

A Machine Learning Approach to Test Pattern Prediction for VLSI Circuits Using Random Forest Regressor

Sima K Gonsai¹ and Usha Mehta²

Department of Electronics and Communication, Institute of Technology, Nirma University, Gujarat, India.

Corresponding author: Usha Mehta (e-mail: usha.mehta@nirmauni.ac.in), **Author(s) Email:** Sima K Gonsai (e-mail: simagonsai@ldce.ac.in)

Abstract: As the need for highly advanced chip technology and high-performance computing grows, semiconductor technology must develop rapidly. Also, as VLSI circuits become more complex, it is more difficult and time-consuming to test circuits using traditional test methods. Hence, Chip design, verification, and testing become extremely complex and critical processes. To confirm the chip as fault-free, testing examines its functionality, timing, and connectivity. Various automatic test pattern generation (ATPG) tools have been available for a long time for fault detection. For large combinational circuits, a lot of computing resources and more time are required to use such tools. To tackle this challenge, machine learning (ML) techniques have recently been introduced as a promising alternative to improve the efficiency of test generation. Machine learning has advanced dramatically over the last few years, and it currently plays an important role in enhancing automation, efficiency, and decision-making in a variety of domains. The main purpose of this work is to explore the possibility of using machine learning to predict test vectors in digital combinational circuits and achieve comparable fault coverage. The Random Forest Regressor method is utilized for generating test vectors. Experimental results showcase that this method performs well compared to traditional tools like ATALANTA on most ISCAS85 benchmark circuits. This approach reduced test pattern generation time to 0.0146 sec, while the standard tool requires 0.02 sec for test set generation, retaining almost similar fault coverage. This research demonstrated the highest fault coverage of 96.523 for the c5315_13 circuit and the lowest coverage of 60.404 for c1908_23, compared to the ATALANTA Tool. The results demonstrate that machine learning is a promising complementary testing method for conventional ATPG tools in VLSI testing.

Keywords Machine Learning; ATPG; ISCAS85 Benchmark Circuits; Test Vectors; Random Forest; Fault Coverage, Semiconductor Testing

1. Introduction

The evolution of integrated circuits has heightened design complexity, making testing an essential element of the chip design life cycle. A reliable and effective testing method is essential for integrated circuits with billions of transistors. To ensure that manufactured chips fulfil the necessary performance and reliability standards and operate as intended, circuit testing is essential. It improves manufacturing quality as well as yield by enabling the detection of manufacturing problems and defects prior to IC deployment. Additionally, efficient testing lowers manufacturing costs by rejecting faulty chips before they reach the market. The primary methods employed to guarantee the circuit is fault-free and functions as per the specifications of the circuits are built-in self-test, design for test (DFT), and automatic test pattern generation (ATPG). Different types of faults circuits may encounter are coupling faults, faults due to delay effects, bridge faults and stuck-at 0 and stuck-at 1

faults. The Automatic test pattern approach generates test vectors capable of detecting faults during the fabrication process. This work is based on generating test patterns for stuck-at 1 and stuck-at 0 faults present in circuits. These faults indicate that an interconnection in a circuit is consistently driven to 0 or 1. After generating test patterns, they are applied to the circuit to check whether the circuit is faulty or fault-free. Standard Automatic test pattern generator employs methodologies and algorithms like the Deterministic algorithm (D-algorithm) and Path-oriented decision making (PODEM). Standard Automatic Test Pattern Generation (ATPG) is an NP-complete problem. Standard ATPG depends on exhaustive search methodologies, which are hindered by exponential growth, resulting in prolonged processing times. Moreover, scalability is a key concern in such a complex design, as it requires more power and time. As Chip design evolves, adaptable ATPG is required; scalability and speed are key concerns for testing.

Hence, we have integrated an ML-based methodology for Automatic test pattern generation.

Recently, AI and ML have been used in a variety of fields, such as image processing, speech identification, automated systems, healthcare diagnosis, and driverless vehicles. The fundamental idea is to direct a machine to execute an intellectual activity, aiming to replace individual decisions or costly and prolonged analytical processes. A machine has inherent learning abilities and can be trained using extensive datasets as training tuples. It is not explicitly programmed; rather, it acquires knowledge from the data throughout the training period. It provides predictions for previously unobserved inputs that were unavailable during the training phase.

Recently, research has begun examining machine learning techniques for test pattern prediction and fault detection tasks at various levels of ATPG; However, attempts based on a direct, end-to-end approach for generating test patterns are still novel in this area. Data-driven techniques have been presented to reduce backtracking and improve test vector search, demonstrating significantly decreased runtime and increased test set quality on a set of benchmark circuits [1]. Comprehensive studies of ML in chip design, verification and testing domains reveal that learning-based methodologies for power estimation, fault detection, and verification support are advancing fast. They further emphasize that machine learning can capture complex, nonlinear relations between input and output that traditional heuristic ATPG techniques cannot clearly represent [2]. Moreover, ML models are predominantly applied for specific tasks such as fault prediction, feature selection, and test pattern sequencing, rather than for a comprehensive Automatic Test Pattern Generation process for widely accepted benchmarks like ISCAS circuits [3]. Thus, a significant research gap exists in establishing direct test pattern generation methods that are assessed on traditional ISCAS-85 benchmark circuits and match up with standard tools. This work addresses that gap by proposing and evaluating a Random Forest-regression-based approach for test pattern estimation, achieving comparable fault coverage.

In the proposed research, the data required to train the model is prepared through a widely used tool by Virginia Tech University, ATALANTA. The supporting files and instructional manual to understand the capabilities and usage of the tool are accessed on GitHub [4]. ATALANTA is an open-source fault simulator tool and generates test patterns with fault coverage. ATALANTA tool is used to generate test patterns for all ISCAS-85 benchmark circuits, which are applied as training input to the machine learning model.

The scalability issue for very large and complex circuits in ATPG, numerous ML, deep learning and CNN-based models have been proposed for testing them. [5] [6] [7] [8]. In particular, designers can use a deep neural network to find design hotspots [9] in a physical-layout architecture. A deep neural network is used to resolve routing in physical design [10]. A deep neural network is more powerful than heuristic or analytical methodologies in approximating functions [11] [12] and also performs faster testing [13]. Reinforcement learning is used to represent advanced circuits in ATPG [14]. To apply machine learning capabilities to testing of digital circuits, engineers should map testing issues into learning-based problems. Selecting appropriate features to apply as the input to the machine learning model requires in-depth knowledge of the testing domain. An appropriate dataset size with a balanced dataset is crucial to prevent model instability and performance deterioration since machine learning models rely heavily on data. ML models are data-driven and data-dependent; in chip testing, machine learning models may fail due to poor performance or an imbalanced dataset [15]. This adversely affects the chip design life cycle and results in delays in wafer production. Hence, ensuring accurate model training and evaluation is very important.

This paper investigates the use of machine learning in digital circuit testing, examining its difficulties and assessing its efficacy. To examine the performance of the machine learning model in chip testing, the model is first trained on a portion of the dataset and then tested on unseen data. The main challenge in Machine Learning is selecting an appropriate input abstraction to acquire robust features. It is necessary to have an adequate test set in order to achieve maximum fault coverage, while it is also desirable to have a reduced number of test patterns in order to reduce the amount of time spent on the test. The Random Forest Regressor machine learning model is effective in facilitating the generation of test patterns for semiconductor testing. The model is applied to ISCAS-85 benchmark circuits. This work is conducted for a dataset of around 980 digital circuits. The complexity of the circuits in the dataset varies in terms of the total count of inputs, total output ports, the number of gates, the circuit level and the total possible faults in the circuit.

Selecting appropriate features for training the Random Forest Regressor model for IC testing is another challenge that was addressed [16] [17]. Nevertheless, this work offered insight into whether machine learning, especially Random Forest Regressor (RFR), may help ATPG generate test sets for combinational digital circuits [18], [19]. This study discusses a new perspective on the practical

implementation of RFR in Semiconductor testing. This research also discusses the constraints of using machine learning in the IC testing sector. In SmartATPG [1], Graph Convolutional Networks and Reinforcement Learning were used to guide the ATPG search process and minimize backtracking, whereas in the proposed work, a supervised Random Forest Regressor is used to directly predict test vectors from extracted circuit features and verify them with the FSIM fault simulator.

This paper focuses on three major machine learning elements in the context of VLSI chip testing.

Data Set Generation: Large datasets are often required to train ML models properly. This dataset has 980 digital circuits with a test pattern set for each circuit. All these circuits are based on ISCAS-85 benchmark circuits. The dataset for this study was created with the ATALANTA tool from Virginia Tech University. A controlled random modification strategy was used to generate the synthetic circuits from the ISCAS-85 original combinational circuits. Up to five logic gates were randomly selected and replaced by functionally equivalent gates (AND, OR, NAND, NOR, XOR, or XNOR) in each circuit generated, without altering the connectivity of the circuit and without introducing cycles. To ensure structural diversity, gate replacements were performed at various locations in the circuit in successive iterations, with the next circuit derived from the previous one. To avoid duplication in the circuit set, gate replacement was performed in a sequential manner, starting from the first gate and continuing through each successive gate up to the final gate.

Test Pattern Generation using Machine Learning: Random Forest Regressor methodology is utilized to prepare test patterns for digital VLSI circuits.

Validation of Predicted Data: To validate the RFR model's outcomes (test pattern set), the test pattern set was applied to the target circuit using the FSIM simulator and the fault coverage was obtained. The main contribution of this work is to directly predict test patterns for digital combinational circuits using a machine learning-assisted framework with a Random Forest Regressor, with circuit-level features as input. A complete workflow includes dataset generation, feature extraction, decimal conversion, prediction, binary conversion, and FSIM-based simulation to validate the effectiveness of the predicted test vectors. This framework is applied to ISCAS85 benchmark circuits and compared with the ATALANTA tool for fault coverage and prediction time. To demonstrate the performance on genuinely unseen circuits, ISCAS89 circuits were used for testing and fault coverage of the predicted test vector set is determined through the FSIM tool. Unlike SmartATPG and reinforcement learning-based ATPG methods that guide the ATPG

search process, the proposed work investigates the feasibility of directly predicting test vectors using supervised Random Forest Regression. It demonstrates that comparable fault coverage can be achieved for several benchmark circuits with significantly lower prediction time.

A machine learning-assisted test vector prediction mechanism using the Random Forest Regressor is implemented for predicting test patterns of digital combinational VLSI circuits. The proposed ML model learns from test vectors produced by ATALANTA and predicts binary test patterns, which are then validated by the FSIM simulator to calculate fault coverage. Finally, the proposed approach is tested on the ISCAS85 benchmark circuits, and subsequently on unseen circuits from the ISCAS89 set, showing comparable fault coverage across several benchmark circuits and demonstrating the feasibility of ML-assisted test vector prediction. The proposed framework serves as the basis for combining data-driven techniques with traditional ATPG, which is a very promising approach to the problem of intelligent and scalable VLSI testing.

Paper Structure: This study identifies substantial issues and prospective research opportunities at the interface of machine learning and semiconductor testing. In Section II, necessary basic information regarding Machine Learning is discussed. ML models and related work for VLSI testing are discussed, and their information is presented. This Section also includes information about the implementation of the Random Forest Regressor. Section III contains results and discussion. Section IV contains the conclusion of the research work.

II. Machine Learning Background and Dataset Preparation:

A. Machine Learning basic concepts:

The main principle underlying Machine learning is to train a mathematical model with a large set of data and test it on unseen tuples. Supervised, unsupervised and reinforcement learning are three main classes of machine learning. Since the dataset consists of input such as digital circuit information and output as test patterns for these circuits, a supervised machine learning method is adopted for this field. For test pattern prediction, which involves estimating output values, a regression model is employed. In a classification model, samples are classified into appropriate classes, whereas in a regression model, binary or numeric values are predicted as per input data. As the ability of the machine learning model to estimate non-linear relationships among input and output, Random Forest Regressor, Decision tree

Table 1 Hyperparameters of Random Forest Regressor Model

Features	Details set for model	Value
N Estimator	Decision tree numbers	8
Split	Splitting strategy at every node	best
Max Depth	Maximum tree depth	10
Min Samples Split	Minimum number of samples to split an internal node.	2
Min Samples Leaf	Minimum number of samples required at a leaf node.	1
Random State	Random seed as input	42
Max Features	All features to be considered while considering best split.	1

Regressor [20] ,[21] support vector machine-SVM [22], multiple linear regression-MLR [23], also Neural Network-NN [24] , Deep neural network-DNN [25] , [26] are preferred over other ML models for VLSI design, verification and testing domain. Furthermore, it is essential to apply feature extraction along with domain expertise to translate the problem into a well-designed machine learning model accurately.

B. Multiple Linear Regression Model:

Linear Regression is a model in which independent input variables $x_1, x_2, x_3, \dots$ Eq. (1) [27] are linearly determining the output variable y Eq. (2), Eq. (3) [27]. α_0 is the intercept, α_1 to α_n are the regression coefficients, and σ is the error term.

$$X = x_1, x_2, x_3, \dots, x_n \quad (1)$$

$$Y = y_1, y_2, y_3, \dots, y_m \quad (2)$$

$$Y = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + \alpha_3 x_3 + \dots + \alpha_n x_n + \sigma \quad (3)$$

C. Random Forest Regression:

The Random Forest Regressor was chosen to learn the non-linear relationship between the input features and the test vector values, while keeping the model simple and efficient in terms of computation. It also provides good generalization and robustness to overfitting; it is an ensemble model based on classification and regression trees. The objective of CART is to figure out the correlation between a dependent variable (Y) and a collection of independent input feature variables ($x_1, x_2, \dots$) Eq. (1) The learning technique utilizes iterative partitioning to choose splits in the x variables, so forming homogeneous groups of Y . The partitioning occurs until either the value of Y at each node is equal or an additional split yields no further enhancement in performance. Random forest varies from the CART methodology by using bootstrap resampling [28] and selecting random variables. An exploration of a regression tree comprised of splits and nodes is shown in Fig. 1. **Random Forest Regressor Architecture using Bootstrap Sampling and Multiple Decision Trees** .This model uses a selected set of x variables to split every node. For continuous output variables, the ensemble prediction is the average of estimated values across trees, $\hat{y}$, whereas the variance is $\text{var}(\hat{y})$. The main concept is to employ bootstrap resampling to parameterize more trees in RFR. Bootstrap resampling

excludes around 37% of observations. An RFR is a machine learning Regressor model composed of a set of trees Regressor $\{h(x, \Theta_m), m = 1, \dots, M\}$, where the Θ_m are independent and identically distributed random vectors, and every tree predicts a single value for the most probable value at the input. This indicates that RFR is an integration involving several tree-structured regressors. In Breiman's RFR approach, all trees are constructed from a set of training data and a random variable (RV), denoted as Θ_m for the m^{th} tree. The RVs are independent and identically distributed, leading to the model representation as $h(x, \Theta_m)$, where x is the input feature. After m iterations of training, we get a sequence of Regression outputs as in Eq. (4) [27] $\{h_1(x), h_2(x), \dots, h_m(x)\}$, which are utilized to form more Regression models. The final output of this system is determined by a straightforward averaging of all predictions, and the final function output is

$$H(x) = \text{Avg} \sum_{i=1}^m I(h_i(x) = Y) \quad (4)$$

$H(\cdot)$ represents the set of Regression methodologies, h_i indicates a single decision tree model, Y is the output variable, and $I(\cdot)$ depicts the indicator function. Each tree can predict the most appropriate result for a test input variable.

Fig. 1 shows the Random Forest Regressor method. As represented in this figure, bootstrap sampling is used to divide the dataset into random subsets for each decision tree. Each tree is trained and tested on this subset of the main dataset. The predicted results from each tree are then averaged to determine the final output of the random forest Regressor. The dataset for RFR is represented through Eq. (5), Eq. (6), Eq. (7) [27].

$$D = \{(x_i, y_i)\}, i = 1 \text{ to } N \quad (5)$$

For t^{th} Decision tree, the training dataset will be $B(\cdot)$, which is bootstrap sampling,

$$D_t = B(D) \quad (6)$$

$$D_t = \{(x_{i1}, y_{i1}), (x_{i2}, y_{i2}), \dots, (x_{iN}, y_{iN})\}, \quad (7)$$

$$i_k \in \{1, 2, 3, \dots, N\}$$

Here, each i_k is selected independently with replacement. N is the number of decision tree. x_i is the input feature of i^{th} tree and y_i is the target variable for i^{th} tree.

D. Steps for Single Decision Tree ML Model:

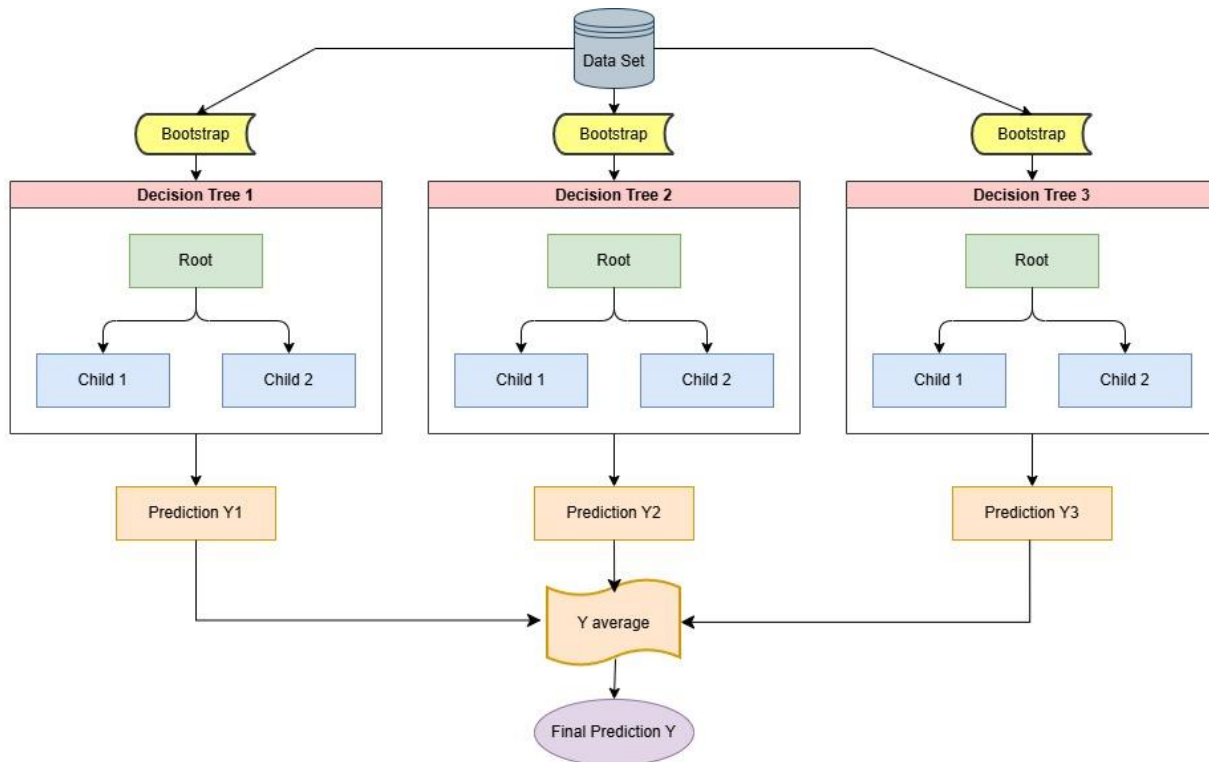


Fig. 1. Random Forest Regressor Architecture using Bootstrap Sampling and Multiple Decision Trees

DTR will build a tree based on observations of the problem and its dataset by creating decision rules from queries. First, some questions are asked to determine the root node, as the main node of the overall tree structure. In a tree, the nodes of the tree are derived from the root node. A decision node is a child node. This node makes a decision by testing an attribute. The node that has to make a decision is also known as the internal node. Then, a few more questions will be asked, and the whole tree structure will be complete up to the leaf nodes. The data set is depicted as in Eq. (8) [27].

$$D = (x_i, y_j), i = 1 \dots n, j = 1 \dots m \tag{8}$$

where X_i is the input feature, Y_j is the output variable, and N is the total number of tuples in a set. The feature space, ψ , is divided into m disjoint regions R_1 to R_m via a decision tree, as in Eq. (9) [27].

$$\psi = R_1, R_2, \dots, R_m, \tag{9}$$

The prediction, y , for input x , in region R_m , is determined by the average of the output values in the same region as in Eq. (10) [27].

$$y(x) = \frac{1}{|R_m|} \sum_{x_i \in R_m} y_i \tag{10}$$

The following steps are used to generate a regression tree Algorithm 1.

Algorithm 1 Regression Tree Generation

(1) **Input:** Training dataset $D (X, Y)$

(2) **Output:** Regression tree T

(3) Initialize root node $\leftarrow D$

(4) **While** stopping criterion is not satisfied **DO**

(5) Evaluate MSE for candidate splits

(6) Select split with minimum MSE

(7) Partition current node into child nodes

(8) **FOR** each child node **DO**

(9) **IF** stopping criterion is satisfied **Then**

(10) Create leaf node

(11) **ELSE**

(12) Continue recursive splitting

(13) **END IF**

(14) **END FOR**

(15) **END While**

(16) Assign mean target value to each leaf node

(17) Generate decision rules

(18) **Return** T

E. Dataset:
Successful deployment of a machine learning model for ATPG demands a substantial number of high-quality, industry-specific, balanced sets of data. Generally, industrial data for testing is not available as open-source for public access. Standard ISCAS-85 benchmark circuits [29] are used to train and test the model. ISCAS-85 set includes 10 combinational digital benchmark circuits. ATALANTA is an open-source test pattern generation tool for digital circuits. It produces test vectors to identify faults and calculates fault coverage effectively. This tool is frequently employed in Digital IC test and research to validate fault detection

techniques and benchmark test-generating methodologies. This tool produces test patterns mainly designed to identify stuck-at faults (SA-0 and SA-1) by activating and propagating these faults to primary outputs. The file "`\circuit_name>.test`" is produced, containing a list of test patterns with their non-faulty outputs. These test patterns of the ISCAS85 benchmark circuits are constructed by the ATALANTA simulator tool and utilized as a training dataset for the machine learning technique in this research. In the beginning, 200 digital-combinational circuits are developed in Verilog. The input format of the ATALANTA tool [30] is a bench file format. Verilog digital circuits are converted to the bench file format and applied to the tool. To train the Random Forest ML model, this tool is employed to generate a test set of test vectors and fault coverage of 200 digital combinational circuits. The first iteration outcomes of the testing demonstrate that a larger set of data is required to adequately train the Random Forest Regressor model.

ATALANTA command to run the benchmark circuit is, ATALANTA -t <out_test_file> <benchmark_file>.

ATALANTA -t c432.test c432.bench.

Where c432.test file is generated by the tool and contains a test vector set, and the input file is c432.bench, which is the Verilog code for a particular circuit.

Approximately 980 digital circuits with comparable connector patterns, gate quantities, circuit levels, and complexities have been devised. Subsequently, these circuits are implemented using the ATALANTA tool for generating test patterns and their corresponding fault coverage for a set of data. The enriched data samples are utilized for efficiently training the Random Forest Regressor model. Table 1 specifies the configuration of the variables for the Random Forest Regressor. The maximum number of trees in the random forest model is set to 8, as there are 8 groups of circuits in the dataset. All remaining parameters are set to default values. To train the RFR model effectively, the circuits in the dataset are divided into 8 groups, as per the total inputs and outputs in the circuits. The distribution of total circuits in each group in the dataset is presented in Fig. 2. A total of 175 circuits is in the c-7552 and c-6288 groups, 150 circuits are in the c-1908 and c-5315 groups, c-3540 has 105 circuits, c-880 has 104 circuits, c-499 has 70 circuits, c-432 has 61 circuits, and c-17 has 6 circuits. The next section presents the detailed implementation procedures for the proposed technique. The effectiveness of a regression model can be measured by its accuracy in predicting a previously unseen set of samples. Typically, statistical performance measures are used for this evaluation. The Mean Squared Error and Mean Absolute Error, Eq.

(11), Eq. (12) [27] are one of the most popular assessment tools.

MSE is the average of the squares of the distances between the predicted point and the actual output, for a set of n points. It is calculated as the average of the squares of the prediction errors for each sample. The lower the MSE, the stronger the prediction and the better the model. Good generalization is expected when a regression model has a relatively low MSE on the training set. However, if the error is very small for the training set, then this can be a sign of overlearning, in which the model remembers the training data and does poorly on data that has not been seen. Hence, Mean Absolute Error, Eq. (11) [27], and Mean Squared Error Eq. (12) [27] should be studied along with the validation and testing results to make sure the models perform well. y_{act}^k is the actual output of k th test point, y_{pre}^k is the predicted output of k th test point.

$$\text{Mean Absolute Error} = \frac{1}{N} \sum_{k=1}^N |y_{act}^k - y_{pre}^k| \quad (11)$$

$$\text{Mean Square Error} = \frac{1}{n} \sum_{K=0}^n (y_{act}^k - y_{pre}^k)^2 \quad (12)$$

When the prediction error is to be measured in the same units as the target variable and requires greater importance to large errors, RMSE Eq. (13) [27] is calculated.

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{K=0}^n (y_{act}^k - y_{pre}^k)^2} \quad (13)$$

Normalized RMSE (NRMSE) Eq. (14), [27] is employed when comparing prediction errors for different datasets, circuits or target variables with different numerical scales. It scales the RMSE; this error is now dimensionless and is easier to compare across different sizes of inputs and output sets in the data. y_{max} and y_{min} are the maximum and minimum values of y , respectively.

$$\text{NRMSE} = \frac{\sqrt{\frac{1}{n} \sum_{K=0}^n (y_{act}^k - y_{pre}^k)^2}}{y_{max} - y_{min}} \quad (14)$$

R^2 Eq. (15) [27] is a measure of the amount of variation in the dependent variable that can be explained by the independent variable. It indicates the proportion of variance in the actual data that is captured by the model. R^2 measures the goodness of fit of the regression model, while MSE, RMSE and NRMSE measure prediction error. y_{mean} is the mean of y_{act} .

$$R^2 = 1 - \frac{\sum_{K=1}^n (y_{act}^k - y_{pre}^k)^2}{\sum_{K=1}^n (y_{act}^k - y_{mean})^2} \quad (15)$$

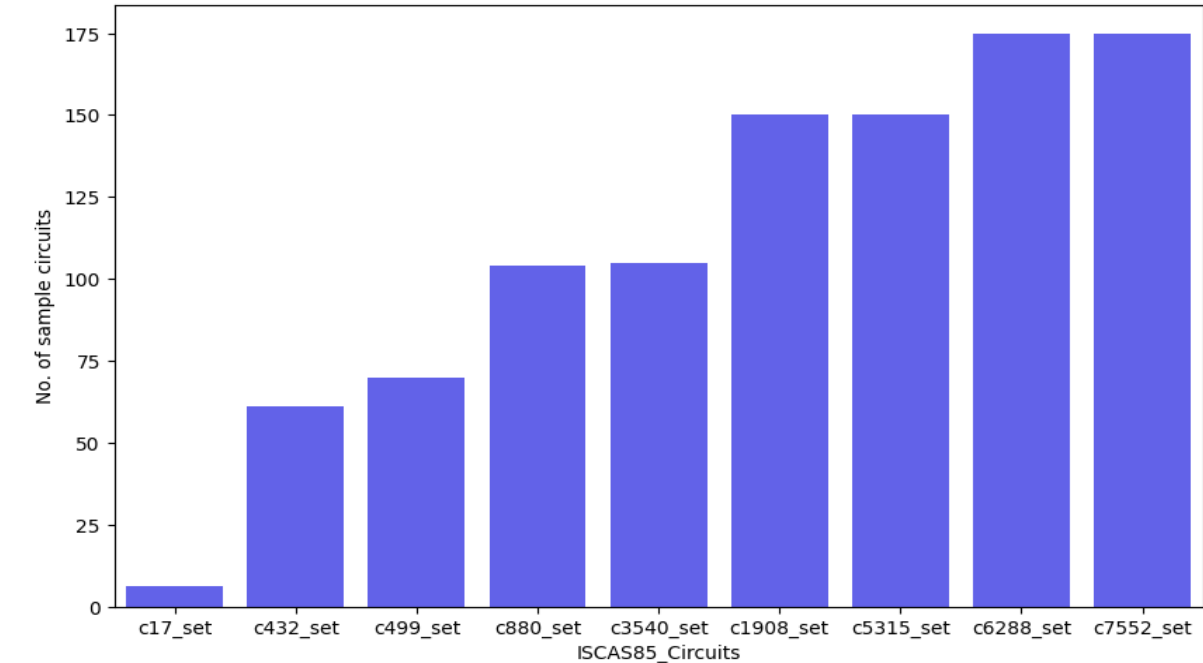


Fig. 2 Distribution of Circuits Across the ISCAS85-Based Dataset Groups

Table 2 Input Features Used for the Random Forest Regressor Model

Circuit Name	Total Inputs	Total Outputs	Total Gates	Total Faults	Total OR Gates	Total AND Gates	Total NOT Gates	Total NAND Gates	Total NOR Gates	Total XOR Gates
C432_3	36	7	160	524	0	7	40	116	19	18
C499_32	41	32	202	758	2	24	40	72	0	104
C880_4	60	26	383	942	29	151	63	142	61	0
C1908_23	33	25	880	1879	0	271	277	608	1	0
C3540_23	50	22	1669	3428	92	767	490	742	68	0
C5315_13	178	123	2307	5350	214	1057	581	1009	27	0
C6288_35	32	32	2416	7744	50	256	32	52	2058	0
C7552_45	207	108	3512	7550	244	1360	876	1814	94	0

The first approach is to assess the ML models with common machine learning metrics such as accuracy, or MSE, RMSE. All of these, however, are insufficient and do not reflect the practical significance in the IC test domain. Some metrics used for a test may be poorly correlated with common metrics used in ML. Then the output of the ML model test pattern prediction is used and applied to the standard tool FSIM simulator to check the fault coverage. The most important parameter used in testing ICs is fault coverage, Eq. (16) [31]. It is the ratio of the number of faults detected to the number of faults present in the circuit.

$$\text{FaultCoverage}(\%) = \frac{\text{Number of detected faults}}{\text{Total number of faults}} \times 100 \quad (16)$$

Fault efficiency, FE Eq. (17) [31], verifies the undetectable faults in the circuit. N_{detected} , $N_{\text{undetected}}$ and N_{total} are the number of detected, undetected and total faults.

$$FE = \frac{N_{\text{detected}}}{N_{\text{total}} - N_{\text{undetected}}} \times 100 \quad (17)$$

Total test patterns Eq. (18) [31], generated is, N_p is test patterns number.

$$T_s = \sum_{i=1}^{N_p} 1 = N_p \quad (18)$$

ATPG simulation time Eq. (19) [31] is, t_j is time needed to generate test pattern for fault j .

$$T_{\text{ATPG}} = \sum_{j=1}^{N_f} t_j \quad (19)$$

N_f , is the total number of target faults. N_{tested} is total the tested faults. Fault detection ratio (FDR) Eq. (20), [31] is defined as,

$$FDR = \frac{N_{\text{detected}}}{N_{\text{tested}}} \quad (20)$$

Test compression ratio Eq. (21), [31] is defined using T_{ORG} is original test set and T_{COM} is compressed test set.

$$TCR = \frac{T_{\text{ORG}} - T_{\text{COM}}}{T_{\text{ORG}}} \times 100 \quad (21)$$

Time required for ATPG Eq. (22), [31] to run and predicts test pattern is a function of, $T_{\text{ATPG}} = f(I, PO, PI, F, G_{\text{AND}}, G_{\text{OR}}, G_{\text{NOT}}, G_{\text{NAND}}, G_{\text{NOR}}, G_{\text{XOR}})$ (22) Here I is the total inputs of the digital circuit, PO and PI is the Primary Inputs and outputs, F is the total faults in the circuit, G_{AND} is total AND gates count in circuit, similarly number of gates present in circuit.

III. Method

Recent research indicates that Naïve Bayes and SVM are effective for defect identification and functional verification. Thus, ML methodologies facilitate the resolution of design issues across multiple levels of abstraction throughout the VLSI design cycle [32],[33],[34],[35]. An unsupervised ML model, specifically Bayesian theory, is employed for scan chain evaluation [35]. The RFR (Random Forest Regressor) is a resilient ensemble learning technique that integrates several decision trees using bagging to enhance precision and generalisation in results. It

efficiently manages non-linearity between input features and output features without requiring significant feature engineering. A primary benefit is its ability to avoid overfitting through averaging multiple trees, and its intrinsic capacity to identify important features. RFR is recommended when interpretability of important input features, stability, and robust performance are essential, particularly in scenarios where linear models underperform and individual decision trees overfit.

Linear Regression framework identifies mostly linear relationships among input and target features, whereas the Decision Tree Regressor is capable of approximating non-linear relations as well. SVM (Support Vector Machine) involves extensive hyperparameter tuning for optimal performance. On the contrary, DTR constructs a single tree, making it simple and interpretable but very susceptible to overfitting and data variations. Random Forest Regressor, on the other hand, is an ensemble approach that builds many trees using bootstrap sampling and feature selection randomly before aggregating their outputs, considerably improving resilience and reducing variance.

Fig. 3 illustrates the overall workflow of the proposed methodology. The dataset is the foundation of any machine learning system, since it directly influences the model's capacity to learn hidden patterns and generalise to unknown inputs. The quality, quantity, and variability of data have a major impact on model accuracy, robustness, and bias. However, noisy, unbalanced, or inadequate data can result in overfitting, underfitting, and inaccurate generalisations. To expand the set of data for the method, 980 digital combinational circuits in bench file format were generated. ATALANTA is an open-source tool that is an Automatic test pattern generator and fault simulator tool for digital combinational circuits.

Each circuit from the 980-circuit set is applied to this tool to generate test pattern sets and fault coverage. Each test pattern is partitioned into 20-bit segments; each segment is converted to decimal form, and a combined CSV file is created for the dataset. This transformation enables prediction to be formulated as a regression task suitable for an RFR model. Although numerical values do not reflect bitwise similarity, the proposed framework evaluates prediction quality based on fault coverage achieved. This test pattern set is available in a text file; in order to enable efficient and seamless utilisation in the ML model, test patterns are converted into a CSV file format. The input features, like the total number of Inputs in a circuit, the total outputs of a circuit, the count of all gates, the total AND, OR, NOT, EX-OR gates and the total faults present in a circuit, as shown in Table 2, were applied to train the RFR model. To test the feasibility of the proposed Random Forest-based ATPG prediction framework,

the feature set was limited to basic circuit features rather than structural features, like SCOAP metrics, logic depth, and reconvergence index. To effectively represent the dataset, it is essential to convert long binary streams into small chunks and transform them into numeric values. All test patterns are fragmented into 20-bit parts, then converted to decimal form. After pre-processing, the dataset is employed for model training. The dataset is then randomly partitioned into a 20%-80% ratio to test and train the model. No leave-circuit-family-out or cross-benchmark validation strategy was used. Thus, the results presented are based on the random partitioning scheme. The partitioned dataset is then used for training and testing the model.

A random forest Regressor is used to forecast test patterns for the circuit applied in the validation set. The original outputs produced by the random forest Regressor model comprise the test pattern set for every circuit, presented in a distributed and numerical manner. As post-processing, to obtain the binary representation of test vectors, numerical values are converted into binary values. Depending on the total number of inputs of the circuit, 3 to 11 columns are

combined to make a single test pattern. All the test pattern lists are combined in different files for each new test circuit.

Each digital combinational circuit's predicted test patterns are saved and transferred to a text file for further processing. This text file is saved in .tst format. Subsequently, this test file (.tst) is applied to FSIM, a fault simulator tool. The FSIM simulator is employed to ascertain the fault coverage of the predicted test vectors for the digital circuit being evaluated. As illustrated in Fig. 3, if the evaluated fault coverage is much lower than the target fault coverage, RFR is fine-tuned, and model parameters are updated to enhance the model's accuracy. The expected fault coverage should be at most a 20% difference in fault coverage compared to ATALANTA's fault coverage. The ML algorithm is retrained and tested in the next iteration with revised parameters.

At the beginning, the experiment was performed on a dataset of 200 digital circuits. However, the achieved fault coverage was below the targeted coverage. Further, the dataset was enriched by increasing the total number of digital circuits to 980 circuits, driven by ISCAS85 digital circuits. Complete workflow steps were

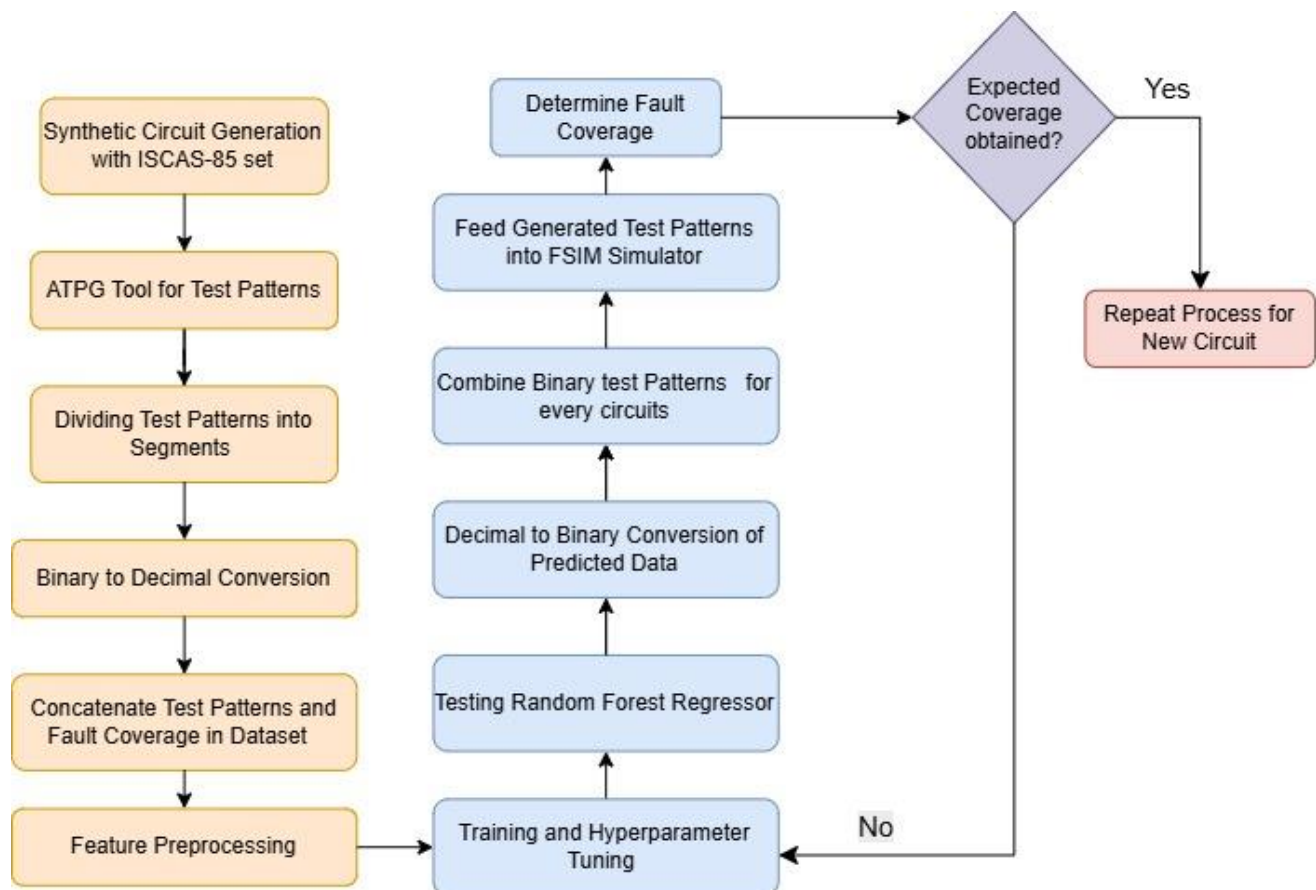


Fig. 3 Proposed RFR-Based Test-Pattern Generation and Validation Workflow

Table 3 Comparison of Test-Pattern Prediction and Fault Coverage Using ATALANTA and RFR

Circuit	# Gates	# Faults	# Test Patterns	% Fault Coverage	# Test Patterns	% Fault Coverage	# Test Pattern	% Fault Coverage	Δ%Fault Coverage	Δ Test Patterns
			ATALANTA		RFR (Best)		Random Patterns		Comparison	
C432_3	160	524	52	99.237	57	87.214	57	81.489	12.023	+5
C499_32	202	758	32	94.591	43	91.821	43	80.765	2.77	+11
C880_4	383	942	47	99.257	55	99.257	55	80.021	0	+8
C1908_23	880	1879	83	77.594	104	60.404	104	55.508	17.19	+21
C3540_23	1669	3428	86	67.153	79	61.610	79	62.048	5.543	-7
C5315_13	2307	5350	122	96.673	125	96.523	125	82.804	0.15	+3
C6288_35	2416	7744	32	93.479	32	93.091	32	82.820	0.388	0
C7552_45	3512	7550	199	89.192	195	81.020	195	78.662	8.172	-4

*RFR(Best): Random Forest Regressor’s Best Result

re-executed, involving feature engineering, preparation of a consolidated .csv dataset for training, and testing the RFR methodology.

Thereafter, post- processing of results was performed. In this phase, the RFR model demonstrated improved performance. Experimental results of fault coverage and discussion are carried out in the subsequent section. These experiments were

conducted on Google Colaboratory using Python (version 3.12.13); the Random Forest Regressor was implemented with the Scikit-learn library, with the parameters in Table 1. The random seed for the experiment was set to 42, and train-test splitting was done using an 80%-20% ratio. Benchmark splitting is not required, as the algorithm randomly selected tuples for training or testing.

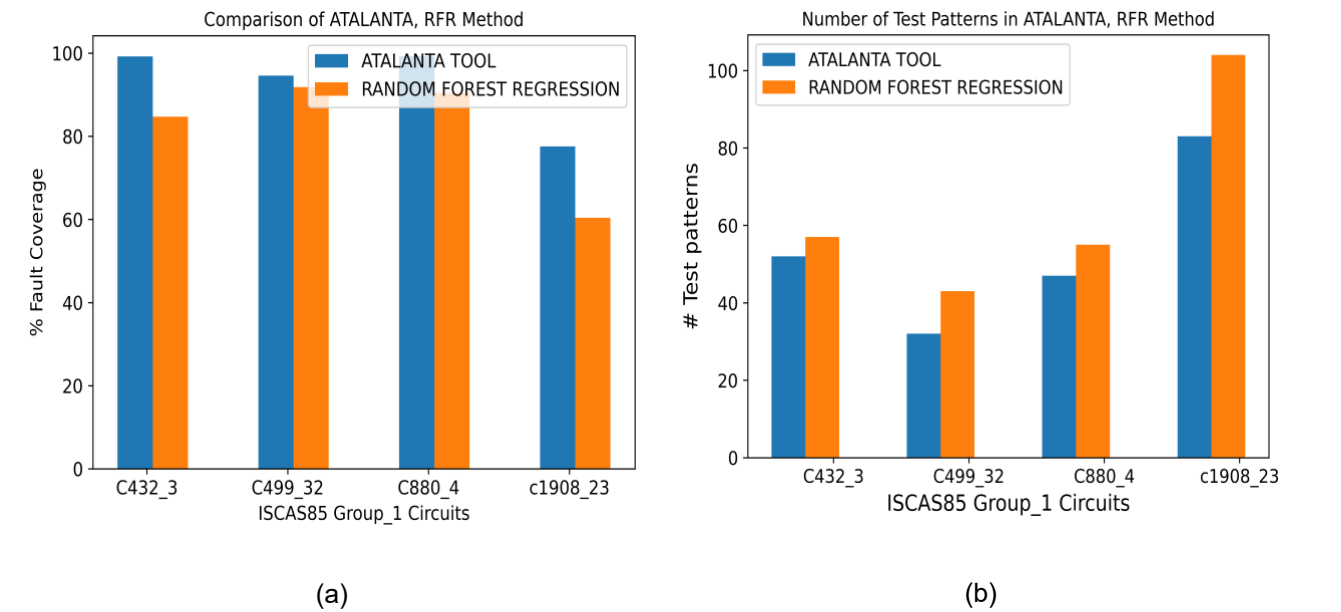


Fig. 4 Set-1 Circuits (a) Fault Coverage of Set-1 Circuits, (b) Test Pattern Count of Set-1 Circuits

Table 4 Prediction Performance of Test Circuits in Terms of Fault Coverage, MSE, RMSE, and Execution Time

Circuit	Total Gates	Total Faults	Total Test Vectors	% Fault Coverage	MSE	RMSE	RFR Time (ms)
C432_3	160	524	57	84.733	13.18X10 ⁴	363.01	14.6
C499_32	202	758	43	91.821	93.93X10 ⁴	969.19	15.2
C880_4	383	942	55	90.446	96.10X10 ⁸	98032.39	16.0
C1908_23	880	1879	104	60.404	59.09X10 ³	243.081	15.3
C3540_23	1669	3428	79	59.772	79.91X10 ⁸	89391.21	15.2
C5315_13	2307	5350	125	96.523	5558X10 ⁹	235748.62	15.5
C6288_35	2416	7744	32	92.936	15.33X10 ⁸	39159.75	15.4
C7552_45	3512	7550	195	81.020	10.06X10 ¹⁰	317183.82	16.9

IV. Result

The present section demonstrates the overall result of RFR. The Random Forest Regressor was executed using Google Colaboratory, a cloud-based platform, for the execution of Python code. Benchmark circuits and other circuits with similar properties were used to evaluate the model. ATALANTA tool served as the foundational platform for the research work. It was used to calculate fault coverage with a test pattern set. Predicted results are compared with actual results by ATALANTA. Random Forest Regressor model predictions are post-processed. Predictions are in numerical format and are converted into binary form to

calculate fault coverage. One text file is created with the test extension for every circuit's test vector set.

The FSIM simulator tool was employed on this test file to accurately compute fault coverage using the entire test pattern set generated by the Random Forest Regressor. Table 3 illustrates the experimental results of RFR and the ATALANTA Tool. It is identified that the circuits C5315_13 and C6288_35 have almost the same fault coverage as the ATALANTA Tool. Also, C880_4 has the same fault coverage as ATALANTA. C5315_13 has 0.15% smaller fault coverage, and C6288_35 has 0.388% smaller coverage compared to the standard tool. C499_32 has a similar order of Fault coverage in the set. C1908_23

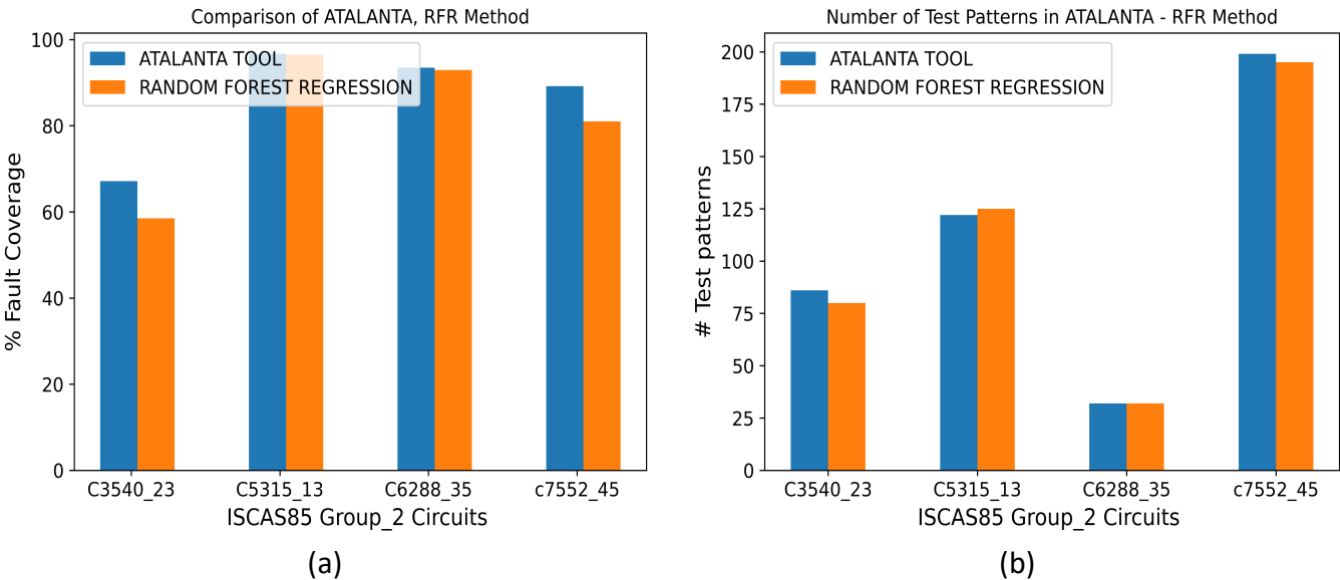


Fig. 5 Set-2 Circuits (a) Fault Coverage of Set-2 circuits, (b) Test Pattern Count of Set-2 Circuits

has the maximum difference in fault coverage at 17.19%.

The difference in Fault coverage for C499_32 is only 2.77. This demonstrates that fault coverage achieved by ATALANTA is slightly higher than the RFR model's Fault Coverage. The best fault coverage of 99.257% is achieved for circuit c880_4. C1908_23 circuit has the lowest fault coverage of 60.404%. The average fault coverage of these circuits is 83.87%. Likewise, highlighted numbers in the test patterns columns indicate that the total test pattern count for circuit C3540_23 and circuit C7552_45 is 7 and 4 fewer test patterns than in the ATALANTA Tool. Fig. 4 represents the comparative Bar chart for test pattern count and fault coverage of the Group-1 circuit. Fig. 5 depicts the comparative bar chart of fault coverage and total test pattern count for Group-2 circuits. Table 4 represents the sensitivity analysis performed by evaluating a Random Forest Regressor with the number of estimators as 5,8, and 50 trees. Results showed that

RFR with 5 estimators performed best for 4 benchmark circuits and was equivalent for 2 circuits, and the remaining two circuits had the best results with 8 estimators. Increasing the number of estimators to 50 did not perform better; rather, fault coverage was lower in all 8 benchmark circuits.

So, selecting 8 estimators is considered sufficient for this proposed work. Also, Table 3 and Table 4 compare RFR (with 5, 8, 50 trees) results with ATALANTA, random patterns and SmartATPG [1], ETFA [36], BIST-CBDC [37], MCAT [38], DTGS [39], DUAL MAC [40] and OCRT [41]. All the methods except MCAT and OCRT outperform RFR results, as it mainly follows the traditional ATPG algorithm and performs variations in the ATPG flow. Table 5 indicates the parameters for the ML model for evaluation of Random Forest Regressor, such as MSE and RMSE, based on mean and root mean squared error Eq. (8), Eq. (9) [27]. The smallest root mean squared error is for circuit C1908_23. In the VLSI

Table 5 Sensitivity Analysis of RFR Fault Coverage for Varying Number of Trees

Author(s), Year	Circuit	C432_3	C499_32	C880_4	C1908_23	C3540_23	C5315_13	C6288_35	C7552_45
Circuits Details	# Gates	160	202	383	880	1669	2307	2416	3512
	# Faults	524	758	942	1879	3428	5350	7744	7550
Proposed Method	All RFR (#TP)	57	43	55	104	79	125	32	195
	RFR (5TREES)	87.214	91.029	99.257	55.402	61.61	96.523	93.091	79.391
	RFR (8TREES)	84.733	91.821	90.446	60.404	59.772	96.523	92.936	81.02
	RFR (50TREES)	81.489	90.897	86.943	52.634	60.502	91.607	92.73	79.258
W. Li et al.,2024 [1]	Smart ATPG	98.851	99.846	---	99.79	99.265	99.813	99.958	99.179
X. Liu et al., 2003 [36]	ETFA	---	---	100	99.67	96.29	99.56	99.42	---
S. Das et al., 2012 [37]	BIST-CBDC	95.918	88.889	92.551	83.571	---	97.988	99.306	---
A. Alamgir et al., 2019 [38]	MCAT	87.59	85.88	87.15	70.14	66.65	73.83	---	80.45
Y. Zhang et al., 2010 [39]	DTGS	99.24	100	100	99.89	100	---	99.56	98.25
Sung-il Bae et al.,2002 [40]	DUAL MAC	99.24	98.94	99.36	99.47	95.74	98.9	99.56	94.98
I. Mrozek et al., 2017 [41]	OCRT	76.33	85.35	82.69	57.26	64.67	70.95	---	78.65

Note: TP-Test Patterns, FC- Fault Coverage, RFR- Random Forest Regressor , “---: not reported/not available”

Verification and Test domain, the total number of faults identified with fewer test patterns is the key concern compared to parameters like MSE and RMSE. As each test pattern is a set of numerical values in binary up to 200 bits, if any test pattern deviates in the MSBs or higher-order bits, the MSE and RMSE of this particular circuit remain higher. This deviation in MSBs or higher-order bits in test patterns compared to the ATALANTA tool leads to higher MSE and RMSE for c880-4 and c7552_45 circuits. Table 5 represents the execution time range for testing different circuits, from 14.6 ms to 16.9 ms. Table 6 presents the experimental results obtained on the ISCAS-89 benchmark circuits using the proposed RFR model. The predicted test vectors were validated using the FSIM fault simulator, and the achieved fault coverage was compared with the reference ATALANTA ATPG tool. This further validates the generalization capability of the proposed framework by external testing on ISCAS-89 benchmark circuits, demonstrating its performance on previously unseen circuit structures. The proposed approach achieved fault coverage ranging from 64.186% to 94.152%, thereby confirming the feasibility of ML-assisted test vector prediction beyond the training benchmark family.

V. Discussion

The results obtained show that the machine learning model can learn the correlation between the characteristics of the circuit and the test pattern sets corresponding to them. The proposed approach shows fault coverage nearly the same as that of a conventional ATPG tool for most circuits, and lower for certain circuits. This model performance can be impacted by different types and levels of circuits in the data sets, the quality of the features and the size of the industrial testing sets. It can be clearly identified that the total number of test patterns in the test set of RFR and ATALANTA is almost the same. The RFR model predicts a similar order of Fault coverage for most of the test circuits, but predicts less fault coverage for the remaining circuits. Overall, the Random Forest Regressor ML model can predict the test pattern set for

ATPG. Compared to the ATALANTA Tool, its performance is slightly lower, with less fault coverage. However, it offers competitive results for the test circuits. Random Forest Regressor identifies intricate and nonlinear correlations between circuit characteristics and their associated test pattern sets using ensemble learning.

The results show the capability of machine learning as a supplementary tool to the traditional ATPG techniques in VLSI testing. The suggested model can help to reduce test generation workload and may enable the creation of an intelligent CAD tool for future applications in semiconductor testing.

The experimental results indicate that the proposed Random Forest Regressor is influenced by circuit-specific characteristics, including structural complexity, logic depth, and testability properties that are not explicitly captured by the current feature set. Consequently, comparable fault coverage is achieved for some benchmark circuits, whereas others exhibit reduced performance. Future work will investigate richer structural features, benchmark-wise validation, and SHAP-based feature importance analysis to improve model robustness and interpretability.

A. Challenges while using ML in Test domain:

There are many steps in IC testing, such as test generation, design for testability, fault simulation, fault isolation, and insertion of test points [42], failure detection, and fault localization, where ML methodologies can simplify the efforts of standard tools and can also help in minimizing computation requirements. In the entire chip design life cycle, a significant amount of time is consumed in testing [43], [44]. As circuit complexity and size increase [45], the test pattern count and testing expenditure increase by a large amount. RFR model is applied for test pattern generations as well as fault identification and achieves comparable results to the ATALANTA tool [46]. To investigate the capabilities of machine learning in integrated circuit testing, several researchers have used ML approaches at various phases of the testing procedure. To effectively train the ML model, balanced, adequate, enriched industry data should be easily accessible. It is unfortunate that standard, balanced industry-based IC testing data for ML models is yet to be developed. The dearth of uniform datasets creates a substantial challenge to the implementation of ML-based methodologies. Primary hindrances are listed here:

- 1) Industrial datasets associated with testing, failure analysis, and diagnosis are largely unavailable in the public domain. Such datasets would serve as valuable resources for training and validating the ML methodologies.
- 2) ML Techniques often require extensive quantities of data for successful training. However, producing

Table 6 Fault-Coverage Performance of the Proposed RFR Model on Unseen ISCAS89 Benchmark Circuits

ISCAS89 Circuit	# Total Faults	%FC ATALANTA	#TP RFR (8TREES)	%FC RFR (8 TREES)
S208	215	100	303	64.186
S298	308	100	303	89.610
S344	342	100	152	94.152
S641	463	100	78	84.017
S713	581	93.46	78	81.411
S5378	4551	99.121	230	80.136

synthetic datasets through simulation is a computationally costly and time-consuming operation.

- 3) To achieve a well-trained model, an extensive benchmark circuit set containing diverse architectures, numbers of gates, different gate types, circuit levels, and interconnection topologies is required.

Effective feature extraction is essential in machine learning, since the quality and relevance of extracted features directly affect the model's prediction ability and generalization performance. Hence, Significant time and effort are required to generate extensive data, preprocess the data, and extract features from it.

B. Limitations and Further Scope:

Although this research work, based on a Random Forest Regressor, predicts well for most of the benchmark group circuits, the approach still exhibits some limitations. The scope of this work is constrained to a Random Forest regression-based approach with a dataset of 980 circuits, which is not suitable for very small datasets; it also requires complex hyperparameter tuning, such as the number of trees, tree depth and minimum samples per split, which reduces interpretability and increases computational complexity. Future work may involve the exploration of advanced methodologies, like Artificial Neural Network Regressor, for the prediction of digital circuit test vectors. These models may enhance prediction accuracy, robustness, and generalization performance relative to other regression methods. In addition, the model performance is affected by dataset imbalance and the selection of relevant features. Furthermore, the integration of larger and more diverse benchmark Datasets, such as ISCAS-89 and ITC-99, along with feature optimization techniques, can improve the scalability and applicability of the model for real-world VLSI testing environments. The incorporation of leave-one-benchmark-out validation is considered an important direction for future work to measure the robustness of the model on unseen tuples.

VI. Conclusion

The integration of machine learning into test vector generation has become an emerging direction and aim of the research work to enhance test set generation and achieve comparable fault coverage in IC testing. Random Forest Regressor approach is employed for test pattern creation focused on fault identification and exhibits performance comparable to the ATALANTA tool, utilizing an equivalent quantity of test patterns. It has achieved fault coverage similar to that of the conventional ATPG tool for 5 circuits, and lower fault coverage for 3 circuits. Experimental results of testing the RFR model have similar fault coverage of 91.821,

99.257, 61.610, 96.523, and 93.091 for C499, C880, C3540, C5315, and C6288 circuits, respectively. A few circuits, such as C432, C1908, and C7552, have lower fault coverage: 87.214, 60.404, and 81.020, respectively. Circuits C5315 and C6288 achieved fault coverage differences of 0.15% and 0.388%, respectively, compared with the ATALANTA-generated results, demonstrating that the selected circuit features adequately characterize their testing behavior. Conversely, C1908 and C3540 exhibited larger differences in fault coverage, with the maximum gap reaching 17.19%. However, the average fault coverage difference is 7.44%. This lower fault coverage in a few circuits is likely due to increased circuit complexity and the limited representational capability of the feature set. As the model relies primarily on global circuit characteristics, it does not capture structural properties that influence ATPG, including reconvergent fanout, logic depth, signal controllability and observability. The proposed system should be seen as a supplementary machine learning-based approach for test pattern prediction, which is not a direct substitute for deterministic ATPG. Incorporating structural features, along with feature-importance or SHAP analyses, may improve the interpretability and predictive performance of the proposed framework. Hence, it could be considered as a future research direction.

Acknowledgment

For the preparation of the dataset and result comparisons, the fault simulator applications ATALANTA (open source, Version 2.0) and FSIM (open source, Version 1.2) were utilized. The authors thank Dong S. Ha, Virginia Polytechnic Institute and State University, for these tools. ATALANTA and FSIM are publicly available as free academic tools, and the ISCAS85 and ISCAS89 circuit sets are standard public domain benchmarks. The open-source tools used in this study are publicly available external resources. In contrast, the synthetically generated dataset comprising approximately 980 circuits, along with the associated preprocessing and processing code, are author-generated materials. Due to the confidentiality of the ongoing research work, the author-generated dataset and code are not publicly available at present; however, they can be made available upon reasonable request after completion of the research work.

Funding

This research project has not received any support/fund from any source.

Author Contribution

Sima K Gonsai completed the Python-based framework, created and evaluated the experimental data, and

conducted a comparative evaluation using a standard ATPG tool. She performed the methodology development, software implementation, validation, formal analysis, investigation, and manuscript writing, including the original draft and subsequent review and editing. Usha Mehta proposed the central research idea and performed the conceptualization, supervision, and manuscript review and editing. She performed conceptualization involving the utilization of ML in VLSI testing, oversaw the overall research tasks, and helped with manuscript assessment and revision. Both authors reviewed and approved the manuscript's final version.

Conflicts of Interest

The authors declare that there are no conflicts of interest in this article.

Ethical Approval

Not applicable for this research work as the study did not involve human participants, animals, biological specimens, or identifiable personal data.

Informed Consent

Not applicable for this research work.

Data and Code Availability

The datasets generated and analyzed during the current study and the codes are not publicly available, as they form part of an ongoing research project. They can be made available at a later stage upon completion of the research, subject to applicable research and confidentiality restrictions.

References

- [1] W. Li, H. Lyu, S. Liang, T. Wang, and H. Li, "SmartATPG: Learning-based Automatic Test Pattern Generation with Graph Convolutional Network and Reinforcement Learning," in *Proceedings of the 61st ACM/IEEE Design Automation Conference*, New York, NY, USA: ACM, Jun. 2024, pp. 1–6. doi: 10.1145/3649329.3656526.
- [2] Y. ATTAOUI, M. Chentouf, Z. E. A. Alaoui Ismaili, and A. Elmourabit, "Machine Learning in VLSI Design: A Comprehensive Review," *Journal of Integrated Circuits and Systems*, vol. 19, no. 2, pp. 1–14, Aug. 2024, doi: 10.29292/jics.v19i2.826.
- [3] W. Fang et al., "A Survey of Circuit Foundation Model: Foundation AI Models for VLSI Circuit Design and EDA," *ACM Trans. Des. Autom. Electron. Syst.*, Aug. 2026, doi: 10.1145/3827602.
- [4] Virginia Polytechnic & State University, "User's Guide for ATALANTA," GitHub. [Online], <https://github.com/hsluoyz/Atalanta>, GitHub.
- [5] [Online]. Accessed: Sep. 06, 2026. [Online]. Available: <https://github.com/hsluoyz/Atalanta>
- [6] H. Dhotre, S. Eggersglub, K. Chakrabarty, and R. Drechsler, "Machine Learning-based Prediction of Test Power," in *2019 IEEE European Test Symposium (ETS)*, IEEE, May 2019, pp. 1–6. doi: 10.1109/ETS.2019.8791548.
- [7] R. Pan, Z. Zhang, X. Li, K. Chakrabarty, and X. Gu, "Black-Box Test-Coverage Analysis and Test-Cost Reduction Based on a Bayesian Network Model," in *2019 IEEE 37th VLSI Test Symposium (VTS)*, IEEE, Apr. 2019, pp. 1–6. doi: 10.1109/VTS.2019.8758639.
- [8] K.-C. Yang, M.-T. Lee, C.-H. Wu, and J. C.-M. Li, "ATPG and Test Compression for Probabilistic Circuits," in *2019 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, IEEE, Apr. 2019, pp. 1–4. doi: 10.1109/VLSI-DAT.2019.8741869.
- [9] S. Millican, Y. Sun, S. Roy, and V. Agrawal, "Applying Neural Networks to Delay Fault Testing: Test Point Insertion and Random Circuit Training," in *2019 IEEE 28th Asian Test Symposium (ATS)*, IEEE, Dec. 2019, pp. 13–135. doi: 10.1109/ATS47505.2019.000-7.
- [10] H. Yang, J. Su, Y. Zou, B. Yu, and E. F. Y. Young, "Layout Hotspot Detection with Feature Tensor Generation and Deep Biased Learning," in *Proceedings of the 54th Annual Design Automation Conference 2017*, New York, NY, USA: ACM, Jun. 2017, pp. 1–6. doi: 10.1145/3061639.3062270.
- [11] A. F. Tabrizi, N. K. Darav, L. Rakai, I. Bustany, A. Kennings, and L. Behjat, "Eh?Predictor: A Deep Learning Framework to Identify Detailed Routing Short Violations From a Placed Netlist," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 6, pp. 1177–1190, Jun. 2020, doi: 10.1109/TCAD.2019.2917130.
- [12] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, Jan. 1989, doi: 10.1016/0893-6080(89)90020-8.
- [13] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. Sohl-Dickstein, "On the Expressive Power of Deep Neural Networks," *Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, PMLR 70*, 2017, Jun. 2017.
- [14] Y. Ma et al., "High Performance Graph Convolutional Networks with Applications in Testability Analysis," in *Proceedings of the 56th Annual Design Automation Conference 2019*, New York, NY, USA: ACM, Jun. 2019, pp. 1–6. doi: 10.1145/3316781.3317838.

- [14] B. Sun *et al.*, "InF-ATPG: Intelligent FFR-Driven ATPG with Advanced Circuit Representation Guided Reinforcement Learning," *56th Annual Design Automation Conference (DAC 2019)*, Nov. 2025, doi: 10.48550/arXiv.2512.00079.
- [15] K. Eykholt *et al.*, "Robust Physical-World Attacks on Deep Learning Visual Classification," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2018, pp. 1625–1634. doi: 10.1109/CVPR.2018.00175.
- [16] Motamarri Venkata Saikumar, Fazal Noorbasha, K. Srinivasa Rao, and K. Girija Sravani, "Enhancing Fault Detection in Digital Circuits Using Machine Learning and LFSR-Based Test Pattern Generation," *International Journal of Computational and Experimental Science and Engineering*, vol. 11, no. 3, Aug. 2025, doi: 10.22399/ijcesen.3231.
- [17] L. Sommeregger and J. Pilz, "Regularizing Lifetime Drift Prediction in Semiconductor Electrical Parameters with Quantile Random Forest Regression," *Technologies by MDPI*, vol. 12, no. 9, p. 165, Sep. 2024, doi: 10.3390/technologies12090165.
- [18] M. Moness, L. Gaber, A. I. Hussein, and H. M. Ali, "Automated Design Error Debugging of Digital VLSI Circuits," *Journal of Electronic Testing*, vol. 38, no. 4, pp. 395–417, Aug. 2022, doi: 10.1007/s10836-022-06020-z.
- [19] L. Gaber, A. I. Hussein, and M. Moness, "Incremental Automatic Correction for Digital VLSI Circuits," in *Computer Science & Information Technology (CS & IT)*, AIRCC Publishing Corporation, Nov. 2020, pp. 83–93. doi: 10.5121/csit.2020.101508.
- [20] Guolin Ke *et al.*, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," in *31st Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, CA, USA., Dec. 2017, pp. 3149–3157.
- [21] S. R. Research Scholar, C. Assistant Professor, and G. Sudhagar, "An Overview on Strategies to Perform VLSI Testing Productively Utilizing BIST," *Int. J. Advanced Networking and Applications*, vol. 16, no. 2, pp. 6342–6348, Aug. 2024, doi: 10.35444/IJANA.2024.16205.
- [22] J. E. Nelson, W. C. Tam, and R. D. Blanton, "Automatic classification of bridge defects," in *2010 IEEE International Test Conference*, Austin, TX, USA: IEEE, Nov. 2010, pp. 1–10. doi: 10.1109/TEST.2010.5699231.
- [23] Z. Li, J. E. Colburn, V. Pagalone, K. Narayanun, and K. Chakrabarty, "Test-cost optimization in a scan-compression architecture using support-vector regression," in *2017 IEEE 35th VLSI Test Symposium (VTS)*, Las Vegas, NV, USA: IEEE, Apr. 2017, pp. 1–6. doi: 10.1109/VTS.2017.7928956.
- [24] K. Fu, Y. Jin, and Z. Chen, "Test Set Optimization by Machine Learning Algorithms," *ArXiv*, Oct. 2020, doi: arXiv:2010.02108.
- [25] N. R. A. Karthikeyan, "Design and implementation of multiple fault diagnosis on VLSI circuits using artificial neural networks," *International Journal of Advanced Engineering Technology*, vol. 3, no. 2, pp. 685–687, Apr. 2012.
- [26] A. B. Chowdhury, B. Tan, S. Garg, and R. Karri, "Robust Deep Learning for IC Test Problems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 1, pp. 183–195, Jan. 2022, doi: 10.1109/TCAD.2021.3054808.
- [27] K. Lad and M. Joshi, "DCRNet: Hybrid Deep Learning Architecture for Forecasting of Blood Glucose," *Journal of Electronics, Electromedical Engineering, and Medical Informatics*, vol. 8, no. 1, pp. 53–68, Jan. 2026, doi: 10.35882/jeeemi.v8i1.1245.
- [28] N. L. Johnson, S. Kotz, and X.-Z. Wu, *Inspection Errors for Attributes in Quality Control*. Chapman and Hall/CRC, 2020. doi: 10.1201/9781003059868.
- [29] H. Fujiwara, "ISCAS'85 Benchmarks: Special Session on ATPG and Fault Simulation, 1985 IEEE International Symposium on Circuits and Systems." [Online]. Available: <http://www.cbl.ncsu.edu:16080/benchmarks/ISCAS85/>
- [30] M. Lipovsky, J. Svarc, E. Gramatova, and P. Fiser, "A new user-friendly ATPG platform for digital circuits," in *2016 IEEE 19th International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS)*, IEEE, Apr. 2016, pp. 1–4. doi: 10.1109/DDECS.2016.7482474.
- [31] Laung-Terng Wang, Cheng-Wen Wu, and Xiaoqing Wen, *VLSI TEST PRINCIPLES AND ARCHITECTURES*. San Francisco, United States: Morgan Kaufmann Publishers Inc, 2006.
- [32] F. Ye, Z. Zhang, K. Chakrabarty, and X. Gu, "Board-Level Functional Fault Diagnosis Using Multikernel Support Vector Machines and Incremental Learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 2, pp. 279–290, Feb. 2014, doi: 10.1109/TCAD.2013.2287184.
- [33] M. Pradhan, B. B. Bhattacharya, and K. Chakrabarty, "Predicting X-Sensitivity of Circuit-Inputs on Test-Coverage: A Machine-Learning Approach," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*,

- vol. 38, no. 12, pp. 2343–2356, Dec. 2019, doi: 10.1109/TCAD.2018.2878169.
- [34] S. Jin, F. Ye, Z. Zhang, K. Chakrabarty, and X. Gu, "Efficient Board-Level Functional Fault Diagnosis With Missing Syndromes," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 6, pp. 985–998, Jun. 2016, doi: 10.1109/TCAD.2015.2481859.
- [35] S. Mittal and R. D. Blanton, "LearnX: A Hybrid Deterministic-Statistical Defect Diagnosis Methodology," in *2019 IEEE European Test Symposium (ETS)*, IEEE, May 2019, pp. 1–6. doi: 10.1109/ETS.2019.8791512.
- [36] X. Liu, M. S. Hsiao, S. Chakravarty, and P. J. Thadikaran, "Efficient Transition Fault ATPG Algorithms Based on Stuck-At Test Vectors," *Journal of Electronic Testing*, vol. 19, no. 4, pp. 437–445, Aug. 2003, doi: 10.1023/A:1024696110831.
- [37] S. Das, L. Jin, M. Assaf, S. Biswas, and E. Petriu, "Implementing built-in self-test environment for cores-based digital circuits with Verilog HDL," *World Journal of Engineering*, vol. 9, no. 6, pp. 519–528, Dec. 2012, doi: 10.1260/1708-5284.9.6.519.
- [38] A. Alamgir, A. Khari Bin A'Ain, U. U. Sheikh, N. Paraman, M. M. Mokji, and I. Grout, "Multiple Controlled Antirandom Testing (MCAT) for High Fault Coverage in a Black Box Environment," *IEEE Access*, vol. 7, pp. 117246–117257, 2019, doi: 10.1109/ACCESS.2019.2937113.
- [39] Y. Zhang and V. D. Agrawal, "A diagnostic test generation system and a coverage metric," in *2010 15th IEEE European Test Symposium*, IEEE, May 2010, pp. 254–254. doi: 10.1109/ETSYM.2010.5512739.
- [40] Sung-il Bae, Sungchul Yoon, Sangwook Kim, and Sungho Kang, "Test Pattern Generation Using a Dual MAC Unit," *Journal of the Korean Physical Society*, vol. 41, no. 6, pp. 974–981, Dec. 2002.
- [41] I. Mrozek and V. Yarmolik, "Optimal Controlled Random Tests," 2017, pp. 27–38. doi: 10.1007/978-3-319-59105-6_3.
- [42] Y. Huang, B. Benware, R. Klöngenberg, H. Tang, J. Dsouza, and W.-T. Cheng, "Scan Chain Diagnosis Based on Unsupervised Machine Learning," in *2017 IEEE 26th Asian Test Symposium (ATS)*, IEEE, Nov. 2017, pp. 225–230. doi: 10.1109/ATS.2017.50.
- [43] S. Roy, S. K. Millican, and V. D. Agrawal, "Training Neural Network for Machine Intelligence in Automatic Test Pattern Generator," in *2021 34th International Conference on VLSI Design and 2021 20th International Conference on Embedded Systems (VLSID)*, IEEE, Feb. 2021, pp. 316–321. doi: 10.1109/VLSID51830.2021.00059.
- [44] G. Thakur, S. Jain, and H. Sohal, "Current issues and emerging techniques for VLSI testing - A review☆," *Measurement: Sensors*, vol. 24, p. 100497, Dec. 2022, doi: 10.1016/j.measen.2022.100497.
- [45] S. Eggersgluss, K. Schmitz, R. Krenz-Baath, and R. Drechsler, "On Optimization-Based ATPG and Its Application for Highly Compacted Test Sets," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 12, pp. 2104–2117, 2016, doi: 10.1109/TCAD.2016.2552822.
- [46] S. R. Das, C. V. Ramamoorthy, M. H. Assaf, E. M. Petriu, W.-B. Jone, and M. Sahinoglu, "Fault Simulation and Response Compaction in Full Scan Circuits Using HOPE," *IEEE Trans. Instrum. Meas.*, vol. 54, no. 6, pp. 2310–2328, Dec. 2005, doi: 10.1109/TIM.2005.858102.

Authors Biography



Sima K Gonsai is currently working as an assistant professor in the Department of Electronics and Communication at L. D. College of Engineering, Ahmedabad. She has received a B.E. in EC Engineering from U. V. Patel college of

Engineering in 2004 and an M.E. in EC with a specialization in Communication Systems Engineering from the L. D. College of Engineering in 2007. Currently, she is a research scholar at the Institute of Technology, Nirma University, Ahmedabad. She has 19 years of teaching experience. She has guided 7 PG (M.E. E.C) students. Her area of research is VLSI Testing for digital circuits, Machine learning, Artificial Intelligence and micro controllers.



Prof. (Dr.) Usha Mehta, currently a Professor and Head at the EC Department, Institute of Technology, Nirma University, Ahmedabad. She received her bachelor degree in Electronics & Communication Engineering in 1994 and master in 2005 and Ph. D.

degree in VLSI Design in 2011. She has more than 23 years of experience for teaching at Under Graduate and Post Graduate level. She has guided more than six Ph.D. students and M. Tech by Research students. She has one Patent on her credit. She has been the Principle Investigator for research projects of ISRO and GUJCOST. RHBD Chip developed by her and her teammates for ISRO has been fabricated at SCL,

Chandigarh, and is in use by the Space Application Centre. She has published more than 55 research papers in the area of VLSI Design and Testing. She has delivered a large number of technical talks at various conferences and workshops.