hypothesis, H_0. r in addition to r specifies the significance of a relationship w.r.t a confidence interval α, typically chosen as 0.05. If $r < \alpha$, then the relationship is significant; otherwise, it is not.

We employ the Pearson coefficient to assess the relationships between input variables and target specifications derived from simulated data of digital/analog circuits. Given that the data distributions are predominantly quantitative, typically adhering to a normal distribution and cleared of any outliers, r emerges as the most appropriate statistical gauge. It assists in pinpointing the primary parameters that impact the target specifications and screening out extraneous input data in a very quick time. Further-

more, it facilitates examining both linear and non-linear relationships among the variables. Depending on the level of non-linearity and the quantity of such variables, this information guides the selection of appropriate machine learning algorithms. The knowledge and understanding of dominant parameters aid in making imminent design trade-offs while building complex circuits from the standard cells. It also benefits design verification and its dependencies in scrutinizing the patterns in simulated data. Sensitivity analysis and dominant parameter extraction are feasible through statistical and visual data exploration methods.

5.1.2 Information Gain

Information gain/entropy is a metric used to gauge diversity among several methods. It quantifies the uncertainty in predicting the value of a target variable, serving as a measure of impurity. Information Gain, on the other hand, assesses the reduction in entropy (uncertainty) for a particular feature when the data is split based on that characteristic [101]. This metric is commonly employed in decision tree algorithms and offers valuable insights. A higher IG associated with a feature indicates its greater utility in the decision-making process. Less probable events convey more information, while more probable events convey less information. Entropy measures the information contained in a random variable's probability distribution. A distribution with a bias towards certain events has low entropy, whereas a distribution where events are equally likely exhibits higher entropy.

Consider a discrete random variable y with two potential outcomes. The binary entropy function, denoted as H and expressed in the base 2 logarithm (also known as the Shannon unit), is defined as follows:

$$H(y) = Plog_2(P) \tag{5.2}$$

The conditional entropy of two events, X and Y, given that X takes on the value x, is defined as:

$$H(y|x) = -\sum_{x\epsilon X}\sum_{y\epsilon Y} p_{xy}(x,y)log_2 p_y(y|x) \tag{5.3}$$

As the degree of impurity diminishes, a class distribution tends to become skewed. In scenarios where the class distribution is uniform, both entropy and misclassification error peak. The minimum entropy is attained when all samples belong to a single class.

Mutual information quantifies the information gained about one random variable when the value of another random variable is known. It is computed between two variables and indicates how much uncertainty is reduced in one variable given knowledge of the other variable. Mathematically, it can be defined as:

$$I(X;Y) = H(X) - H(X|Y) \tag{5.4}$$

It is always non-negative, with higher values indicating a stronger relationship between the variables. A value of zero implies independence. Mutual information is a versatile correlation measure showing

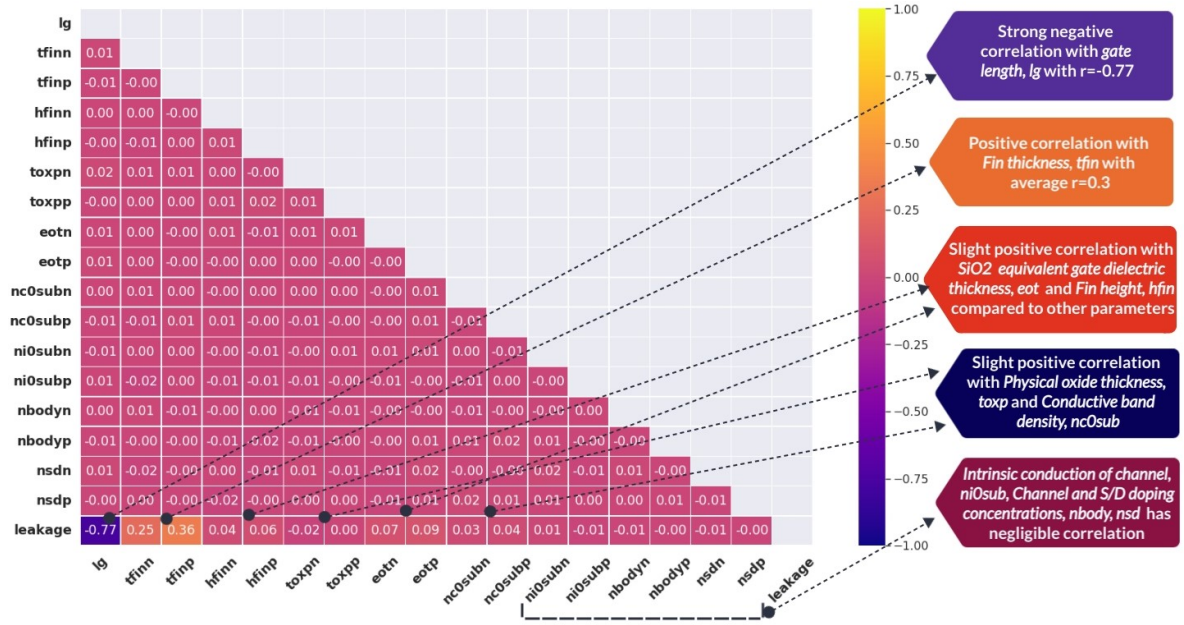


Figure 5.2: Analysis of process parameters' impact on leakage power in 16nm CMOS Full adder using heatmaps

the dependence between random variables. It is also applied in ML algorithms, such as Independent Component Analysis (ICA), which identifies statistically independent components in a dataset.

5.2 Sensitivity Analysis on Digital Circuits

We conducted a sensitivity analysis on statistical simulation data spanning various datasets across the technology nodes. This was accomplished through exploratory data analysis as outlined in [99]. The aim is to discern the influence of PVT parameters on leakage and delay in digital circuits. This approach provides rapid insights into the pivotal factors that shape the electrical behavior of the circuit under examination. It is also helpful in understanding the inter-relationships among the PVT distributions.

The key parameters were identified based on the values of Pearson's coefficient (r), which are visually represented on the heatmaps. Heatmaps illustrate the statistical dependence between two parameters, primarily a linear correlation. We exemplify this sensitivity analysis using the full adder dataset. However, the observations presented encompass all datasets (refer to Fig. 5.2 for the heatmap illustrating 16nm FinFET leakage, and Fig. 5.3 for the heatmap depicting 10nm FinFET delay). Additionally, the influence of various processes is annotated in these figures.

After scrutinizing the heatmaps (correlation coefficients) across all the targeted FinFET technology nodes, we have gathered the following insights:

1. The influence pattern of the process on leakage remains consistent from 16nm to 7nm FinFET technology nodes.

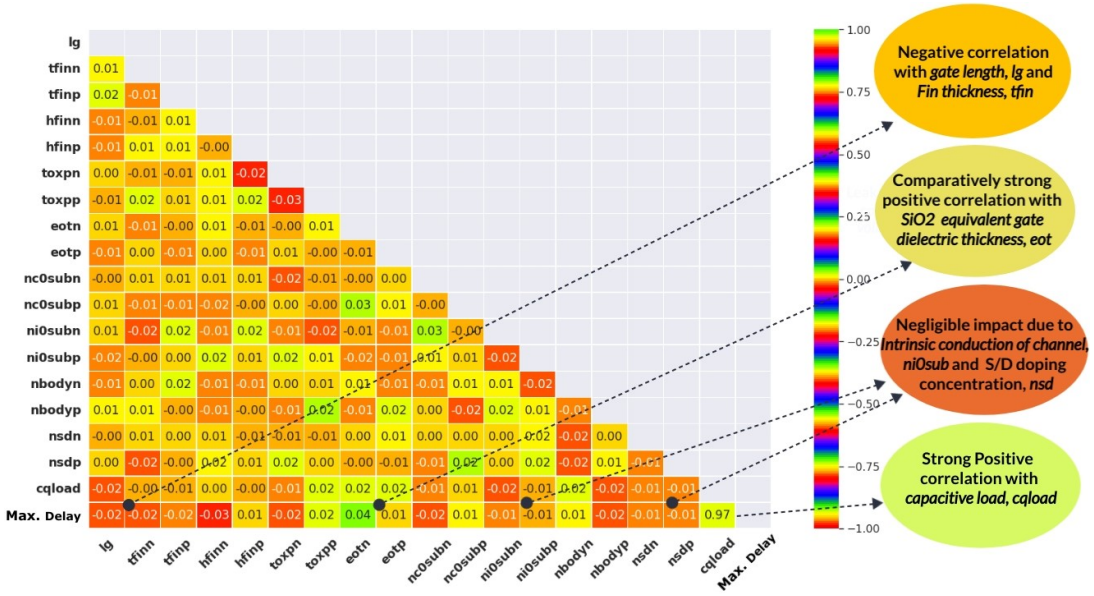


Figure 5.3: Analysis of process parameters' impact on propagation delay in 16nm CMOS Full adder using heatmaps

2. Gate length (*lg*) and Fin thickness (*tfin*) emerge as the most dominant factors in leakage estimation.

3. Negligible variations in leakage are observed due to changes in Intrinsic conduction of channel (*ni0sub*), Channel doping concentration (*nbody*), and Source/Drain doping concentration (*nsd*). Therefore, these factors are considered non-dominant and are omitted from the leakage modeling process.

4. The process influence pattern exhibits minor differences in delay as we transition from 16nm to 7nm technology nodes.

5. FinFET delay is primarily governed by capacitive load (*cqload*) and is further influenced by gate length (*lg*), Fin thickness (*tfin*), and fin height (*hfin*) across all technology nodes.

6. SiO2 equivalent gate dielectric thickness has a greater impact on delays at 10nm and 7nm nodes than the 16nm node.

7. The processes *ni0sub*, *nbody*, and *nsd* have a minimal impact on delay across all three nodes. As a result, they are deemed non-dominant and are excluded from the delay modeling process.

8. Operating temperature and supply voltage have fundamentally a strong influence on leakage and delay. Hence, they are considered the dominant parameters and are included for modeling.

A similar examination of CMOS process nodes led to the identification of *Toxref* and *Xj* as non-dominant parameters for both leakage and delay. Consequently, we excluded them from the DNN training (in

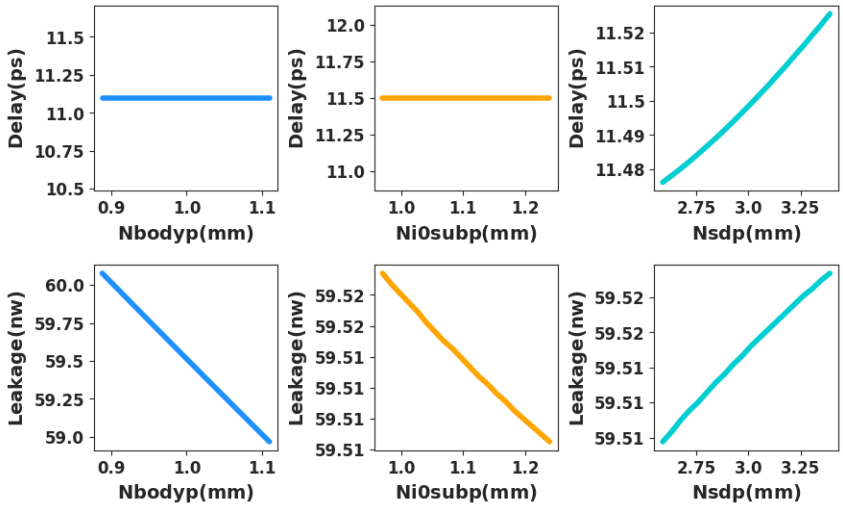


Figure 5.4: Impact of the non-dominant FinFET process parameters on circuit performances in 16nm Full adder standard cell

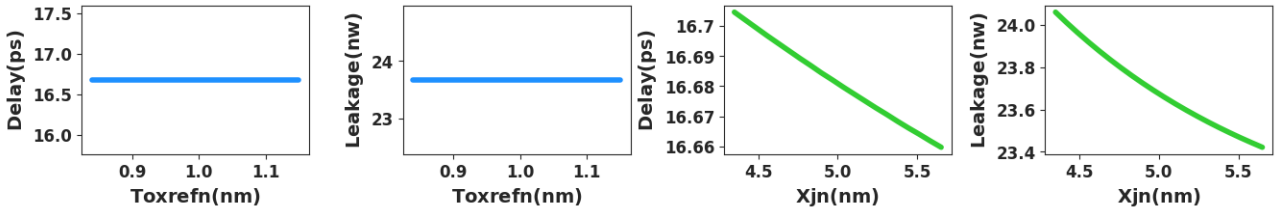


Figure 5.5: Impact of the non-dominant CMOS process parameters on circuit performances in 16nm Full adder standard cell

Chapter 6). To validate the accuracy of the proposed dominant parameter extraction, we conducted a thorough analysis involving SPICE MC simulations for each PVT scenario under consideration, and the impact of excluded non-dominant parameters is pictured in Figs 5.4, 5.5. The results of this analysis perfectly align, affirming the reliability and quality of our sensitivity analysis.

These findings align with the simulation-driven analysis detailed in chapter 3, where we examined the inter-relationships between PVT factors and their effects on leakage and delay. As a result, we affirm that employing statistical analysis through the Pearson coefficient followed by heatmap visualization offers a swift means to discern the influence of different input and design parameters on target specifications. This approach aims to circumvent the need for laborious and time-consuming simulated experiments.

Another crucial aspect to address in sensitivity analysis is the presence of multicollinearity among the parameters under examination. Multicollinearity arises when two or more input variables are highly correlated with each other, rendering them unable to offer independent information in the construction of a regression model [12]. Such variables can exert an undue influence on the regression model and

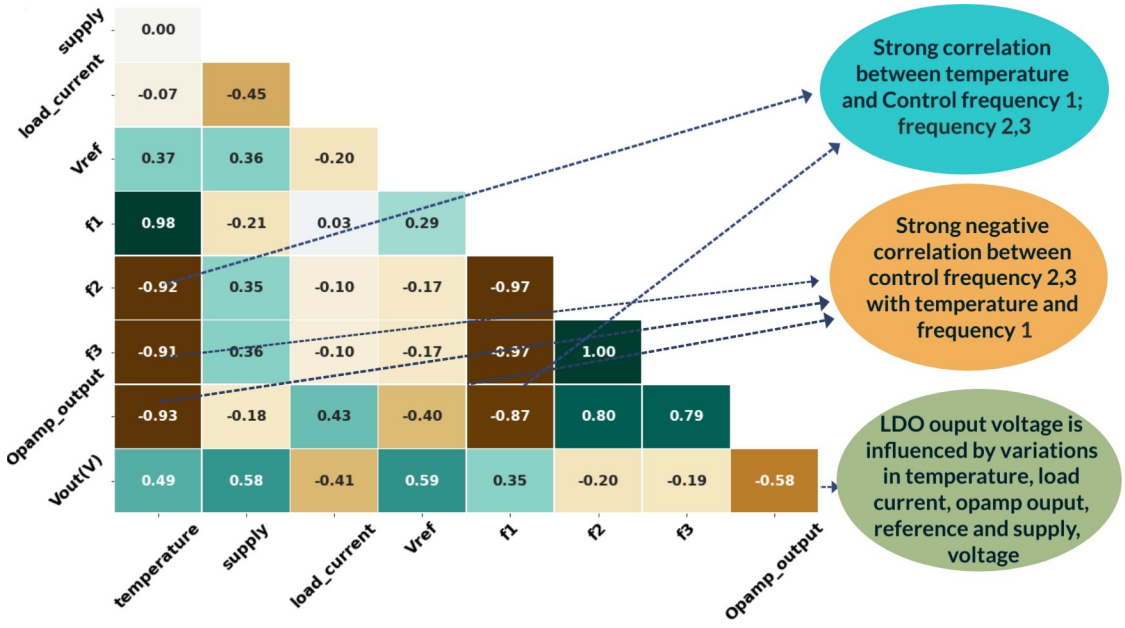


Figure 5.6: Correlation plot of a low dropout regulator

may result in inflated standard errors for the regression coefficients and extended computation time. After evaluating the Pearson coefficient values, we have confirmed no evidence of multicollinearity in the leakage/delay simulations across FinFET nodes from 16nm to 7nm and CMOS nodes from 45nm to 16nm.

5.3 Dominant Parameter Extraction in Analog/Mixed/RF Circuits

The idea is to delve into an emerging data-driven paradigm, wherein historical design data and simulation outcomes are utilized to train machine learning models. These models can subsequently pinpoint crucial design components, forecast circuit behaviors, and offer valuable insights for design adaptation. This paradigm shift not only accelerates the design process but also enhances the designer's ability to innovate by exploring unconventional design alternatives that might have been overlooked using traditional methods. An analog designer often has a crucial interest in comprehending how various internal electrical parameters affect the output of a circuit, which helps in making necessary design trade-offs. By employing statistical and visual data exploration methods on simulation results, it becomes feasible to extract the most influential parameters of an analog circuit and perform predictive analysis using machine learning techniques. We investigate the interrelationships among input and output parameters and their collinearity by calculating correlation coefficients (equ. 5.1).

In complex analog circuits with multiple degrees of freedom, employing machine learning inference on dominant parameters proves more effective than human design analysis. Based on these insights, the analog designer can proceed with the necessary design changes that address the identified dominant

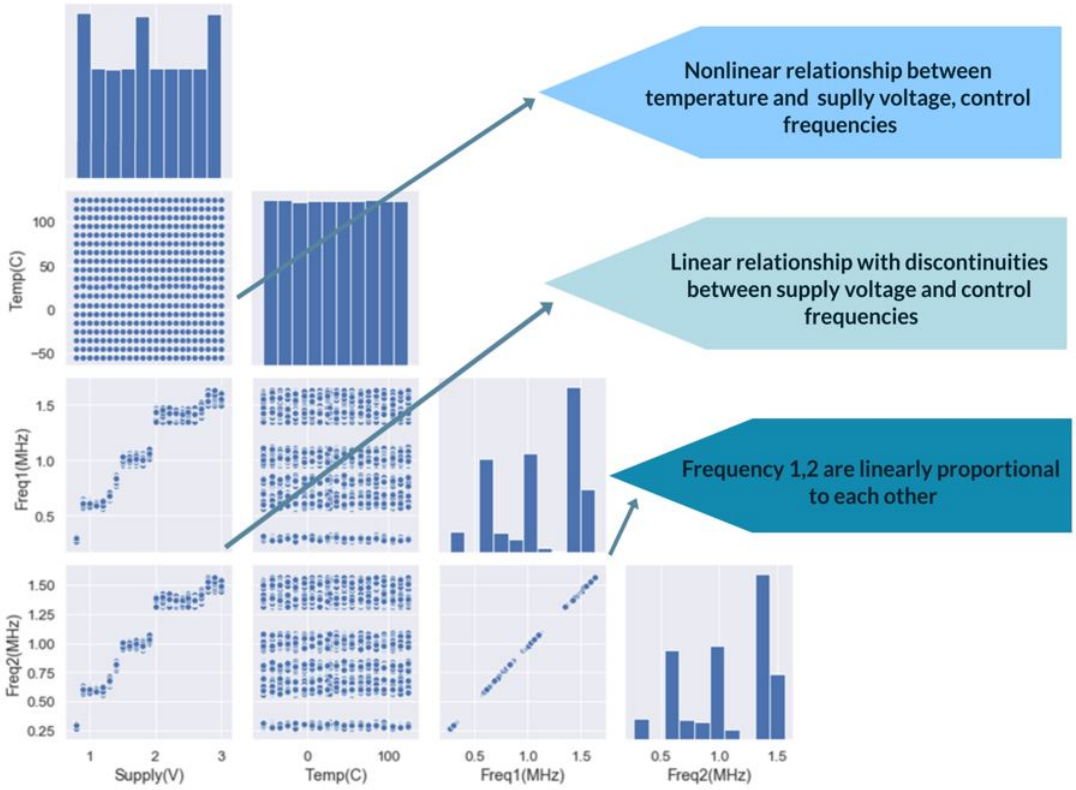


Figure 5.7: Pair plot of a current reference generator

parameter. For high-complexity circuits, dimensionality reduction techniques such as principal component analysis and auto-encoder [12] can reduce computational complexity. However, in our study, dimensionality reduction does not significantly impact the demonstrated circuits.

Visualizing the data through scatter plots, pair plots [11], and other techniques aids in understanding the patterns, complex structures, and relationships within the identified dominant parameters. Consequently, it facilitates the identification of suitable machine-learning algorithms to model such data.

The dominant parameter extraction using a heat map is illustrated by analyzing simulated data of an LDO [88]. The correlation matrix depicted in Fig. 5.6 highlights that variations in temperature and manufacturing process play a pivotal role in influencing the output voltage (Vout) of the LDO. Additionally, it reveals a notable negative linear correlation between the frequencies (f2, f3) of the process monitor (PMON) and temperature. The PMON frequencies are design representations for simulating manufacturing process variations in this context.

A useful method for visualizing the relationships among input parameters and their impact on output specifications is by employing a pair plot. Fig. 5.7 showing a pair plot of current reference design proposed in [87] serves as an illustrative example of this approach.

Furthermore, parameter selection holds paramount importance in analog circuit design, necessitating thorough analysis, simulation, and validation to ensure optimal operation under varying conditions.

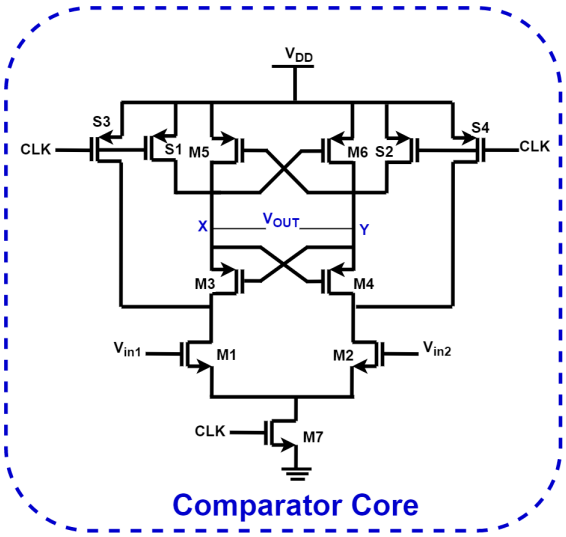


Figure 5.8: Strong arm latch based comparator [102]

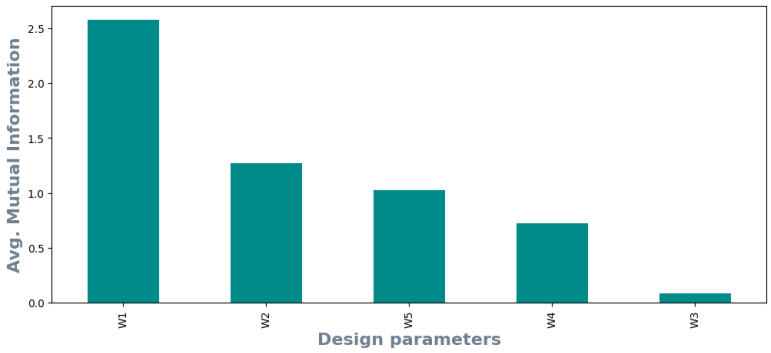


Figure 5.9: Dominant parameter extraction using Information gain

By meticulously choosing and optimizing these parameters, designers can attain desired specifications, meet performance targets, and enhance the overall reliability and stability of analog circuits. For instance, the selection of appropriate transistor sizes and biasing currents can directly influence amplifiers' gain, bandwidth, and linearity. Similarly, the choice of suitable resistor and capacitor values can determine the circuit's frequency response, filtering characteristics, and noise performance. Hence, to identify the most influential parameter(s) in an analog circuit, we also utilize IG. An illustration of extracting dominant parameters from a strong arm comparator (Fig. 5.8) is provided here.

The strong arm latch-based comparator assumes a crucial role in high-speed Analog-to-digital converter (ADC)s and other precision analog circuits, pivotal for achieving accurate and efficient conversion. The selection of transistor widths directly impacts the comparator's performance, denoted as W1 to W5. Among these, W1 significantly influences the input offset voltage, crucial for ensuring ADC accuracy. Meanwhile, W2 and W3 primarily affect the comparator's speed and delay, essential for applications requiring high-speed operation. The switches (S1 to S4) in the comparator must pull their

drain nodes to supply (VDD), ensuring proper functionality. W4's width affects the comparator's delay, while W5's width is vital for ensuring proper latch operation and reducing meta-stability, crucial for reliable digital output. Additionally, W5's width also affects the comparator's power consumption. The selection of dominant parameters is based on their effectiveness in impacting overall output parameters, including delay, offset, power consumption, and meta-stability. The mapping of the choice of W1 to W5 with respect to transistors M1 to M7 is as mentioned below.

W1 $\leftarrow$ M1, M2

W2 $\leftarrow$ M3, M4

W3 $\leftarrow$ M5, M6

W4 $\leftarrow$ S1, S2, S3, S4

W5 $\leftarrow$ M7

Based on our analysis utilizing IG, W1 stands out as a dominant parameter, as variations in its width notably impact all output specifications, scoring 2.509 compared to other design parameters (Fig. 5.9). Therefore, the dominant design parameter W1 can be utilized to adjust the output specifications of a strong arm latch-based comparator and further adapt to any variations in performance.

5.4 Conclusion

In summary, sensitivity analysis utilizing both Pearson coefficient and information gain emerges as a potent methodology for comprehensively assessing the impact of design and PVT variations on both digital and analog VLSI circuits. By employing the Pearson coefficient, we gain insights into the linear relationship between input variations and circuit performance, facilitating targeted adjustments to enhance robustness. Meanwhile, information gain provides a holistic view of the significance of each parameter variation, aiding in effectively prioritizing mitigation strategies. By combining these methodologies, designers attain a nuanced understanding of circuit performance across varied operating conditions. This knowledge empowers informed decision-making, fostering the development of robust VLSI designs capable of navigating PVT variations within the ever-evolving technological landscape.

Chapter 6

Circuit Performance Estimation Using Transfer-Learning Approach

A significant limitation of traditional ML algorithms lies in their demanding training data requirements, posing challenges in domains like VLSI due to resource limitations, privacy considerations, and the high cost of acquiring labeled data. To mitigate these challenges, we present a novel approach for surrogate modeling of digital VLSI circuits using DNNs with a focus on Transfer Learning (TL). TL enables the utilization of pre-trained models or knowledge from related tasks, effectively reducing the necessity for extensive labeled data. Our methodology delves into the interdependencies within CMOS/FinFET technologies across various nodes, facilitating precise estimations of PVT-aware circuit performance at advanced process nodes. By leveraging TL, we significantly alleviate the substantial data requirements typically associated with such modeling endeavors, thereby advancing the efficiency and feasibility of VLSI circuit design and optimization.

6.1 Transfer Learning and its Application for Circuit Analysis

Transfer learning is an optimization technique harnessing knowledge obtained in one domain to improve performance in a related domain. It proves especially beneficial when learning from limited datasets, utilizing insights from pre-trained models of similar nature. Unlike many ML approaches, which presuppose that training and test data share identical feature spaces and distributions, TL relaxes this constraint. Consequently, the model in the target domain no longer requires full training from the ground up, thereby mitigating the demand for extensive data. Moreover, TL yields a decrease in training duration.

A domain, denoted as D is characterized by $D = \{\chi, P(X)\}$, where χ represents the space of all possible feature vectors, and $P(X)$ stands for the marginal probability distribution. X constitutes a learning sample with n features denoted as $X = \{x_1, x_2, ..., x_n\} \epsilon \chi$, where x_i signifies the i_{th} feature vector. Features serve as distinctive attributes upon which the output relies, and the feature space encompasses the potential values of a feature within a given dataset. In a specified domain D, a task T is defined as $T = \{Y, f(.)\}$, where Y denotes the label space, and $f(.)$ represents a predictive function derived from pairs of feature vectors and labels $\{x_i, y_i\}$, with $x_i \epsilon$ X, $y_i \epsilon Y$. With these domain and task definitions, transfer learning is formally articulated below.

Given a source domain, D_S, along with its corresponding task, T_S, as well as a target domain, D_T, and a target task, T_T, the primary aim of transfer learning is to facilitate the acquisition of the conditional probability distribution $P(Y_T|X_T)$ within D_T, utilizing insights obtained from D_S and T_S, even in cases where $D_S \neq D_T$ or $T_S \neq T_T$ [103, 104]. According to the TL definition, $D_S = \{\chi_S, P(X_S)\}$ and $D_T = \{\chi_T, P(X_T)\}$. The condition $D_S \neq D_T$ implies that $\chi_S \neq \chi_T$ and/or $P(X_S) \neq P(X_T)$. In most instances, the size of D_S greatly surpasses that of D_T, denoted as $N_S >> N_T$.

6.1.1 Transfer Learning for Digital Circuit Analysis

Transfer Learning emerges as a promising solution to face the data scarcity challenge of VLSI circuits at times of resource limitations. Our modeling paradigm enables the statistical characterization of digital standard cells at a specific technology node (the source node) using DNNs. This knowledge can then be efficiently transferred to other nodes utilizing the Transfer learning concept, incurring minimal computational costs. Standard cell characterization becomes a one-time endeavor at the source technology node, backed by ample simulation data. Our proposed transfer learning approach significantly diminishes the need for simulation data for standard cell characterization at related nodes, while still ensuring accurate performance estimations. Furthermore, the modeling framework facilitates the estimation of complex circuit performances without the necessity to individually model each circuit, thus drastically reducing the computational simulations typically required by traditional tools. In this study, we conduct a comprehensive analysis of transfer learning for leakage and delay estimation across various FinFET nodes (16nm, 10nm, and 7nm) and demonstrate its applicability to CMOS nodes (45nm, 32nm, 22nm, and 16nm). It is worth noting that our methodology is a versatile technique adaptable to a wide range of nodes. The TL approach is analogous to the framework proposed in Chapter 4, with a methodology to drastically reduce the data requirements as we move across the technology nodes.

6.2 Correlation in PVT-aware Behavior of Circuits across the Technology Nodes

Conducting a comprehensive sensitivity analysis is crucial in comprehending the impact of PVT variations on circuit performance across diverse technology nodes, while also revealing the inherent correlations among them. The efficacy of transferred learning models hinges upon the meticulous selection of input variables, particularly the PVT parameters. Hence, we identify the dominant and correlated process parameters across various technology nodes within a specific technology, such as FinFET or CMOS. This endeavor aims to bolster modeling accuracy and diminish the high-dimensional variation space, thereby curtailing computational time, notably training time. Furthermore, it furnishes valuable insights into the pivotal factors influencing the electrical characteristics of the examined circuit. Understanding these dominant parameters is instrumental in making informed design trade-offs when constructing complex circuits from standard cells. Moreover, it facilitates design verification and its associated dependencies by scrutinizing patterns in simulated data.

Table 6.1: Comparison of model performance with the dominant process

Standard cell	Parameter	All process		Dominant process	
		R^2	σ_E	R^2	σ_E
2 input And	Leakage	0.992	0.169	0.990	0.561
	Delay	0.997	0.807	0.999	0.448
2x1 Multiplexer	Leakage	0.989	0.991	0.990	0.970
	Delay	0.995	0.331	0.998	0.979
Full adder	Leakage	0.998	0.982	0.992	0.512
	Delay	0.997	0.200	0.994	0.224

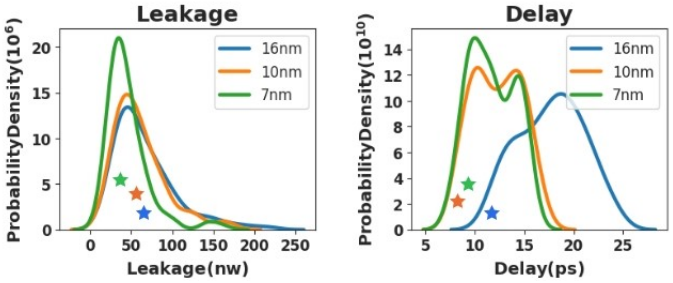


Figure 6.1: Correlation in statistical leakage and delay distributions due to PVT variations across different FinFET technology nodes

We conducted an extensive data exploration and statistical analysis to investigate the relationship between process variations and leakage as well as delay in digital standard cells. Correlation coefficients between process variations and these parameters were estimated, with dominant parameters identified based on Pearson's coefficient values (r) and visually depicted using heatmaps. Detailed analysis is provided in Chapter 5, specifically in sections 5.1.1 and 5.2. The examination revealed negligible variations in FinFET leakage and delay attributed to variations in Intrinsic conduction of channel (*Ni0sub*), Channel doping concentration (*Nbody*), and Source/Drain doping concentration (*Nsd*). Similarly, an assessment of CMOS process nodes led to the identification of gate oxide thickness (*Toxref*) and junction depth (*Xj*) as non-dominant parameters for both leakage and delay. Consequently, these factors were deemed non-dominant and excluded from the modeling process. Subsequent investigation delved into the ramifications of neglecting non-dominant processes during leakage and delay modeling. The results, summarized in Table 6.1 for a sample of FinFET standard cells at the 16nm node, demonstrate a negligible impact on accuracy when non-dominant processes are excluded from modeling. This observation remains consistent across various technology nodes.

We also illustrate the influence of PVT variations across different technology nodes in Fig. 6.1. Here, * indicates the performance at nominal PVT. This visualization unveils correlations that facilitate generalized learning and knowledge transfer, thereby enhancing training efficiency and reducing the necessity for extensive training data. Analyzing the variations from 16nm to 10nm to 7nm, we observe overlapping trends in leakage and power variability (Fig. 6.1), which offer valuable insights into performance attributes and characteristics, consequently enhancing prediction accuracy. This indicates that such insights can supplement data for upcoming nodes (e.g., 5nm and 3nm) even in the absence of labeled performance data. Thus, we can gradually approach the problem as a zero-shot regression learning task [105], employing a method that infers parameters for unobserved target nodes.

6.3 Transfer Learning Framework for Circuit Performance Estimation

In response to the challenge of devising a swift and precise surrogate modeling framework with statistical awareness for estimating circuit performance under the influence of PVT variations, we introduce our Transfer Learning-enabled learning paradigm. The core aim of this proposed modeling approach is to alleviate the computational load, reduce data prerequisites, and augment overall computational efficiency. Our modeling paradigm is outlined in Fig 6.2. It involves three key steps:

1. Conduct a dominant parameter analysis on the datasets to identify the most influential and correlated PVT factors affecting both leakage and delay across all technology nodes (as demonstrated in section 6.2)

2. Accumulate a sufficient amount of simulation data to create precise PVT-aware models for each standard cell at the source node and subsequently train DNNs.

3. Through transfer modeling, we conserve the observed PVT variant behavior in the cell at the source node in the trained DNN structure and extend its application to another node (target node) with a limited amount of simulation data.

We employ a hybrid framework that combines a comprehensive sensitivity analysis across different nodes, the utilization of BO to construct DNNs for precise modeling of standard cells in the source domain, and the effective transfer of knowledge to subsequent target nodes through meticulous fine-tuning.

6.3.1 Dense Neural Networks for Statistical Circuit Performance Estimation

In this study, we leverage DNNs to construct efficient surrogate models for the prediction of leakage and delay characteristics in standard cells, a crucial aspect in VLSI design. The architecture of these neural networks is defined by fully connected layers, wherein connections are established between every node in one layer and every other node in the subsequent layer. Our approach begins with a customized

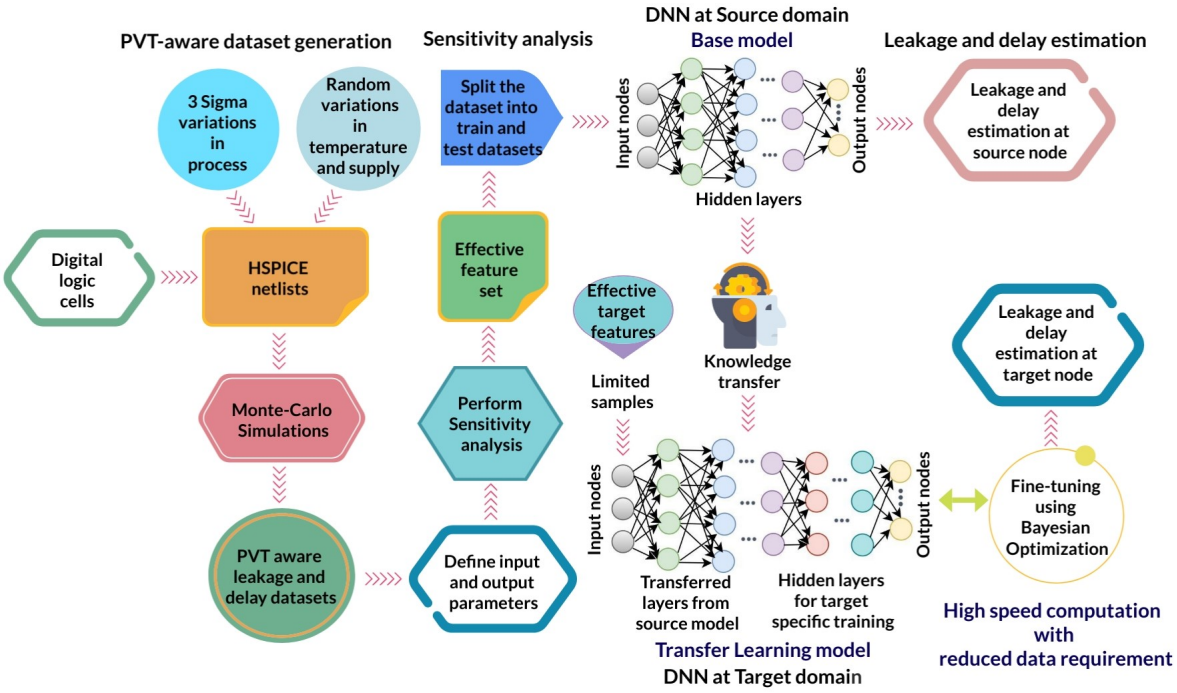


Figure 6.2: Transfer Learning modeling paradigm for the statistical digital circuit performance estimation

Table 6.2: Comparison of digital cell modeling and performance metrics with the State-of-the-art works

	[79]	[37]	[106]	[107]	[108]	[39]	[44]	**TL model**
Technology node (nm)	16	16	NS	45 - 7	22	45 - 14	180 - 7	**45 - 7**
Transistor type	CMOS	FinFET	CMOS	CMOS/ FinFET	CMOS	CMOS/ FinFET	CMOS/ FinFET	**CMOS/ FinFET**
Performance measure	Leakage & total power	Leakage	Power	Leakage & delay	Leakage & delay	Delay & slew time	Area, delay, power & energy	**Leakage & delay**
Sensitivity analysis (dimensionality reduction)	Yes	NA	NA	NA	NA	NA	NA	**Yes**
Sample count	50K	NS	NS	15K	50K	NS	NS	**≤1.5K (TL model)**
Temperature variations (°C)	NA	-55–125	NA	-55–125	-55–125	NA	NA	**-55–125**
Voltage variations (V)	NA	5%	NA	±10%	±10%	NA	NA	±10%
Process variations	Yes	Yes	NA	Yes	Yes	Yes	NA	**Yes**
Modeling algorithm	LR, PR	PR, ANN	RF	GBM	Residual Neural Networks (ResNN)	BI	PR	**DNN (TL)**
Accuracy metric	R^2 >0.99	R^2 >0.99	R^2 >0.99	R^2 >0.99	R^2 >0.99	Avg. error 4.3%	R^2 >0.95	R^2 **>0.99**
Speed improvement	NS	NS	NS	10000x	NA	15x	NS	**10000x**
Knowledge transferability to other technology nodes	NA	NA	NA	NA	NA	Yes	Yes	**Yes**

*NS - not specified; NA - not applicable

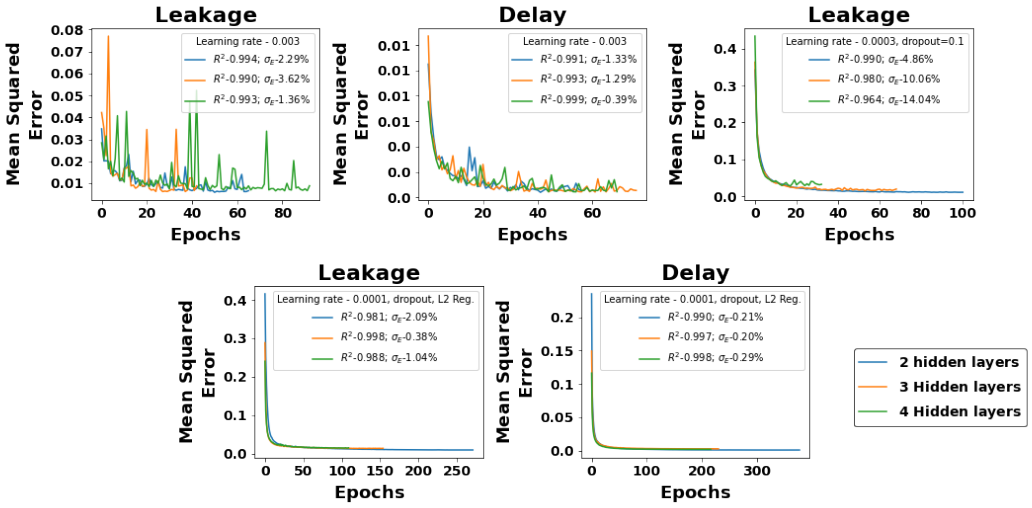


Figure 6.3: Performance of DNN model with different hidden layers and ADAM optimizer w.r.t SPICE simulations of 16nm FinFET Full adder

initial hidden layer tailored to accommodate the input feature size, followed by a sequence of appropriately sized hidden layers. The final layer incorporates an activation function that facilitates the mapping of continuous output. Due to their layered architecture, DNNs proficiently encapsulate hierarchical representations of features at various levels, thereby serving as a potent tool for capturing complex relationships within datasets. While widely recognized for their efficacy in classification tasks, DNNs also demonstrate exceptional performance in modeling continuous data, a result of deliberate design and experimentation [109]. We apply the DNNs to model the PVT-aware performance of digital circuits, offering insights into their robustness and efficiency under varying PVT conditions.

In our work, the goal of the DNN with L layers is to minimize the loss function $J_R(\theta)$ w.r.t the parameters, θ over a set of network inputs $D = \{(x_1, y_i), ..., (x_n, y_n)\}$, where $y_1, ..., y_n$ are labels in the dataset and θ represents the parameter set $\{\theta_1, ..., \theta_L\}$. The output of the DNN for a generic input x is $F(x; \theta)$. Here, we chose $L2norm$ (Mean squared error) as the loss function.

$$J_R(\theta) = \frac{1}{2}||y - F(x; \theta)||^2 \tag{6.1}$$

Extensive experimentation was conducted on various configurations of DNNs modeling the digital standard cells. Factors such as hidden layer architectures, activation functions, and fine-tuning hyperparameters were systematically varied to optimize both leakage and delay estimations while accommodating diverse PVT conditions. To attain globally optimized hyperparameters suitable for a wide array of standard cells, BO was employed. Hyperparameter tuning, a critical aspect of network optimization, involves searching for the most effective settings defining the network architecture, including parameters

such as learning rate, epochs, and optimizer selection. The training process of the DNN with different hidden layer configurations is shown in Fig. 6.3. MSE for validation data and R^2 and $\%\sigma_E$ for test data are employed for evaluation. Results indicate that a three-hidden-layer DNN, utilizing a learning rate of E^{-4}, incorporating $L2$ regularization and dropout layers, yielded optimal performance for both leakage and delay estimation tasks. The Rectified Linear Unit (*Relu*) activation function coupled with the *ADAM* optimizer was utilized. During the training process, BO suggested the optimal number of neurons within each layer (ranging from 32 to 100 neurons), serving as an outer optimizer, while the *ADAM* optimizer managed the inner optimization within the DNN, adjusting weights and biases throughout the fitting process.

6.3.1.1 Bayesian Optimization

Bayesian optimization is one of the best approaches for globally optimizing black-box functions, f, denoted as f, which are often challenging to evaluate due to their computational expense. This method adeptly handles stochastic noise in function assessments and operates as a sequential optimization algorithm, iteratively seeking the optimal solution. The crux of the Bayesian optimization challenge resembles that of finding the global minimum (or maximum) of an unknown function, f.

$$x^* = \arg\min_{x \in A} f(x) \tag{6.2}$$

Where, A represents a d-dimensional simplex $\{x \epsilon \mathbb{R}^d : \sum_i x_i = 1\}$ (typically $d < 20$), $f(x)$ denotes the objective function ($L2norm$ in this case), and x^* is the set of hyper-parameters that yield lowest value of $f(x)$.

BO utilizes GPR, a Bayesian statistical technique, to construct a probability model of the objective function. This model, denoted as $P(f)$, is characterized by a Gaussian process prior with mean μ and covariance K, representing the underlying continuous function f. Mathematically, this prior belief can be expressed as:

$$P(f) = GP(f; \mu, K) \tag{6.3}$$

Where μ and K are the mean and covariance of the objective function, f. The model with this prior belief is sequentially modified using Bayesian posterior updating for a maximum number of iterations to produce an optimum solution [75]. Bayesian optimization chooses the optimal set of hyper-parameters as it considers prior optimization history to evaluate the objective function in each iteration, resulting in a quick and efficient solution.

6.3.2 Transfer Learning enabled Statistical Circuit Performance Estimation

Transfer learning emerges as a highly effective and efficient technique for enhancing circuit performance estimation across diverse technology nodes, particularly in cases where ample simulated data is scarce, costly, or time-intensive. In essence, TL extracts underlying features from abundant labeled

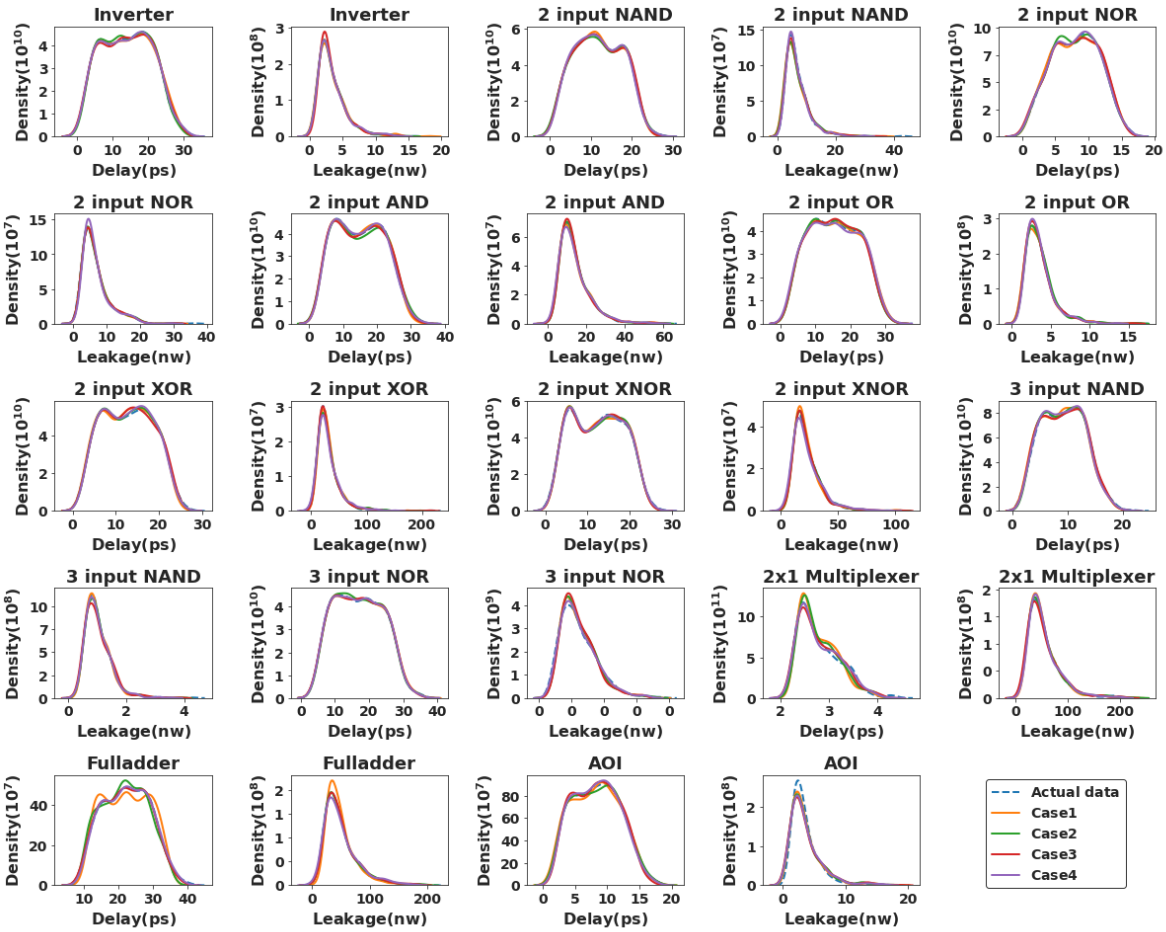


Figure 6.4: Performance of sequential transfer learned models from 16nm $\rightarrow$ 10nm $\rightarrow$ 7nm

training data in one task (such as leakage/delay at a specific node) to bolster the generalization performance in another related task (for instance, leakage/delay at the subsequent node). A transfer learning task, defined by $< D_S, D_T, T_S, T_T, f_t(.) >$, involves a deep learning network function $f_T(.)$ that embodies a non-linear function within a deep learning framework [110]. The approach entails training the base network in the source domain and transferring the network structure and parameters from selective initial layers (L) to the target network. Further layers (T) are then added to the target network, initialized, and trained for the specific target task. Any errors are subsequently back-propagated from the new task into the base model to fine-tune it for optimal performance in the new task. Our research focuses on implementing knowledge transfer from a higher technology node (source node) to a lower node (target node) via DNNs. This process involves incorporating selected top layers from the well-trained source DNN, which remain fixed, and appending additional bottom layer(s) to construct a new neural network trained on the target dataset, which typically contains a substantially smaller training sample. The chosen top layers of the pre-trained source DNN encapsulate critical transferable information concerning

PVT features relevant to the target node. We call this kind of modeling as Few Shot Learning (FSL) or sequential transfer learning. The critical steps involved in our modeling paradigm are organized within Algorithm 2.

Fine-tuning the transferred layers within the target DNN is crucial to ensure effective adjustment when necessary. Bayesian optimization is also employed to fine-tune the transferred network, with the entire process, including hyperparameter tuning and fine-tuning, being automated within the transfer learning framework. This entails the option to freeze either the upper or lower network layers or unfreeze the entire network for re-training if the specified test criteria (outlined in Section 6.5.1) are not satisfied. The evaluation process relies on comprehensive test criteria to determine the optimal DNN model that yields accurate leakage/delay predictions in the transferred or target nodes. The fine-tuning process consists of four phases, as outlined in Table 6.3. Phase 1 precedes any fine-tuning, while phases 2 to 4 involve fine-tuning using Bayesian optimization. In Phase 2, the transferred top layers remain frozen while only the bottom layers are fine-tuned to ascertain appropriate hyperparameters. Phase 3 maintains the frozen bottom layers from Phase 2 while fine-tuning only the top layers. Phase 4 involves fine-tuning all layers in the network using a very low learning rate. These fine-tuning phases are executed sequentially based on the accuracy achieved in each phase, corresponding to Cases 1 to 4 in Fig. 6.4. Here, Case 1 to Case 4 corresponds to the fine-tuning adapted via Bayesian optimization over the layers. (Case 1: Freezing top and bottom layers, Case 2: Fine-tuning bottom layer(s), Case 3: Fine-tuning top layer(s), Case 4: Fine-tuning all layers). The most effective model selected from these cases for each standard cell is chosen for performance evaluation or potential further transfer learning. The complete process of transfer learning and fine-tuning is automated using Python.

In summary, the proposed modeling paradigm unfolds in three distinct stages. The initial stage focuses on crafting proficient DNN models at the source node, exemplified by 16nm FinFET. The subsequent stage involves transferring knowledge from the source node to the subsequent node, such as 10nm, and subsequently evaluating its performance following automated hyperparameter tuning. The ultimate stage revolves around fine-tuning the transferred model.

6.4 Transfer Learning Framework for Performance Estimation at Future Nodes

Observing the statistical correlation in FinFET nodes due to PVT variations (as depicted in Fig. 6.1) suggests a promising avenue for extending this understanding to future nodes through a learning network, potentially circumventing the dependency on simulation data. While it is recognized that forthcoming nodes, such as 5nm and 3nm, may introduce additional influencing factors, the chosen PVT parameters are meticulously selected from classical leakage and delay transistor equations, anticipated to significantly impact upcoming nodes' performance. This modeling approach is referred to as Zero Shot Learning (ZSL).

Algorithm 2 Proposed TL-Modeling algorithm

Require: D_S and D_T Source and Target datasets

Dominant Parameter Analysis

$X \leftarrow \{x_1, ...x_k\} \forall$ PVT variations
$Y \leftarrow$ target
for $i \leftarrow 1, k$ **do**
  Compute ρ for $Y|x_i$
  if $\rho_i > \rho_{th}$ in D_S, D_T **then**
    $X_e \leftarrow \{x_i\}$ ▷ Effective feature set, X_e in D_S, D_T
  end if
end for

DNN Training at the Source node

Require: $X_e \leftarrow \{x_1, ...x_m\} \forall$ dominant PVT, Y
Define Search space for all hyper-parameters over $J_R(\theta)$
$f_s : X_e \leftarrow Y_S$ ▷ DNN model M_s to model $Y_S|D_S$
Split D_S as D^{train}, $D^{Validation}$, D^{test}
Call BO [111] $\forall \arg\min_{x \in X} f_s(x)$ over D^{train}, $D^{Validation}$
Extract p^* ▷ Optimal hyper-parameter set
Train M_s on D^{train} using p^* [112]
Compute ${E_s}^*|M_s$ on D^{test} ▷ Test error set
if $E_s > {E_{sc}}^*$ **then** ▷ Test error criteria
  Hyper-parameter tuning
  Re-train M_s
end if
Save M_s

Transfer Learning to the Target node

Require: $X_e \in D_T, Y_T, M_s$
$f_t : X_e \leftarrow Y_T$ ▷ Transferred model M_t to model $Y_T|D_T$
  $L \leftarrow$ frozen layers of M_s
  $T \leftarrow$ additional trainable layers of M_t
  $\alpha \leftarrow \alpha|M_s$ ▷ α - Case 1 learning rate
Split D_T as D^{train}, $D^{Validation}$, D^{test}
Train M_t on D^{train}
Compute ${E_s}^*|M_t$ on D^{test} ▷ Test error set
if $E_s > {E_{st}}^*$ **then**
  Case 2: Call BO, extract P^*, retrain M_t
  Case 3: Fine-tune top layers of M_t using BO
  Case 4: Fine-tune all layers of M_t using BO
end if
Save the best performing M_t

* - E_s, E_{sc} are a composite set of test errors, and its criteria - $R^2 > 0.95$, $\{\%\mu_E, \%\sigma_E\} \leq 1\%$ and MPE - Mean Percentage Error for 500 unseen test samples $\leq 2\%$. E_{st} is same as E_{sc} except for 1% additional tolerance in error(s) at the target node.

To validate ZSL, experiments were undertaken to forecast circuit performances at the 7nm node utilizing knowledge derived from DNN and transferred models at 16nm and 10nm, respectively. The trained DNN models of standard cells, utilized during knowledge transfer from 16nm to 10nm nodes as discussed in the preceding sections, are leveraged for predicting PVT-aware leakage and delay at a subsequent future node, specifically the 7nm node in this context, without the need for any training data at the 7nm node. The results of ZSL are demonstrated in section 6.5.2.5.

6.5 Results and Discussion

We scrutinize and deliberate on the outcomes of DNN training and TL concerning circuit performance estimation in this section. Throughout all experiments, the datasets were partitioned in an 80:20 ratio for training and testing, reserving 20% of the training data for validation purposes. We conducted experiments with varying sizes of simulation data to ascertain the optimal training set at the source node. was to identify a sufficient quantity of labeled data at the source node, as this significantly impacts the effectiveness of knowledge transfer to other nodes. Notably, through experimentation, we observed that the most favorable performance for delay estimation was achieved with 5,000 training samples and for leakage with 15,000 samples (refer to Fig. 6.5). Furthermore, we selected a subset comprising 10% of the training samples from the target node for the transfer learning process, validating its accuracy by comparing it with the baseline performance (see Fig. 6.5). The results are demonstrated with 2-input And probability density; 5K and 15K samples (Baseline models) show good performance for 16nm FinFET delay and leakage modeling; 500 and 1.5K samples with transfer learning show good agreement with baseline models (marked as 5K) at 10 and 7nm nodes. The best-performance set is $\{5K, 500\}$ for delay and $\{15K, 1.5K\}$ for leakage while modeling at source and transferred nodes, respectively. The necessity for a larger simulation dataset in leakage estimation can be attributed to the substantial deviation from its mean, primarily due to PVT variations (refer to Fig. 3.9). All the experiments were implemented in Python 3.9 utilizing the Keras and the Tensorflow deep learning library on a computer with an i5 core CPU and 8GB of RAM.

We constructed DNN models for all 12 standard cells (listed in Table 6.3) at the specified source node (say, 16nm in FinFET and 45nm in CMOS) to predict both leakage and delay, utilizing a minimum of 5000 samples for each cell. While achieving such precise modeling in alternative technology nodes is indeed possible, as evidenced by various existing surrogate methodologies, it often involves resource-intensive simulations and lengthy training durations. In contrast, our approach leverages TL to effectively train digital cells across diverse technology nodes.

6.5.1 Test Criteria for Performance Evaluation

For a comprehensive performance assessment, we have opted for a composite set of evaluation criteria. This includes a combination of R^2, $\%\mu_E$, $\%\sigma_E$, and MaPE for the testing data as discussed in Chapter 4. We utilize the MaPE for 500 unseen test samples (referred to as N). Once the model success-

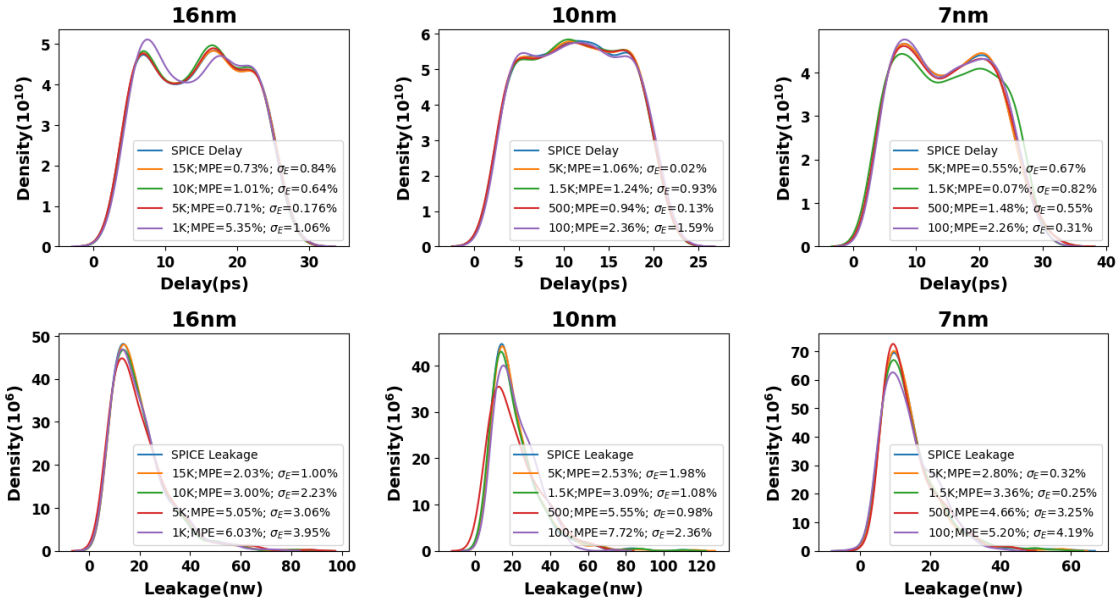


Figure 6.5: Performance of DNN model with different samples w.r.t SPICE leakage/delay

fully meets the predefined test criteria on the test split data w.r.t R^2, $\%\mu_E$, $\%\sigma_E$, it undergoes further testing in terms of MaPE for these 500 unseen test samples to confirm model's generalization capability.

By carefully selecting and tuning hyperparameters, we attained an error of less than 1% error in $\%\mu_E$, $\%\sigma_E$ and $< 2\%$ in MPE for estimations at the source node for all the 12 CMOS/FinFET standard cells. This high level of accuracy was also achieved at the target node through TL facilitated by BO. On average, $\%\mu_E$ and $\%\sigma_E$ were both less than 1% and MaPE was below 3% at the target node with only 10% of the samples compared to what was required at the source node. We showcase detailed results about FinFET cells and illustrate the applicability of our proposed modeling framework to CMOS technology.

6.5.2 Set of Experiments over Transfer Learning

We formulated five distinct experiments to evaluate the effectiveness of the proposed TL methodology for estimating statistically variable circuit performance across different nodes, as outlined below.

6.5.2.1 Single-node Transfer

With a focus on accommodating the broad need for variability analysis, we aim to transfer knowledge from higher to the consecutive lower/advanced technology nodes. The outcomes of experiments conducted on FinFET 16nm to 10nm standard cells, as detailed in Table 6.3, exhibit highly favorable accuracy in comparison to SPICE simulations. These results report an R^2 greater than 0.99 and an average mean percentage error of 2% for the unseen test samples under consideration. Fig 6.6 provides a

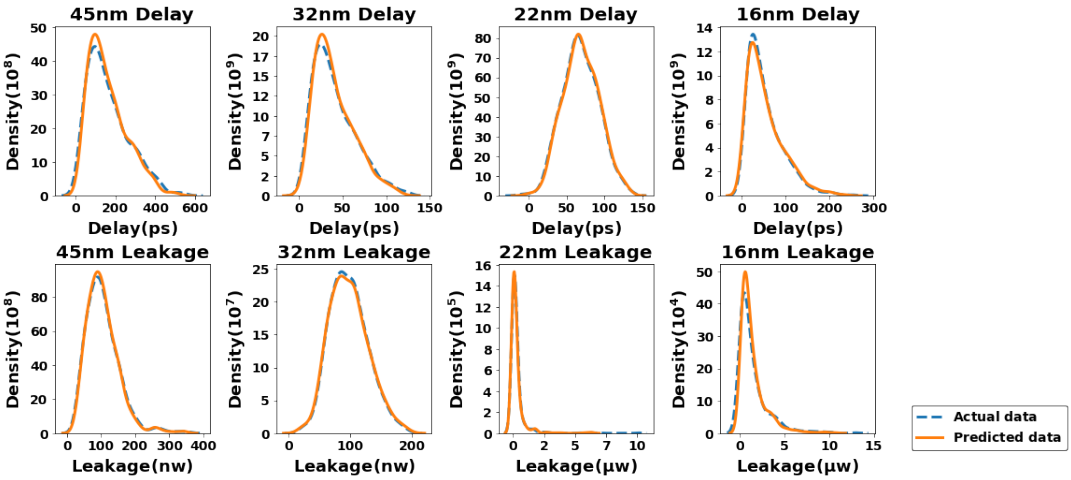


Figure 6.6: Performance of CMOS high-performance 45 → 32nm → 22nm → 16nm sequential transfer learned models w.r.t SPICE (actual) data.

parallel investigation on CMOS technology nodes 45nm to 32nm. The distribution of actual simulations of leakage and delay at 45nm from SPICE perfectly aligns with the DNN source model predictions, deliberately showcasing the efficacy of the devised neural network model. Furthermore, the overlapping distributions of SPICE and the transfer-learned model's leakage and delay at 32nm demonstrate the accuracy of the transfer-learned network. Additionally, we evaluate the performance of baseline models (trained with 5,000 and 15,000 samples) for FinFET alongside SPICE and transfer models in Fig. 6.5. The results indicate a negligible level of error. In Fig 6.7, a comparison is presented between SPICE simulation time and the training times of DNN and TL models, encompassing the prediction times of the trained models. The trained DNN models require only 0.03 seconds to predict leakages and delays for 5,000 random PVT conditions. This showcases the significant enhancement in computational speed brought about by the proposed methodology.

In each experiment, we intentionally incorporated MaPE assessment using unseen samples at every node. This was done to meticulously examine the designed models' ability to generalize and to scrutinize for any negative transfer effects thoroughly. The precise alignment between the predictions and actual simulations strongly suggests the absence of negative transfer. This underscores that the transfer-learned models have been finalized undergoing comprehensive experimentation.

6.5.2.2 Sequential Transfer

We introduce a method of sequential transfer learning for PVT-aware circuit analysis across multiple technology nodes, exemplified by the 16nm to 10nm transferred model serving as the source node to model 7nm leakage/delay (16nm → 10nm → 7nm). The numerical outcomes of this experiment, conducted on FinFET nodes ranging from 16nm to 7nm, are detailed in Table 6.3. The most proficient model among the four fine-tuning cases is highlighted in bold and designated as the final model. The

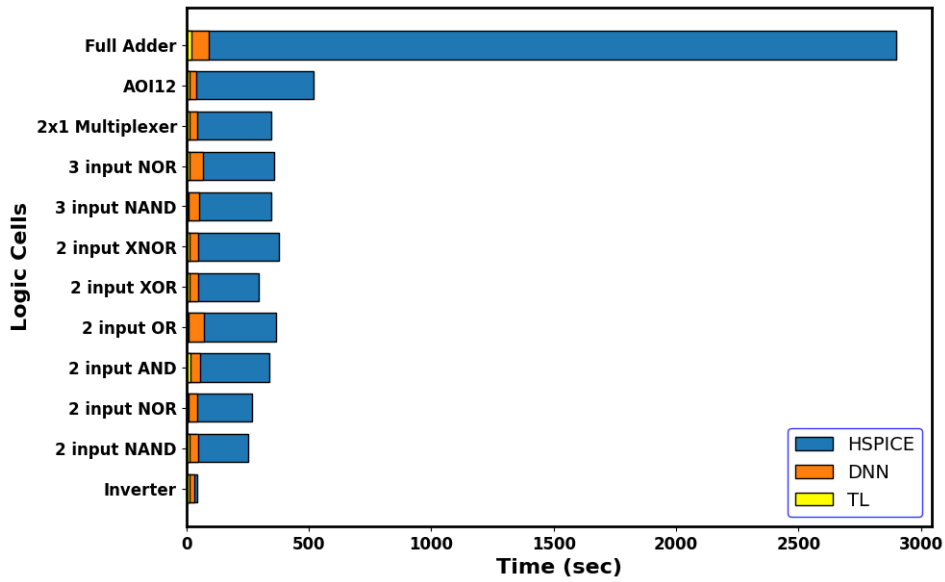


Figure 6.7: Comparison of DNN training time with SPICE Simulation time on FinFET 10nm delay with 5000 samples and TL training time with 500 samples.

results of the transfer-learned network exhibit an R^2 greater than 0.98, with an average variation of $\%\mu_E$ and $\%\sigma_E$ less than 1% in all standard cell estimations, along with MaPE of 2%. These findings are in alignment with the outcomes of single-node transfer. Notably, in sequential transfer, the training data for the second node (7nm) has been reduced by an additional 10%, comprising only 400 samples for delay and 1350 samples for leakage. The congruence between SPICE simulations and predictions from the sequential transfer model at 7nm across all standard cells underscores the precision of the modeling process (Fig. 6.4). The figure also provides comparative insights into the results of different fine-tuning phases.

Similar experiments have been extended to CMOS technology. The results for CMOS 45nm $\rightarrow$ 32nm $\rightarrow$ 22nm $\rightarrow$ 16nm are illustrated using a 2x1 multiplexer cell (Fig. 6.6). Average R^2 is 0.98, $\%\mu_E < 1\%$, $\%\sigma_E < 2\%$ and 3% MaPE are the reported metrics. These results demonstrate accurate estimations of leakage and delay, even at the transferred leaf nodes, i.e., FinFET 7nm and CMOS 16nm. Our comprehensive experiments on CMOS/FinFET cells substantiate that sequential transfer learning facilitates precise circuit estimation, further reducing training requirements.

6.5.2.3 Skip-node Transfer

We have also investigated TL towards a lower technology node, bypassing one intermediary node. For instance, taking FinFET 16nm as the source node and 7nm as the target node, bypassing the 10nm technology node. We name it Skip-node transfer. The findings reveal that, across all cells, the average $\%\sigma_E$ amounts to 1% for delay and 3% for leakage, with MaPE values of 2% and 5% respectively. In

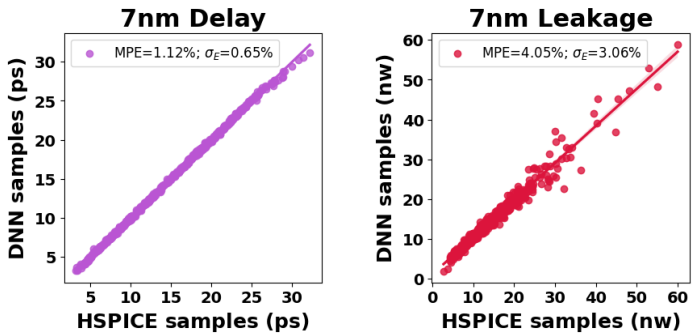


Figure 6.8: Performance of skip-node transfer learning model from 16 $\rightarrow$ 7nm

Fig. 6.8, the SPICE and TL predictions for a 2-input AND cell are depicted. The ML metrics reported are $\%\sigma_E$ is 1% and 3%; MaPE is 2% and 5% for delay and leakage, respectively, on an average across all the cells. The outcomes demonstrate exceptional accuracy in delay estimation; however, there is a slight deviation in leakage estimation.

6.5.2.4 Reverse-node Transfer

Additionally, we perform experiments on reverse-node transfer learning, addressing scenarios where a trained model from a lower technology, such as 7nm, is employed to model the performance of a higher technology node, for example, 10nm. The conducted experiments, visualized in Fig 6.9 (using 2-input AND cell), yield encouraging outcomes for the delay, achieving a $\%\sigma_E$ of approximately 1%, though for leakage, it slightly increases to 3%.

6.5.2.5 Zero-shot Learning

We demonstrate the ZSL implementation for futuristic performance prediction on 7nm statistical library cells. This relies solely on prior knowledge from the 10nm node (Case 3: 10nm $\rightarrow$ 7nm) and both the 16nm and 10nm nodes (multi-node as in Case 4: 16nm $\rightarrow$ 10nm $\rightarrow$ 7nm). No training data from the 7nm node is utilized in these estimations. The experimental outcomes are plotted in Fig. 6.10. Comparisons are made with predictions from a DNN model trained with 5,000 data samples at the 7nm node (Case 1: DNN model) and a TL (FSL) model constructed using only 500 data samples (Case 2: 10nm $\rightarrow$ 7nm, with limited training samples at 7nm). The close alignment of zero-shot (Cases 3 and 4), transfer-learned (Case 2), and DNN (Case 1) leakage/delay predictions at the 7nm node with SPICE MC simulations underscores the accuracy and potential of our ZSL model and tailored DNN architecture, showcasing its modeling capabilities and enhanced generalization.

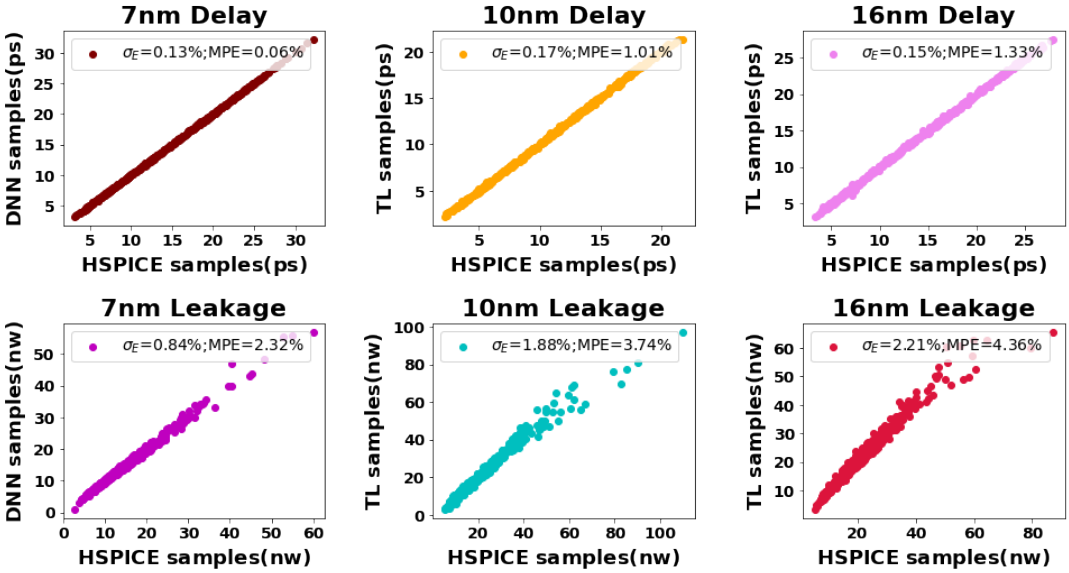


Figure 6.9: Performance of sequential transfer learning models from lower to higher technology nodes 7nm $\rightarrow$ 10nm $\rightarrow$ 16nm

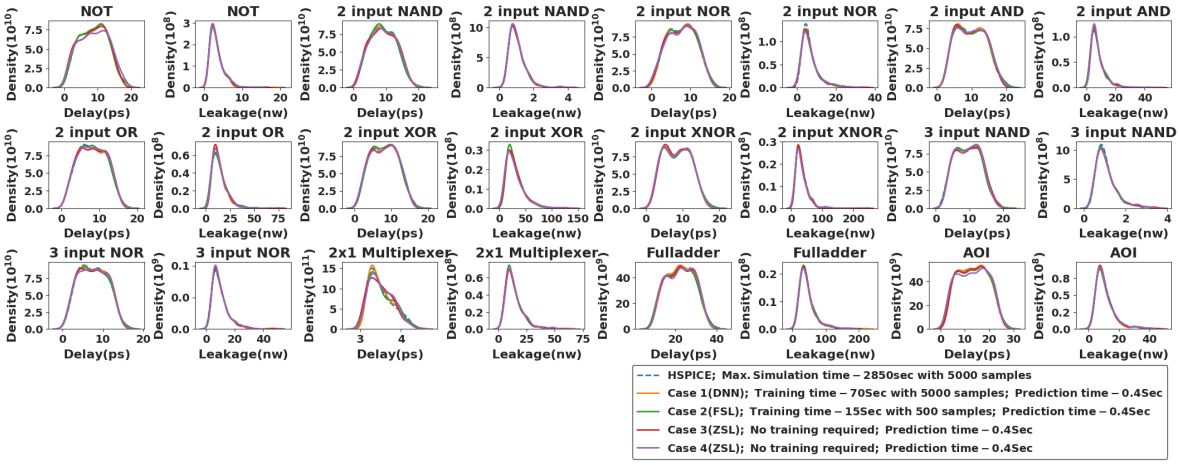


Figure 6.10: Performance of the leakage/delay of the proposed methodology for 7nm FinFET standard cells w.r.t SPICE simulations for 500 unseen test cases

6.6 Conclusion

This chapter presents a rapid, precise, and computationally efficient surrogate modeling approach for characterizing PVT-aware digital circuit performance. The method involves developing highly efficient DNN models, which are subsequently utilized for TL across diverse technology nodes, thereby minimizing data requirements. The proposed TL framework effectively integrates information from dominant PVT conditions, the DNN at the source node, and a limited set of samples from the target node. Additionally, in another configuration of ZSL, accurate modeling is achieved with zero modeling complexity and without the need for any simulation data, making it a valuable tool for preliminary analysis of circuit variability for future nodes.

This facilitates the implementation of effective countermeasures to ensure reliable and optimal designs. The experimental results, both before and after fine-tuning via Bayesian optimization, showcase enhanced performance of the transfer-learned DNN. Through comprehensive demonstrations across various digital standard cells, our proposed methodology achieves a speed-up of 10^4 times compared to SPICE and up to 10x cost reduction compared to other state-of-the-art modeling techniques, all while maintaining heightened model accuracy. Moreover, the estimated leakage and delay of complex circuits, obtained through the modeled standard cells, closely align with simulation results, eliminating the need for additional computations. It's worth emphasizing that the proposed framework functions as a black-box model, indicating its equal applicability to modeling any set of performances at any desired technology node, provided with associated performances for modeling.

Table 6.3: Validation of sequential transfer learning framework with fine-tuning through Bayesian Optimization for each standard cell

Standard cell	Performance	16nm				Fine-Tuned		10nm				7nm			
		R^2	$\%\mu_E$	$\%\sigma_E$	MPE	Upper layer	Lower layer	R^2	$\%\mu_E$	$\%\sigma_E$	MPE	R^2	$\%\mu_E$	$\%\sigma_E$	MPE
Inverter	Leakage	0.993	0.503	0.127	1.475	False	False	0.974	1.667	3.725	5.472	0.990	0.693	0.217	4.122
						False	True	0.987	0.708	1.752	3.514	0.986	0.297	1.241	3.265
						True	False	0.990	0.110	1.506	3.418	0.981	0.114	3.321	3.646
						True	True	**0.992**	**0.157**	**0.368**	**2.772**	**0.977**	**0.561**	**0.605**	**3.349**
	Delay	0.999	0.186	0.381	0.871	False	False	0.994	0.513	0.179	2.567	0.996	0.093	1.418	2.728
						False	True	0.995	0.513	1.177	2.411	0.995	0.396	0.202	2.783
						True	False	0.998	0.689	1.206	1.563	0.996	1.054	2.548	2.255
						True	True	**0.998**	**0.301**	**0.659**	**1.266**	**0.998**	**0.170**	**0.897**	**1.532**
2 input Nand	Leakage	0.993	0.638	0.851	2.055	False	False	0.969	0.665	4.661	6.244	0.974	0.742	4.044	5.579
						False	True	0.988	0.176	3.463	4.201	0.987	0.197	2.413	4.133
						True	False	0.989	0.495	2.100	4.117	0.984	0.129	2.453	3.113
						True	True	**0.986**	**0.283**	**1.303**	**4.084**	**0.982**	**0.270**	**1.075**	**3.277**
	Delay	0.999	0.365	0.428	0.993	False	False	0.995	0.002	0.293	2.699	0.993	0.883	3.919	3.477
						False	True	0.994	0.559	0.961	2.491	0.993	0.123	0.831	3.568
						True	False	**0.999**	**0.122**	**0.107**	**0.974**	**0.999**	**0.045**	**0.199**	**1.685**
						True	True	0.999	0.453	2.621	1.179	0.999	0.066	2.313	1.551
2 input Nor	Leakage	0.993	0.319	1.012	2.063	False	False	0.962	2.034	4.742	6.743	0.952	2.231	10.006	4.252
						False	True	0.988	0.239	0.920	3.862	0.975	0.975	5.260	3.420
						True	False	0.988	0.606	2.640	4.217	**0.990**	**0.688**	**0.340**	**2.975**
						True	True	**0.991**	**0.086**	**1.093**	**3.305**	0.987	0.812	3.613	4.226
	Delay	0.999	0.144	0.303	0.805	False	False	0.994	1.906	2.308	3.043	0.994	0.381	1.178	3.121
						False	True	0.996	0.835	0.276	2.282	0.993	0.177	2.116	3.360
						True	False	0.997	0.279	2.280	1.577	0.999	1.193	3.287	2.107
						True	True	**0.999**	**0.573**	**0.467**	**1.475**	**0.998**	**0.616**	**0.848**	**1.295**
2 input And	Leakage	0.994	0.223	1.004	2.026	False	False	0.968	2.012	2.217	5.967	0.966	0.049	3.591	4.027
						False	True	0.985	0.958	2.325	3.317	0.978	0.948	1.717	3.433
						True	False	0.986	0.664	2.025	3.182	**0.991**	**0.350**	**0.252**	**3.363**
						True	True	**0.987**	**0.359**	**1.080**	**3.085**	0.982	0.245	0.648	3.632
	Delay	0.999	0.335	0.176	0.706	False	False	0.995	0.545	0.390	2.194	0.992	0.702	2.861	3.462
						False	True	0.993	0.949	1.009	2.244	0.992	0.328	0.385	3.345
						True	False	0.998	0.231	2.328	0.863	0.998	0.337	0.731	1.991
						True	True	**0.999**	**0.005**	**0.127**	**0.936**	**0.998**	**0.197**	**0.548**	**1.479**
2 input Or	Leakage	0.993	0.323	0.155	1.967	False	False	0.967	1.146	3.065	4.458	0.966	0.227	5.918	3.814
						False	True	0.987	0.556	2.393	2.899	0.968	1.351	4.646	3.083
						True	False	0.990	0.756	2.875	3.455	0.977	0.354	1.257	2.366
						True	True	**0.993**	**0.697**	**0.651**	**3.432**	**0.985**	**0.481**	**0.143**	**2.291**
	Delay	0.999	0.645	0.438	1.07	False	False	0.992	1.151	4.068	2.278	0.995	0.703	0.041	2.473
						False	True	0.992	0.763	3.230	2.407	0.991	0.496	1.192	2.362
						True	False	0.998	0.419	1.461	1.092	0.999	0.907	3.707	1.514
						True	True	**0.999**	**0.454**	**0.956**	**1.025**	**0.997**	**0.578**	**0.255**	**1.836**
2 input Xor	Leakage	0.992	0.560	0.701	2.075	False	False	0.971	0.506	3.958	5.977	0.967	1.890	2.838	4.906
						False	True	0.984	0.413	2.159	4.023	0.972	0.108	2.642	4.908
						True	False	**0.985**	**0.077**	**1.341**	**3.658**	0.981	0.993	4.418	3.872
						True	True	0.986	0.596	3.939	4.851	**0.982**	**0.507**	**0.001**	**3.573**
	Delay	0.999	0.091	0.752	0.604	False	False	0.996	0.069	0.357	1.788	0.993	1.012	0.247	2.875
						False	True	0.997	0.195	1.781	1.796	0.995	0.183	0.881	2.631
						True	False	**0.999**	**0.205**	**0.021**	**1.128**	0.999	0.042	0.792	1.612
						True	True	0.999	0.435	1.713	0.968	**0.999**	**0.004**	**0.772**	**1.495**

Standard cell	Performance	16nm				Fine-Tuned		10nm				7nm			
		R^2	$\%\mu_E$	$\%\sigma_E$	MPE	Upper layer	Lower layer	R^2	$\%\mu_E$	$\%\sigma_E$	MPE	R^2	$\%\mu_E$	$\%\sigma_E$	MPE
2 input Xnor	Leakage	0.992	1.065	1.095	2.097	False	False	0.9633	0.527	6.337	5.215	0.976	0.065	2.880	4.738
						False	True	0.988	0.182	1.473	3.262	0.982	0.934	2.745	3.438
						True	False	0.989	0.259	2.216	3.031	0.965	2.099	5.321	3.268
						True	True	**0.991**	**0.992**	**1.387**	**2.025**	**0.980**	**0.960**	**1.771**	**3.722**
	Delay	0.999	0.389	0.275	1.09	False	False	0.997	0.129	1.486	1.676	0.992	0.271	3.616	3.771
						False	True	**0.999**	**0.017**	**0.641**	**1.571**	0.991	0.253	0.505	3.808
						True	False	0.999	0.100	0.299	1.583	0.997	0.077	0.477	2.517
						True	True	0.999	0.490	0.338	1.602	**0.997**	**0.229**	**0.142**	**2.163**
3 input Nand	Leakage	0.991	1.025	1.095	1.719	False	False	0.978	1.597	0.852	3.565	0.986	0.335	0.502	2.277
						False	True	0.992	0.179	1.865	2.692	0.987	0.483	1.838	2.703
						True	False	**0.994**	**0.007**	**0.949**	**1.692**	**0.994**	**0.229**	**0.366**	**1.466**
						True	True	0.994	0.103	2.252	2.619	0.9808	0.041	1.112	2.429
	Delay	0.999	0.182	0.798	0.604	False	False	0.992	0.630	3.888	2.258	0.994	1.544	2.266	2.818
						False	True	0.994	0.406	0.512	2.022	0.994	0.367	2.467	2.623
						True	False	0.996	0.848	3.830	1.120	0.998	0.141	1.152	1.930
						True	True	**0.999**	**0.152**	**0.666**	**1.064**	**0.999**	**0.118**	**0.114**	**1.458**
3 input Nor	Leakage	0.989	1.022	1.088	1.231	False	False	0.957	0.765	4.866	5.754	0.955	0.383	0.605	5.953
						False	True	0.966	0.328	3.453	4.127	0.990	0.688	0.340	2.975
						True	False	**0.981**	**0.001**	**1.809**	**3.650**	0.982	0.304	2.986	4.005
						True	True	0.975	0.919	3.686	4.304	**0.975**	**0.158**	**1.654**	**3.099**
	Delay	0.999	0.032	0.409	0.646	False	False	0.990	0.972	4.636	2.564	0.995	0.167	3.479	2.298
						False	True	0.991	0.417	4.303	2.494	0.994	0.251	2.853	2.239
						True	False	0.999	0.530	4.246	1.003	0.996	0.458	3.672	1.094
						True	True	**0.999**	**0.213**	**0.138**	**0.820**	**0.999**	**0.643**	**0.036**	**1.020**
2x1 Multiplexer	Leakage	0.993	0.189	0.030	2.089	False	False	0.962	0.369	4.510	6.96	0.985	0.693	4.712	3.382
						False	True	0.991	1.499	4.087	4.019	0.986	0.084	0.062	3.879
						True	False	0.986	0.270	2.691	3.735	0.986	0.449	1.822	3.490
						True	True	**0.985**	**0.635**	**1.283**	**3.202**	**0.977**	**0.927**	**0.035**	**3.077**
	Delay	0.999	0.042	1.080	0.334	False	False	0.996	0.198	1.995	0.456	0.976	0.062	6.67	2.652
						False	True	**0.997**	**0.077**	**0.032**	**0.369**	0.953	0.106	7.292	2.552
						True	False	0.994	0.364	1.810	0.456	0.964	0.277	9.658	1.935
						True	True	0.995	0.128	1.441	0.421	**0.995**	**0.278**	**1.375**	**1.702**
Full adder	Leakage	0.995	0.589	1.041	0.845	False	False	0.949	0.464	3.630	6.347	0.971	1.405	8.762	6.164
						False	True	0.977	1.895	2.181	3.795	0.968	0.192	3.353	3.746
						True	False	**0.989**	**0.153**	**1.743**	**3.217**	**0.977**	**0.723**	**1.953**	**3.213**
						True	True	0.989	0.858	2.262	4.347	0.955	0.495	1.975	4.463
	Delay	0.997	0.969	1.192	1.285	False	False	0.989	2.374	1.472	2.453	0.966	0.462	1.739	3.871
						False	True	0.967	4.305	3.912	4.649	0.979	2.112	3.075	3.686
						True	False	0.993	1.609	2.824	2.320	**0.996**	**0.627**	**1.566**	**1.267**
						True	True	**0.992**	**0.760**	**1.071**	**2.184**	0.994	0.780	2.075	1.071
AOI12	Leakage	0.992	1.074	0.442	1.874	False	False	0.962	1.964	4.726	5.705	0.977	0.098	3.486	5.596
						False	True	0.987	1.999	0.939	4.553	0.963	1.547	0.325	6.736
						True	False	0.988	1.225	1.442	3.869	0.988	0.364	2.217	5.642
						True	True	**0.990**	**0.505**	**1.382**	**2.805**	**0.975**	**0.493**	**1.952**	**3.668**
	Delay	0.999	0.126	0.906	0.807	False	False	0.996	0.335	2.753	1.877	0.992	0.655	2.863	3.693
						False	True	**0.996**	**0.121**	**0.294**	**1.699**	0.986	0.553	5.007	3.808
						True	False	0.999	0.158	2.058	0.981	0.998	0.017	1.549	1.522
						True	True	0.999	0.182	2.216	1.566	**0.999**	**0.011**	**0.024**	**1.403**

Chapter 7

Applications of Digital Circuit Surrogate Models

Digital standard cell characterization is a crucial step in IC design, involving the assessment of standard cell behavior under various conditions. Our PVT-aware standard cell library, featuring 22 commonly used digital cells, serves diverse purposes in the design process, from performance enhancement to ensuring reliability in diverse environments. This includes:

- **Variability Analysis:** Characterization offers insights into how standard cells respond to process variations, supply fluctuations, and temperature changes. This aids in assessing their resilience to variations and reliability in different settings.
- **Timing Parameter Estimation:** It facilitates the estimation of crucial timing parameters like propagation delay, ensuring the digital circuit meets timing specifications.
- **PVT-aware Power Estimation:** Enables the evaluation of power consumption, particularly critical for assessing leakage power, a dominant source in sub-nanometer technology.
- **Optimization Library Creation - OptiMo:** PVT-characterized models are used to build a library of optimized standard cells, considering factors like power and performance. These can then be effectively utilized by designers in specific applications.

7.1 PVT-aware Complex Cell Estimation

Our aim is to expedite and enhance the evaluation of intricate digital circuit performance across diverse PVT conditions using pre-characterized standard cells. Our approach facilitates precise estimation of leakage and delay in complex digital cells by exclusively utilizing surrogate models constructed from standard digital cells. These models offer both accuracy and computational efficiency, eliminating the need for additional simulations, a common practice in traditional methods for each complex cell. We have developed estimators for complex cell leakage and delay, seamlessly integrating them into our framework to streamline performance estimation.

The process of estimating the performance of complex cells, outlined in Fig. 7.1, begins by inputting the circuit's Verilog structural model. It then proceeds with a structural mapping of the Verilog de-

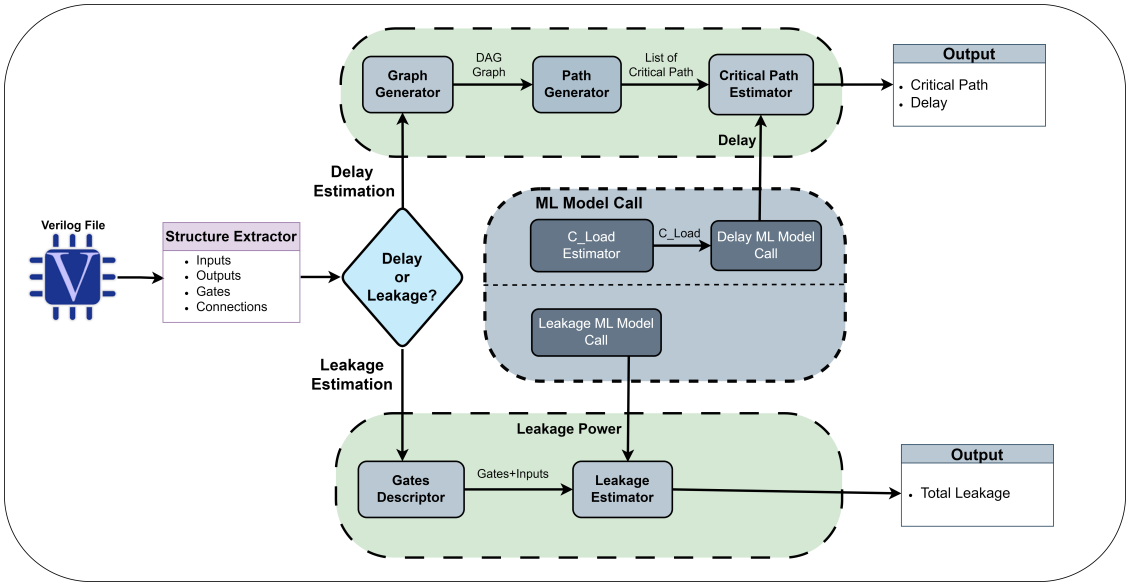


Figure 7.1: The comprehensive flow of the automated complex cell framework

scription to the digital library cells. Next, the estimator calls upon the trained ML power/delay models selected by the user and predicts the overall circuit power/delay. Our approach simplifies the assessment of complex cell performance, enhancing accessibility and efficiency for digital designers.

7.1.1 Complex Cell Leakage Estimation

We have designed a top-level model for estimating complex cell leakage across the PVT, denoted as the Complex Cell Leakage (CCL) estimator. The leakage of the complex cell relies on the statistical leakage estimations of the individual cells that comprise it.

$$CCL_{Leak} = \sum_{i=1}^{M} (P_{Leak-Stat}) \tag{7.1}$$

Here, CCL_{Leak} represents the leakage of a complex cell for a specific input combination, with M denoting the number of distinct standard cells within it. Our estimator initiates by decomposing the complex cell into a collection of standard cells, associating the corresponding $P_{Leak-Stat}$ values with the relevant inputs obtained from the primary inputs of the complex circuit. This structural information is systematically organized into a Python dictionary [113], with gates serving as keys and their respective inputs as values. Following this dictionary structure, commencing from standard cells linked to the primary inputs, surrogate leakage models (trained ML models) of standard cells are utilized to evaluate $P_{Leak-Stat}$ for a given PVT and input(s). This iterative process extends to all dictionary elements, concluding at gates connected to the primary outputs. Subsequently, a leakage calculator function computes the total leakage power of the complex cell by aggregating individual leakages.

In essence, our methodology entails the methodical arrangement of complex cell components, leveraging a dictionary structure for streamlined processing, and employing ML models to predict PVT-aware leakage for each cell within the circuit, culminating in the computation of the overall leakage.

7.1.2 Complex Cell Delay estimation

The delay of a complex cell is contingent upon the critical delay of the circuit. To identify the critical path in a complex circuit, we transform the circuit into a Directed Acyclic Graph (DAG) [114] by considering the circuit inputs, outputs, gates, and their connections. In this representation, nodes correspond to inputs, outputs, and gates, while edges depict directed connections between the gates. Initially, all edges are assigned equal weights, typically set to 1. Identifying critical paths involves extracting paths with the highest weights, resulting in a list of critical paths. We evaluate the delay of more than one critical path to ensure maximum delay estimation, accounting for potential variations resulting from PVT effects. Each critical path is deconstructed into a pair of driver-driven cells. By determining the standard cell type serving as the load, and obtaining its associated C_L from the capacitive load matrix (section 3.4.3), the Complex Cell Delay (CCD) estimator leverages surrogate delay models within each critical path, considering PVT conditions and C_L as inputs to determine the statistical delay of the driver. The overall circuit delay is then computed as the linear combination of all driver cell delays, given by:

$$C_{Delay-Stat} = \sum_{i=1}^{K} (G_{Delay-Stat}) \tag{7.2}$$

Here, $C_{Delay-Stat}$ signifies the delay of a complex cell, and K denotes the count of standard cells in the critical path. This process iterates for each cell within the critical path, updating the weights of the edges along the way. Following this, critical path extraction and maximum delay calculation functions are employed to derive the critical path and its associated maximum delay from the list of identified critical paths.

7.1.3 Results and Discussion

The CCL and CCD estimators yield the final leakage and delay values of the complex cell for a specific PVT condition in seconds. The outcomes of the estimations in FinFET technology nodes for a randomly selected PVT test case across different complex cells/circuits, with the transfer learning methodology (Chapter 6) are outlined in Tables 7.2 and 7.3. The percentage error, as defined by equ. 7.3, is used to gauge the accuracy of the estimations with the SPICE simulations.

$$\%Error = \frac{MODEL_{Prediction} - SPICE_{value}}{SPICE_{Value}} * 100 \tag{7.3}$$

The results demonstrate an average error of less than 1% for all complex circuits regarding leakage and delay estimations compared to SPICE simulations, with estimations running several orders of magni-

Table 7.1: OptiMo PVT variations

Process Parameter Variations (Gaussian distributions $\pm 3\sigma$ from mean)						**Uniform Random Variations (URV)**		**Design Parameters (URV)**	
Toxe (nm)	Toxm (nm)	Toxref (nm)	Toxp (nm)	Xj (nm)	Ndep (E+17/cm^3)	T_{op} ($^{\circ}C$)	V_{dd} (V)	L (nm)	W (nm)
0.65	0.67	0.67	0.4	7.2	120	-55 to 125	0.8 $\pm 10\%$	22 to 26	22 to 440

tude faster. This improvement is even more pronounced for complex cells, due to the utilization of pre-characterized standard cell characteristics, abstaining from any further simulations. In traditional E-CAD tools, the complexity of complex cell PVT analysis increases linearly, potentially affecting operating speed and accuracy. However, in our framework, this aspect remains unaffected, ensuring that speed and accuracy remain at their peak regardless of circuit complexity.

7.2 PVT-aware Sizing and Optimization Engine

Our digital standard cell characterization finds another valuable application in creating a swift and effective optimization engine. Our optimization framework constitutes a composite structure comprising a series of multi-objective Pareto-optimal algorithms that operate in conjunction with a set of ML algorithms. We name it as "OptiMo". The primary objective is to achieve rapid and efficient optimization for the sizing of standard digital logic cells. Embracing a bottom-up approach, the methodology begins with leaf cells representing optimized digital logic cells that exhibit resilience across a broad spectrum of PVT variations. These cells are designed to meet power and delay constraints under nominal conditions as well as within the range of $\pm 10\%$ of PVT variations from nominal values. Subsequently, an automated complex circuit performance estimator (CCL and CCD) leverages these optimized cells contributing to the power-delay optimization at the circuit level. OptiMo addresses the computational demand of the traditional optimization processes that rely heavily on simulators. The distinctive aspect of our work lies in integrating state-of-the-art optimization algorithms on top of the accurately trained ML models, resulting in a substantial improvement in computational speed compared to simulator-dependent approaches.

The methodology begins with the generation of training data, incorporating Gaussian variations in the manufacturing process and Uniform Random Variations (URV) in temperature, supply voltage, and transistor design parameters (channel length and width). The nominal process values and URV ranges are detailed in Table 7.1. Logic cell netlists, integrating the generated statistical variations and technology library files, undergo simulation using SPICE to estimate leakage and delay variation data. The generation of training dataset goes inline with the methodology discussed in Chapter 3 (section 3.4), in addition we include design parameter variations for each transistor in the logic cell in this context. Further, the circuit performances modeled include leakage power and propagation delay. This PVT-aware simulation data serves as the training dataset for generating surrogate ML models. It's noteworthy that generating such datasets is a one-time requirement for developing surrogate ML models, applicable across a wide spectrum of digital circuit designs.

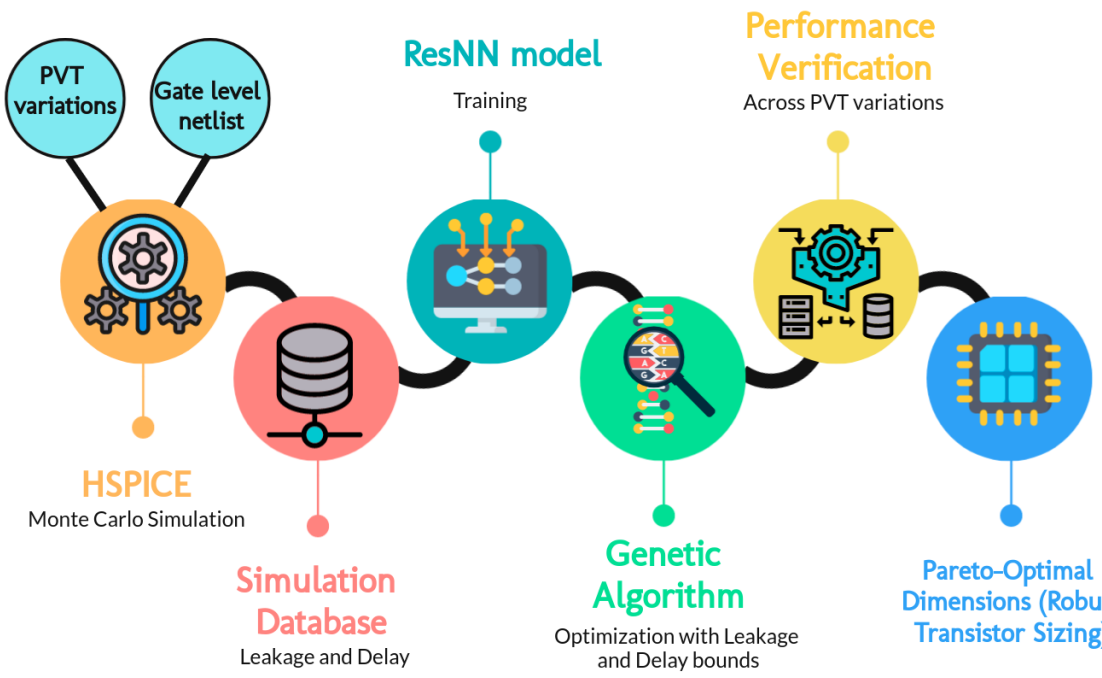


Figure 7.2: The PVT aware Pareto-optimal transistor sizing methodology

7.2.1 The Key Idea

A bank of ML algorithms is trained on the selected statistical training data and the best-performing algorithm is identified and applied to all the standard digital cells. The accurate ML models are proficient in prototyping the SPICE simulations at a significantly accelerated pace. Subsequently, a set of optimization algorithms is deployed to operate over the trained ML models, to size optimization of the logic cell to minimize power and delay while adhering to the bounds on minimum sizing leakage and delay at extreme PVT values ($\pm 10\%$ from the mean). The robustness of the obtained sizing is tested over 10000 randomly generated PVT values. The optimal combination of the optimization algorithm and the ML model is determined through a defined composite criterion, and this combination is then applied to model all other cells. The optimized sizing is subsequently employed in an automated routine for complex circuit power-delay optimization.

Fig. 7.2 shows the PVT aware Pareto-optimal transistor sizing methodology with ResNN model and Genetic Algorithm (GA) as an example. The consecutive stages involve training a ResNN model using the standard cells database to accurately predict delay and leakage values for a given PVT variation. Subsequently, a GA-based optimization engine is constructed on top of the ResNN model to determine the Pareto-optimal point with minimum delays and leakages by adjusting each transistor gate's dimensions (L and W). Estimations for leakage and delay are then conducted for the optimized gate dimensions across the PVT spectrum, validating the nominal bounds under standard operating conditions

Table 7.2: Comparison of propagation delay estimations (ps) of DNN/TL model w.r.t SPICE simulations on Complex/multi-stage cells

Technology node		16nm			10nm					7nm				
Complex cell		HSPICE	MODEL	%Error	HSPICE	Base MODEL	%Error	TL MODEL	%Error	HSPICE	Base MODEL	%Error	TL MODEL	%Error
2NOT	Inverter chain	25.87	25.87	0.00	22.81	22.83	-0.09	22.82	-0.04	20.48	20.51	-0.15	20.61	-0.63
4NOT	Inverter chain	51.74	51.74	0.00	41.98	41.95	0.07	42.42	-1.05	41.26	41.42	-0.39	41.71	-1.09
2NAND	2-input nand chanin	24.6	24.45	0.61	22.9	22.72	0.79	22.91	-0.04	24.5	24.75	-1.02	24.3	0.82
4NAND	2-input nand chanin	49.2	48.89	0.63	55.2	54.97	0.42	54.92	0.51	42.67	42.4	0.63	42.57	0.23
4PARITY	4-bit Parity generator	20.35	20.3	0.25	32.77	32.73	0.12	32.69	0.24	35.3	35.15	0.42	35.33	-0.08
8PARITY	8-bit Parity generator	27.7	27.61	0.32	44.68	44.64	0.09	44.63	0.11	48.3	48.22	0.17	48.34	-0.08
2MULT	2-bit multiplier	41.2	41.3	-0.24	39.39	39.29	0.25	39.76	-0.94	34.53	34.8	-0.78	35.01	-1.39
4MULT	4-bit multiplier	149.7	149.87	-0.11	130.4	131.2	-0.61	128.69	1.31	108.58	108.48	0.09	107.97	0.56
4FA	4-bit adder	87.5	86.7	0.91	78.6	78.86	-0.33	77.59	1.28	78.23	78.35	-0.15	77.48	0.96
8FA	8-bit adder	174.4	173.2	0.69	147.4	147.81	-0.28	145.47	1.31	151.33	150.56	0.51	150.01	0.87
16FA	16-bit adder	333.7	332.1	0.48	303	304.6	-0.53	298.53	1.48	364.2	363.52	0.19	360.55	1.00
32FA	32-bit adder	639.2	634.6	0.72	544.8	548.5	-0.68	538.2	1.21	645.3	643.09	0.34	641.17	0.64

Table 7.3: Comparison of leakage power estimations (nw) of DNN/TL model w.r.t SPICE simulations on Complex/multi-stage cells

Technology node		16nm			10nm					7nm				
Complex cell	Input combination(s)	HSPICE	MODEL	%Error	HSPICE	Base MODEL	%Error	TL MODEL	%Error	HSPICE	Base MODEL	%Error	TL MODEL	%Error
2NOT	1'D0	5.09	5.05	0.79	8.6	8.68	-0.93	8.62	-0.70	3.16	3.18	-0.63	3.16	0.00
4NOT	1'D1	16.05	15.82	1.43	13.09	13.19	-0.76	12.99	-1.54	6.69	6.73	-0.60	6.61	1.20
2NAND	2'D2	10.69	10.54	1.40	22.69	22.81	-0.53	22.37	-1.97	7.93	7.92	0.13	7.91	0.25
4NAND	2'D3	14.66	14.52	0.95	23.77	23.9	-0.55	23.47	-1.83	9.57	9.56	0.10	9.56	0.10
4PARITY	4'D11	48.55	47.58	2.00	56.11	56.26	-0.27	55.81	-0.81	34.46	34.45	0.03	35.06	-1.74
8PARITY	8D'20	235.1	233.94	0.49	213.22	212.46	0.36	213.07	0.29	130.88	130.85	0.02	128.81	1.58
2MULT	{2'D1,2'D3}	86.3	86.6	-0.35	79.35	78.99	0.45	79.47	0.60	89.64	90.29	-0.73	89.81	-0.19
4MULT	{4'D12,4'D10}	954.43	951.64	0.29	837.2	838.5	-0.16	813.56	-3.07	594.57	595.15	-0.10	599.73	-0.87
4FA	{4'D2,4'D15}	226.7	226.03	0.30	146.2	143.61	1.77	146.21	1.78	143.1	146.59	-2.44	147.51	-3.08
8FA	{8'D15,16'D128}	535.1	528.3	1.27	508.4	507.77	0.12	504.67	-0.61	477.7	481.43	-0.78	489.8	-2.53
16FA	{16'D19840,16'D45327}	1125.7	1121.4	0.38	895.2	882.59	1.41	892.83	1.15	661.4	667.71	-0.95	675.56	-2.14
32FA	{32'D7359872,32'D15577359}	2045.6	2047.5	-0.09	1619.3	1594.5	1.53	1577.5	-1.08	1234	1232.1	0.15	1250.2	-1.31

7.2.2 Machine Learning Algorithms for Circuit Optimization

Within our framework, we strategically integrate a varied array of regression algorithms adept at analyzing and modeling PVT-aware leakage and delay phenomena. These algorithms encompass decision trees, extra trees, light gradient boosting regressors, artificial neural networks, and residual neural networks. Each algorithm is meticulously applied with suitable scaling functions using one or more datasets to identify the most effective one, streamlining its subsequent efficient utilization by optimization algorithms. Below is a concise overview of these algorithms.

7.2.2.1 Decision Tree Regressor

The DT regressor constructs a predictive model resembling a tree structure based on input features, where nodes represent features and thresholds recursively divide the data [115]. Specific predictions are provided at terminal nodes, and the model highlights the importance of features. DT is known for its ability to capture non-linear patterns in the data. We utilized the L2 norm (mean-squared error)

for optimal feature and threshold selection. The regressor generates continuous predictions, effectively capturing nonlinear relationships. Nonetheless, to mitigate the risk of over-fitting, we implemented pre-pruning measures to restrain excessive complexity and ensure improved generalization on unseen data. Balancing model complexity and regularization, hyperparameter tuning was conducted using grid search, a commonly used technique for optimizing hyperparameters in machine learning models [77]. This process yielded a configuration with a maximum depth of 3, a maximum of 5 leaf nodes, a minimum of 3 samples per leaf, and a minimum weight fraction of 0.1, selected based on the best split.

7.2.2.2 Extra Tree Regressor

ET, an advancement of the Random Forest algorithm tailored for regression tasks, harnesses ensemble learning principles [116]. Distinguished by its randomized tree construction process, ET enhances robustness and mitigates over-fitting risks. To optimize its hyperparameters effectively, we employed BO [75, 76], a sophisticated technique that utilizes probabilistic models to guide the search process toward promising regions of the hyperparameter space. This process yielded a model configuration with 100 estimators, a minimum sample leaf of 2, a minimum of 4 samples necessary for node splitting, and parallelization across eight jobs.

7.2.2.3 Light Gradient Boosting Regressor

LGBM is an ensemble learning algorithm that has shown exceptional performance in terms of both speed and accuracy [107,117]. Its unique tree construction algorithm adopts a leaf-wise growth strategy, resulting in faster training and reduced memory consumption. This characteristic ensures efficiency, particularly with large datasets and high-dimensional features. LGBM has been selected for its proficiency in enhancing the predictive capability of surrogate models. After hyperparameter tuning using BO, the final settings comprise a learning rate of 0.025, a maximum depth of 8, a bagging fraction of 1, a bagging frequency of 50, a feature fraction of 0.6, a sub-sample of 0.8, 100 leaves, employing 2000 boosting rounds, with early stopping enforced after 200 rounds to alleviate concerns regarding over-fitting.

7.2.2.4 Artificial Neural Network

ANNs are versatile models well-known for their ability to capture complex data relationships [12]. Leveraging ANN, we uncover and exploit the nuanced, non-linear dependencies present in PVT-aware leakage and delay data.The chosen architecture, determined through experimentation and trials, consists of five hidden layers with configurations of 320x250x100x50x50 neurons. Additionally, we utilize a learning rate of 0.05, employ the ReLU activation function, utilize the ADAM solver, and limit the maximum iterations to 500. To fine-tune these hyperparameters and optimize the model's performance, we employed grid search than BO to reduce the computational overhead.

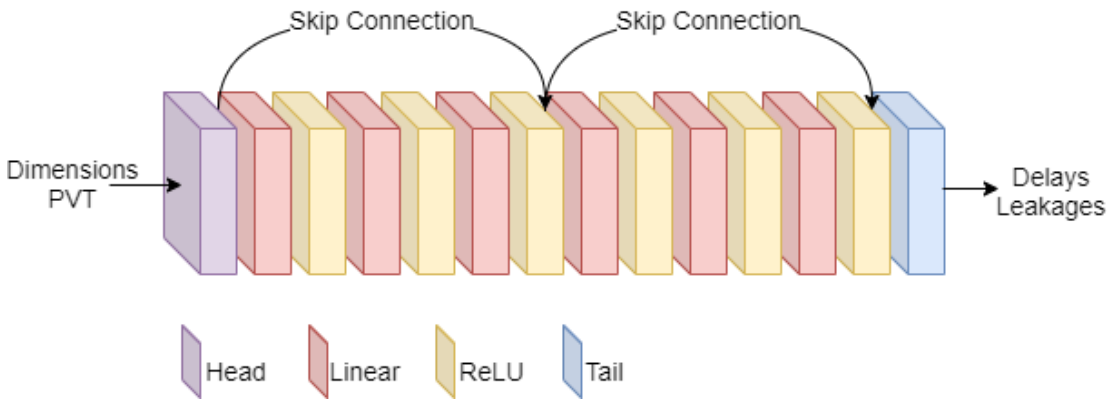


Figure 7.3: Residual neural network to model delays and leakages against variations in design and PVT parameters

7.2.2.5 Residual Neural Networks

Inspired by ResNet's architecture, engineered to mitigate vanishing gradient challenges in deep neural networks, ResNN [118] are adopted to capture subtle intricacies in PVT-aware characteristics of leakage and delay. Utilizing ResNet's residual blocks, we designate x as input parameters, f as the learned function, and y as output parameters. In ResNN, the output and gradients are derived through $y = f(x) + x$, fostering a more seamless gradient flow and augmenting learning speed and accuracy by assimilating residuals.

$$y = f(x) + x \implies \frac{dy}{dx} = \frac{df(x)}{dx} + 1$$

Here, $f(x) = y - x$ represents the function aimed at learning the residual. Our ResNN architecture consists of stacked linear and ReLU layers, with hyperparameters such as 'block size' dictating the number of layers per skip connection and 'num blocks' indicating the total skip connections. Specifically, we configure 'block size' to three, resulting in skip connections composed of three stacked Linear + ReLU blocks, and 'num blocks' to two, yielding two skip connections as shown in Fig. 7.3. The optimizer employed is ADAM with a fixed learning rate of 0.001. During training, we employ cross-validation and early-stopping techniques to mitigate the risk of over-fitting [119] and under-fitting. Additionally, we utilize grid search and BO methods to identify optimal hyperparameters, systematically ensuring superior model performance.

The performance assessment of the ML algorithms is depicted in Table 7.5 and Fig. 7.4. According to the ML metrics (R^2, $\%\mu_E$ $\%\sigma_E$, MaPE), LGBM and ResNN emerge as the top-performing algorithms, meeting the criteria E_c delineated in Algorithm 3. This observation is visually apparent in Fig. 7.4, where the predictions of LGBM and ResNN closely match the actual (SPICE) leakage and delay values across 500 randomly chosen unseen test cases. A detailed comparison of the predictions generated by the LGBM and ResNN models is presented in Fig. 7.5 over 500 random PVT scenarios. Both models exhibit strong alignment with SPICE simulations, although LGBM demonstrates a slight advantage in generalization, particularly in predicting the leakage and delay of unseen PVT scenarios (Table 7.5).

Table 7.4: LGBM model training metrics across all the PVT and Sizing-aware digital logic cells for leakage and delay

S.No.	Logic Cells	Leakage								Delay							
		Training time(sec)	**Testing time(sec)**	R^2	**MSE**	**MAE**	μ_E	σ_E	MPE_U	**Training time**	**Testing time**	R^2	**MSE**	**MAE**	μ_E	σ_E	MPE_U
1	Buffer (BUF)	30.97	0.06	1	4.01E-16	1.14E-08	0.04	0.74	3.01	11.9	0.03	1	6.85E-25	3.49E-13	0.02	0.67	1.24
2	Inverter (NOT)	28.02	0.05	1	1.05E-16	4.93E-09	0.03	0.22	2.45	12.14	0.03	1	4.99E-25	2.35E-13	0	0.22	1.09
3	2-input NAND(NAND2)	61.63	0.05	1	2.94E-16	8.92E-09	0.05	0.3	3.51	12.28	0.03	1	7.96E-25	3.50E-13	0.01	0.42	1.23
4	2-input NOR (NOR2)	64.82	0.06	1	3.62E-16	1.12E-08	0.01	0.26	3.96	16.04	0.05	1	2.02E-24	5.08E-13	0.04	0.48	1.31
5	2-input AND (AND2)	69.56	0.06	1	6.83E-16	1.53E-08	0.06	0.6	3.29	20.7	0.06	1	8.68E-25	4.32E-13	0.04	0.76	1.27
6	2-input OR (OR2)	109.83	0.07	1	7.99E-16	1.75E-08	0	0.49	3.58	18.09	0.04	1	1.05E-24	4.92E-13	0.05	0.95	1.32
7	2-input XOR (XOR2)	156.91	0.09	1	2.71E-15	3.18E-08	0.03	0.58	3.28	21.79	0.05	1	2.67E-24	6.72E-13	0.03	0.59	1.37
8	2-input XNOR (XNOR2)	145.15	0.06	0.99	3.93E-15	3.91E-08	0.02	0.72	3.27	30.65	0.06	1	1.60E-24	6.44E-13	0.05	0.99	1.4
9	3-input NAND (NAND3)	208.88	0.09	1	4.73E-16	1.19E-08	0.01	0.32	3.86	20.18	0.05	1	1.28E-24	4.93E-13	0.03	0.56	1.24
10	3-input NOR (NOR2)	237.42	0.07	1	5.37E-16	1.43E-08	0.02	0.36	3.73	14.69	0.03	1	3.22E-24	7.77E-13	0.03	0.56	1.31
11	3-input AND (AND3)	245.44	0.07	1	7.21E-16	1.57E-08	0.02	0.36	3.1	20.32	0.05	1	1.15E-24	5.28E-13	0.05	0.89	1.38
12	3-input OR (OR3)	236.92	0.08	1	9.97E-16	1.83E-08	0	0.34	3.14	21.11	0.05	1	1.89E-24	6.79E-13	0.07	1.43	1.42
13	2x1 Multiplexer (MUX21)	233.95	0.08	1	1.07E-13	1.53E-07	0.03	0.48	4.71	16.7	0.04	1	8.60E-25	4.54E-13	0.05	0.74	1.22
14	Full Adder (FA)	716.75	0.12	1	1.24E-14	6.75E-08	0.02	0.93	4.56	18.72	0.06	0.99	5.24E-24	1.00E-12	0.08	0.99	2.74
15	And-Or-Invert (AOI12)	259.15	0.11	1	9.16E-16	1.92E-08	0.03	0.28	3.14	21.89	0.05	1	1.23E-24	5.52E-13	0.05	0.95	1.37
16	Latch	167.61	0.07	1	6.31E-16	1.79E-08	0.19	0.01	0.19	20.05	0.01	0.98	1.72E-23	1.12E-12	0.11	2.27	2.07
17	D Flip-flop (DFF)	224.23	0.11	0.99	1.78E-13	3.12E-07	0.01	0.09	0.02	21.89	0.09	0.97	7.52E-22	3.98E-12	0.04	2.95	1.54

Algorithm 3 OptiMo-ML Modeling

Require: D, a PVT-aware standard cell training data
  Split D as D^{train}, $D^{Validation}$, D^{test} ▷ 60,20, 20 Split
  Define $X \leftarrow \{x_1, ...x_m\}$ $\forall$ PVT, $Y \leftarrow$ power/delay
  Training the bank of ML algorithms
  for bank of ML algorithms **do**
    Initialize the hyper-parameters, p
    $f : X \mapsto Y$ ▷ ML model M to model $Y|X$
    Train M on D^{train} using p
    Compute $E^*|M$ on $D^{Validation}$,D^{test} ▷ Test error set
    if $E_s > E_c{}^*$ **then** ▷ Test error criteria
      Hyper-parameter tuning
      Re-train M
      Repeat tuning for N iterations
      Choose the optimal hyperparameter set p^*
    end if
    save M
  end for
  Best ML algorithm $\leftarrow E_{cMin}$

* - E, E_c are a composite set of test errors, and its criteria - $R^2 > 0.95$, $\{\%\mu_E, \%\sigma_E\} \leq 1\%$ and MPE - Mean Percentage Error for 500 unseen test samples $\leq 5\%$

Consequently, LGBM is identified as the superior algorithm, and surrogate ML models utilizing it are developed for all standard cells. The performance metrics of these models are provided in Table 7.4, showcasing accurate results across all standard cell models. ML and optimization algorithms, and SPICE simulations are executed on the 12th Gen Intel(R) Core(TM) i9 processor with 32GB RAM, while the training of ResNN is conducted on a standalone GPU core featuring 4GB of VRAM, utilizing an RTX 2080 Ti graphics card. Libraries such as Scikit-learn [77] and PyTorch [120] are utilized for training ML algorithms, with Scikit-learn being employed for DT, ET, LGBM, ANN and PyTorch for ResNN.

7.2.3 Optimization Algorithms with Machine Learning for Transistor Sizing

In IC design, adhering to standard gate sizing, typically guided by Logical Effort principles, ensures a balanced distribution of pull-down and pull-up resistances, mirroring the behavior of a reference CMOS inverter unit. This conventional approach effectively addresses concerns related to drive strength and mobility. However, deviations from this standard sizing serve as a starting point for systematic exploration in optimizing transistor width and length [121], with the ultimate goal of identifying configurations that surpass the reference based on predefined metrics.

The utilization of meta-heuristic algorithms proves indispensable for navigating multidimensional search spaces during optimization tasks [122]. Traditional techniques like Direct Search or Gradient-based approaches become impractical when confronted with high-dimensional vectors [123] [124], a

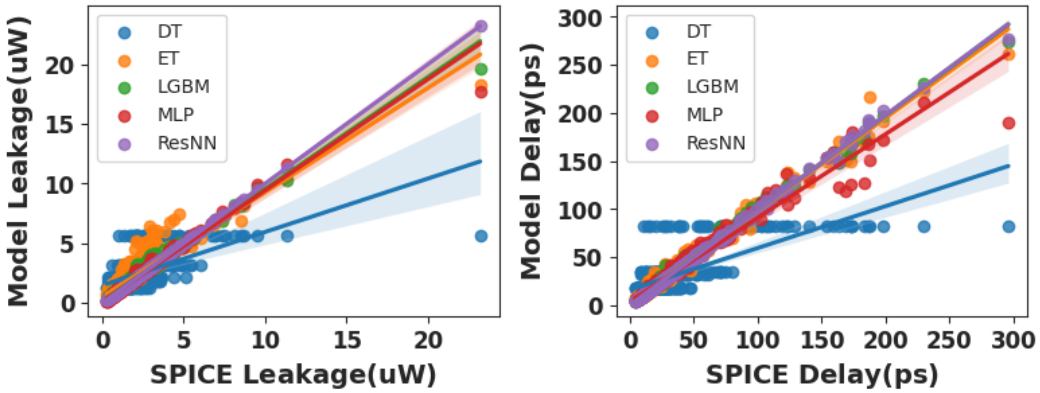


Figure 7.4: Comparison of regression models w.r.t SPICE leakage and delay values

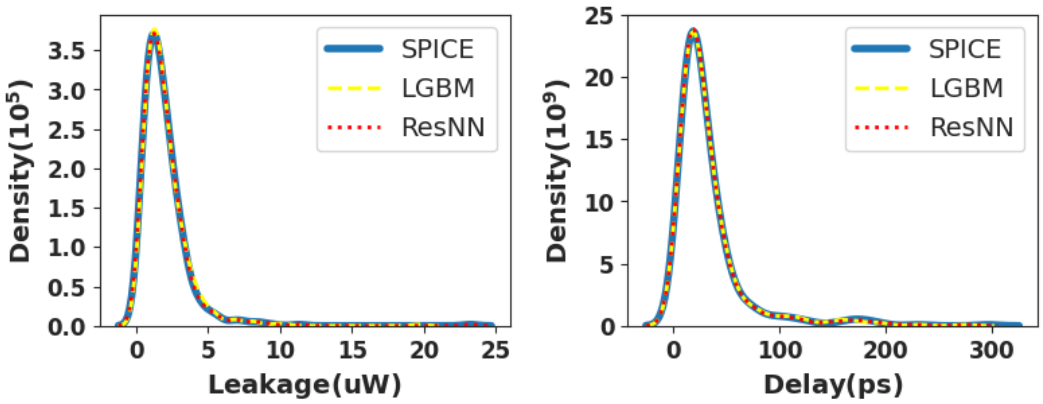


Figure 7.5: Comparison of LGBM and ResNN model leakage and delay predictions w.r.t SPICE

Table 7.5: ML model training metrics on training PVT and Sizing-aware Full adder logic cell for leakage and delay

Model/ Metrics	**Training and Validation**		**Testing**				**Unseen testcases**	
	R^2	**Runtime**	R^2	**MSE**	$\%\mu_E$	$\%\sigma_E$	**MPE**	**Runtime**
Leakage								
DT	0.572	3.77	0.571	1.4E-06	0.15	24.51	68.85	0.22
ET	0.994	1182.07	0.963	7.1E-07	0.10	4.66	29.40	0.05
LGBM	0.998	716.75	0.998	2.5E-07	0.02	0.93	4.56	0.12
ANN	0.998	516.91	0.995	2.8E-07	0.96	1.73	6.42	0.01
ResNN	0.999	869.91	0.997	8.2E-16	0.08	0.37	5.02	0.19
Delay								
DT	0.529	0.34	0.524	7.4E-07	0.11	27.56	43.30	0.00
ET	0.994	11.23	0.958	6.9E-08	0.30	5.08	10.79	0.02
LGBM	1.000	18.72	0.992	5.2E-24	0.08	0.99	2.74	0.06
ANN	0.992	210.56	0.975	9.3E-08	0.64	3.11	5.17	0.02
ResNN	0.995	192.68	0.984	6.4E-11	0.98	2.13	3.98	0.09

scenario frequently encountered in endeavors such as full adder cell optimization, where complexity escalates due to numerous transistor configurations. Furthermore, the execution of SPICE simulations for leakage and delay computations exacerbates the challenge, demanding significant computational resources and time. To illustrate, optimizing a single full adder cell encompassing 28 transistors (resulting in 58 dimensions) typically consumes around 30 to 300 minutes [125] [126] [127].

To surmount these hurdles, the employment of ML models presents a promising avenue for expediting the evaluation process, circumventing the need for intricate SPICE simulations. Nonetheless, the selection of an efficient algorithm is paramount to circumvent the risk of becoming ensnared in local optima. Nature-inspired meta-heuristic algorithms, renowned for their simplicity and adaptability to NP-hard problems, emerge as compelling solutions to these challenges. The OptiMo framework seamlessly integrates ML models with meta-heuristic optimization algorithms using a multi-output objective function. This fusion results in a remarkable acceleration of the optimization process, achieving the completion of full adder cell optimization in approximately 2 minutes, representing a significant advancement over conventional methodologies. Moreover, extending this approach to the circuit level promises even more substantial speed enhancements.

This research implements five evolutionary algorithms: GA [128], Particle Swarm Optimization (PSO) [129], Artificial Bee Colony (ABC) [130], Elephant Herd Optimization (EHO) [131], and Harris Hawk Optimization (HHO) [132], tailored for transistor sizing. These algorithms operate in conjunction with the most effective trained ML model—namely, the LGBM model—associated with each standard cell. Additionally, we comprehensively evaluate the performance of various optimization algorithms concerning the power-delay trade-off. The most effective algorithm is then chosen for optimizing complex digital circuits, with further elaboration on these methodologies provided subsequently.

Algorithm 4 Objective Function

Require: Design parameters (width, length) vector: x

$w_d \leftarrow$ delay weight vector
$w_l \leftarrow$ leakage weight vector
$l, d \leftarrow$ Calc_Leakage_Delay(x)
Compute leakage fitness: $f_l = \sum_i l_i \cdot w_{l_i}$
Compute delay fitness: $f_d = \sum_i d_i \cdot w_{d_i}$
Return (Leakage fitness f_l, Delay fitness f_d) vector
CALC_LEAKAGE_DELAY(X)
  Concatenate input: $input = [vin(i/p), x, pvt, C_{Load}]$
  Predict leakage: $leak \leftarrow Leak\_ML\_Model(input)$
  Predict delay: $delay \leftarrow Delay\_ML\_Model(input)$
  Return: $leak$, $delay$
End

7.2.3.1 Objective Function

Meta-heuristic algorithms rely on objective functions to navigate optimization landscapes, serving as mathematical guides directing searches toward solutions aligned with specific problem goals. Crafting effective objective functions necessitates careful consideration of problem attributes, computational efficiency, and desired solution structure. The choice of an appropriate objective function is crucial as it determines the algorithm's ability to efficiently discover optimal solutions, making it pivotal in optimization success. Typically represented as a real number indicating cost or fitness, the objective function assesses the quality of a candidate solution. In this context, the function is tailored to minimize a performance metric involving leakage power and propagation delay. Through iterative evaluation and prioritization based on objective values, meta-heuristic algorithms navigate the search space to converge upon solutions optimizing the objective for the sizing problem.

Algorithm 4 outlines the objective function tailored for optimizing channel length and width with power-delay trade-off using meta-heuristic algorithms. By utilizing the design parameter values(x), PVT conditions (pvt), input combinations ($vin(i/p)$) (for leakage estimation) and C_{Load} (for delay estimation), the $CALC\_LEAKAGE\_DELAY()$ function invokes trained ML models to compute leakage ($leak$) and delay ($delay$) measures. Weight assignment (w_d&w_l) is customized to meet problem specifications, with the flexibility to prioritize leakage reduction or delay reduction as required. The fitness function, aggregating the output (f_l&f_d), sums weighted values for leakage (l_i) and delay (d_i), guiding optimization algorithms toward optimal solutions. The attained solution adheres to power-delay reduction bounds under extreme and nominal PVT variations. Notably, unlike previous approaches that prioritized either leakage or delay reduction, our method employs uniform weights, ensuring a balanced reduction of both metrics across various triggers and inputs.

7.2.3.2 Optimization Algorithms

All the algorithms initiate by assessing performance parameters based on initial standard gate sizing, presuming it to be the optimal solution (Fig. 7.6). In the figure, 'a' represents the starting point for optimization is based on Logical Effort initial standard gate sizings. Simultaneous Leakage and Delay Optimization are performed at every Iteration. The plot follows the performance metric of the best fittest individual in the population search space, corresponding to each algorithm. During optimization, algorithms tend to prioritize optimization of one metric ('b' represents delay@7^{th} iteration) and ('c' represents leakage@11^{th} iteration) ignoring the other, unlike GA which can balance the trade-off better. The optimization converges at 'd' at which optimal gate sizings are extracted. During initialization within a population of 100, 99 additional members are introduced throughout the entire search space. Among these algorithms, PSO and EHO, incorporate mechanisms for members to track the best solution among peers, adjusting positions based on past performance and the current best member. In PSO, maintains personal best (pbest) and global best (gbest) positions, inspired by swarm intelligence in nature. Conversely, EHO mimics the herd behavior of elephants, with members updating positions based on the matriarch and clan leader. Upon reaching a suitable local optimum, a member becomes the new best

member, subsequently influencing the entire population. HHO implements a dynamic learning process with multiple reference points, emphasizing exploration, fine-tuning positions, and soft besiege maneuvers. ABC relies on a probabilistic dance language where employed bees communicate promising food sources to onlooker bees without explicitly revealing locations. Despite their effectiveness in navigating complex spaces, these algorithms commonly demonstrate a slight sensitivity to local optima. On the other hand, GA, employing principles of 'natural selection,' eliminates unfit members, thereby preventing the spread of undesirable traits and facilitating the efficient propagation of promising features. The 'crossover' and 'mutation' features within GA aid in exploring diverse niche regions. Convergence in GA occurs when the best gene remains unchanged for several generations.

Algorithm 5 Genetic Algorithm

Require: ℓ: Gene length, N: Population size, e: Elite count, c: Cross-count, $\mathcal{F}$: Fitness function, $\mathcal{I}$: Initial population, $\mathcal{S}$: Selection operator, $\mathcal{C}$: Crossover operator, $\mathcal{M}$: Mutation operator, $\mathcal{E}$: Elitism selection, $\mathbf{x}_0$: Gene Seed;

Ensure: $\mathbf{x}_f$: (Global Best Fitness, Best individual);

$\mathbf{x}_f \leftarrow (\mathcal{F}(\mathbf{x}_0), \mathbf{x}_0)$
$\mathcal{P} \leftarrow \mathcal{I}()$
while $convergence$ **do**
    $\mathcal{P}_{fit} \leftarrow \emptyset$
    for $i = 1, \ldots, N$ **do**
        if $\mathcal{F}_P[i] \leq \mathbf{x}_f[0]$ **then**
            $\mathcal{P}_{fit} \cup \mathcal{P}[i]$
        end if
    end for
    if $\mathcal{P}_{fit} \neq \emptyset$ **then**
        $\mathbf{x}_f \leftarrow (\min(\mathcal{F}_P), \arg\min(\mathcal{F}_P))$
        $\mathcal{P}_{select} \leftarrow \mathcal{S}(\mathcal{P}_{fit})$
        $\mathcal{P}_{cross} \leftarrow \mathcal{C}(\mathcal{P}_{select}, c)$
        $\mathcal{P}_{elite} \leftarrow \mathcal{E}(\mathcal{P}_{fit}, e)$
        $\mathcal{P} \leftarrow \mathcal{M}(\mathcal{P}_{cross}) \cup \mathcal{P}_{elite}$
        $converge\_count, c_c \leftarrow 0$
    else
        $c_c \leftarrow c_c + 1$
        $\mathcal{P}_{select} \leftarrow \mathcal{S}(\mathcal{P})$
        $\mathcal{P}_{cross} \leftarrow \mathcal{C}(\mathcal{P}_{select}, c)$
        $\mathcal{P}_{elite} \leftarrow \mathcal{E}(\mathcal{P}, e)$
        $\mathcal{P} \leftarrow \mathcal{M}(\mathcal{P}_{cross}) \cup \mathcal{P}_{elite}$
    end if
    $generation \leftarrow generation + 1$
end while

Initial experimentation is focused on the full adder standard cell due to its complex structure, consisting of 28 transistors compared to other digital cells. The primary objective was to ensure that at least one algorithm could optimize the gate under worst-case PVT scenarios. Upper bound references for

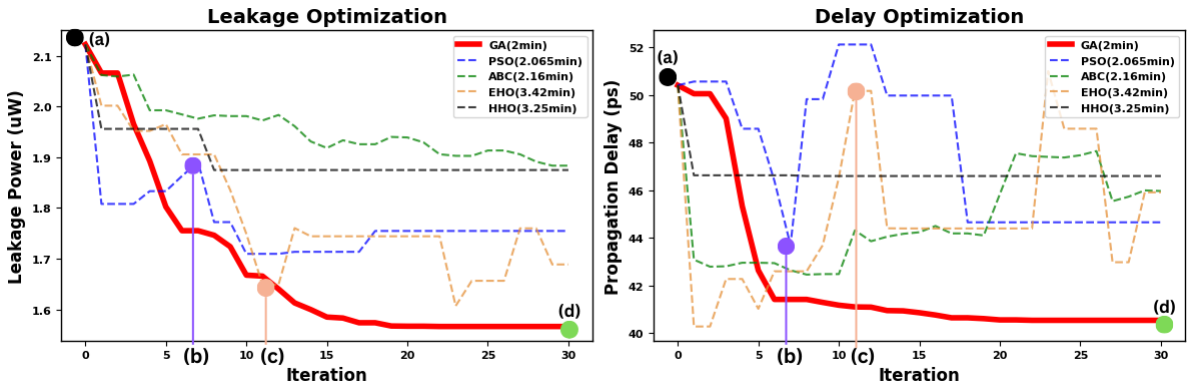


Figure 7.6: Comparison of full adder cell optimization with different evolutionary algorithms (a) Leakage (b) Delay

leakage and delay were determined based on worst-case PVT conditions, where the process and voltage were within 10% of nominal, and the temperature ranged from $-55 \circ C$ to $125 \circ C$. Evaluation of algorithms was based on their computational time and their ability to effectively balance the trade-off between leakage and delay reduction. This evaluation is visually depicted in Fig. 7.6 which illustrates the average leakage and delay measures of the best-fit individual or population member at each iteration. We can observe the aggregated convergence behavior of these algorithms in Fig. 7.6. Notably, GA exhibited superior performance, achieving faster convergence time and effectively managing the trade-off between leakage and delay reduction compared to other algorithms, as illustrated in 'b' and 'c'. In these plots, attempting to decrease one parameter led to an increase in the other. Consequently, GA was selected as the optimal algorithm for optimization and subsequently applied to all other cells.

We present the operating details of GA in Algorithm 5. To incorporate a multi-objective approach, fit genes were selectively introduced into the fit population (P_{fit}) based on their demonstrated reductions in both leakage and delay compared to the reference value. The fittest gene in the next generation becomes the new upper bound. Within the population, a mutation step was implemented, assigning random numbers within the ranges of $[-2, 2]nm$ for length (L) and $[-3, 3]nm$ for width (W). Despite the broader range of W spanning from 22 nm to 440 nm, this mutation step was deemed optimal through extensive experimentation for traversing the search space without overlooking optimal solutions or risking entrapment within sub-optimal regions. Within P_{fit}, the top 20% were classified as elite and exempted from mutation, while the remaining 80% underwent selection (S) and gene crossover (C) operations, followed by mutation. In each mutation iteration, gene values were rounded to the nearest integral value within the nanometer range, aligning with the ML model's constraint of utilizing only integral values. If the fittest gene remains unchanged, indicating a lack of improvement and resulting in an empty P_{fit}, the fit population nears convergence. These steps collectively refine the optimization process, enabling the discovery of solutions that simultaneously address both leakage and delay reduction objectives while adhering to the constraints of the ML model. In our training process, a significant accomplishment was

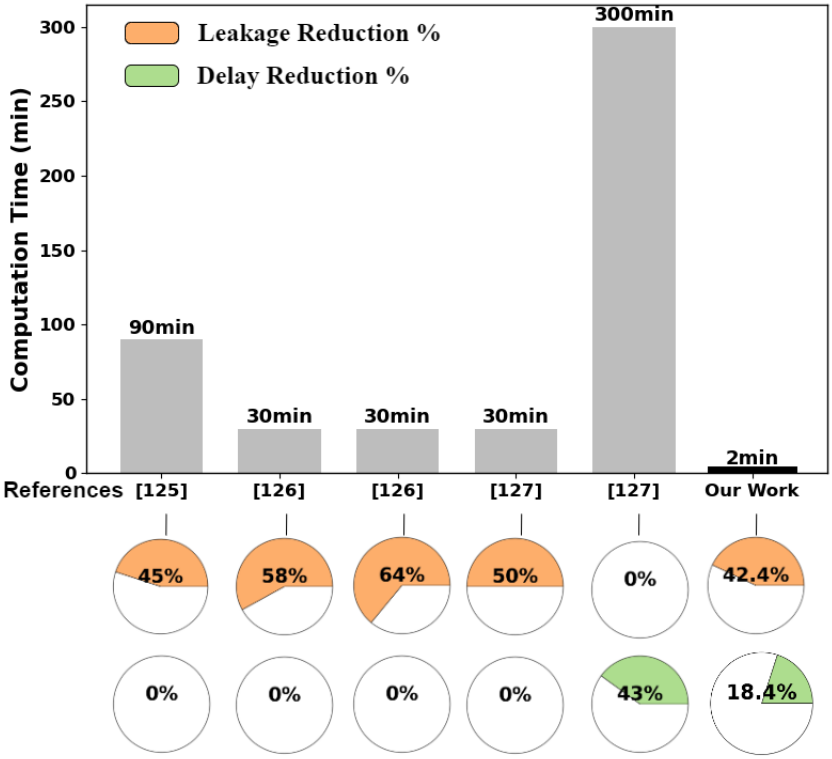


Figure 7.7: Comparison of the OptiMo with State-of-the-art works in terms of %reductions in leakage and delay and computation time

achieving nominal channel length (L) values of 22 nm after optimization, despite initially considering variations in channel length to enhance the search space. This achievement is particularly noteworthy considering the constraints associated with increasing channel length in a specific technology, which often leads to disadvantages such as increased transistor size and diminished performance metrics.

We concentrate on the dual objective of reducing both leakage and delay, recognizing the trade-off between these metrics. The algorithm is customizable, enabling adjustments to the threshold and termination criteria based on achieving desired levels of leakage or delay reduction. This flexibility accommodates various user preferences in IC design.

Following the optimization process where optimal width (W) values were derived from the best performing algorithm (GA), a comprehensive evaluation was conducted across 10,000 random PVT variations, as illustrated in Fig 7.9. This evaluation involved comparing the resulting performance metrics with the initial user-provided metric aimed to be minimized across the entire PVT range. The resulting histogram plots visually demonstrate a significant shift in numerous distributions towards lower metric values, particularly around nominal conditions (Table 7.1). Under extreme PVT conditions ($\pm 10\%$ from nominal for voltage, process, and $[-55°C, 125°C]$ for temperature), metrics deviate notably from their initial values, as observed from the tail-ends in Fig. 7.9. The reduction in metric values is proportional

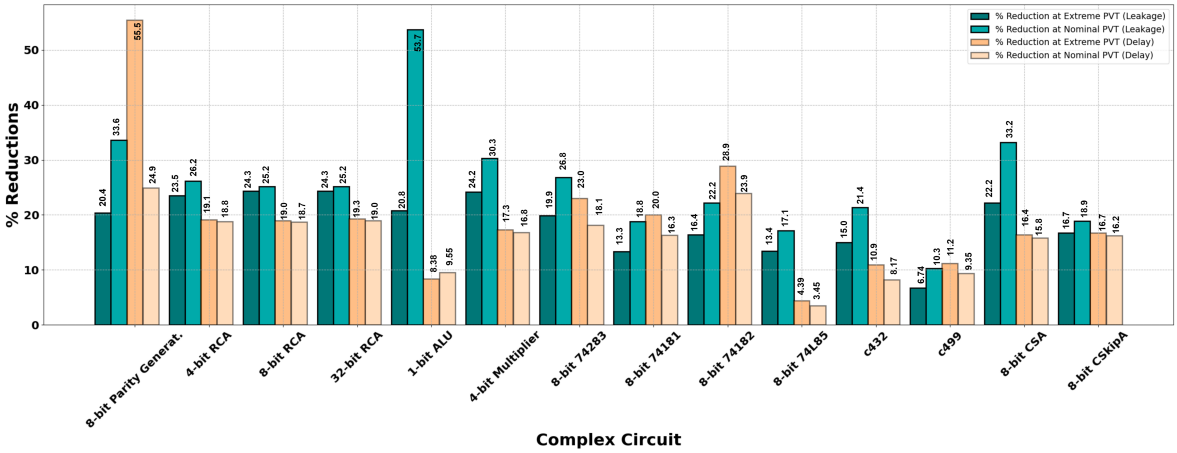


Figure 7.8: Visualization of %reduction in leakage and delay of complex Cells with optimized transistor sizing

to the number of transistors present in a standard cell due to the larger dimension of the search space provided. Consequently, compared to the FA, XNOR2 cells in Fig. 7.9, other gates exhibit relatively lower reduction values. Smaller gates (BUF, NAND, NOR) often experience reductions in either leakage or delay, with simultaneous reductions proving challenging.

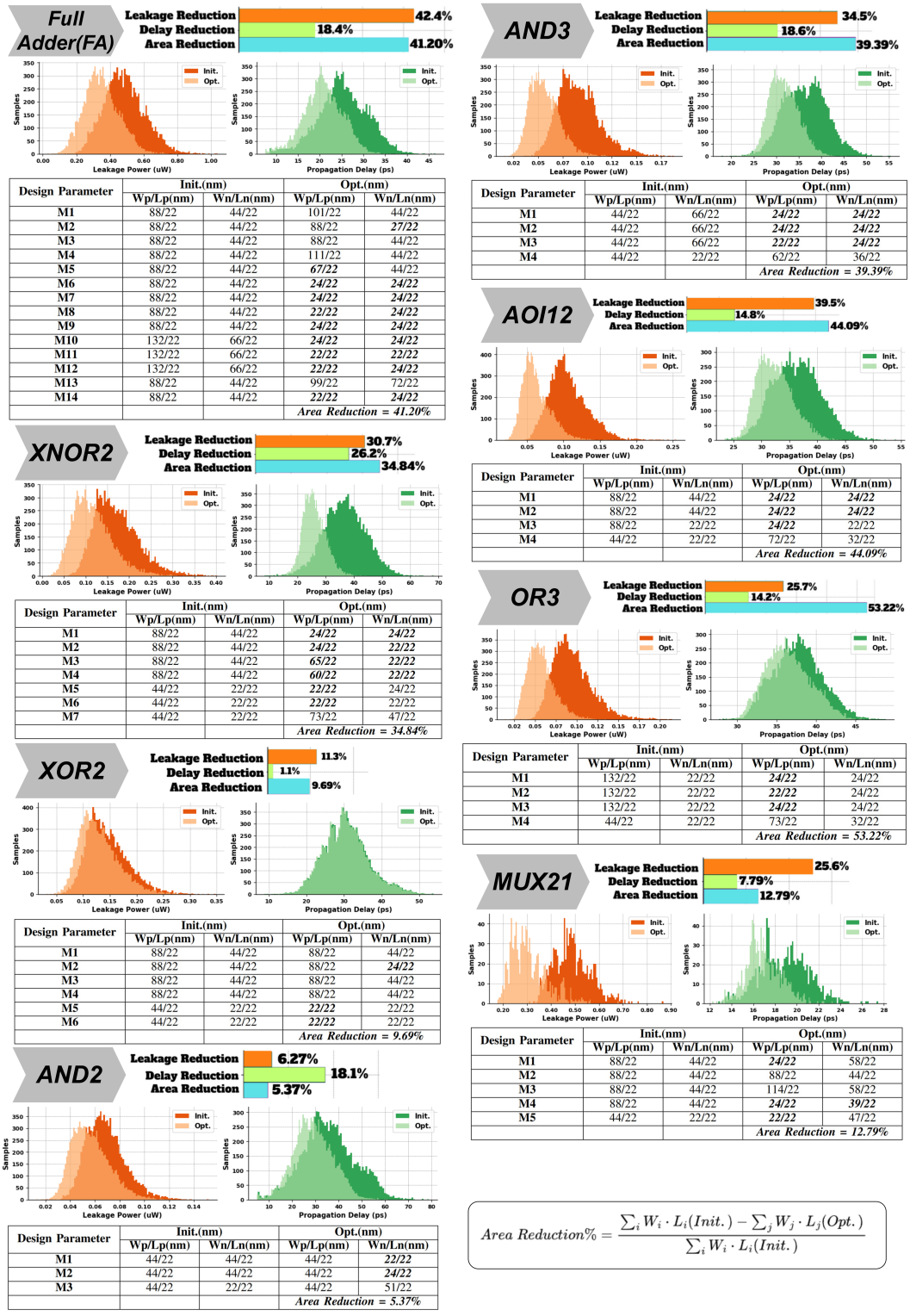
The optimized gate sizing(s) obtained, as presented in Tables within Fig. 7.9 for corresponding standard cells, serve as the foundation for constructing various complex cells within the Complex Cell Framework. This establishes a comprehensive workflow for optimizing complex cells, eliminating the necessity of tedious netlist generation and intensive simulations for every design being modeled. Consequently, our approach ensures the universality of optimization, irrespective of the specific complexity of the circuit under consideration. Moreover, our ML-model-based optimization methodology is designed to be technology-agnostic, suiting any process technology.

7.2.4 Performance Verification across PVT Variations

During the execution of the genetic algorithm, all PVT parameters were maintained at their nominal values. To verify that the determined gate dimensions were indeed PVT-aware, we compared the initial gate dimensions (estimated under nominal PVT) across 1000 iterations. In each iteration, temperature, and voltage values were randomly selected from a uniform distribution within specified ranges, while process parameter variations were sampled from a Gaussian distribution with a variance of $\pm 3\sigma$. Subsequently, we compared the delay and leakage values for the initially sized and optimized gates.

7.2.5 Results and Discussion

We have meticulously demonstrated the outcomes of digital standard cell optimization and the application of ML models in sections 7.2.3 and 7.2.2. This section presents the results obtained from evaluating the performance of complex circuits using the optimized standard cells. The sizing of standard



Design Parameter	Init.(nm)		Opt.(nm)	
	Wp/Lp(nm)	Wn/Ln(nm)	Wp/Lp(nm)	Wn/Ln(nm)
M1	88/22	44/22	101/22	44/22
M2	88/22	44/22	88/22	***27/22***
M3	88/22	44/22	88/22	44/22
M4	88/22	44/22	111/22	44/22
M5	88/22	44/22	***67/22***	44/22
M6	88/22	44/22	***24/22***	***24/22***
M7	88/22	44/22	***24/22***	***24/22***
M8	88/22	44/22	***22/22***	***24/22***
M9	88/22	44/22	***24/22***	***24/22***
M10	132/22	66/22	***24/22***	***24/22***
M11	132/22	66/22	***22/22***	***22/22***
M12	132/22	66/22	***22/22***	***24/22***
M13	88/22	44/22	99/22	72/22
M14	88/22	44/22	***22/22***	***24/22***
			Area Reduction = 41.20%	

Design Parameter	Init.(nm)		Opt.(nm)	
	Wp/Lp(nm)	Wn/Ln(nm)	Wp/Lp(nm)	Wn/Ln(nm)
M1	88/22	44/22	***24/22***	***24/22***
M2	88/22	44/22	***24/22***	***22/22***
M3	88/22	44/22	***65/22***	***22/22***
M4	88/22	44/22	***60/22***	***22/22***
M5	44/22	22/22	***22/22***	24/22
M6	44/22	22/22	***22/22***	22/22
M7	44/22	22/22	73/22	47/22
			Area Reduction = 34.84%	

Design Parameter	Init.(nm)		Opt.(nm)	
	Wp/Lp(nm)	Wn/Ln(nm)	Wp/Lp(nm)	Wn/Ln(nm)
M1	88/22	44/22	88/22	44/22
M2	88/22	44/22	88/22	***24/22***
M3	88/22	44/22	88/22	44/22
M4	88/22	44/22	88/22	44/22
M5	44/22	22/22	***22/22***	22/22
M6	44/22	22/22	***22/22***	22/22
			Area Reduction = 9.69%	

Design Parameter	Init.(nm)		Opt.(nm)	
	Wp/Lp(nm)	Wn/Ln(nm)	Wp/Lp(nm)	Wn/Ln(nm)
M1	44/22	44/22	44/22	***22/22***
M2	44/22	44/22	44/22	***24/22***
M3	44/22	22/22	44/22	51/22
			Area Reduction = 5.37%	

Design Parameter	Init.(nm)		Opt.(nm)	
	Wp/Lp(nm)	Wn/Ln(nm)	Wp/Lp(nm)	Wn/Ln(nm)
M1	44/22	66/22	***24/22***	***24/22***
M2	44/22	66/22	***24/22***	***24/22***
M3	44/22	66/22	***22/22***	***24/22***
M4	44/22	22/22	62/22	36/22
			Area Reduction = 39.39%	

Design Parameter	Init.(nm)		Opt.(nm)	
	Wp/Lp(nm)	Wn/Ln(nm)	Wp/Lp(nm)	Wn/Ln(nm)
M1	88/22	44/22	***24/22***	***24/22***
M2	88/22	44/22	***24/22***	***24/22***
M3	88/22	22/22	***24/22***	22/22
M4	44/22	22/22	72/22	32/22
			Area Reduction = 44.09%	

Design Parameter	Init.(nm)		Opt.(nm)	
	Wp/Lp(nm)	Wn/Ln(nm)	Wp/Lp(nm)	Wn/Ln(nm)
M1	132/22	22/22	***24/22***	24/22
M2	132/22	22/22	***22/22***	24/22
M3	132/22	22/22	***24/22***	24/22
M4	44/22	22/22	73/22	32/22
			Area Reduction = 53.22%	

Design Parameter	Init.(nm)		Opt.(nm)	
	Wp/Lp(nm)	Wn/Ln(nm)	Wp/Lp(nm)	Wn/Ln(nm)
M1	88/22	44/22	***24/22***	58/22
M2	88/22	44/22	88/22	44/22
M3	88/22	44/22	114/22	58/22
M4	88/22	44/22	***24/22***	***39/22***
M5	44/22	22/22	***22/22***	47/22
			Area Reduction = 12.79%	

$$Area\ Reduction\% = \frac{\sum_i W_i \cdot L_i(Init.) - \sum_j W_j \cdot L_j(Opt.)}{\sum_i W_i \cdot L_i(Init.)}$$

Figure 7.9: Leakage and delay distributions across 10000 random PVT conditions before (Init.) and after optimization (Opt.), verified with SPICE simulations. % reductions are against the initial sizing

cells, optimized through the best-performing algorithm (GA), is applied in optimizing complex circuits. These circuits undergo evaluation using our systematically devised automated procedure to generate estimations for leakage and delay based on the pre-characterized PVT-aware standard cells. Utilizing standard cells for estimating complex circuits offers the advantage of significantly reducing the extensive simulations typically required across PVT variations, as demanded by traditional estimation tools. Traditional surrogate methodologies described in the literature model complex circuit performances by depending on PVT-aware datasets, requiring the creation of such datasets for every circuit of interest. Conversely, our innovative approach advocates for the reuse of standard cell PVT-aware characterizations for estimating a wide array of complex cells. Moreover, the integration of trained ML models leads to a substantial acceleration, resulting in speed-ups of several orders. Furthermore, our methodology is automated, substantially decreasing the manual effort and time needed. This automation yields efficiency gains that surpass those achievable through traditional methods by several orders of magnitude.

Table 7.6: Optimized leakage estimations through OptiMo at extreme and nominal PVT for various complex Cells

S.No	**Complex Cell**	**Extreme PVT (10%)**			**Nominal PVT**		
		Predicted Leakage (Initial) (μW)	**Predicted Leakage (Optim.) (μW)**	**% Reduction**	**Predicted Leakage (Initial) (μW)**	**Predicted Leakage (Optim.) (μW)**	**% Reduction**
1	8-bit Parity Generator	5.38	4.28	20.44	1.10	0.73	33.63
2	4-bit RCA	7.82	5.98	23.53	1.84	1.36	26.09
3	8-bit RCA	17.89	13.55	24.26	4.23	3.16	25.30
4	32-bit RCA	71.55	54.2	24.25	16.91	12.66	25.13
5	1-bit ALU	8.62	6.82	20.88	3.86	1.79	53.63
6	4-bit Multiplier	28.83	21.87	24.14	6.06	4.22	30.36
7	8-bit 74283	17.92	14.36	19.87	3.53	2.58	26.91
8	8-bit 74181	32.32	28.02	13.30	6.31	5.13	18.70
9	8-bit 74182	11.18	9.35	16.37	1.99	1.55	22.11
10	8-bit 74L85	16.51	14.29	13.45	2.94	2.43	17.35
11	c432	53.64	45.6	14.99	10.91	8.58	21.36
12	c499	106.94	99.73	6.74	20.19	18.12	10.25
13	8-bit CSA	46.56	36.24	22.16	12.21	8.15	33.25
14	8-bit CSkipA	26.35	21.94	16.74	5.68	4.61	18.84
15	s27	10.43	6.98	33.08	8.48	5.83	31.25

Fig. 7.7 shows the comparison of the proposed methodology with Previous Works in terms of %reductions in leakage and delay and computation time. LP and High Performance (HP) specify low power and high performance respectively for sub-45nm or 22nm CMOS technology.

Table 7.7: Optimized delay estimations through OptiMo at extreme and nominal PVT for various complex Cells

S.No	Complex Cell	Extreme PVT (10%)			Nominal PVT		
		Predicted Delay (Initial) (ps)	Predicted Delay (Optim.) (ps)	% Reduction	Predicted Delay (Initial) (ps)	Predicted Delay (Optim.) (ps)	% Reduction
1	8-bit Parity Generator	27.99	12.45	55.51	28.66	21.53	24.88
2	4-bit RCA	38.71	31.33	19.06	45.37	36.84	18.80
3	8-bit RCA	79.11	64.10	18.97	92.39	75.15	18.66
4	32-bit RCA	326.39	263.32	19.32	381.28	308.92	18.98
5	1-bit ALU	19.95	18.28	8.37	24.84	22.46	9.58
6	4-bit Multiplier	68.35	56.54	17.28	80.58	67.03	16.82
7	8-bit 74283	45.63	35.16	22.95	58.72	48.07	18.14
8	8-bit 74181	79.50	63.58	20.03	96.53	80.83	16.26
9	8-bit 74182	35.40	25.15	28.95	42.63	32.46	23.86
10	8-bit 74L85	38.98	37.27	4.39	51.88	50.09	3.45
11	c432	119.16	106.14	10.93	161.26	148.08	8.17
12	c499	88.31	78.40	11.22	108.28	98.15	9.36
13	8-bit CSA	85.20	71.26	16.36	100.14	84.35	15.77
14	8-bit CSkipA	87.53	72.93	16.68	103.87	87.05	16.19
15	s27	148.88	138.68	6.85	239.11	225.61	5.65

We rigorously validated our automated framework by scrutinizing the leakage and delay of 14 distinct complex circuits, encompassing both the ISCAS-74X and ISCAS-85 series. We examined two scenarios: one with nominal PVT conditions and another with extreme PVT variations ($\pm 10\%$ from nominal). For both scenarios, we assessed the circuits using initial (W/L) sizing (pre-optimized sizing) and optimized (W/L) sizing. The results, compared against SPICE simulations, are meticulously presented in Tables 7.6 and 7.7. The consistent reductions in both leakage and delay across all cells under nominal and extreme PVT conditions vividly underscore the efficacy of our optimized transistor sizing. The results report maximum delay and leakage reduction of 55.51% and 20.44%; 24.88% and 33.63% at extreme and nominal PVT respectively in the 8-bit parity generator. The average reductions of leakage and delay across 14 complex circuits report 16.2% and 19.28%; 25.92% and 15.63% at extreme and nominal PVT respectively.

We visualized the percentage reductions in leakage and delay for both nominal and optimized PVT conditions in Fig. 7.8, which affirm the substantial impact of our approach. Moreover, we showcase the transformative effect of optimization with optimized W values for each standard cell under various PVT conditions, in Fig. 7.10 for by the 4-bit RCA and 4-bit multiplier. Notably, the optimized sizing yields reductions in leakage and delay across all 300 random PVT variations, unequivocally validating the effectiveness of our proposed OptiMo approach.

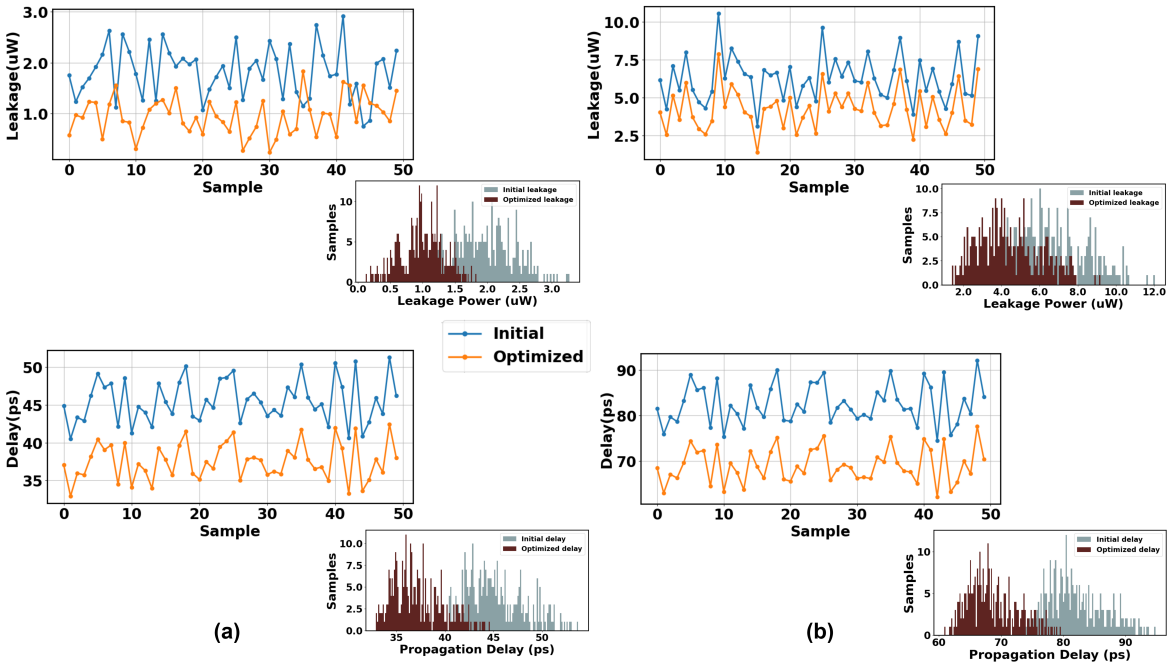


Figure 7.10: Distribution Plot for leakage and delay across random 50 PVT (Line Plot) and 300 PVT (Bar Plot) for (a) 4-bit RCA (b) 4-bit Multiplier.

7.3 Conclusion

By leveraging efficiently trained ML models of standard digital logic cells, we calculate leakage and delay estimations for various complex cells through a meticulously designed automated procedure. Furthermore, through the utilization of multi-objective and multi-directional optimization algorithms on these trained ML models, we achieve a significant enhancement in computational efficiency compared to conventional simulators, all while maintaining accuracy. Our approach intelligently selects the optimal combination of ML and optimization algorithms tailored to each circuit, ensuring optimal outcomes. Moreover, our methodology, operating as a black-box approach, is adaptable to any technology node with appropriate modeling. Experimental validation conducted on 22nm CMOS High-K technology complex digital circuits demonstrates substantial reductions of up to 55.51% in delay and 20.44% in leakage, alongside consistent area reduction. Our optimization engine, OptiMo, offers a fast, reliable, and accurate solution for a diverse range of intricate circuits through an automated approach. Application results on ISCAS-74X and ISCAS-85 benchmark circuits showcase remarkable reductions of up to 64.6% in power-delay-product (PDP).

Chapter 8

Application of Surrogate Analog Models

The unique challenges posed by process variations, temperature fluctuations, and other environmental factors make the design of analog circuits particularly complex. Conventional design methods often entail manual adjustments, extensive post-production fine-tuning, and generous design margins to account for these uncertainties. However, as technology shrinks to nano-meter scales and the demand for heightened performance grows, traditional approaches are proving increasingly inadequate and inefficient. By combining profound expertise in the analog domain with the scalability and effectiveness of machine learning, self-adaptive analog design opens up new avenues for advancement in the realm of analog electronics.

8.1 AI-driven Self-adapting Microelectronic circuits

We designed a novel Artificial Intelligence based solution to implement a wide range of microelectronic circuits that can adapt by themselves to varying usage conditions, manufacturing discrepancies, or defects such as process variations, device parameter mismatches, inaccuracies in device models, as well as environmental conditions (e.g., voltage and temperature) to mitigate their impact on the circuit performance characteristics [133]. This involves utilizing an ML model that encapsulates the microelectronic circuit's response to alterations in operational conditions. Derived from simulation-generated data, this ML model deduces appropriate internal modifications to circuit components (e.g., sizes, functionalities, connections) to counteract the effects on circuit performance characteristics stemming from condition changes. The self-adaptive microelectronic circuit is equipped with supplementary hardware, including sensors for process, voltage, and temperature, to monitor environmental conditions. It also incorporates control circuits to execute adjustments, as the ML model advises.

Fig. 8.1 provides an illustrative representation of the self-adapting approach, employing a block diagram featuring multiple components and their implementation. This methodology encompasses the initial step of training a ML model offline, utilizing statistical characterization data from off-chip analog/mixed-signal/RF circuits. This training covers a range of operating and environmental conditions following application specifications. Subsequently, the ML model can predict circuit performances for any parameter variations (PVT). In simpler setups, the ML model that prototypes the microelectronic

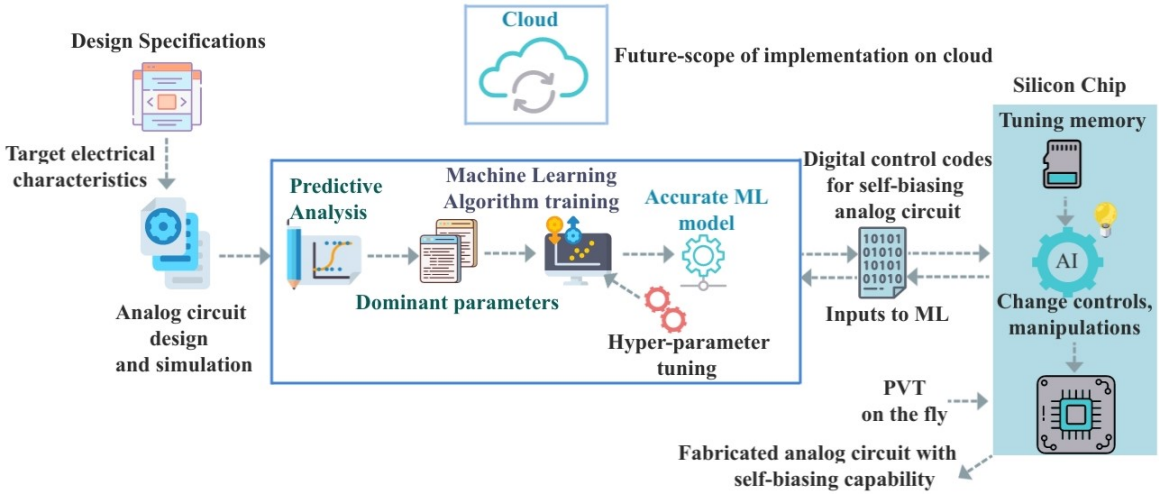


Figure 8.1: Block diagram of Self-adapting microelectronic circuit design

circuit is stored in the local memory of the CPU. However, in more complex implementations involving extensive simulation data and training, the data is stored in a remote configuration, and the trained ML model is stored in the cloud. Computational tasks may also be executed in the cloud if necessary.

The microelectronic circuit receives real-time electrical signals from the PVT sensors, which are integrated during the design phase as an integral part of the chip. These sensed PVT conditions are then relayed to the trained ML model to obtain corresponding circuit performance data. By comparing the ML-predicted circuit performances with the desired specifications stored in the on-chip memory, AI generates change control codes. Consequently, the self-adapting microelectronic chip generates an appropriate output signal through the change control register driven by the co-located microcontroller, effecting alterations to the functional section of the circuit. These functional modifications can span from straightforward adjustments like device size changes to more complex additions such as gates, data bits, or supplementary functions like amplifier stages and current sources. The core microelectronic circuit may be simple, while the adoptable add-on circuitry makes the desired adaptations based on the current operating conditions.

A repository of add-ons or a collection of adaptable base circuits can be established to streamline and generalize the self-adapting process. This AI-powered self-adapting methodology thereby guarantees that intricate microelectronic circuits autonomously adjust to variations in environmental conditions while upholding their originally specified functionality.

The benefits of self-adapting microelectronic circuit design are outlined below:

- Introduces a novel design approach for microelectronic circuits utilizing machine Learning models based on electronic signals, usage patterns, and environmental conditions.

- Enables microelectronic circuits to autonomously adjust to usage conditions while preserving their initially specified functionality.

- Employs an artificial intelligence engine, essentially an inference engine, to deduce necessary modifications to the microelectronic circuits based on specific inputs.
- Attains significantly more precise circuit specifications across various operating conditions.
- Offers the potential for extending this design methodology to encompass all types of complex functions (such as memories, processor cores, etc.), subsystems, and system-on-chip configurations.

The concept of self-adaptive design is demonstrated by modeling fundamental analog components, as outlined in Chapter 4. These circuits find widespread application and are selected as illustrative examples to underscore the effectiveness of our approach. In our setup, integrating change control switches within the analog design is pivotal in enabling the circuit to self-adapt, receiving pertinent inputs or change control codes from the AI engine. While incorporating the control circuit may introduce additional complexity, it's crucial to recognize that, to achieve comparable precision specifications under PVT variations, a traditional analog circuit would necessitate intricate designs and supplementary calibration circuitry, sometimes even external components off-chip. These alternatives would lead to a larger chip footprint and increased cost implications.

8.2 Comparison with the State-of-the-art Works

In the literature, techniques for addressing variations are categorized based on the nature of variations considered [61] as follows:

1. *Self-calibration:* These methods aim to compensate for static variations exclusively, such as manufacturing discrepancies or defects.
2. *Self-healing:* Capable of addressing both static and quasi-static variations, self-healing techniques tackle challenges such as aging or stress-induced variations.
3. *Self-adaptation:* These approaches offer the broadest scope, addressing static, quasi-static, and dynamic variations. They excel at handling dynamic operating conditions and channel impairments, providing robust compensation across a wide range of scenarios.

An on-chip self-calibrating die using ML is proposed in [62]. A LNA was designed and fabricated in IBM's 130 nm radio frequency CMOS process, and a training set for ML modeling was generated after fabrication with a substantial sample count of 444,528 samples, which is time-consuming. [63] presents a one-shot off-chip self-calibration method that identifies the best combination of tuning knobs with a unique test iteration using ML, including both process and tuning variations. The technique was demonstrated on a 65nm radio frequency power Amplifier. Since the calibration is post-fabrication, test pattern generation would take more time. The calibration approach may fail in the presence of defects

within the circuit as they use non-intrusive sensors. [134] proposes an analog/radio-frequency interference cancellation technique that can autonomously adapt itself to time-varying interference channels using a Least-mean square adaptive loop. This technique has been prototyped on a 0.5-to-2.5-GHz full-duplex multi-input multi-output receiver, designed and fabricated in a 65nm CMOS process. Although the above technique has advantages over ML optimization techniques in terms of time and power, it is limited to a specific architecture. These works are confined to self-calibration, dealing with process variation alone.

Our methodology presents a novel self-adaptation approach considering dynamic operating conditions across a wide range of operating voltages, temperatures, and processes, aiming for minimal implementation cost and power requirements. We design fast and accurate ML models with as few as 2000 training samples and employ them in our adaptation methodology. Through self-adaptation, we enhance the performance of analog circuits and adjust circuit variations to typical values within seconds. Moreover, our methodology does not introduce additional pins for adaptation, which is a necessity in traditional self-calibration techniques [61].

8.3 DUT 1: Current Reference Generator

In our implementation (depicted in Fig. 8.3), the current-defining resistor serves as a key component for controlling changes. It consists of a series of unit resistor cells, resembling an n-bit resistor chain commonly found in digital-to-analog converters. Resistor variations contribute to approximately 90% of process variations, underscoring the importance of process monitors for passive resistance tracking. Two ring oscillators illustrated in Figs. 8.2(a) and 8.2(b) represent PMON frequencies f1 and f2, respectively. While f1 is influenced by both nMOS and pMOS process corners, f2 depends on resistance, nMOS, and pMOS process corners, exhibiting higher sensitivity towards resistance. The chosen resistance values are specifically tailored to amplify resistance sensitivity and reduce sensitivity to nMOS and pMOS variations in f2. Furthermore, the disparity between f2 and f1 helps offset the impact of other variations, resulting in a resistance-exclusive process.

During the self-adaptation process, the ML model (specifically LGBM) utilizes input data from sensors monitoring PVT conditions to predict the relevant output specification, such as current. The resulting output is then compared with the typical (desired) specification to measure the deviation. Subsequently, an appropriate code is generated to nullify this deviation, which is transmitted via a LUT method by the AI engine to the control circuitry. In this process, digital code activates the necessary switches for resistance adjustment. Consequently, the performance of the current reference (specifically the current) affected by PVT is corrected to its typical specifications through a self-adaptive process, as illustrated in Fig. 8.3. The enhancements achieved in various specifications with our self-adaptation methodology compared to traditional designs are summarized in Table 8.1. The 50% spread of current due to PVT is reduced to 3%, accuracy is improved by a factor of 21, line sensitivity is 16 times lower, PSSR is better by 23dB, and power consumption is also reduced by 2.5 times.

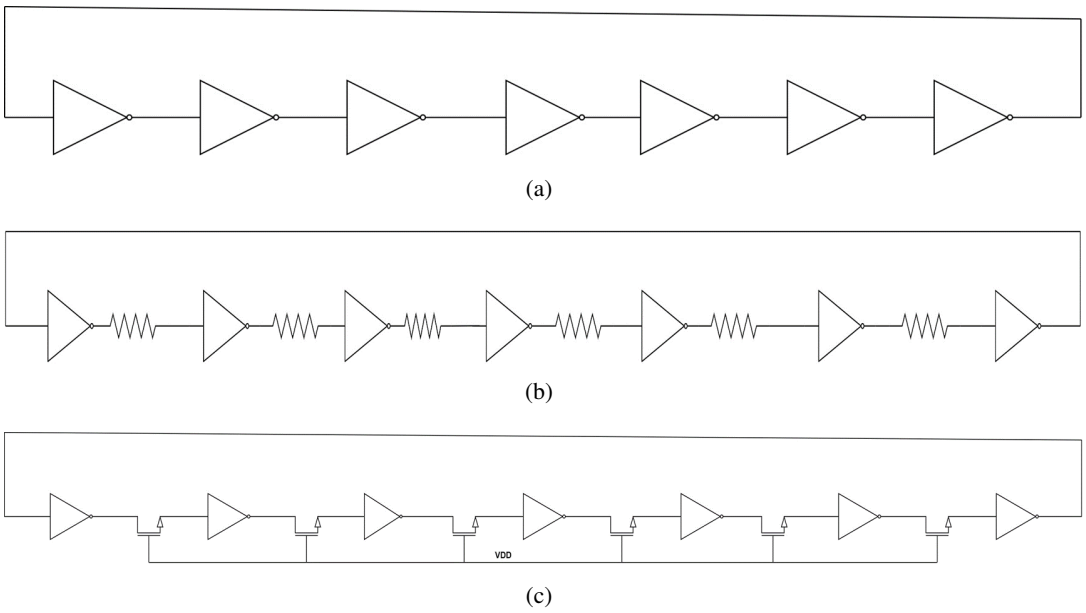


Figure 8.2: Process monitor design (a) Ring oscillator (b) Ring oscillator with resistance (c) Ring oscillator with nMOS

Table 8.1: Current reference performance improvement with Self-adaptation

Specifications	**Traditional design**			**With Self-adaptation**			**Improvements**
	Min	**Typ**	**Max**	**Min**	**Typ**	**Max**	
I_{ref} **(uA)**	8	10	13	9.8	10	10.1	50% spread reduced to 2-3%
Accuracy (ppm)	6.13	10.25	221.7	10.1	10.25	10.24	21x improvement
Line Sensitivity (%/V)	0.06	0.07	0.16	0.01	0.01	0.01	16x reduction
PSRR (dB)	-68	-91	-93	-90	-91	-91	Better by 23dB
Power (uW)	18.82	54.2	140.2	54	55	55	2.5x reduction

8.4 DUT 2: Low Dropout Regulator

The second DUT corresponds to the modeled LDO detailed in Chapter 4. In the LDO circuit, the pass transistor assumes a critical role as the primary component. Approximately 90% of process variations originate from the resistor and power FET, necessitating process monitors for nMOS and passive resistance tracking. As illustrated in Fig 8.4, numerous pass transistor multipliers are interconnected with series switches, controlled by codes derived from a control register through machine learning models. In this context, three PMON frequencies are under consideration. While frequencies f1 and f2 align with those mentioned for DUT 1, frequency f3 exhibits a stronger dependence on nMOS characteristics. Resistance values are strategically chosen to heighten resistance sensitivity and diminish nMOS and pMOS sensitivity in f2. Similarly, for f3, nMOS aspect ratio is adjusted to augment its sensitivity and reduce

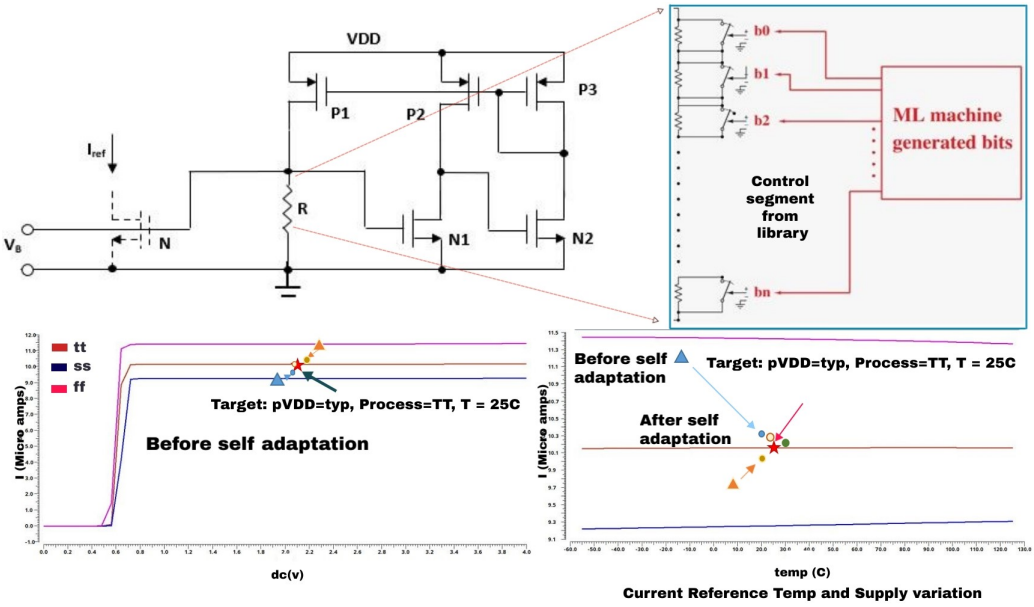


Figure 8.3: Low voltage precise PVT tolerant current reference generator in 180nm process

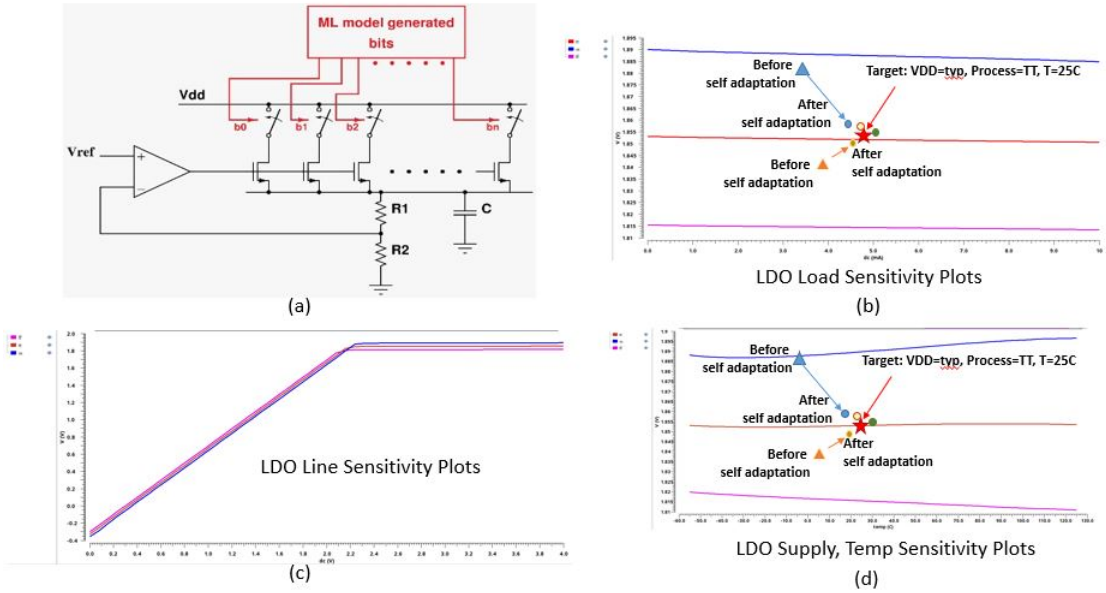


Figure 8.4: Circuit implementation of LDO with ML adaptive control and results

pMOS sensitivity. Ultimately, the differences between f2, f1, and f3, f1 further counteract the influence of other variations, resulting in process monitors exclusive to resistance and nMOS characteristics.

Similar to DUT 1, the voltage regulator's specifications are significantly improved through the self-adaptation process. Table 8.2 outlines the traditional design specifications of the LDO, along with the enhancements achieved through self-adaptation. The sensitivity plots depicting the status of the specifications before and after adaptation are illustrated in Fig. 8.4.

Table 8.2: LDO performance improvement with Self-adaptation

Specifications	Traditional design			Improvements with Self-adaptation
	Min	Typ	Max	
V_{out} (V)	1.6	1.853	2.145	Spread reduced by 50%
Temperature Stability (ppm)	4.6	15.2	105	200x lowered
Line Sensitivity (%/V)	0.005	0.0045	0.103	18x lowered
PSRR (dB)	-44.6	-57.4	-71.4	Better by 23dB
Power (mW)	0.28	0.62	1.5	Worst case reduced by 2.4x
Load Sensitivity	0.008	0.013	0.095	Maintains the same
Phase margin (○)	59	75	85	Maintains the same

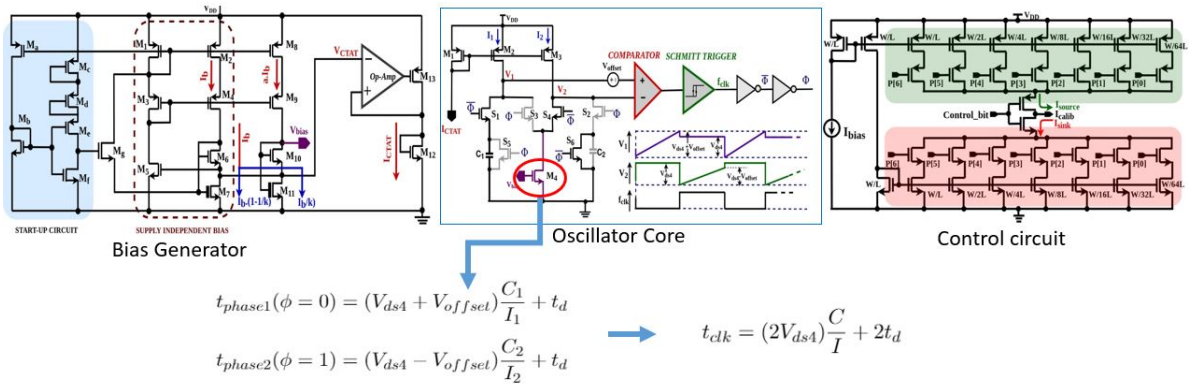


Figure 8.5: A free-running oscillator circuit with dominant variation sources

8.5 DUT 3: A Free-running Oscillator

In the free-running RC oscillator detailed in Chapter 4, the oscillation frequency is primarily influenced by the active resistor M4 and capacitor C1 (as shown in Fig. 8.5). Consequently, the selected PMONs are designed to monitor the properties of these dominant components. The control circuit, employing parallel current sources (pMOS) and sinks (nMOS), injects or withdraws trimming current to the core as necessary, guided by ML-generated codes obtained during the self-adaptation process. Throughout PVT variations, , the performance specifications of the oscillator have been finely adjusted to closely match its typical specification, leveraging inferred control codes from an ML-based model. The substantial enhancements made to the oscillator's performance metrics are summarized in Table 8.3.

Table 8.3: Oscillator performance improvement with Self-adaptation

Specifications	Traditional design			Improvements with Self-adaptation
	Min	Typ	Max	
Frequency (kHz)	31.6	43.3	70.5	80% spread reduced by 2%
Temperature coefficient (ppm/C)	1900	2037	2360	4.6x lowered
Line sensitivity (kHz/V)	3	10	14	20x lowered
Power (mW)	0.25	0.268	0.28	Maintains the same

8.6 DUT 4: Phase Locked Loop - Kvco

The methodology for modeling PLL-Kvco using classification to identify the operating corner is detailed in section 4.4.2.6. Following the identification of the operating corner via our precise classifier (SVM), an automated control routine accesses the control codes through a designed LUT approach for tuning the PLL via control switches. The control circuitry design illustrated in Fig. 8.6 comprises a series of pass transistor switches that attempt to adjust the size of the transistor controlling the control voltage input (Vctrl) to bring the Vctrl value back near to its typical value. By varying the voltage applied to the Vctrl input, we generate control switch codes from different operating temperatures and corners and store them as an LUT, from which the automated routine post-classification provides the control codes. In the FF corner, typically, we reduce the transistor size so that 1 GHz (operating frequency) can generate the necessary Vctrl to tune the circuit back to its typical state, whereas we increase the transistor size in the case of the SS corner. The self-adaptation results are depicted in Fig. 8.7. Fig. 8.7(a) illustrates a scenario with a fixed Vctrl node voltage of 610 mV with frequency variation of 1 GHz (from 3.5 GHz to 4.45 GHz) and Kvco variation from 5.8 GHz/V to 7.3 GHz/V for temperatures ranging from -40$^{\circ}C$ to 125$^{\circ}C$. With the self-adaptation process described earlier, the same conditions are improved, as shown in Fig. 8.7(b). Now, the frequency varies only by 400 MHz (from 3.8 GHz to 4.28 GHz) and Kvco varies from 6.3 GHz/V to 6.99 GHz/V across the same temperature range.

8.7 DUT 5: Low Noise Amplifier

In this particular investigation, we utilize ML-driven self-adjusting biasing methods to dynamically regulate the biasing conditions of the LNA based on real-time temperature readings. The designed LNA achieves a gain exceeding 14 dB, a noise figure below 3.21 dB, and a linearity of -13 dB for the specified frequency of 1 GHz. Illustrated in Fig. 8.8, the LNA circuit incorporates additional control circuitry for tuning the MOSFET at the common-source amplifier. This configuration offers the advantage of adjusting the gain with minimal effort and without impacting other parameters. The control circuit is equipped with a 6-bit tuning mechanism for the MOSFET multiplier.

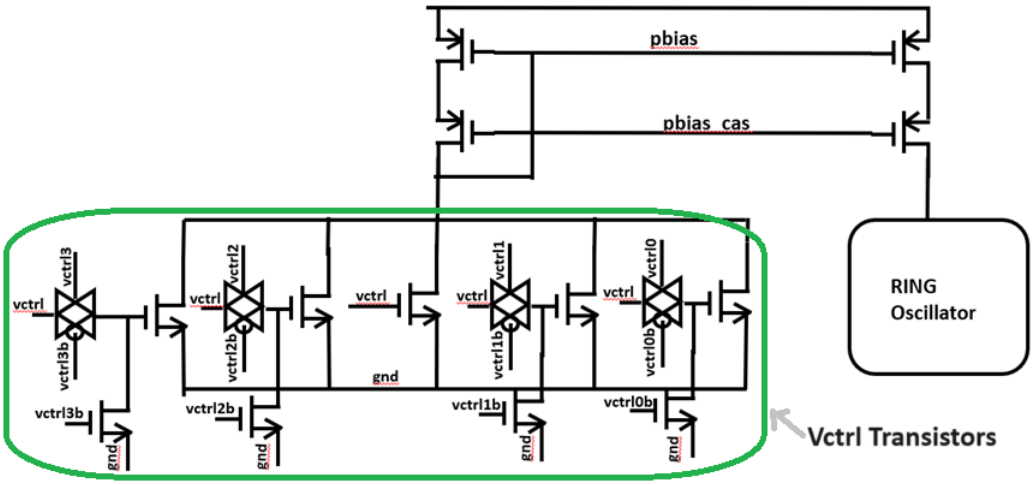


Figure 8.6: PLL control circuitry

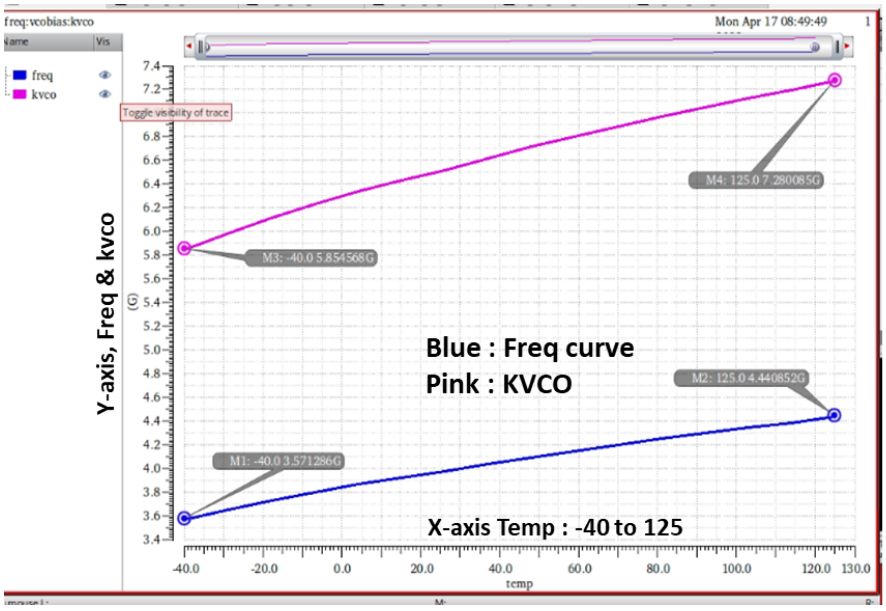
Here, we utilize two regression models to facilitate dynamic self-adaptation, which offers flexibility not constrained by predefined conditions, as seen in LUT-based approaches. Initially, we employ a trained MPR with a degree of 3 to forecast gain across varying PVT conditions, serving as our first model. Subsequently, the second regression model is trained using PVT variations and their corresponding gain values as inputs, aiming to capture the changing finger values (the decimal representations of digital control codes). This second regression model is then employed to gauge the impact of PVT in conjunction with gain, allowing us to estimate the requisite control codes for adjusting the circuit's performance to typical levels through the control circuitry. The effectiveness of the first regression model is illustrated in the LNA section of Chapter 4.

The performance of the second regression model is depicted in Figs. 8.10 and 8.11. Among the three algorithms—LGBM, XGBM, and MPR—used for training fingers over PVT and gain, LGBM exhibits superior performance, boasting an R^2 value of 0.99 and an MaPE of less than 1%. The training duration of the LGBM model amounts to approximately 2 minutes, involving 1953 training samples. Our ML-based self-adaptation method has successfully reduced the PVT spread of gain from $\pm$ 20

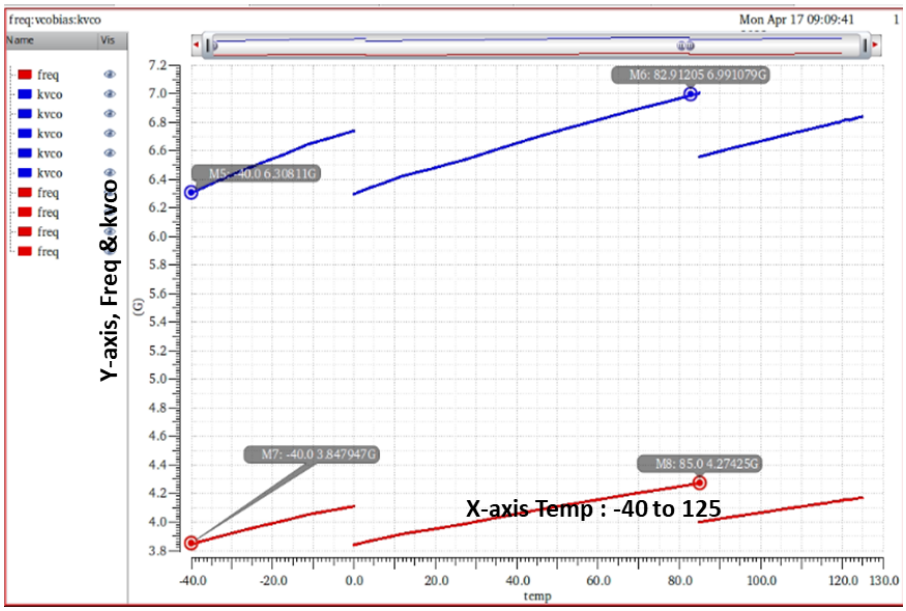
8.8 DUT 6: High-Speed Low Noise Amplifier

The high-speed LNA, as depicted in Fig. 4.31, is designed to operate at 25GHz in a 28nm technology node, with specifications outlined in Table 8.4. This table also showcases the enhancements in performance achieved through self-adaptation. Operating at a nominal supply voltage of 0.9V, the design consumes 7.8mW of power. The design ensures that both gain and noise figure are tuned simultaneously, ensuring a third-order intercept point greater than -20dB. Fig. 8.9 illustrates the high-speed LNA circuit alongside the control circuitry.

Similar to the previous section on the LNA, this case also involves the utilization of two regression models. The ML training outcomes for gain and noise figure are visualized in Fig. 8.12. The final model,



(a)



(b)

Figure 8.7: PLL performances with temperature (a) Before self-adaptation (c) After self-adaptation

LGBM, is deployed for prediction. It takes PVT, gain, and noise margin as inputs and forecasts the fingers as outputs. These finger values are then converted into their binary equivalents to be applied to the control circuitry as digital control codes for circuit tuning. The comparison between pre-adaptation and post-adaptation gain and noise figure values across varying PVT conditions is illustrated in Fig. 8.13.

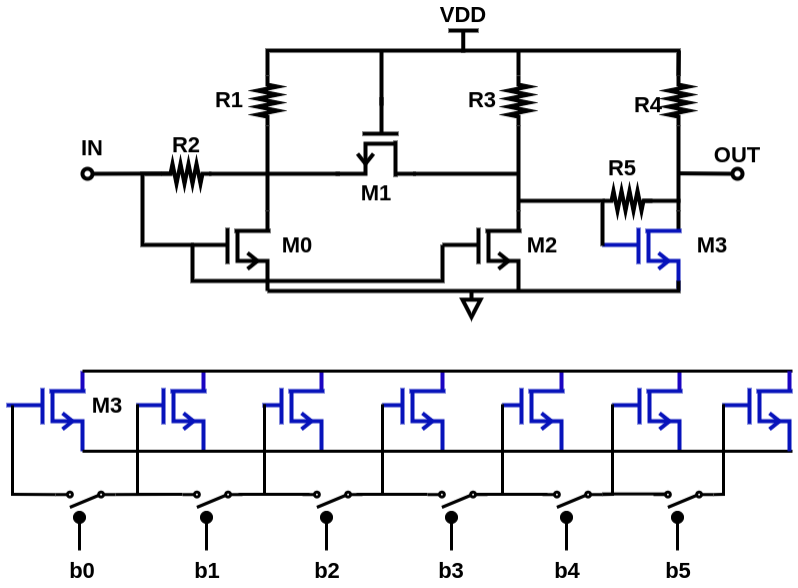


Figure 8.8: LNA circuit with self-adaptation control circuitry

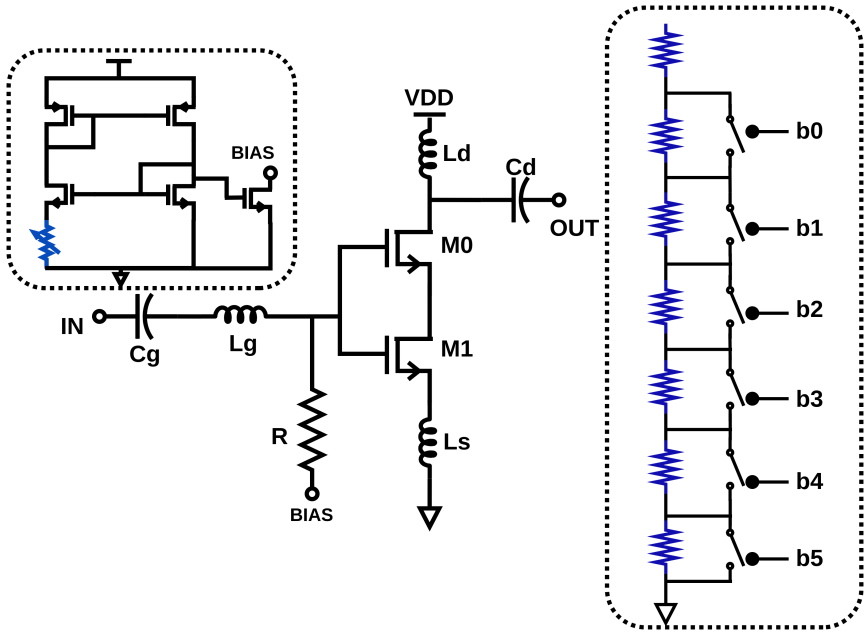


Figure 8.9: High-speed LNA circuit with self-adaptation control circuitry

8.9 DUT 7: Mixer

The Mixer, designed in a 65nm technology node with a 900MHz input frequency, features control circuitry for self-adaptation, as depicted in Fig. 8.14. Similar to the LNA circuits, the second regression model, which takes as inputs the predicted outputs from the first regression models along with PVT data

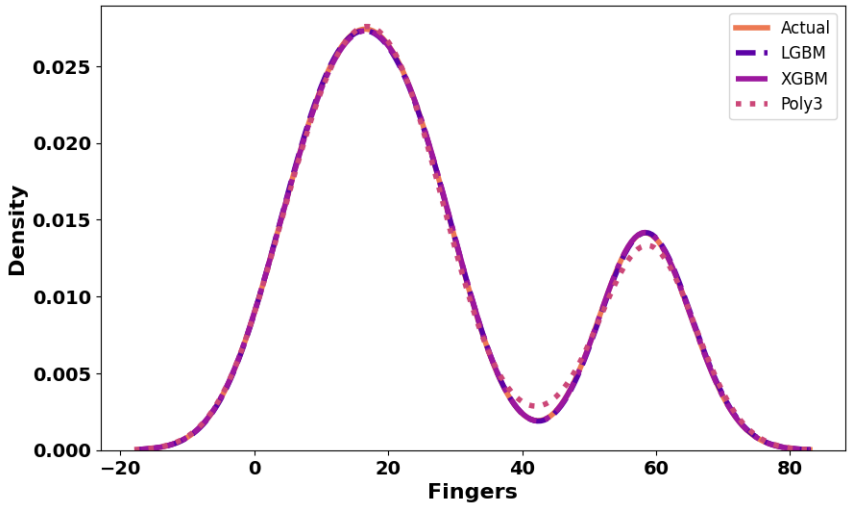


Figure 8.10: Comparison of ML models' predicted fingers (decimal control codes) of LNA Vs actual values

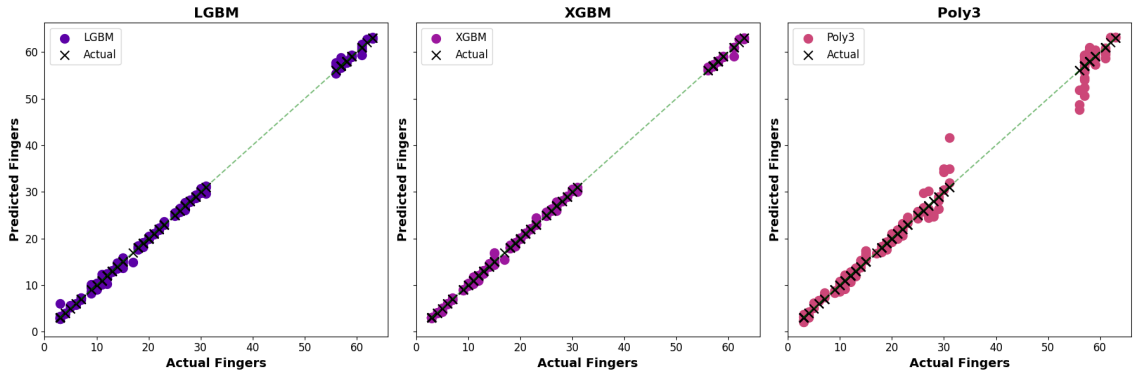


Figure 8.11: Scatter plot of actual and ML models' predicted fingers of LNA

Table 8.4: High-speed LNA performance improvement with Self-adaptation

Specifications	**Traditional design**			**Improvements with Self-adaptation**
	Min	**Typ**	**Max**	
Gain (dB)	15.443	18	19.542	21% spread reduced by 7.8%
Noise figure (dB)	1.015	3	2.484	59% spread reduced by 25%
Third order intercept point (dBm)	-19.23	-15	-8.005	$<$ -20

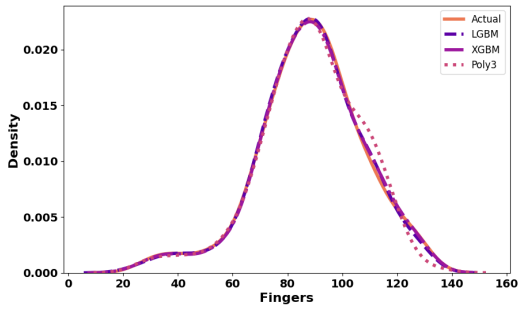
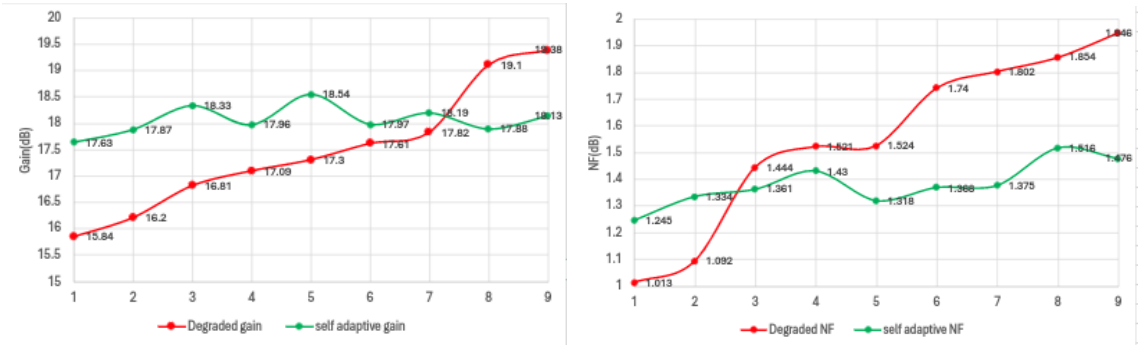


Figure 8.12: Comparison of ML models' predicted fingers (decimal control codes) of high-speed LNA Vs actual values



(a) (b)

Figure 8.13: Pre-adaptation and Post-adaptation values over different PVT conditions (a) Gain (dB) (b) Noise Figure (dB)

(as shown in Chapter 4), forecasts the decimal control codes (fingers). The performance comparison after training is visualized in Fig. 8.15.

The design boasts a third-order intercept point (IIP3) of -3.263dBm and an input-referred 1dB compression point (P1dB) of -13.18dBm. Its variation across PVT corners was noted to be between -2.2dBm to 8dBm for IIP3 and -19.28dBm to -12.56dBm for P1dB. Post-adaptation, the variation is constrained to -3.66dBm to -3.33dBm for IIP3 and -14.38dBm to -11.33dBm for P1dB.

8.10 Conclusion

As evidenced by the various analog designs tested, our self-adaptation methodology has effectively fine-tuned performance attributes to closely match their typical specifications across diverse PVT conditions, utilizing inferred control codes derived from trained ML models. We have demonstrated two approaches: Firstly, the LUT approach utilizes fixed conditions to cover all operating conditions and processes, accessed through an automated routine following classification model predictions. Secondly, a dynamic approach employs a regression model to predict the decimal equivalent of control codes,

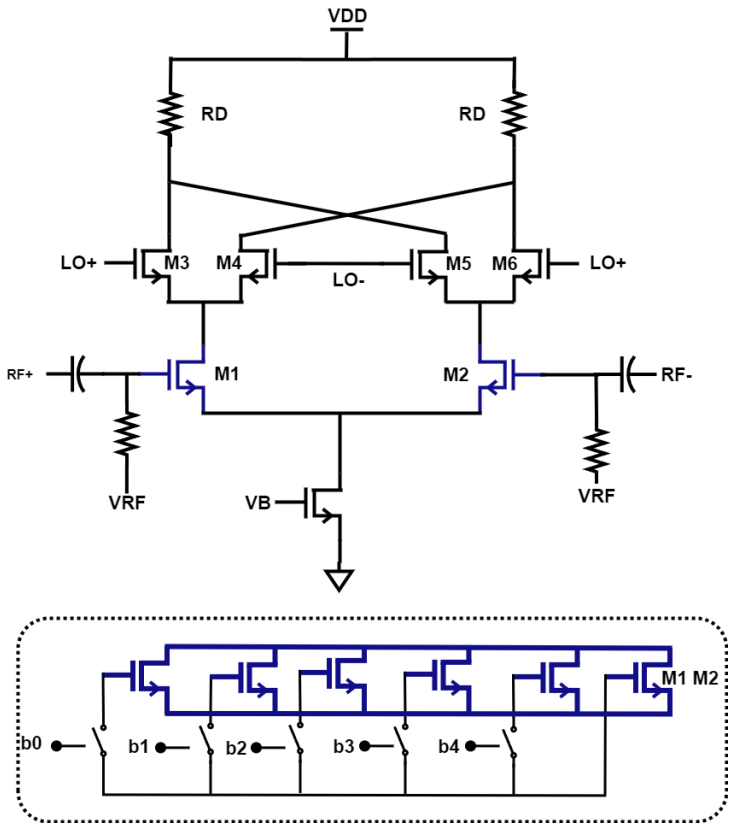


Figure 8.14: Mixer circuit with self-adaptation control circuitry

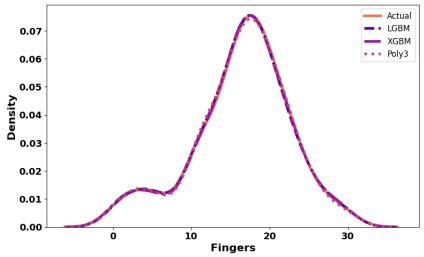


Figure 8.15: Comparison of ML models' predicted fingers (decimal control codes) of Mixer Vs actual values

subsequently converted into binary for application to digital switches. This dynamic method effectively spans all operating regions of the analog circuits. Both approaches yield highly accurate circuit tuning to typical conditions within a very short computation time (within few seconds). The precision of ML modeling and predictions, tailored for specific PVT conditions, underlies these accurate outcomes. It is crucial to underscore that generating the dataset for ML model training and determining dominant components for an analog functional circuit is a one-time endeavor integrated into the self-adaptive design library. Furthermore, self-adaptation markedly enhances circuit performance compared to traditional methodologies, improving it by several orders of magnitude.

Chapter 9

Conclusion

The thorough analysis of the impact of design, process, and environmental variations on digital and analog VLSI circuits reveals their vulnerability to unpredictable variations during fabrication processes, temperature fluctuations, and bias conditions. The emergence of PVT variations can lead to significant performance degradation, sometimes making the circuit non-functional. This underscores the need for comprehensive worst-case analysis before fabrication and the implementation of appropriate countermeasures during the design phase. Conventional mathematical modeling and simulation tools struggle to address all scenarios and temporal changes, resulting in prolonged design cycles, delayed time-to-market, and increased production costs. Moreover, the heavy dependence on designers' intuition, skills, and expertise, coupled with a lack of formalization, hampers the dissemination and reuse of knowledge in this domain.

In response, our research introduces automated circuit modeling, representing a paradigm shift that accelerates design and development processes while enhancing performance. We develop a rapid and efficient surrogate modeling framework for accurately estimating PVT-aware circuit performance in VLSI circuits. This framework, versatile and platform-independent, leverages AI/ML algorithms to create surrogate models capable of accurately capturing the effects of design, process, and environmental variations across multiple technology nodes. Through meticulous training dataset generation, incorporating varied PVT and design parameter variations, the regression and classification algorithms in framework builds application-specific surrogate models considering the effects of design, process, and environmental variations across multiple technology nodes. Digital circuit modeling encompasses high-performance CMOS technology nodes, namely 45nm, 32nm, 22nm, and 16nm, and FinFET technology nodes, such as 16nm, 10nm, and 7nm. Analog, mixed-signal, and radio-frequency circuit modeling ranges from 180nm to 65nm, then progresses to 28nm designs. Through our composite test criteria, we determined LGBM algorithm as the best performer, both in the case of regression and classification in modeling digital and AMS circuits. Polynomial regression, dense neural networks for regression and support vector machine for classification are the other notable well-trained models. Across all our experiments in various technology nodes, the best performing regression models achieve a very high R^2 of > 0.99 for digital circuits and more than 0.95 in the case of AMS circuits. Further, we report μ_E, σ_E $< 1\%$ and an average MaPE $< 1\%$ for the test split data in all the designs, validating the accuracy of

the trained regression models. Remarkably, the classification models report 100% accuracy in multiple analog designs.

Our research also introduces the Unified Deep-learning Neural Network (U-DNN) architecture for the reuse of its architecture acquired through modeling of PVT-aware AMS circuits in CMOS 180nm and 65nm technologies, enabling accurate modeling at other technology nodes such as CMOS 28nm. The U-DNN serves as a versatile platform for modeling various AMS circuits for different applications, significantly reducing the extensive design exploration time typically required to select appropriate NN structures.

Furthermore, our research introduces the novel implementation of transfer learning for data-efficient circuit modeling, effectively investigating the relationships of PVT and leakage and delay of standard digital cells within CMOS/FinFET technologies across different nodes. The experimental outcomes, both before and after fine-tuning via Bayesian optimization, demonstrate an enhanced performance of the transfer-learned DNN. Through comprehensive demonstrations of the different digital standard cells, our proposed methodology attains a speed-up of 10^4 times than SPICE and up to 10x cost reduction compared to other cutting-edge modeling techniques, all while achieving heightened model accuracy. Additionally, in another configuration of Zero-Shot Learning, accurate modeling is achieved with zero modeling complexity and without the need for any simulation data, making it a valuable tool for preliminary analysis of circuit variability for future nodes.

Further, the research accelerates and improves the estimation of PVT-aware performance for complex digital circuits. It achieves this by utilizing predictions from surrogate standard cell models in an automated manner. The experimental results exhibit a remarkably close alignment with simulation results, obviating the need for additional computations. Moreover, through the utilization of multi-objective and multi-directional optimization algorithms on these trained ML models, we achieve a significant enhancement in computational efficiency compared to conventional simulators, all while maintaining accuracy. Our approach intelligently selects the optimal combination of ML and optimization algorithms tailored to each circuit, ensuring optimal outcomes. Experimental validation conducted on 22nm CMOS High-K technology complex digital circuits demonstrates substantial reductions of up to 55.51% in delay and 20.44% in leakage, alongside consistent area reduction. Our optimization engine, OptiMo, offers a fast, reliable, and accurate solution for a diverse range of intricate circuits through an automated approach. Application results on ISCAS-74X and ISCAS-85 benchmark circuits showcase remarkable reductions of up to 64.6% in PDP.

In addition, this research integrates a rapid sensitivity analysis and a methodology for identifying dominant parameters using the Pearson coefficient and Information gain for both digital and analog components. Moreover, the innovative self-adaptation approach has successfully optimized the performance characteristics of AMS circuits to closely align with their typical specifications across a wide range of PVT conditions, leveraging inferred control codes derived from trained ML models. Both the LUT approach, which maintains fixed conditions across all operating scenarios and processes, and the dynamic approach, which employs a regression model to predict the decimal equivalent of digital con-

trol codes, demonstrate highly accurate circuit tuning to typical conditions within a brief computation time. The precision of ML modeling and predictions, customized for specific PVT conditions, underpins the accuracy of these outcomes. It is imperative to emphasize that generating the dataset for ML model training and identifying dominant components for an analog functional circuit is a one-time effort integrated into the self-adaptive design library. Furthermore, self-adaptation significantly enhances circuit performance compared to traditional methodologies, improving it by several orders of magnitude.

By offering a holistic PVT-aware surrogate modeling approach that bridges the gap between digital and analog realms, our framework holds promise for accelerating the design process and enhancing overall circuit performance in increasingly complex semiconductor ecosystems. In the future, implementing the surrogate methodologies on silicon and verifying the speed and accuracy of the developed AI/ML models would be our focus.

Bibliography

[1] A. Jacob, R. Xie, M. Sung, W. LiebmannLars, R. Lee, and B. Taylor, "Scaling Challenges for Advanced CMOS Devices," *International Journal of High Speed Electronics and Systems*, vol. 26, p. 1740001, 2017.

[2] A. Zjajo, *Stochastic Process Variation in Deep-Submicron CMOS*. Springer, 2016.

[3] K. Kuhn, "Chapter 1 - CMOS and Beyond CMOS: Scaling Challenges," in *High Mobility Materials for CMOS Applications*, ser. Woodhead Publishing Series in Electronic and Optical Materials, N. Collaert, Ed. Woodhead Publishing, 2018, pp. 1–44. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B978008102061600001X

[4] H. Iwai, "Technology roadmap for 22nm and beyond," in *2009 2nd International Workshop on Electron Devices and Semiconductor Technology*, 2009, pp. 1–4.

[5] J.-A. Carballo, W.-T. J. Chan, P. A. Gargini, A. B. Kahng, and S. Nath, "Itrs 2.0: Toward a re-framing of the Semiconductor Technology Roadmap," in *2014 IEEE 32nd International Conference on Computer Design (ICCD)*. IEEE, 2014, pp. 139–146.

[6] D. Bhattacharya and N. K. Jha, "FinFETs: From Devices to Architectures," *Advances in Electronics*, vol. 2014, 2014.

[7] X. Wang, A. R. Brown, B. Cheng, and A. Asenov, "Statistical variability and reliability in nanoscale FinFETs," in *2011 International Electron Devices Meeting*. IEEE, 2011, pp. 5–4.

[8] B. Vincent, R. Hathwar, M. Kamon, J. Ervin, T. Schram, T. Chiarella, S. Demuynck, S. Baudot, Y. K. Siew, S. Kubicek, E. D. Litta, S. Chew, and J. Mitard, "Process Variation Analysis of Device Performance Using Virtual Fabrication: Methodology Demonstrated on a CMOS 14-nm Finfet Vehicle," *IEEE Transactions on Electron Devices*, vol. 67, no. 12, pp. 5374–5380, 2020.

[9] Z. Abbas, A. Zahra, M. Olivieri, and A. Mastrandrea, "Geometry Scaling Impact on Leakage Currents in Finfet Standard Cells Based on a Logic-Level Leakage Estimation Technique," in *Microelectronics, Electromagnetics and Telecommunications*, J. Anguera, S. C. Satapathy, V. Bhateja, and K. Sunitha, Eds. Singapore: Springer Singapore, 2018, pp. 283–294.

[10] D. S. Boning, I. A. M. Elfadel, and X. Li, "A Preliminary Taxonomy for Machine Learning in VLSI CAD," in *Machine Learning in VLSI Computer-Aided Design*. Springer, 2019, pp. 1–16.

[11] T. Hastie, R. Tibshirani, and J. Friedman, "The Elements of Statistical Learning – Data Mining, Inference, and Prediction."

[12] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[13] S. Shukla, S. S. Gill, N. Kaur, H. Jatana, and V. Nehru, "Comparative Simulation Analysis of Process Parameter Variations in 20 nm Triangular FinFET," *Active and Passive Electronic Components*, vol. 2017, 2017.

[14] Z. Abbas and M. Olivieri, "Impact of technology scaling on leakage power in nano-scale bulk CMOS digital standard cells," *Microelectronics Journal*, vol. 45, no. 2, pp. 179–195, 2014.

[15] Y. S. Chauhan, S. Venugopalan, M. A. Karim, S. Khandelwal, N. Paydavosi, P. Thakur, A. M. Niknejad, and C. C. Hu, "BSIM — Industry standard compact MOSFET models," in *2012 Proceedings of the ESSCIRC (ESSCIRC)*, 2012, pp. 30–33.

[16] P. Cox, Ping Yang, S. S. Mahant-Shetti, and P. Chatterjee, "Statistical modeling for efficient parametric yield estimation of MOS VLSI circuits," *IEEE Transactions on Electron Devices*, vol. 32, no. 2, pp. 471–478, 1985.

[17] A. R. Alvarez, B. L. Abdi, D. L. Young, H. D. Weed, J. Teplik, and E. R. Herald, "Application of statistical design and response surface methods to computer-aided VLSI device design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 7, no. 2, pp. 272–288, 1988.

[18] D. L. Young, J. Teplik, H. D. Weed, N. T. Tracht, and A. R. Alvarez, "Application of statistical design and response surface methods to computer-aided VLSI device design II. desirability functions and Taguchi methods," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 10, no. 1, pp. 103–115, 1991.

[19] M. A. H. Khan, A. S. M. Z. Rahman, T. Muntasir, U. K. Acharjee, and M. A. Layek, "Multiple polynomial regression for modeling a MOSFET in saturation to validate the Early voltage," in *2011 IEEE Symposium on Industrial Electronics and Applications*, 2011, pp. 261–266.

[20] A. A. Mutlu and M. Rahman, "Statistical methods for the estimation of process variation effects on circuit operation," *IEEE Transactions on Electronics Packaging Manufacturing*, vol. 28, no. 4, pp. 364–375, 2005.

[21] S. Basu, P. Thakore, and R. Vemuri, "Process Variation Tolerant Standard Cell Library Development Using Reduced Dimension Statistical Modeling and Optimization Techniques," in *8th International Symposium on Quality Electronic Design (ISQED'07)*, 2007, pp. 814–820.

[22] L. Brusamarello, G. I. Wirth, P. Roussel, and M. Miranda, "Fast and accurate statistical characterization of standard cell libraries," *Microelectronics Reliability*, vol. 51, no. 12, pp. 2341–2350, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0026271411001879

[23] M. Miranda, P. Roussel, L. Brusamarello, and G. Wirth, "Statistical characterization of standard cells using design of experiments with response surface modeling," in *2011 48th ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2011, pp. 77–82.

[24] M. Miranda, P. Zuber, P. Dobrovolný, and P. Roussel, "Variability aware modeling for yield enhancement of SRAM and logic," in *2011 Design, Automation Test in Europe*, 2011, pp. 1–6.

[25] S. Chaudhuri, P. Mishra, and N. K. Jha, "Accurate leakage estimation for FinFET standard cells using the response surface methodology," in *2012 25th International Conference on VLSI Design*. IEEE, 2012, pp. 238–244.

[26] L. Cao, "Circuit power estimation using pattern recognition techniques," in *Proceedings of the 2002 IEEE/ACM international conference on Computer-aided design*, 2002, pp. 412–417.

[27] L. Cheng, P. Gupta, and L. He, "Efficient Additive Statistical Leakage Estimation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 11, pp. 1777–1781, 2009.

[28] "MCNC Designers' Manual," 1993. [Online]. Available: https://www.carolana.com/NC/NC_Manuals/NC_Manual_1993_1994.pdf

[29] H. Chang and S. Sapatnekar, "Full-chip analysis of leakage power under process variations, including spatial correlations," in *Proceedings. 42nd Design Automation Conference, 2005.*, 2005, pp. 523–528.

[30] A. Moshrefi, H. Aghababa, and O. Shoaei, "Statistical estimation of delay in nano-scale CMOS circuits using Burr Distribution," *Microelectronics Journal*, vol. 79, pp. 30–37, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S002626921730770X

[31] T.-T. Liu and J. M. Rabaey, "Statistical analysis and optimization of asynchronous digital circuits," in *2012 IEEE 18th International Symposium on Asynchronous Circuits and Systems*. IEEE, 2012, pp. 1–8.

[32] D.-W. Kim and T.-Y. Choi, "Delay Time Estimation Model for Large Digital CMOS Circuits," *VLSI Design*, vol. 11, no. 2, pp. 161–173, 2000.

[33] A. B. Kahng, M. Luo, and S. Nath, "SI for free: machine learning of interconnect coupling delay and transition effects," in *2015 ACM/IEEE International Workshop on System Level Interconnect Prediction (SLIP)*, 2015, pp. 1–8.

[34] V. Janakiraman, A. Bharadwaj, and V. Visvanathan, "Voltage and Temperature Aware Statistical Leakage Analysis Framework Using Artificial Neural Networks," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 7, pp. 1056–1069, 2010.

[35] L. Garg and V. Sahula, "Variability aware support vector machine based macromodels for statistical estimation of subthreshold leakage power," in *2012 International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, 2012, pp. 253–256.

[36] D. Helms, R. Eilers, M. Metzdorf, and W. Nebel, "Leakage models for high-level power estimation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 8, pp. 1627–1639, 2017.

[37] V. Govindaraj and B. Arunadevi, "Machine Learning Based Power Estimation for CMOS VLSI Circuits," *Applied Artificial Intelligence*, vol. 35, no. 13, pp. 1043–1055, 2021. [Online]. Available: https://doi.org/10.1080/08839514.2021.1966885

[38] "ISCAS Benchmark Circuits," (Accessed 13 March. 2024). [Online]. Available: https://web.eecs.umich.edu/~jhayes/iscas.restore/benchmark.html

[39] L. Yu, S. Saxena, C. Hess, I. A. M. Elfadel, D. Antoniadis, and D. Boning, "Statistical library characterization using belief propagation across multiple technology nodes," in *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2015, pp. 1383–1388.

[40] F. Wang, M. Zaheer, X. Li, J.-O. Plouchart, and A. Valdes-Garcia, "Co-Learning Bayesian Model Fusion: Efficient performance modeling of analog and mixed-signal circuits using side information," in *2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2015, pp. 575–582.

[41] Z. Gao, F. Wang, J. Tao, Y. Su, X. Zeng, and X. Li, "Correlated Bayesian Model Fusion: Efficient High-Dimensional Performance Modeling of Analog/RF Integrated Circuits over Multiple Corners," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2022.

[42] Z. Abbas and M. Olivieri, "Optimal transistor sizing for maximum yield in variation-aware standard cell design," *International Journal of Circuit Theory and Applications*, vol. 44, no. 7, pp. 1400–1424, 2016.

[43] "Predictive Technology Model," 2012. [Online]. Available: http://ptm.asu.edu/

[44] A. Stillmaker and B. Baas, "Scaling equations for the accurate prediction of CMOS device performance from 180 nm to 7 nm," *Integration*, vol. 58, pp. 74–81, 2017.

[45] 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, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167926020302947

[46] 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, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/3/435

[47] G. e. a. Liñán-Cembrano, "Design Automation of Analog and Mixed-Signal Circuits Using Neural Networks – A Tutorial Brief," *IEEE Transactions on Circuits and Systems II: Express Briefs*, pp. 1–1, 2023.

[48] S. Devi, G. Tilwankar, and R. Zele, "Automated Design of Analog Circuits using Machine Learning Techniques," in *2021 25th International Symposium on VLSI Design and Test (VDAT)*, 2021, pp. 1–6.

[49] G. Wolfe and R. Vemuri, "Extraction and Use of Neural Network Models in Automated Synthesis of Operational Amplifiers," *Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol. 22, no. 2, p. 198–212, nov 2006. [Online]. Available: https://doi.org/10.1109/TCAD.2002.806600

[50] O. e. a. Garitselov, "Fast-Accurate Non-Polynomial Metamodeling for Nano-CMOS PLL Design Optimization," in *Proceedings of the 2012 25th International Conference on VLSI Design*, ser. VLSID '12. USA: IEEE Computer Society, 2012, p. 316–321. [Online]. Available: https://doi.org/10.1109/VLSID.2012.90

[51] N. e. a. Lourenço, "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)*, 2018, pp. 133–136.

[52] Z. e. a. Wang, "Application of Deep Learning in Analog Circuit Sizing," in *Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence*, ser. CSAI '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 571–575. [Online]. Available: https://doi.org/10.1145/3297156.3297160

[53] N. e. a. Lourenço, "Using Polynomial Regression and Artificial Neural Networks for Reusable Analog IC Sizing," in *2019 16th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, 2019, pp. 13–16.

[54] G. e. a. İslamoğlu, "Artificial Neural Network Assisted Analog IC Sizing Tool," in *2019 16th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, 2019, pp. 9–12.

[55] N. Kahraman and T. Yildirim, "Technology independent circuit sizing for fundamental analog circuits using artificial neural networks," in *2008 Ph. D. Research in Microelectronics and Electronics*. IEEE, 2008, pp. 1–4.

[56] J. P. S. Rosa, D. J. D. Guerra, N. C. G. Horta, R. M. F. Martins, and N. C. C. Lourenço, *Using ANNs to Size Analog Integrated Circuits*. Cham: Springer International Publishing, 2020, pp. 45–66. [Online]. Available: https://doi.org/10.1007/978-3-030-35743-6_4

[57] A. Canelas, R. Martins, R. Póvoa, N. Lourenço, and N. Horta, "Efficient yield optimization method using a variable K-means algorithm for analog IC sizing," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, 2017, pp. 1201–1206.

[58] K. Settaluri, A. Haj-Ali, Q. Huang, K. Hakhamaneshi, and B. Nikolic, "Autockt: Deep Reinforcement Learning of Analog Circuit Designs," in *Proceedings of the 23rd Conference on Design, Automation and Test in Europe*, ser. DATE '20. San Jose, CA, USA: EDA Consortium, 2020, p. 490–495.

[59] S. Zhang, W. Lyu, F. Yang, C. Yan, D. Zhou, and X. Zeng, "Bayesian Optimization Approach for Analog Circuit Synthesis Using Neural Network," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2019, pp. 1463–1468.

[60] Z. Gao and D. S. Boning, "A Review of Bayesian Methods in Electronic Design Automation," 2023.

[61] M. Andraud and M. Verhelst, "From on-chip self-healing to self-adaptivity in analog/RF ICs: challenges and opportunities," in *2018 IEEE 24th International Symposium on On-Line Testing And Robust System Design (IOLTS)*, 2018, pp. 131–134.

[62] G. Volanis, D. Maliuk, Y. Lu, K. S. Subramani, A. Antonopoulos, and Y. Makris, "On-die learning-based self-calibration of analog/RF ICs," in *2016 IEEE 34th VLSI Test Symposium (VTS)*, 2016, pp. 1–6.

[63] M. Andraud, H.-G. Stratigopoulos, and E. Simeu, "One-Shot Calibration of RF Circuits Based on Non-Intrusive Sensors," in *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2014, pp. 1–6.

[64] J. C. Park and V. J. Mooney III, "Sleepy Stack Leakage Reduction," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 14, no. 11, pp. 1250–1263, 2006.

[65] D. Helms, E. Schmidt, and W. Nebel, "Leakage in CMOS Circuits – An Introduction," in *Integrated Circuit and System Design. Power and Timing Modeling, Optimization and Simulation*, E. Macii, V. Paliouras, and O. Koufopavlou, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 17–35.

[66] M.-B. Lin, "Introduction to VLSI Systems: A Logic, Circuit, and System Perspective (1st ed.), CRC Press," 2011.

[67] "BSIM4 Model," 2017. [Online]. Available: https://bsim.berkeley.edu/models/bsim4/

[68] "BSIM-CMG Model," 2023. [Online]. Available: http://bsim.berkeley.edu/models/bsimcmg/

[69] M. Olivieri and A. Mastrandrea, "Logic drivers: A propagation delay modeling paradigm for statistical simulation of standard cell designs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 6, pp. 1429–1440, 2013.

[70] E. Alpaydin, *Introduction to Machine Learning*. MIT press, 2020.

[71] A. J. Ferreira and M. A. T. Figueiredo, *Boosting Algorithms: A Review of Methods, Theory, and Applications*. New York, NY: Springer New York, 2012, pp. 35–85.

[72] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

[73] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[74] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised Machine Learning: A Review of Classification Techniques," *Emerging artificial intelligence applications in computer engineering*, vol. 160, no. 1, pp. 3–24, 2007.

[75] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the Human Out of the Loop: A Review of Bayesian Optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.

[76] F. Hutter, L. Kotthoff, and J. Vanschoren, *Automated Machine Learning: Methods, Systems, Challenges*. Springer Nature, 2019.

[77] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[78] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian Optimization of Machine Learning Algorithms," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'12. Red Hook, NY, USA: Curran Associates Inc., 2012, p. 2951–2959.

[79] D. Amuru, A. Zahra, and Z. Abbas, "Statistical Variation Aware Leakage and Total Power Estimation of 16 nm VLSI Digital Circuits Based on Regression Models," in *VLSI Design and Test: 23rd International Symposium, VDAT 2019, Indore, India, July 4–6, 2019, Revised Selected Papers 23*. Springer, 2019, pp. 565–578.

[80] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: A System for Large-Scale Machine Learning," in *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, ser. OSDI'16. USA: USENIX Association, 2016, p. 265–283.

[81] T. pandas development team, "pandas-dev/pandas: Pandas," Feb. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.3509134

[82] Wes McKinney, "Data Structures for Statistical Computing in Python," in *Proceedings of the 9th Python in Science Conference*, Stéfan van der Walt and Jarrod Millman, Eds., 2010, pp. 56 – 61.

[83] Z. Wu, J. Zhang, and S. Hu, "Review on Classification Algorithm and Evaluation System of Machine Learning," in *2020 13th International Conference on Intelligent Computation Technology and Automation (ICICTA)*, 2020, pp. 214–218.

[84] W. Xu, Z. Zhu, and L. Wang, "Comparative Analysis of Different Machine Learning Algorithms in Classification," in *2022 International Conference on Big Data, Information and Computer Network (BDICN)*, 2022, pp. 257–263.

[85] D. R. Cox, "The Regression Analysis of Binary Sequences," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 21, no. 1, pp. 238–238, 12 2018. [Online]. Available: https://doi.org/10.1111/j.2517-6161.1959.tb00334.x

[86] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, Sep 1995. [Online]. Available: https://doi.org/10.1007/BF00994018

[87] B. B. Yadav, K. Mounika, A. Bathi, and Z. Abbas, "67ppm/°c, 66na PVT Invariant Curvature Compensated Current Reference for Ultra-Low Power Applications," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020, pp. 1–5.

[88] B. B. Yadav, K. Mounika, K. De, and Z. Abbas, "Low Quiescent Current, Capacitor-Less LDO with Adaptively Biased Power Transistors and Load Aware Feedback Resistance," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020, pp. 1–5.

[89] M. Kelam, B. Y. Battu, and Z. Abbas, "A Compact, Power Efficient, Self-Adaptive and PVT Invariant CMOS Relaxation Oscillator," in *2020 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2020, pp. 1–6.

[90] C. W. Bae, D. Choi, K. Ahn, and C. Yoo, "A 6-gb/s differential voltage mode driver with independent control of output impedance and pre-emphasis level," *JSTS:Journal of Semiconductor Technology and Science*, vol. 13, pp. 423–429, 2013.

[91] H. Banba, H. Shiga, A. Umezawa, T. Miyaba, T. Tanzawa, S. Atsumi, and K. Sakui, "A CMOS bandgap reference circuit with sub-1-V operation," *IEEE Journal of Solid-State Circuits*, vol. 34, no. 5, pp. 670–674, 1999.

[92] R. Rifin, M. B. I. Reaz, S. Amin, and H. Husain, "Design of an Inductor-Less LNA Using Resistive Feedback Topology for UWB Applications," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 5, pp. 2196–2202, 02 2013.

[93] K. L. Fong and R. Meyer, "Monolithic RF active mixer design," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, no. 3, pp. 231–239, 1999.

[94] J. Liu, M. Hassanpourghadi, Q. Zhang, S. Su, and M. S.-W. Chen, "Transfer Learning with Bayesian Optimization-Aided Sampling for Efficient AMS Circuit Modeling," in *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, 2020, pp. 1–9.

[95] J. Liu, S. Su, M. Madhusudan, M. Hassanpourghadi, S. Saunders, Q. Zhang, R. Rasul, Y. Li, J. Hu, A. K. Sharma, S. S. Sapatnekar, R. Harjani, A. Levi, S. Gupta, and M. S.-W. Chen, "From Specification to Silicon: Towards Analog/Mixed-Signal Design Automation using Surrogate NN Models with Transfer Learning," in *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, 2021, pp. 1–9.

[96] P. E. e. a. Allen, *CMOS Analog Circuit Design*. Elsevier, 2011.

[97] Z. A. Chetan Mittal, Arnab Dey, "A 37nw, All-in-One Trim-free Voltage/Current Reference without using Resistors and Amplifiers," *IEEE MWSCAS 2023*, 2023.

[98] M. A. et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," [Online]. Available: https://www.tensorflow.org/, 2015.

[99] R. D. Peng and E. Matsui, *The Art of Data Science: A guide for anyone who works with Data*. Skybrude consulting LLC, 2016.

[100] G. C. Cawley and N. L. C. Talbot, "On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation," *J. Mach. Learn. Res.*, vol. 11, pp. 2079–2107, 2010. [Online]. Available: https://dl.acm.org/doi/10.5555/1756006.1859921

[101] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, Mar 1986. [Online]. Available: https://doi.org/10.1007/BF00116251

[102] R. K. Siddharth, Y. Jaya Satyanarayana, Y. B. Nithin Kumar, M. H. Vasantha, and E. Bonizzoni, "A 1-v, 3-ghz Strong-Arm Latch Voltage Comparator for High Speed Applications," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, no. 12, pp. 2918–2922, 2020.

[103] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[104] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A Survey of Transfer Learning," *Journal of Big Data*, vol. 3, no. 1, pp. 1–40, 2016.

[105] X. Sun, J. Gu, and H. Sun, "Research progress of zero-shot learning," *Applied Intelligence*, vol. 51, no. 6, pp. 3600–3614, Jun 2021. [Online]. Available: https://doi.org/10.1007/s10489-020-02075-7

[106] S. Gourishetty, H. Mandadapu, A. Zahra, and Z. Abbas, "A Highly Accurate Machine Learning Approach to Modelling PVT Variation Aware Leakage Power in FinFET Digital Circuits," in *2019 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, 2019, pp. 61–64.

[107] D. Amuru, M. S. Ahmed, and Z. Abbas, "An Efficient Gradient Boosting Approach for PVT Aware Estimation of Leakage Power and Propagation Delay in CMOS/FinFET Digital Cells," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020, pp. 1–5.

[108] K. Agarwal, A. Jain, D. Amuru, and Z. Abbas, "Fast and efficient ResNN and Genetic optimization for PVT aware performance enhancement in digital circuits," in *2022 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*. IEEE, 2022, pp. 1–4.

[109] B. Ramsundar and R. B. Zadeh, *TensorFlow for Deep Learning: From Linear Regression to Reinforcement Learning*, 1st ed. O'Reilly Media, Inc., 2018.

[110] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A Survey on Deep Transfer Learning," in *Artificial Neural Networks and Machine Learning – ICANN 2018*, V. Kůrková, Y. Manolopoulos, B. Hammer, L. Iliadis, and I. Maglogiannis, Eds. Cham: Springer International Publishing, 2018, pp. 270–279.

[111] J. Snoek, O. Rippel, K. Swersky, R. Kiros, N. Satish, N. Sundaram, M. Patwary, M. Prabhat, and R. Adams, "Scalable Bayesian Optimization Using Deep Neural Networks," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 2171–2180. [Online]. Available: https://proceedings.mlr.press/v37/snoek15.html

[112] C. C. Aggarwal, *Training Deep Neural Networks*. Cham: Springer International Publishing, 2018, pp. 105–167. [Online]. Available: https://doi.org/10.1007/978-3-319-94463-0_3

[113] "Python Dictionaries," 2024. [Online]. Available: https://docs.python.org/3/tutorial/

[114] Y. Song and M. Yu, "On finding the longest antisymmetric path in directed acyclic graphs," *Information Processing Letters*, vol. 115, no. 2, pp. 377–381, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020019014002294

[115] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in Python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[116] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine learning*, vol. 63, pp. 3–42, 2006.

[117] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A Highly Efficient Gradient Boosting Decision Tree," *Advances in neural information processing systems*, vol. 30, 2017.

[118] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *CVPR*, 2016, pp. 770–778.

[119] G. James, D. Witten, T. Hastie, R. Tibshirani *et al.*, *An Introduction to Statistical Learning*. Springer, 2013, vol. 112.

[120] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative Style, High-Performance Deep Learning Library," *Advances in neural information processing systems*, vol. 32, 2019.

[121] R. Kukreti and K. Singh, "Performance Optimization of Digital CMOS Integrated Circuits using LE Theory and APSO," in *2019 International Conference on Computing, Power and Communication Technologies (GUCON)*, 2019, pp. 600–604.

[122] Y. Jin and J. Branke, "Evolutionary optimization in uncertain environments-a survey," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 3, pp. 303–317, 2005.

[123] A. Starnes, A. Dereventsov, and C. Webster, "Gaussian smoothing gradient descent for minimizing functions (gsmoothgd)," 2024. [Online]. Available: https://arxiv.org/abs/2311.00521

[124] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ser. ICML'13. JMLR.org, 2013, p. III–1139–III–1147.

[125] P. Gupta et al., "PVT Variations Aware Robust Transistor Sizing for Power-Delay Optimal CMOS Digital Circuit Design," in *ISCAS*. IEEE, 2019, pp. 1–5.

[126] P. Saha, H. S. Kalluru, and Z. Abbas, "Transistor Sizing based PVT-Aware Low Power Optimization using Swarm Intelligence," in *2021 34th VLSID and 2021 20th International Conference on Embedded Systems*. IEEE, 2021, pp. 234–239.

[127] H. Kalluru et al., "Algorithm driven Power-Timing Optimization Methodology for CMOS Digital Circuits considering PVTA Variations," in *2021 IEEE ISCAS*. IEEE, 2021, pp. 1–5.

[128] K. Deb, K. Deb, and S. Gupta, "A survey of the state-of-the-art in genetic algorithms," *Journal of Genetic Algorithms and Evolutionary Computation*, vol. 6, no. 2, pp. 191–231, 2000.

[129] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, 1995, pp. 1942–1948.

[130] X.-S. Yang, "Artificial bee colony algorithm for global optimization," *Nature Inspired Computing and Optimization*, pp. 281–314, 2009.

[131] G. G. Beasley, X.-S. Yang, and S. Z. Nabney, "Elephant herding optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 1082–1102, 2009.

[132] A. H. Gandomi, A. H. Alavi, and S. Mirjalili, "A comprehensive review of Harris Hawks Optimization: Its variants and applications," *Archives of Computational Methods in Engineering*, pp. 1–22, 2022.

[133] K. M. Le, K. De, D. Amuru, and Z. Abbas, "AI-driven self adapting microelectronic circuits," Aug. 16 2022, uS Patent 11,416,664.

[134] Y. Cao and J. Zhou, "Integrated Self-Adaptive and Power-Scalable Wideband Interference Cancellation for Full-Duplex MIMO Wireless," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 11, pp. 2984–2996, 2020.