Journal of Electronic Testing (2024) 40:139–158
https://doi.org/10.1007/s10836-024-06117-7

A Survey and Recent Advances: Machine Intelligence in Electronic Testing

Soham Roy[1] · **Spencer K. Millican**[2] · **Vishwani D. Agrawal**[3]

Received: 20 November 2023 / Accepted: 27 March 2024 / Published online: 15 April 2024

Abstract
Integrated circuit (IC) testing presents complex problems that for large circuits are exceptionally difficult to solve by traditional computing techniques. To deal with unmanageable time complexity, engineers often rely on human “hunches" and “heuristics" learned through experience. Training computers to adopt these human skills is referred to as machine intelligence (MI) or machine learning (ML). This survey examines applications of such methods to test analog, radio frequency (RF), digital, and memory circuits. It also summarizes ML applications to hardware security and emerging technologies, highlighting challenges and potential research directions. The present work is an extension of a recent paper from IEEE VLSI Test Symposium (VTS’21), and includes recent applications of artificial neural network (ANN) and principal component analysis (PCA) to automatic test pattern generation (ATPG).

Keywords Machine intelligence (MI) · Machine learning (ML) · Analog testing · Digital testing · Memory test and repair · RF testing · Hardware security · Artificial neural network (ANN) · Principal component analysis (PCA).

1 Introduction

Integrated circuit (IC) defects behave differently depending on the type of circuit, requiring separate test methodologies. Analog and radio frequency (RF) tests are functional and derived from high-level specifications [82], digital tests are structural and target modeled faults [18], and memory tests also target modeled faults but test them in a functional manner [3]. For any circuit type, increasing integration reduces cost, but testing must address the increased complexity and test for nuanced faults not seen in previous generations of circuit technology.

Problems like digital test pattern generation are computationally complex while those such as integrated circuit (IC) yield enhancement are not easily addressable by simple algorithms. Human intuition often helps but the cost of employing teams of experienced engineers to apply their intuition can be nontrivial. In this situation, engineers can apply machine learning (ML), also known as machine intelligence (MI), to create novel solutions for test problems. Besides, ML also makes programming easier and reduces software development cycles and costs.

Previous surveys [119, 152] have discussed ML applications to testing. Our recent article at the VLSI Test Symposium (VTS’21) [128] explored additional areas absent from the previous surveys. The present article provides some details from previous publications. In addition, recent applications of ML to automatic test pattern generation (ATPG) are summarized in Section 3.9. These are,

- Establish the feasibility of training artificial neural network (ANN) to guide an ATPG algorithm [126].
- Optimize the training of ANN for ATPG [129].
- Use principal component analysis (PCA) [69, 117] to combine multiple heuristics in ATPG [130].
- Impact of ML guidance on the performance of ATPG [127].
- Use PCA to combine multiple heuristics for backtracing and *D*-drive [48] in a practical ATPG system (i.e., random patterns followed by algorithmic vectors) [131].

Responsible Editor: H.-G. Stratigopoulos

✉ Soham Roy
soham.roy@intel.com

Spencer K. Millican
spencer.k.millican@dynetics.com

Vishwani D. Agrawal
agrawvd@auburn.edu

[1] Intel Corp., Santa Clara, CA 95054, USA

[2] Dynetics Inc., Huntsville, AL 35806, USA

[3] Auburn Univ., Auburn, AL 36849, USA

Section 3.9 is derived from authors' recent research in which they try to follow the elusive goal of zero backtracks in ATPG [124]. The results show improvement from the past but cannot claim ultimate optimality. Indeed, they point to a possible path for the future, and that is the purpose of this survey.

Rest of this article is organized as follows. Section 2 discusses ML applications in testing of analog and RF circuits. Section 3 explores new ML techniques for digital circuits, which is an additional contribution beyond the previous surveys. Memory testing is the subject of Section 4. Section 5 concludes the survey by listing some open test-related challenges yet to be addressed by ML.

2 Analog and RF Testing

Analog and radio frequency (RF) components are integral parts of modern electronics, and testing them requires sophisticated equipment and methods. Such devices demand more time and indirectly increase manufacturing costs. A common belief among engineers is that even though the analog and mixed-signal parts may occupy around 10% of the chip area, the rest being digital, they take 90% of the testing effort. This is mainly because analog testing is specification-based while digital testing relies on fault models permitting effective use of computer tools.

Efforts to reduce test time have led to alternate test strategies: generating signatures that differentiate between faulty and fault-free circuits [2, 144]; built-in test (BIT) or the use of an on-chip tester [55, 134] that switches the device under test (DUT) into test mode by fetching signals from sensors [1, 31, 51, 73, 104, 169]; built-off test (BOT) or converting RF signals to DC signals using an interface (placed on a load board) between the DUT and tester [13, 40]; and implicit test, i.e., statistical model-based test that can make an off-line PASS/FAIL decision [5, 155, 170].

Complete automation in this area has been an elusive goal, and that is where machine learning has begun to play a role [37, 153].

2.1 Use of Machine Learning

Machine learning can play an important role in testing of analog and radio frequency devices, because here the decision of a test passing or failing is not as straightforward as in a digital test. We use built-in self-test (BIST) for an RF device under test (DUT), such as a low noise amplifier (LNA), for illustration. A proposed architecture [101] consists of a stimulus generator, measurement acquisition sensors, and an artificial neural network (ANN) to provide PASS/FAIL decision. During the offline training or test phase, ANN translates measured (test) data into a one-bit output, indicating whether it is in compliance with the DUT specification (see Fig. 1). The training phase selects a suitable ANN topology, e.g., number of hidden layers, number of neurons per hidden layer, etc., as well as the weights assigned to the internal synapses. The weights are saved in a local memory and downloaded during the test. Self-test is applied by connecting the DUT with a test stimulus generator. On-chip sensors provide the ANN with relevant data from the DUT. Analyzing the test data in relation to the learned classification boundary is how the ANN classifies the DUT. Beside training on fabricated chips, the technique has also been further enhanced [156].

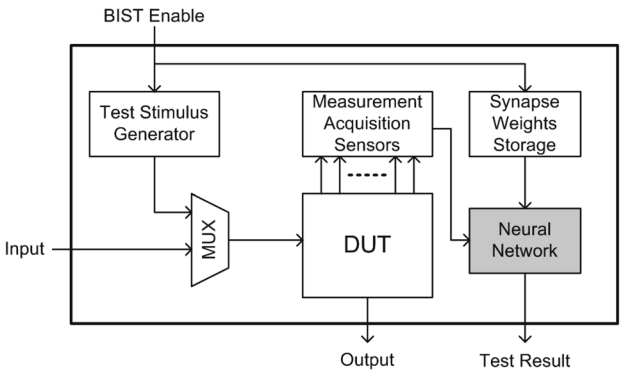

Fig. 1 Built-in self-test (BIST) of a radio frequency (RF) device under test (DUT) [101]

For an effective implementation of the BIST circuit shown in Fig. 1, area and power consumption of the ANN hardware should be low. An analog ANN on silicon densely packs synapses and computing elements for superior parallel processing ability, robustness, and fault tolerance. Compared to a digital implementation it is faster, smaller, easy to reconfigure and train, and consumes less power. However, analog ANN design must consider 1) topology, 2) training algorithm, and 3) weight/bias storage. Fabrication technology makes implementing analog ANN on silicon difficult since conventional CMOS technologies have significant parameter variations [67, 99, 106, 110].

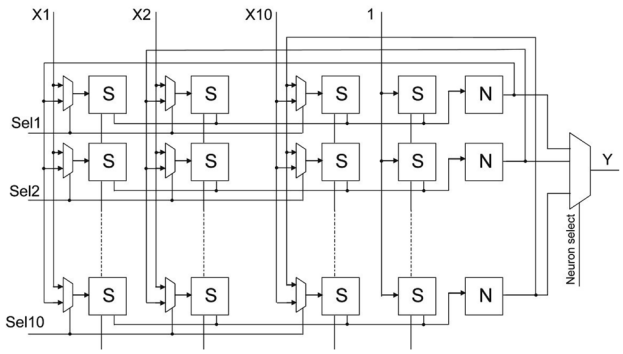

Fig. 2 Reconfigurable ANN [101]

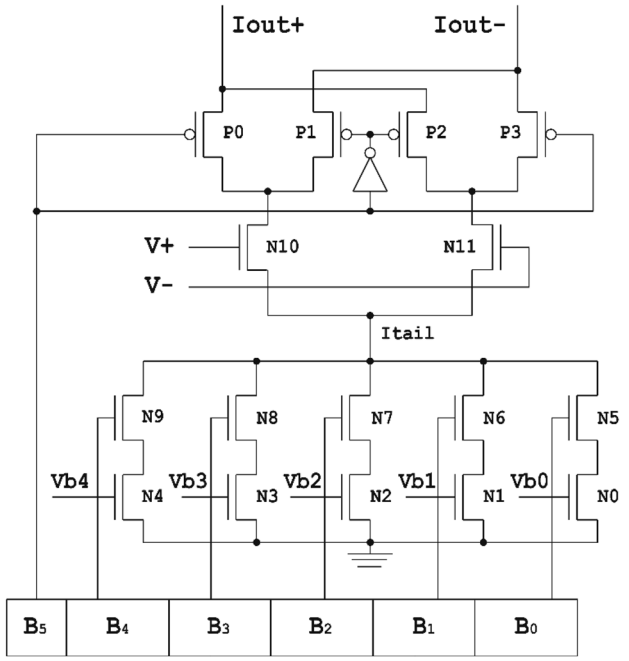

Fig. 3 Schematic diagram of synapse [101]

Figure 2 illustrates an architecture of a reconfigurable, single hidden layer ANN [101]. It comprises of synapses (*S*), multiplexers, and neurons (*N*) in a matrix. Each synapse is mixed-signal hardware that performs computation in analog mode while storing weights and biases in a digital random access memory (RAM). The schematic of a typical synapse circuit, shown in Fig. 3, illustrates multiplication implemented through a digital-to-analog converter (DAC) [86], a combination of differential input voltages, and programmable tail currents. The upper half of Fig. 3 is a differential pair "N10-N11" performing multiplication while switching transistors "P0-P3" controlled by bit "B5", steer the current and define the sign of the multiplication. In the lower half, five switching transistors digitally program the tail currents "N5-N9" and binary-weighted current sources "N0-N4". Thus, the tail current depends on the digital word "B0-B4". Since multiplication in analog circuitry is area-expensive, approximate multiplication is common but may be non-linear, which can be mitigated by using customized backpropagation algorithms [99]. Multiplexers select input sources from previous layers, and the summation of synapses is fed into a neural circuit, as illustrated in Fig. 4. This neuron circuit converts synapse outputs, i.e., the differential currents, into differential voltages. The common-mode cancellation circuit produces a positive difference from "I_{in}^{+}" and "I_{in}^{-}." The next stage is a current–voltage converter made up of two p-channel MOSFETs. The last stage, a level shifter, is a source follower circuit that shifts the output voltage from the previous stage upward to match the high voltage requirement of synapses in the next layer(s). This architecture has following advantages:

1. It is modular and can expand to any number of neurons and inputs within the chip area.
2. Output multiplexer reduces the number of pins and analog-to-digital converter (ADC) devices.
3. All signals are differential with broad input ranges thus providing improved noise resiliency.

Conventional training algorithms (i.e., backpropagation algorithms) for on-chip ANNs suffer from low precision and high area overhead. A parallel stochastic weight perturbation technique [76] may be preferred since it does not require on-chip support and provides a compact solution. In this method, random vectors perturb all edge weights of the ANN. The mean squared error (MSE) is calculated over the entire training set to check the error status. If the error decreases, the new random vector with weights is accepted, otherwise it is discarded. This method is likely to get trapped in local minima, which can be avoided by using a simulated annealing technique, allowing the state of the network to move "uphill."

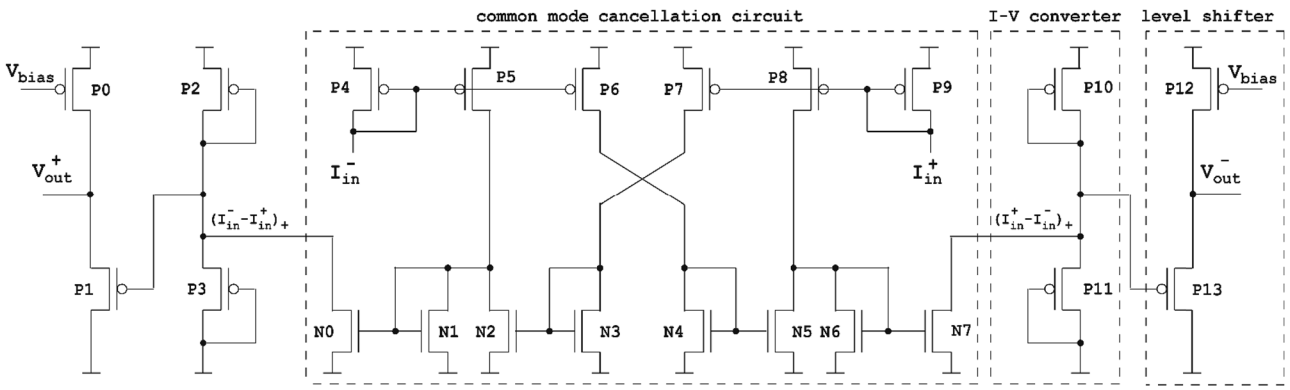

Fig. 4 Schematic diagram of neuron [101]

In an experiment on low noise amplifier (LNA) circuits two RF amplitude detectors placed at the input and output produced DC signals proportional to RF power at detector inputs [101]. These DC signals were fed to an analog ANN classifier, trained in different configurations with 2, 4, and 8 neurons in a single hidden layer. This was repeated five times to average out randomness of the training algorithm's stochastic nature. Additional experiments replaced the hardware classifier with a software classifier using the Matlab neural network toolbox trained by a resilient backpropagation algorithm. It was observed that the software classifier training error outperforms the hardware classifier, but the validation error was comparable in both cases. However, for more neurons in the hidden layer the hardware classifier's validation error is substantial, compared to the software classifier. Several future research directions were reported by this study:

1. The accuracy of the hardware classifier is lower than the software classifier due to non-linearity in synapse multiplication, limited resolution and dynamic range of weight values, and the training algorithm's limitations.
2. The dynamic range of synapses can be improved using adjustable gains, i.e., by changing gain when weights become too low or saturated [66].
3. Weight resolution is problem-specific and depends on network architecture. However, it can be increased in the presence of high non-linearity for minimal size devices but may lead to mismatch and parameter variation in the manufacturing process [95].
4. The training algorithm demonstrates significant convergence properties with minimal variance of the final error, but this requires increased training time.
5. Weight storage is large since it is implemented as digital memory. However, in built-in self test (BIST), these weights need to be stored permanently, which may require memories using floating gate transistors [56, 59]. Nevertheless, using floating gate memories to store weights of analog neural networks may further raise issues like handling of high voltage, accurate programming schemes, and weight updates.
6. Further investigation is needed on whether the ML-based approach considers the effects of DUT degradation during device lifetime, which includes the ANN as well.

2.2 Parametric Test Metrics Based on Machine Learning

Test engineers have procedures to estimate analog parameters related to test costs, yield loss, and test escapes. However, these metrics should be accurately estimated by simulation ahead of silicon manufacturing. A test strategy [152, 154] that includes ML algorithms is shown in Fig. 5 and illustrates the following points:

1. A trained ANN classifies circuits whose parametric metrics are estimated closer to the specification, known as "extreme" instances.
2. Circuit netlists are synthesized using a process design kit (PDK) [152] from intellectual property (IP) vendors. The procedure simultaneously trains an ANN with process metrics to classify "extreme" circuits. These represent rare occurrences identified by a special Monte Carlo technique known as statistical blockade [147].
3. The ANN is re-trained with new simulated circuits to push the boundary such that performance of the extreme class of circuits matches even closer to the specification. This process continues with the re-trained boundary in the pursuit of collecting true "extreme" instances (circuits having performance values marginally satisfying to the specification), and push the training boundary to generate more such "extreme" circuits.

The "extreme" instances can serve as fault models based on parameters, that examine high-performance, and are obtained from an alternative test scheme [9, 157]. This method speeds up the Monte Carlo transistor-level

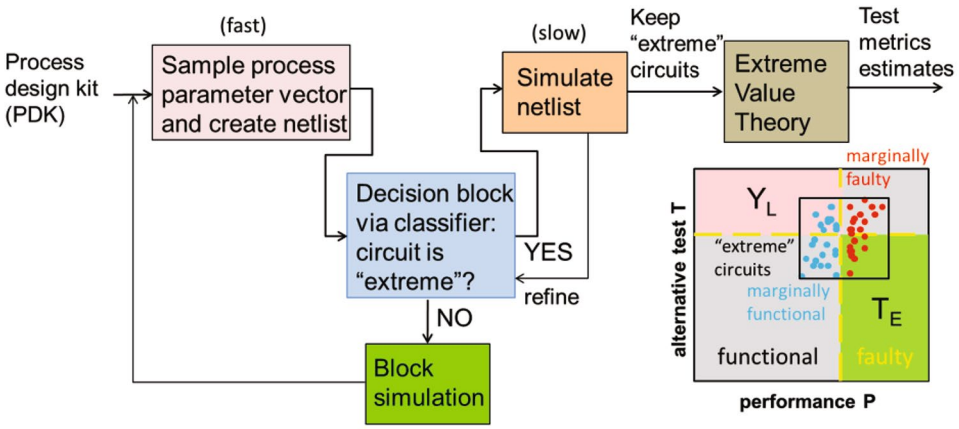

Fig. 5 Simulation flow for parametric test metrics estimation [152]

simulation. Typically, fault models account for process parameters based on a joint distribution as given in their respective PDK [157]. Finally, the fault model is verifiable after performing transistor-level simulation. The algorithm [154] outputs a refined parametric fault model compared to the generalized fault models and helps estimate fault coverage and yield loss more precisely. This method was applied to a low noise amplifier (LNA) [157] and reduced simulation run-time by eliminating the redundant specification tests and replacing them with the proposed ML-based parametric measurements. The technique was also applied to data-converters [9], but is yet to be explored for other analog ICs whose simulation run-time is high, such as phase-locked-loops (PLLs).

3 Digital Testing

In a modern electronic system, digital parts cover most area. Similarly, digital testing occupies most pages in a book on electronic test [18]. As chips become more complex, two types of problems emerge. One, whose complexity is beyond the economically available computing capability, and the other for which the problem itself is too ill-defined is to find an algorithmic solution. Some of these problems have benefited from machine learning.

3.1 Wafer Testing

In general, logic defects occur on wafers in physical clusters [114]. Thus, clustering algorithms [158] can identify defect concentrations across the wafer. They work in two steps: 1) cluster containment and 2) learning. The first step identifies wafers with cluster patterns and screens out passing dies having no defect within these clusters. Those dies are marked for high risk of failure. This process repeats based on cluster size, cluster location on the wafer, and failure composition across multiple wafers to avoid additional yield loss and failure analysis. Recent work [176] proposes a similar cluster-detecting ML algorithm using support vector machine (SVM) [62, 136]. The SVM kernel is a radial basis function, generally a Gaussian function, for distance computation to identify the die from the defective clusters during classification. The corresponding process flow diagram is shown in Fig. 6.

3.2 Scan Chain Defects

Defective scan latches can fail with permanent faults (which are easy to model) or intermittent faults (which are difficult to model). A recent survey [72] points to Bayesian learning [168] for identifying faulty scan cells in the presence of intermittent faults using an unsupervised learning approach.

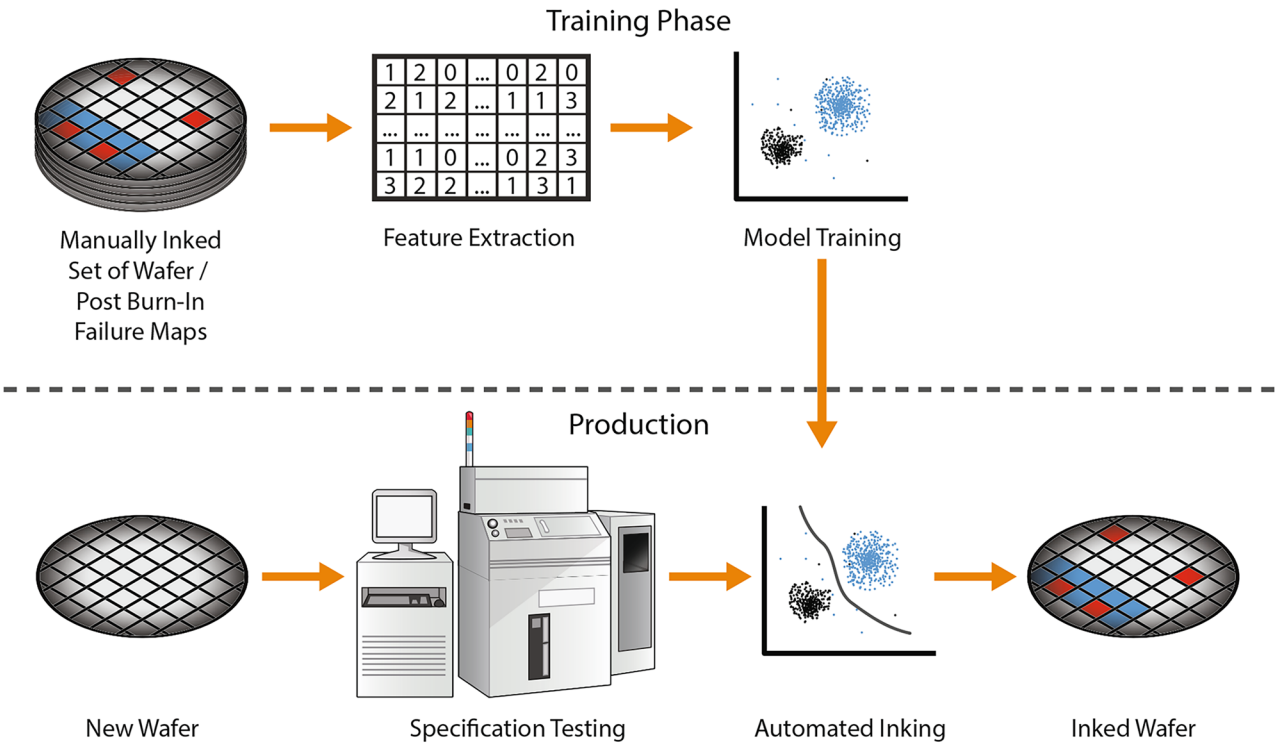

Fig. 6 ML-based die inking process [176]

The method analyzes a test set and the corresponding failure log of the scan chain [71]. The details of this algorithm are explained by assuming a chain fault expressed by a dataset. This dataset contains count of patterns for the respective scan cell that is “sensitive" to the fault (“sbits") and the count of patterns for which a sensitive bit failed (“fbits"). In case of perfectly modeled permanent fault, one would expect any upstream cells (scan cells between the scan chain input and a scan cell’s scan input terminal) will fail on sensitive patterns, and any downstream cell (scan cells between the scan chain output and a scan cell’s scan output terminal) to pass on all the sensitive patterns. However, if defects do not behave similar to the modeled faults, upstream defective cells are likely to have failure rate below 100% and downstream cells, a failure rate above 0%. This unsupervised learning-based approach has been applied to diagnose designs containing intermittent faults with positive results.

Another work [30] proposes a different ML-based scan chain diagnosis technique using supervised learning. This uses ANN to diagnose intermittent faults in a scan chain. Various multi-level ANNs with proper topologies (termed in this study as coarse global neural network (CGNN) and refined local neural network (RLNN)) provide high-resolution scan diagnosis. By evaluating in multiple stages, the investigators were able to zoom into the faulty location with higher accuracy. They also incorporated comprehensive ANN training vectors to have lower chances for unseen data deviating from trained patterns and experimental results showed encouraging results. The ANN has the following input features: fault type, faulty cell’s identification number, and the probability of a test pattern activating the fault. The output layer represents scan cells of a particular scan chain. These input features are modeled in the form of binary response vectors, further compressed into a single integer failure vector (IFV) computed by performing bit-wise addition of all binary response vectors. The number of scan latches in the scan chain determines the length of the IFV. The computation of the output node of CGNN indicates the candidate scan cell being faulty in the scan chain.

This work [30] also proposed a novel solution for compressing binary response vectors into a single vector. An affine group comprises of scan cells whose euclidean distance between their IFV and candidate scan cell is minimal. The length of the modified IFV, known as “reduced cascaded vector (RCV)," can be reduced by removing bits at certain positions based on the affine group (a group of scan cells having similar characteristics). This updated CGNN comprises of two layers whose number of input nodes equals the length of RCV, and the number of nodes in the output layer equals the number of scan cells in the affine group. The resulting scan diagnosis procedure could achieve reasonably high accuracy.

3.3 Printed-Circuit Board (PCB) Testing

Fault modeling and test methodology for a printed circuit board (PCB) differs from that of VLSI chips [18]. Modern PCBs consist of multilevel substrates with interconnects and mounted packages of digital and analog components. Typical tests are applied either from the edge connectors to check the board’s function, or to test components directly through in-circuit test (ICT) probes.

In-circuit testing (ICT) is a crucial aspect of identifying defects in electronic components [10]. Two primary methods of ICT are currently in use: analog and digital. Analog ICT entails measuring the electrical properties of components, such as resistance and capacitance, to identify subtle issues in passive components like resistors and capacitors. On the other hand, digital ICT employs digital signals to stimulate and monitor the performance of components and is suitable for digital components such as microcontrollers, memory chips, and other integrated circuits. By sending specific input signals and comparing the measured responses against expected values, digital ICT can quickly identify defects like incorrect logic states or malfunctioning components.

The choice of the most appropriate ICT method, analog or digital, depends on the nature of the PCB and its components. Many modern ICT systems now offer a combination of both methods, which comprehensively evaluate all aspects of the PCB’s functionality.

Testing each component on a board is vital from the real-time testing perspective. Even when an in-circuit test [10] of components using automatic test equipment (ATE) passes, the board-level functional test can fail. This phenomenon is foreboding and needs a structured way of testing to guarantee the reliability of the PCB (or SoC) and its continual maintenance. Typically, board-level functional fault diagnosis is based on the past root-cause analysis of faulty boards, which is also used as training data to predict defective components on new boards. The syndromes for faulty boards serve as a set of features, and the diagnosed root-causes serve as labels for the training data set.

A reasoning-based approach [113] is effective in functional debugging since it continuously learns during debugging and development. However, it is difficult to fix the problem if reasoning-based learning incorrectly identifies the faulty component on the board. Replacement of the entire reasoning model is trivial, but could adversely affect the correct detection of an observed failure. The investigators [113] kept the fixing of their approach as an open problem for the future.

Another ML application [184] proposed a technique to debug and repair board-level functional failures. It exploits the connection between failure syndromes and repair actions to train an ANN not to infer from visual inspection of log files and data sets.

An SVM-based technique [181, 185] diagnoses boards by learning incrementally to locate the root causes of failures. The learning tunes the SVM kernel to achieve high accuracy in diagnosis. The overall system training time improves with the continuous incremental learning of SVM.

ANNs and SVMs are combined to have a diagnosis system using a meta learning technique called weighted-majority voting (WMV) [96]. A proposed system combines the weights of different repair suggestions generated by respective machines to identify single pair of recommended repair suggestions. WMV using ANN or SVM can further optimize repair [162, 180]. There are three types of voting mechanisms: 1) unanimous voting, i.e., all experts agree on the same output, 2) at least one or more than half of the experts agree on the same output, i.e., simple voting, and 3) certain experts are qualified and their votes are weighted to improve the overall performance, i.e., weighted-majority voting.

Limited access to training data on the history of board failures and the feature vector size for training the ML models to diagnose failures are major concerns. A syndrome merging technique has been proposed [163] to reduce feature vector size. However, some syndromes that are not easily computable do not allow merging. Another technique [80] can still diagnose a system with a non-computable or missing syndrome using label-imputation and the so-called two-feature-selection methods.

3.4 Fault Diagnosis

Defective ICs can provide failure logs for fault diagnosis, but logging substantial data can be memory-expensive. Besides, the analysis of the entire dataset is time-consuming and may even be infeasible. ML can help decide when data collection can be stopped without sacrificing the efficiency of fault diagnosis [171]. The idea has been demonstrated by using different types of ML approaches, namely, *k*-nearest neighbor (kNN) [90], support-vector machine (SVM) [62, 136], and decision trees [64]. Both, unsupervised and supervised learning methods can cooperate in identifying design bugs [109]. A survey [70] of diagnosis using machine learning examines the relevancy of failure log information for fault diagnosis, defect location in scan chain or functional logic block, and diagnosis time.

Fault diagnosis plays a vital role in physical failure analysis (PFA), also known as failure mode analysis (FMA), where too many candidate faults may diminish diagnostic efficacy leading to low diagnostic resolution. For a diagnostic procedure, the average size of group within which faults cannot be distinguished from each other is referred to as the diagnostic resolution (DR) [183]. The ideal resolution, $DR = 1$, is often difficult to achieve. ML techniques try to meet specific objectives such as, 1) mapping of diagnosed faults onto corresponding defects based on the failure response of the circuit [47, 50], and 2) tuning the set of candidate faults to further improve the diagnostic resolution [178]. The ANN used in these studies get help from the layout and logic information of the circuit and failure response.

Conventional diagnostic tools claim to be highly accurate, but fail to identify certain faults because they may not consider layout information. Such faults occur due to systematic defects, and EDA tools and yield learning methods such as physical failure analysis (PFA) are incapable of handling them. This can be addressed by analyzing the fail-logs of multiple ICs, known as volume diagnosis. This involves analysis of large amount of data, and is time-consuming and expensive.

An ML-based technique [74] can be included in the yield-learning process to identify systematic defects and distinguish them from random defects. Here, failure responses of defective ICs are clustered using a procedure known as the farthest-neighbor method [36]. Later work [173] extended this technique to identify defect locations in fanout-free regions by observing how systematic faults affect the same set of outputs. The circuit is first decomposed into fanout-free regions for a specific kind of defect or defect class, which are then classified based on failure outputs using SVM. When many ICs fail due to a particular defect class, it is assumed that the ICs have systematic defects. Volume diagnosis also produces multiple failure features for an IC. At least two methods, namely, statistical-learning approach [166] and Bayesian network approach [29], can evaluate the failure feature probability.

An ML-based volume diagnosis technique [173] has several advantages: 1) It relies on certain decision-based subroutines, and computation complexity is much lower than traditional volume diagnosis methods; 2) It provides high-resolution diagnosis and statistical data, which classifies defective chips based on the defect location; and 3) The ML-based technique also works for scan designs using test compression and locates defects in most faulty ICs. The diagnosis methodology has been compared with respect to run time to the traditional analysis. Basic assumptions made are that faults in fanout free regions can be activated, propagated through common paths, and observed at common scan latches. According to the available experimental results [173] the technique can detect more than 90% of defective chips in a 50X output compacted design, which is faster than the traditional diagnosis methods. Besides, it could also detect 86% of defective chips with 100X outputs compacted designs in a few milliseconds.

An ML-based method that assists PFA [142] provides high-resolution detection of defects. Defects are grouped in “defect modes". A statistical test, such as χ^2 independence test, is applied to the data obtained from layout-aware

scan diagnosis. This test evaluates the amount of correlation between the defect and the “defect modes". The “defect modes" have corresponding *p-values* and rank the respective modes to capture the correct systematic defects and eliminate the effects of random defects (also treated as noise in this context of statistical analysis).

3.5 Test Compression

Due to the continuing technology node shrinkage, the increasing testing cost of high-density ICs has become a primary concern. This cost includes test application time, which is proportional to test data volume, and the cost of generating test data. Traditionally, compressor/decompressor architecture, i.e., pseudo-random pattern generator (PRPG) along with decompressor reduces the test cost by loading scan chains through decompressors and compacting test responses in multiple input signature registers (MISRs) [18]. However, the length of a PRPG does impact the test time irrespective of various circuit parameters [94]. The problem of PRPG length may be resolved by using ATPG, but that too is time-consuming. A PRPG length selection method is shown in Fig. 7. It uses a predictor based on the support vector regression (SVR) model, which reduces test costs in the CODEC architecture. The authors of that work [94] give a correlation-based feature selection method applied to industrial designs for reducing the test time with high prediction accuracy [120].

3.6 Testability Analysis

Testability analysis generally refers to linear, or at most polynomial but not exponential, complexity procedures that can identify test bottlenecks in a circuit [4]. The analysis determines numerical measures representing controllability and observability of signals. “Distance" or logic depth through the circuit has been the simplest measure that was used in an ATPG algorithm [48]. Here the distance of a signal site in terms of logic gates between PI and the site is considered the controllability measure, and that to PO as the observability measure. Some of the other testability measures are TMEAS [151], Sandia Controllability/Observability Analysis Program (SCOAP) [49], CAMELOT [12], and controllability and observability program (COP) [16]. The first four examine the circuit topology and the last one, signal probabilities. They have been used for improving digital circuit design or for selecting one out of multiple choices that occur within complex test generation programs. We discuss three areas where machine learning has been applied.

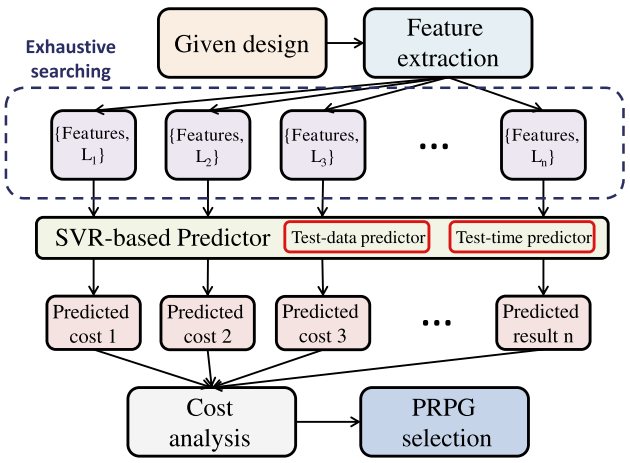

Fig. 7 Pseudo-random pattern generation (PRPG) methodology [94]

3.6.1 Combining Testability Measures

Several of the testability measures listed above have been combined into a composite measure using unsupervised learning [125]. The goal of the learning algorithm is to explore the data and find some structure or pattern within it. Popular learning models include *k*-means clustering [57], partitioning around medoids (PAMs) [81], ordering points to identify the clustering structure (OPTICS) [6], principal component analysis (PCA) [69, 117], minimum redundancy maximum relevance (mRMR) [118], and self-organizing maps (SOMs) [85]. These methods are typically used to segment text topics, classify items, and identify data outliers.

In our illustration, for every signal node, four testability measures have been defined, 0-controllability, 1-controllability, 0-observability, and 1-observability [77]. The last two measures are often replaced by a single measure, observability, leading to three measures per node. The combination process has the following steps:

- For testability measures, e.g., distances, SCOAP [49], etc., to be combined, compute relevant values corresponding to each signal node in the circuit.
- Normalize all quantities to the range [0,1].
- Phase correction - Consider SCOAP, which is a measure of effort. Thus, low or closer to 0 0-controllability means that the node is easy to set to 0. On the other hand, COP [16] estimates probability and for the same node the 0-controllability will be closer to 1. Assuming that the combined measure is to have the probability interpretation, the normalized SCOAP values should be subtracted from 1.0 in order to align with other measures.
- All measures are combined using the principal component analysis (PCA). If *n* measures are being combined, then PCA computes *n* values for each node of the circuit. The largest of these is the principal component and is used as the combined measure. The analysis is repeated three times to generate the combined 0-controllability, 1-controllability and observability for each node.

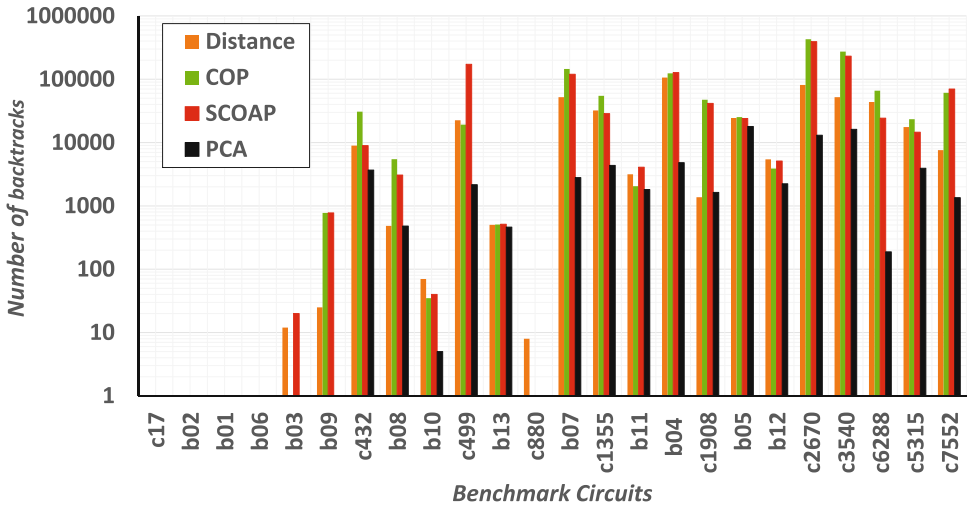

Fig. 8 Total backtracks used while finding a test or proving redundancy for the checkpoint faults left after the random ATPG phase applied to ISCAS'85 [17] and ITC'99 [32] benchmark circuits [131]

The PCA combined testability measure has been used to guide the ATPG with notable performance improvement as shown in Figs. 8 and 9, and discussed in Section 3.9. Other applications such as finding hard to detect (HTD) faults or test point insertion (TPI) candidate nodes are yet to be attempted. Also, the effects of combining larger number of measures may be explored in the future.

3.6.2 X-Sensitivity

Don't care or *unknown* signal state (denoted as *X*), when present in simulation, degrades the quality of fault detection. Their sources can be uninitialized memory cells, bus contentions, anomalous analog-to-digital conversion, and manufacturing defects during post-silicon validation. *X*-sensitivity of a signal is a measure of degrading effect on fault coverage from *X* on that signal. The support vector procedure, a machine learning technique, has been shown [120] to predict the sensitivity of *X*'s in a digital circuit. The method ranks circuit nodes according to *X*-sensitivity, which is beneficial in the post-silicon validation phase.

3.6.3 Signal Probability

Savir [135] conjectured that it would be impossible to calculate a simple testability measure based on signal controllabilities and observabilities in a circuit containing re-convergent fanouts such that the measure will truly represent the probability of fault detection. This is because the reconvergence introduces signal correlations not accounted for in simple testability measures. The difficulty is that almost all industrial circuits contain re-convergent fanouts. Topological analyses [122, 140] can detect re-convergent fanouts, but they can be computationally burdensome. Toward application of ML, recent work [75] has used ANN to predict signal probabilities from minimal fanout information, resulting in increased accuracy with reasonably small computation time.

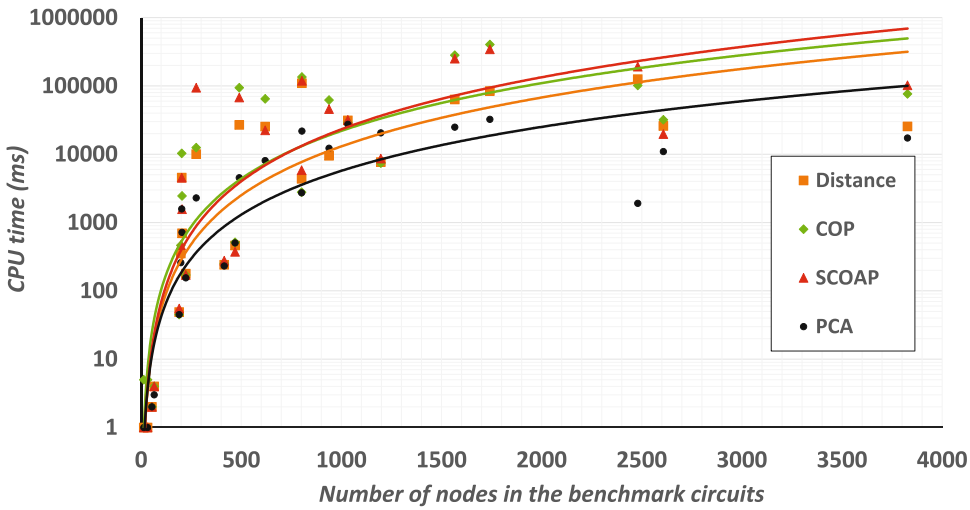

Fig. 9 Total CPU times on Intel i7-8700 based workstation with 8-GB RAM to find a test or prove redundancy for the checkpoint faults left after random ATPG phase applied to ISCAS'85 [17] and ITC'99 [32] benchmark circuits. Four types of data points and trend curves are for PODEM ATPG programs guided, by respectively, logic distance (square dot, third curve from top), COP (diamond dot, second curve from top), SCOAP (triangular dot, third curve from top), and PCA-combined measure (circular dot, bottom curve) [131]

3.7 Built-In Self-Test (BIST) and Test Point Insertion (TPI)

Logic built-in self-test (LBIST) often relies on pseudo-random patterns, which may be economically generated in hardware by a linear feedback shift-register (LFSR) [18]. However, an LFSR may not generate specific patterns to detect random pattern-resistant (RPR) faults. As an ML solution to this problem, an ANN can be used to generate test patterns to detect RPR faults as well as easy-to-detect faults.

The self-learning capability, suitable for a system-on-chip (SoC), also deals with aging-induced degradation. This proposed flow uses existing LBIST and an ML-based software predictor to remedy the problems arising from the wear-out or aging of IC in the field [39]. An ANN is developed using LBIST patterns (converted from ATPG-generated tests for transition delay faults) that activate critical or near-critical paths. The results demonstrate that a gate-overlap and path delay-aware algorithm can select the optimum set of test vectors. This methodology is area and test-time efficient.

To improve the fault coverage of LBIST, designers insert test points (TPs) modifying the circuit's internal signal values to detect random pattern resistant (RPR) faults. Test point insertion (TPI) [60] techniques find high-quality TPs to improve fault coverage or reduce test vector count. These techniques are classified based on the form of analysis used, namely, fault simulation, probabilistic testability measures, or multiple measurements [107, 161].

A deep learning technique to solve the TPI problem of logic circuits has been proposed [100]. It uses a graph convolutional network (GCN) to classify signal nodes as either easy-to-observe or difficult-to-observe. This ANN analyzes attributes of each node and its neighbors, based on a testability measure such as SCOAP [49]. Further work [132, 133, 159] used fully-connected neural networks to evaluate the impact of control-0, control-1, and observe test points on fault coverage and found that an iterative TPI process improved the fault coverage and significantly reduced TPI time. In another extension [108], when randomly generated circuits were used for training, the ANN still yielded a performance comparable to that of ANN trained on benchmark circuits.

A more recent investigation [160] has shown that optimizing the complexity of the neural network can improve the LBIST performance with higher fault coverage, fewer test points, and shorter test length, while reducing the computation time to find test points.

3.8 Power Supply Noise (PSN) and Signal Integrity

Reliability problems of integrated circuits center around operating conditions, such as, temperature, speed, voltage, and circuit aging. Many of these remain uncovered during the conventional testing and may be found during the burn-in test [78]. Some related concerns are power supply noise (PSN), signal integrity, and timing failures.

IR drop is a significant concern in IC design and is often referred to as power supply noise (PSN) [143, 167]. Unrestrained PSN can lead to performance glitches and impact timing [25, 79]. Also, excessive PSN during test can cause false failure if a test pattern induces PSN that substantially exceeds the functional mode behavior [46, 93, 172, 174]. Hence, PSN simulation, though a nontrivial effort, is important.

Timing analysis is vital because it determines the clock frequency for the IC. However, circuit timing depends on static and dynamic characteristics, because PSN impacts the supply voltage reaching individual gates, it affects propagation delays and slows down switching.

Applications of ML in this area include the use of support vector machine (SVM) [179] to predict voltage droop in field-programmable gate array (FPGA) and dynamically adjust the clock frequency of the circuit. However, without feature extraction, the method is applicable only to small ICs. Another ML-based technique [97] includes feature extraction methods, such as ANN [34], SVM [15, 179], and least-square boosting (LSBoost) [15]. Here, ANNs are found to be the best predictors of circuit timing for test patterns.

A recent paper [112] gives a machine learning (ML) solution for small delay fault (SDF) detection problem of resistive opens. Such defects may not cause a failure of timing specification but still present a reliability challenge. The method uses tests at multiple voltages and frequencies to examine the latent faults considering three ML techniques: support vector machine [62, 136], *k*-nearest neighbors [90], and random decision forests [64]. The results show that the learning scheme based on random decision forest classifies the embedded faulty cells with higher accuracy.

3.9 Machine Intelligence Applied to ATPG

An ATPG algorithm searches for an input vector to detect a given fault. For a combinational circuit, the search space consists of $2^{\#PI}$ vectors, where $\#PI$ is the number of primary inputs (PIs). Thus, ATPG is a search algorithm whose size of search space increases exponentially with circuit size, in terms of $\#PI$.

Roth's *D*-algorithm [123] conceptualizes ATPG by defining *D*-algebra and giving a complete search algorithm. The symbol D represents a composite state of a signal in the fault-free and faulty circuits. Thus, D means 1 in fault-free circuit and 0 in faulty circuit. $\overline{D}$ is the opposite condition.

D-algorithm has high complexity as it manipulates all internal signals of the circuit. It can be particularly inefficient for large circuits containing *XOR* gates and reconvergent fanouts. The path oriented decision making (PODEM) [48] algorithm improves the search efficiency

by focusing on PIs. In general, ATPG implementations use heuristics to speed up the search. In summary, the relevant features of the PODEM algorithm are,

- The search space is reduced from 2^n for *D*-algorithm, where n is the total number of signals (gates and PI) in the circuit, to $2^{\#PI}$ for PODEM.
- A concept of *X*-path-check is introduced, where *X* refers to an unknown or yet undetermined value of a signal. *D*-algorithm may try to find a test even when the entire *D*-frontier is blocked, but PODEM's *X*-path-check verifies that there is at least one *D*-frontier gate with access to a primary output. Otherwise, it will backtrack to the previous stage in the search process where an alternative signal choice is available. *D*-frontier is the set of all gates that have a D or $\overline{D}$ at their input but the output is still *X*, i.e., undetermined.
- PODEM originally proposed a distance-based heuristic to identify easy or hard to control inputs of logic gates while backtracing to primary inputs, as opposed to *D*-algorithm that traditionally chose any gate input. Several other heuristics based on the circuit topology have been used in the programmed implementations of both algorithms. Similarly, while propagating the fault effect to an observable primary output (PO), the gate closest to PO will be selected from the *D*-frontier.

Many other ATPG algorithms, e.g., FAN [43], TOPS [84], SOCRATES [137–139], EST [19, 27, 28], recursive learning [88], TRAN [23], GRASP [103], NEMESIS [92], TEGUS [150], and Boolean satisfiability (SAT) [11, 20, 22, 91, 92], have been reported. Although the search space size remains $2^{\#PI}$, researchers [63, 164] attempt to find tests faster either by special subroutines to filter the search space, or through heuristics to select from available choices. It is this second aspect of the ATPG that the ML techniques focus on.

Before machine learning was applied to ATPG, artificial neural networks (ANN) were used to model digital circuits where a bidirectional binary neuron would represent the state of a signal [20, 22]. Each neuron has a threshold value and its interconnects to other neurons have weights, which together determine the energy of the ANN for any set of neuron states. For any binary [0,1] states of primary input (PI) neurons, the minimum energy of the ANN is attained only when all neurons assume valid signal states corresponding to the digital circuit. Given a target fault, the ANN for the corresponding arbiter circuit is first constructed. The minimum energy state of this ANN is then determined and the states of PI neurons provide a test vector. The ATPG requires either a physical neural network or a software model. In either case, the network energy function depends on a large number of variables (all signals) and may have many local minima, making the search for a test (minimum energy state) for some faults rather difficult. A program, TRAN [21, 23], makes this algorithm computable by using graph theoretic principle of transitive closure.

Applications of quantum computing, although not exactly considered machine intelligence, have also been reported [145, 146]. While we discuss recent developments in this section, one can find discussion of machine learning in the context of ATPG as far back as 1987 [87].

The application of ML is related to the heuristic part of the ATPG algorithm. All programmed algorithms have used heuristics to speed-up the search. Typical heuristics base decisions on distance, in terms of logic gates, from PIs or POs to signal sites, testability measures, voting on fanout stems depending on branches, learning techniques using implication graphs, etc. In 1985, Patel and associate [115, 116] conducted experiments to study the effectiveness of various testability measures as heuristics in PODEM and proposed a strategy for test generation. They observed that instead of using a single testability measure with a high backtrack limit, it is more efficient to use multiple testability measures successively and with a low backtrack limit. Considering this a traditional approach, machine learning (ML) as discussed next will be quite different; multiple testability measures will be combined and used all together. The result will be even greater efficiency over the successive application approach [115, 116].

Recent work [124] uses ANN and principal component analysis (PCA) [69, 117] as ML models, relies on the conventional gate-level circuit description, and uses a search algorithm that, given unlimited computing resources, would guarantee a test in significantly reduced CPU time by making fewer unproductive algorithmic decisions requiring backtracks. The ANN and PCA combine circuit topology information and testability measures to create a novel heuristic to guide the search. Since several available heuristics are being applied together, we do not need a low backtrack limit as a stopping criterion to avoid unproductive decisions.

PODEM [48] offers an ideal ATPG environment to apply ML-based heuristic to choose a backtrace path to a primary input (PI) for justifying a desired signal value at an objective site. The ATPG benefits from the ML-based guidance, which is found to reduce backtracks. Three approaches have been reported to provide successively higher performances. All use a conventional PODEM program with backtrace guidance provided either by a PCA-combined testability measure [130, 131], as described in Section 3.6.1, or by a trained ANN [126, 127, 129]. The former is called unsupervised learning, while the latter is called supervised learning.

Considering the present context, the PCA can combine any number of data types relevant to the ATPG algorithm, such as input–output distance (logic depths), and testability

measures from COP [16] and SCOAP [49] values into a set of principal components (PCs). Then the largest (major) PC would guide the PODEM ATPG backtraces [130], also known as “PCA-guidance" methodology. The next case we examine is a “optimally-trained-ANN" feature reduction methodology to improve the ANN complexity and guide decisions that otherwise would rely on heuristics, also known as “PCA-trained-ANN" [127]. The result, not surprisingly, is the best achieved among the aforesaid ML-based ATPG options studied.

The preceding evaluation is based on a combined ATPG performance (number of backtracks and CPU time) for all or a target subset of faults. However, in practical ATPG implementation an important criteria is the performance with respect to the hardest-to-detect or even redundant faults. Thus, a fault-by-fault micro-evaluation of the ATPG guidance techniques is recommended for the future, and what follows next offers a preview.

Statistical analysis of fault coverage for random and deterministic vectors [141] can assess circuit testability from fault simulation, predict coverage from detection probabilities, estimate test length for required coverage, and help generate test vectors by fault sampling. On these lines, we discuss a practical ATPG system where easy-to-detect faults are covered by random vectors and hard-to-detect faults are left for a PODEM-based ATPG with backtrace guidance coming from either MI [124, 126, 127, 129, 130], or distance (logic depth) heuristic [48], or controllability and observability program (COP) [16], or SCOAP [49]. We find that MI-guided ATPG shows significantly improved performance over others.

Unsupervised learning or PCA was applied only in the backtrace step [130] in the early work, while the *D*-drive used the conventional distance (logic depth) heuristic [48]. The ATPG system we will examine now [131] applies principal component (PC) to direct both backtrace and *D*-drive. In addition, this is a complete ATPG system with random and algorithmic phases and a fault simulator. The ML based ATPG was applied only to faults left uncovered after the random pattern fault simulation phase. The results showed the effectiveness of guidance provided by PCA to PODEM ATPG.

- In Figs. 8 and 9 circuits are arranged left to right in the order of increasing number of nodes. Figure 8 shows combined backtracks and Fig. 9 gives total CPU time for all stuck-at faults left over from the random phase. Number of backtracks (four bars in Fig. 8) and CPU milliseconds (ms) (four data points in Fig. 9) for each circuit correspond to the four versions of PODEM. These are PODEM programs where backtraces and *D*-drives are directed, respectively, by distance (logic depth), COP, SCOAP, and the major principal component from PCA combining distance, COP and SCOAP measures. In Fig. 9, a trend curve is obtained by power-law fit to the experimental data from each PODEM version. Notably, shorter black bars and lower black curve indicate consistent improvement provided by PCA guidance [131].

These results demonstrate that guidance from PCA-generated linear combination of multiple heuristics can reduce ATPG backtracks and CPU times when compared with conventional single heuristic guidance. Circuits b03, c432, b10, b13, c880, b07, b05, b12, c5315, c7552, c1355, c2670, c3540, b04, b11, b08, c499 and c6288 exhibit significant reductions in backtracks and CPU times in Figs. 8 and 9. PCA is most frequently the best guidance for ATPG, but even when it is not, it is never the worst. There are no reconvergent fanouts in c17, b02, b01, and b06, and so there is no scope for reducing backtracks as there would be none. An example of zero backtracks by PCA-based PODEM ATPG is circuit b09 in Fig. 8.

An obvious advantage of this procedure is its simplicity. Besides, any number of testability measures can be combined by PCA. For example, a measure that includes the information on reconvergent fanouts may give additional benefit to the ATPG. In general, all measures may have linear complexity approximations, each retaining a different piece of information. Thus, adding more measures in PCA should continue to increase the benefit.

Another form of ML application employs artificial neural networks (ANN) and is referred to as supervised learning. Here, models are trained with input data where the desired outputs are known. Supervised learning uses patterns to predict labels on unlabeled data and is used in applications where the history of data predicts likely future events. A supervised learning algorithm receives inputs along with corresponding correct outputs; the algorithm learns by comparing its outputs against and correct outputs to find errors and modifies the learning model accordingly to minimize errors. Some known learning models are support vector machine (SVM) [62, 136], one-class SVM [62] or one-class neural network [148], decision trees (DT) [121], random forest (RF) [65], linear regression (LR) [111], multivariate adaptive regression splines (MARS) [182], logistic regression [33], adaboosts [41], ANNs [61], convolutional neural network (CNNs) [61], autoencoders [7], recurrent neural network (RNNs) [61], long short-term memories (LSTMs) [45], and half-space trees (HS-Trees) [165].

A recent study examined MI’s supervised learning ability to enhance ATPG by reducing backtracks [126], also called here as “basic training of ANN" methodology, by replacing conventional heuristic to decide backtracing direction using an ANN trained with PODEM data on hard-to-detect faults. The training of ANN can be tuned for ATPG application [129], which we will call “optimally trained ANN" methodology. In this case, supervised ML uses sample

ATPG data and circuit information to train an artificial neural network (ANN), which then provides the backtrace decisions for ATPG. In contrast, unsupervised ML was more direct as it used neither sample ATPG data nor the ANN.

Figure 10 [127] shows the ATPG results for the same benchmark circuits (as in Figs. 8 and 9) now using supervised learning. As the fitted trend curves show, these results are similar to the unsupervised learning results of Fig. 9 [131], although they cannot be numerically compared. In the unsupervised learning case the ATPG was applied only to the checkpoint faults left undetected by random vectors, whereas in the supervised learning experiment all checkpoint faults were used. Another difference is that the data are arranged, respectively, according to the number of signal nodes in Fig. 9 and logic depth in Fig. 10. Noticeable difference is seen, however, for several small and medium size circuits in Fig. 10 where the ATPG CPU time with the PCA trained ANN guidance is negligibly small.

4 Memory Test and Repair

4.1 ML-Based Built-In Self-Repair of DRAM

Device and interconnect geometries of VLSI circuits are decreasing rapidly. As a result, manufacturing yield continues to drop, owing to higher component density, complicated fabrication process, and greater susceptibility of shrunk features to defects. Some faulty parts on chips are rescued by incorporating redundant components, and a reconfiguration scheme that replaces the faulty component with a redundant one. A dynamic random access memory (DRAM) is densely packed, and redundant rows and columns are added to reconfigure faulty cell rows and columns of memory sub-arrays using electronically programmable latches. Optimal reconfiguration and redundant component allocation is a classical problem widely studied by researchers [42]. However, these algorithms are not directly applied to memory sub-arrays as they are neither controllable nor observable by external testers. This problem is resolved by the introduction of built-in self-test (BIST) that comprehensively tests memory arrays and discards them if they fail. The scheme is further modified as “built-in self-repair (BISR)", and is used to salvage faulty memory arrays.

Memory repair was first introduced in 64 kbit DRAM to improve the chip yield using redundant rows and columns [149]. With technology advances, increasing memory size has made the search space too large and the types of faults have also become complex. Therefore, conventional repair algorithms, both greedy [38] and exhaustive [35], became ineffective. Since memory repair problem is NP-complete [54], heuristic algorithms were introduced. These included branch and bound [89], approximation [89], best-first search [58], and others [98, 175]. They have worst-case complexities that are nearly exponential and are not easily implementable in the built-in self-repair (BISR) mode. Focus next shifted [105] to: (1) an efficient algorithm so that overall throughput improves with the chip yield, and (2) hardware implementable algorithms. A self-repair scheme using BISR [105] repairs memory subarrays by reconfiguring redundant rows/columns. As “Repair Most (RM)" is a simple and easily implementable hardware, the performance of ANN-based memory repair algorithm has been compared against RM [105].

ANNs have been used to tackle optimization problems, e.g., the famous *traveling salesman problem* [44], for which a solution was proposed by Hopfield [68]. Lyapunov’s energy function can represent an optimization cost function, and the convergence property of the ANN from a random initial state to a local minimum state can reduce this cost by using a gradient descent algorithm. However, this kind

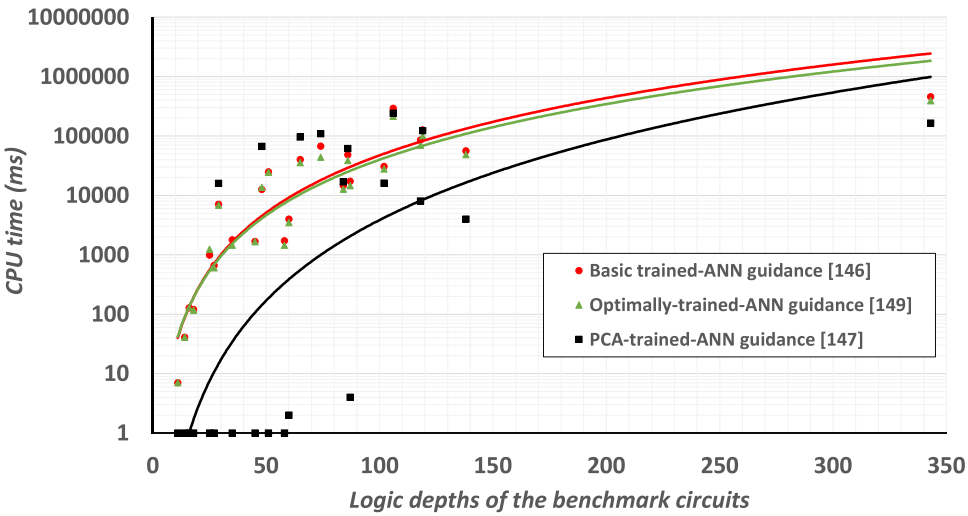

Fig. 10 Total CPU times on Intel 8700 processor based workstation with 8-GB RAM to find a test or prove redundancy for *all* checkpoint faults in ISCAS’85 [17] and ITC’99 [32] benchmark circuits. Three types of data points and trend curves are for PODEM ATPG guided by, respectively, basic-trained ANN (round points, top curve), optimally-trained ANN (triangular points, middle curve), and PCA-trained ANN (square points, bottom curve) [127]

of ANN formulation has low-quality, and therefore another proposed algorithm [105] modifies the existing gradient descent to a hill-climbing algorithm. This improves the solution quality and raises the probability of finding a globally optimal solution.

Also, it is found that conventional repair algorithms run slow on digital computers, whereas ANN's collective computational property provides a faster solution. A gradient descent algorithm [105] can be 2-to-4 times better than conventional "RM" algorithms in repair schemes as gradient descent minimizes the network's cost function in the locality of the starting energy value, and the hill-climbing algorithm further bypasses the local minima traps. It was empirically observed that the hill-climbing algorithm can repair almost 98% of faults in a large memory array as opposed to other conventional and gradient descent algorithms with a certainty of approximately 20%. Both hill-climbing and gradient descent algorithms using ANNs take minimal area overhead of approximately 3%. It was also reported [105] that the chip yield increased from 10% to 100% by improved repair. Additionally, the ANN hardware is more fault-tolerant and robust than conventional logic circuits and therefore is the best candidate for a self-repair circuit. However, three types of component failures have been identified in neural networks, namely synapse-stuck-at fault, bias fluctuations, and neuron stuck-faults to serve as fault model. For each faulty synapse, either of synaptic weights can be assumed as stuck, due to transistor-stuck faults or defective memory cells that control the programmable synapses. Faulty bias generators are modeled to fluctuate within one unit of the pre-determined biases, and faulty neurons will have stuck-at firing or stuck-at non-firing states. For unknown reasons, if the ANN neurons are stuck-at firing or non-firing state, then its ability to repair faulty memory cells degrades gracefully and supports continual operation despite multiple faulty neurons in the ANN.

4.2 Software-Assisted Self-Test of Flash Memory

Among the application domains of flash memory, automobile industry is an important one. Embedded flash memory cores occupy substantial portion in automotive SoCs with significant impact on the final yield of devices. Automotive IC testing must ensure correct chip function after calibration, test, and repair of flash memories [102]. This requires redundant memory cells, i.e., spare word lines (WLs) and bit lines (BLs), and activation mechanism for the redundant structures. Redundant component analysis can be done on-line in software-assisted in-chip self-test (SIST) [102], but a major bottleneck is efficient reconfiguration of redundant components quickly and accurately. A bitmap scheme was originally used to reconfigure faulty memory cells by downloading the cell coordinates, but later it proved to be ineffective and time-consuming, which prevented it from becoming a regular industry practice. The strategies that maintain a trade-off between test time and memory costs with accurate reconfiguration to spare components may lead to false-positive behavior and yield loss such as, (1) identifying uncorrectable faulty memory by a repair algorithm, which is not feasible or (2) discarding the correctable faulty memory despite the availability of suitable spare components owing to the repair algorithm's inability.

One must deal with false fail identifications and prevent unnecessary repairs [102]. If a replacement algorithm is heavily constrained with execution time it may classify a repairable memory core as irreparable, known as false fail. A vital step in using an ML-based predictor to identify false fails is to extract training features. Training features have been extracted using a coloring algorithm [53], where every fault is assigned a unique color and its occurrence is evaluated statistically. This algorithm combines different faults with unique colors to provide a chunk of datasets to the ANN.

A machine learning technique [102] works in two steps: (1) in development phase, bitmaps are collected for selected devices that compose training/test datasets, and (2) in production phase, training features are extracted using the coloring algorithm [53], and the discarded devices are labeled as false failures. A detailed analysis is used to extract training features and results are fed back to the coloring algorithm designers. Supervised and unsupervised training techniques are deployed to assess the false fails and to determine whether or not they are correctly discriminated against in training features. Artificial bitmaps are added to original bitmaps to keep unaltered fail signature characteristics: bitwise AND, OR, XOR, noise, and many more. These additional bitmaps provide more comprehensive training datasets including false fails, leading to significantly better prediction accuracy.

It is also found [102] that the training data sets are highly unbalanced and therefore a confusion matrix is used. This is a table whose rows resemble predicted labels and columns represent actual labels. The resultant square matrix provides useful information, i.e., the correct prediction lies on the diagonal and misclassifications, elsewhere. The best ML-based predictor must be fast, reliable, easily hardware implementable, and interpretable so that it does not affect the overall test time of the IC production flow.

Experimental results [102] have shown that the ML-based predictor of a model "decision tree" compared to other models such as "random forest" and "feed-forward" have a better score and minimal variance with the fastest and easy-to-implement subroutine. The overall approach is empirically proven on real-time data and demonstrates that it is feasible to predict a false fail device with better accuracy.

4.3 SRAM Yield Improvement using Statistical Blockade

As transistor size shrinks, the statistical blockade technique [147], discussed in Section 2.2, is found to improve the yield of SRAM ICs since they contain highly repeatable components. Statistical blockade is conceptually an improvised Monte Carlo method, employs ideas from non-traditional sources such as extreme value theory (EVT) and machine learning. This novel technique, proven to be more efficient than the conventional Monte Carlo method, provides accuracy and speedup of approximately two orders of magnitude across circuits despite parametric variation.

5 Conclusion and Future Work

This survey has highlighted key aspects of machine learning (ML)-based testing of analog, digital, memory, and radio frequency (RF) devices. It motivates the integration of ML into the IC testing process of the future. It is expected that the use of ML would become routine in testing of the ICs of the future in the defense, healthcare, space, and automotive industries:

- Recently, ML has been at the cutting-edge in the IC test industry, but the accuracy in classifying test data after training an ANN is not fully convincing (i.e., may not be close to 100%). Moreover, 100% training and test accuracy will not fetch the correct classification of data in a real-time, which may lead to a catastrophe in critical systems.
- ML in counterfeiting detection can prove dangerous if the attackers use ML-based model to attack either good (false positive) or bad (false negative) ICs. The research on defense techniques needs to stay ahead!
- Emerging technology designs and their testing is in a nascent stage and hence full of imperfections and ongoing variations. However, quick adoption of the existing methods and tools to a new technology can be effective [186, 187]. While any ML applications in the existing tools are immediately adopted this way, future investigations may lead to unconventional benefits.

ML applications fall in two categories, experimental and algorithmic. The first category includes defect diagnosis, i.e., locating and identifying defects, which is a part of manufacturing. Diagnosing defects, often different from the modeled faults targeted by tests, is a problem that requires experience and skill. Section 2 discusses analog and RF testing that does not rely on fault models and signal parameter ranges must be interpreted during test. This often requires human intervention, which ML can automate. Section 3.4 and several other subsections on digital circuit diagnosis and Section 4 on memory test and repair bring out the ML potential.

The second category consists of algorithms that need to be programmed. Algorithms for digital test generation are complex and grow exponentially with the circuit size. A typical program uses heuristics to select among multiple choices to direct the execution toward a quick solution. Several subsections around Section 3.4 point to some very practical applications of ML in the supervised and unsupervised learning modes.

We must mention items that our reader will find missing in this survey. The references in this paragraph are nowhere comprehensive but are given to get the search started. We believe there is ample research scope in new areas like hardware security [14, 52], wafer map failure pattern classification [26], test escape detection for digital circuits [83], intelligent lithographic hotspot detection [177], expediting device-level testing followed by circuit and SoC-level testing using ML for some of the emerging technologies such as carbon nanotube field-effect-transistor (CNTFET) devices [8], monolithic 3D (M3D) devices such as resistive RAM (ReRAM) [24] and many more.

Data Availability Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

Declarations

Conflict of Interest We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work, there is no professional or other personal interest of any nature or kind in any product, service and/or company that could be construed as influencing the position presented in, or the review of, the manuscript entitled. Also, one of our co-authors is an editor of the journal.

References

1. Abdallah L, Stratigopoulos H, Kelma C, Mir S (2010) Sensors for built-in alternate RF test. In Proc. 15th IEEE European Test Symposium (ETS), pp. 49–54
2. Acar E, Ozev S (2008) Defect-oriented testing of RF circuits. IEEE Trans Comput Aided Des Integr Circuits Syst 27(5):920–931
3. Adams RD (2003) High performance memory testing. Frontiers in Electronic Testing Book Series, Springer
4. Agrawal VD, Mercer MR (1985) Testability measures – what do they tell us? In Proc. International Test Conf., (Philadelphia, PA), pp. 391–396
5. Akbay SS, Torres JL, Rumer JM, Chatterjee A, Amtsfield J (2006) Alternate test of RF front ends with IP constraints: frequency domain test generation and validation. In Proc. IEEE International Test Conference, pp. 1–10
6. Ankerst M, Breunig MM, Kriegel HP, Sander J (1999) OPTICS: Ordering points to identify the clustering structure. In Proceedings of the 1999 ACM SIGMOD

International Conference on Management of Data, SIGMOD '99, (New York, NY, USA), Association for Computing Machinery, p. 49-60
7. Baldi P (2012) Autoencoders, unsupervised learning, and deep architectures. In Guyon I, Dror G, Lemaire V, Taylor G and Silver D, editors, Proceedings of ICML Workshop on Unsupervised and Transfer Learning, volume 27 of Proceedings of Machine Learning Research, (Bellevue, Washington, USA), PMLR, pp. 37–49
8. Banerjee S, Chaudhuri A, Chakrabarty K (2020) Analysis of the impact of process variations and manufacturing defects on the performance of carbon-nanotube FETs. IEEE Trans Very Large Scale Integr VLSI Syst 28(6):1513–1526
9. Barragan MJ, Stratigopoulos H, Mir S, Le-Gall H, Bhargava N, Bal A (2016) Practical simulation flow for evaluating analog/mixed-signal test techniques. IEEE Des Test 33(6):46–54
10. Bateson J (1985) In-circuit testing. Van Nostrand Reinhold Company, New York
11. Becker B, Drechsler R, Eggersgluess S, Sauer M (2014) Recent advances in SAT-based ATPG: Non-standard fault models, multi constraints and optimization. In Proc. 9th IEEE International Conference on Design & Technology of Integrated Systems in Nanoscale Era (DTIS), pp. 1–10
12. Bennetts RG, Maunder CM, Robinson GD (1981) CAMELOT: A computer-aided measure for logic testability. IEE Proceedings E - Computers and Digital Techniques 128(5):177–189
13. Bhattacharya S, Chatterjee A (2006) A DFT approach for testing embedded systems using DC sensors. IEEE Des Test Comput 23(6):464–475
14. Bhunia S, Tehranipoor M (2018) Hardware security: A hands-on learning approach, 1st edn. Morgan Kaufmann
15. Bishop C (2006) Pattern recognition and machine learning. Springer Publishing Company, Incorporated
16. Brglez F (1984) On testability analysis of combinational circuits. Proc International Symp Circuits and Systems 221–225
17. Brglez F, Fujiwara H (1985) A neutral netlist of 10 combinational benchmark circuits and a targeted translator in FORTRAN. Proceedings of the IEEE Int. Symposium on Circuits and Systems (ISCAS), pp. 677–692
18. Bushnell ML, Agrawal VD (2013) Essentials of electronic testing for digital, memory and mixed-signal VLSI circuits. Springer Publishing Company, Incorporated
19. Bushnell ML, Giraldi J (1997) A functional decomposition method for redundancy identification and test generation. J Electronic Testing 10:175–195
20. Chakradhar ST (1991) Neural network models and optimization methods for digital testing. PhD thesis, Rutgers University, USA
21. Chakradhar ST, Agrawal VD (1991) A transitive closure based algorithm for test generation. In Proceedings of the 28th ACM/IEEE Design Automation Conference, DAC '91, pp. 353–358
22. Chakradhar ST, Agrawal VD, Bushnell ML (1991) Neural models and algorithms for digital testing. Springer
23. Chakradhar ST, Agrawal VD, Rothweiler SG (1993) A transitive closure algorithm for test generation. IEEE Trans Comput Aided Des Integr Circuits Syst 12(7):1015–1028
24. Chaudhuri A, Banerjee S, Park H, Kim J, Murali G, Lee E, Kim D, Lim SK, Mukhopadhyay S, Chakrabarty K (2020) Advances in design and test of monolithic 3-D ICs. IEEE Des Test 37(4):92–100
25. Chen HH, Ling DD (1997) Power supply noise analysis methodology for deep-submicron VLSI chip design. In Proceedings of the 34th Annual Design Automation Conference, p. 638-643
26. Chen S, Zhang Y, Hou X, Shang Y, Yang P (2022) Wafer map failure pattern recognition based on deep convolutional neural network. Expert Syst Appl 209:118254
27. Cheng KT (1991) On removing redundancy in sequential circuits. In Proceedings of the 28th ACM/IEEE Design Automation Conference (DAC), 1991, pp. 164–169
28. Cheng KT, Agrawal VD (1989) Unified methods for VLSI simulation and test generation. Springer
29. Cheng W, Tian Y, Reddy SM (2017) Volume diagnosis data mining. in Proc. 22nd IEEE European Test Symposium (ETS), pp. 1–10
30. Chern M, Lee SW, Huang SY, Huang Y, Veda G, Tsai KHH, Cheng WT (2019) Improving scan chain diagnostic accuracy using multi-stage artificial neural networks. In Proceedings of the 24th Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 341–346
31. Cimino M, Lapuyade H, De Matos M, Taris T, Deval Y, Begueret JB (2006) A robust 130nm-CMOS built-in current sensor dedicated to RF applications. In Proc. Eleventh IEEE European Test Symposium (ETS'06), pp. 151–158
32. Corno F, Reorda MS, Squillero G (2000) RT-level ITC'99 benchmarks and first ATPG results. IEEE Des Test Comput 17:44–53
33. Cox DR (1958) The regression analysis of binary sequences. J R Stat Soc B Methodol 20(2):215–242
34. Daasch WR, Madge R (2005) Data-driven models for statistical testing: measurements, estimates and residuals. In Proc. IEEE International Test Conference, pp. 10 pp.–322
35. Day JR (1985) A fault-driven, comprehensive redundancy algorithm. IEEE Des Test Comput 2(3):35–44
36. Dillon WR, Goldstein M (1984) Multivariate analysis: Methods and applications. Wiley Publishing Company, Incorporated
37. Ellouz S, Gamand P, Kelma C, Vandewiele B, Allard B, Combining internal probing with artificial neural networks for optimal RFIC testing. In Proc. IEEE International Test Conference, pp. 1–9
38. Evans RC (1981) Testing repairable RAMs and mostly good memories. In Proceedings International Test Conference, pp. 49–55
39. Fagot C, Girard P, Landrault C (1997) On using machine learning for logic BIST. In Proc. IEEE International Test Conference, pp. 338–346
40. Ferrario J, Wolf R, Moss S, Slamani M (2003) A low-cost test solution for wireless phone RFICs. IEEE Commun Mag 41(9):82–88
41. Freund Y, Schapire RE (1997) A decision-theoretic generalization of on-line learning and an application to boosting. J Comput Syst Sci 55(1):119–139
42. Fuchs WK, Chang MF (1989) Diagnosis and repair of large memories: a critical review and recent results, pp. 213–225. Boston, MA: Springer US
43. Fujiwara H, Shimono T (1983) On the acceleration of test generation algorithms. IEEE Trans Comput C–32(12):1137–1144
44. Garey MR, Johnson DS (1990) Computers and intractability. A guide to the theory of NP-completeness. W. H, Freeman and Co
45. Gers FA, Schraudolph NN, Schmidhuber J (2003) Learning precise timing with LSTM recurrent networks. J Mach Learn Res 3(null)115-143
46. Girard P, Nicolici N, Wen X, editors (2010) Power-aware testing and test strategies for low power devices. Springer
47. Gómez LR, Wunderlich H (2016) A neural-network-based fault classifier. In Proc. IEEE 25th Asian Test Symposium (ATS), pp. 144–149
48. Goel P (1981) An implicit enumeration algorithm to generate tests for combinational logic circuits. IEEE Trans Comput C–30(3):215–222
49. Goldstein L (1979) Controllability/observability analysis of digital circuits. IEEE Transactions on Circuits and Systems, vol. CAS-26, no. 9, pp. 685–693, Sept. 1979

50. Gómez LR, Cook A, Indlekofer T, Hellebrand S, Wunderlich HJ (2014) Adaptive bayesian diagnosis of intermittent faults. J Electron Test 30(5):527–540
51. Gopalan A, Margala M, Mukund PR (2005) A current based self-test methodology for RF front-end circuits. Microelectron J 36(12):1091–1102
52. Guin U, DiMase D, Tehranipoor M (2014) Counterfeit integrated circuits: Detection, avoidance, and the challenges ahead. J Electron Test Theory Appl 30(1):9–23
53. Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. J Mach Learn Res 3(null)1157-1182
54. Haddad RW, Dahbura AT, Sharma AB (1991) Increased throughput for the testing and repair of RAMs with redundancy. IEEE Trans Comput 40(2):154–166
55. Hafed MM, Abaskharoun N, Roberts GW (2002) A 4-GHz effective sample rate integrated test core for analog and mixed-signal circuits. IEEE J Solid-State Circuits 37(4):499–514
56. Harrison RR, Bragg JA, Hasler P, Minch BA, Deweerth SP (2001) A CMOS programmable analog memory-cell array using floating-gate circuits. IEEE Trans Circuits Syst II Analog Digit Signal Process 48(1):4–11
57. Hartigan JA, Wong MA (1979) Algorithm AS 136: A k-means clustering algorithm. J Roy Stat Soc 28:100–108
58. Hasan N, Liu CL (1988) Minimum fault coverage in reconfigurable arrays. In Digest of Papers 18th International Symposium on Fault-Tolerant Computing, pp. 348–353
59. Hasler P, Lande TS (2001) Overview of floating-gate devices, circuits, and systems. IEEE Trans Circuits Syst II Analog Digit Signal Process 48(1):1–3
60. Hayes JP, Friedman AD (1974) Test point placement to simplify fault detection. IEEE Trans Comput C–23(7):727–735
61. Haykin SS (2009) Neural networks and learning machines. Pearson Education, third edition, Upper Saddle River, NJ
62. Hearst MA, Dumais ST, Osuna E, Platt J, Schölkopf B (1998) Support vector machines. 13:18–28
63. Henftling M, Wittmann H, Antreich KJ (1995) A formal non-heuristic ATPG approach. Proceedings of the Conference on European Design Automation, pp. 248–253
64. Ho TK (1995) Random decision forests. in Proc International Conference on Document Analysis and Recognition (ICDAR), pp. 278–282
65. Ho TK (1995) Random decision forests. In Proceedings of 3rd International Conference on Document Analysis and Recognition, volume 1, pp. 278–282 vol. 1
66. Hoehfeld M, Fahlman SE (1992) Probabilistic rounding in neural network learning with limited precision. Neurocomputing 4(6):291–299
67. Holler MA, Tam SM, Castro HA, Benson R (1989) An electrically trainable artificial neural network (ETANN) with 10240 'floating gate' synapses. In Proc. International Joint Conference on Neural Networks, pp. 191–196
68. Hopfield J, Tank D (2004) Neural computation of decisions in optimization problems. Biol Cybern 52:141–152
69. Hotelling H (1933) Analysis of a complex of statistical variables into principal components. J Educ Psychol 24(6):417–441
70. Huang Q, Fang C, Mittal S, Blanton RD (2018) Improving diagnosis efficiency via machine learning. In Proc. IEEE International Test Conference (ITC), pp. 1–10
71. Huang Y, Benware B, Klingenberg R, Tang H, Dsouza J, Cheng WT (2017) Scan chain diagnosis based on unsupervised machine learning. In 2017 IEEE 26th Asian Test Symposium (ATS), pp. 225–230
72. Huang Y, Guo R, Cheng W, Li JC (2008) Survey of scan chain diagnosis. IEEE Design Test Comput 25(3):240–248
73. Huang Y, Hsieh H, Lu L (2007) A low-noise amplifier with integrated current and power sensors for RF BIST applications. In Proc. 25th IEEE VLSI Test Symposium (VTS'07), pp. 401–408
74. Huisman LM, Kassab M, Pastel L (2004) Data mining integrated circuit fails with fail commonalities. In Proc. International Test Conference, pp. 661–668
75. Immanuel J, Millican SK (2020) Calculating signal controllability using neural networks: improvements to testability analysis and test point insertion. In Proc. IEEE 29th North Atlantic Test Workshop (NATW), pp. 1–6
76. Jabri M, Flower B (1991) Weight perturbation: An optimal architecture and learning technique for analog VLSI feedforward and recurrent multilayer networks. Neural Comput 3(4):546–565
77. Jain SK, Agrawal VD (1985) Statistical fault analysis. IEEE Design Test Comput 2:38–44
78. Jensen F, Petersen NE (1982) Burn-in. John Wiley & Sons Inc, Chichester, UK
79. Jiang YM, Cheng KT (1999) Analysis of performance impact caused by power supply noise in deep submicron devices. In Proceedings Design Automation Conference, pp. 760–765
80. Jin S, Ye F, Zhang Z, Chakrabarty K, Gu X (2016) Efficient board-level functional fault diagnosis with missing syndromes. IEEE Trans Comput Aided Des Integr Circuits Syst 35(6):985–998
81. Kaufman L, Rousseeuw PJ (2008) Partitioning around medoids (Program PAM), pp. 68–125. John Wiley & Sons, Inc
82. Kelly J, Engelhardt M (2007) Advanced production testing of RF, SoC, and SiP devices. Artech House Inc, Boston
83. Butler KM, Carulli Jr JM, Saxena J, Vasavada AP (2011) System and method for estimating test escapes in integrated circuits. U.S. Patent 7865849B2
84. Kirkland T, Mercer MR (1987) A topological search algorithm for ATPG. Proceedings of the 24th ACM/IEEE Design Automation Conference, pp. 502–508
85. Kohonen T (2002) The self-organizing map. Proc IEEE 78(9):1464–1480
86. Koosh VF, Goodman RM (2002) Analog VLSI neural network with digital perturbative learning. IEEE Trans Circuits Syst II Analog Digit Signal Process 49(5):359–368
87. Krishnamurthy B (1987) Hierarchical test generation: Can AI help? In Proc, International Test Conf
88. Kunz W, Pradhan DK (1992) Recursive learning: An attractive alternative to the decision tree for test generation in digital circuits. In Proceedings of the IEEE International Test Conference, pp. 816–825
89. Kuo S, Fuchs WK (1987) Efficient spare allocation for reconfigurable arrays. IEEE Des Test Comput 4(1):24–31
90. Laaksonen J, Oja E (1996) Classification with learning k-nearest neighbors. In Proc. International Conference on Neural Networks (ICNN) 3:1480–1483
91. Larrabee T (1989) Efficient generation of test patterns using Boolean difference. In Proceedings International Test Conference, pp. 795–801
92. Larrabee T (1992) Test pattern generation using boolean satisfiability. IEEE Trans CAD 11(1):4–15
93. Li YH, Lien WC, Lin IC, Lee KJ (2014) Capture-power-safe test pattern determination for at-speed scan-based testing. IEEE Trans Comput Aided Des Integr Circuits Syst 33(1):127–138
94. Li Z, Colburn JE, Pagalone V, Narayanun K, Chakrabarty K (2017) Test-cost optimization in a scan-compression architecture using support-vector regression. In Proc. IEEE 35th VLSI Test Symposium (VTS), pp. 1–6
95. Linares-Barranco B, Serrano-Gotarredona T, Serrano-Gotarredona R (2003) Compact low-power calibration

mini-DACs for neural arrays with programmable weights. IEEE Trans Neural Netw 14(5):1207–1216
96. Littlestone N, Warmuth MK (1989) The weighted majority algorithm. In Proc. 30th Annual Symposium on Foundations of Computer Science, pp. 256–261
97. Liu Y, Han C, Lin S, Li JC (2017) PSN-aware circuit test timing prediction using machine learning. IET Comput Digit Tech 11(2):60–67
98. Lombardi F, Huang WK (1988) Approaches for the repair of VLSI/WSI RRAMs by row/column deletion. In Digest of Papers, 18th International Symposium on Fault-Tolerant Computing, pp. 342–347
99. Lont JB, Guggenbuhl W (1992) Analog CMOS implementation of a multilayer perceptron with nonlinear synapses. IEEE Trans Neural Netw 3(3):457–465
100. Ma Y, Ren H, Khailany B, Sikka H, Luo L, Natarajan K, Yu B (2019) High performance graph convolutional networks with applications in testability analysis. In Proc. 56th ACM/IEEE Design Automation Conference (DAC), pp. 1–6
101. Maliuk D, Stratigopoulos HG, Huang H, Makris Y (2010) Analog neural network design for RF built-in self-test. In Proc. International Test Conference (ITC), pp. 23.2.1–23.2.10
102. Manzini A, Inglese P, Caldi L, Cantero R, Carnevale G, Coppetta M, Giltrelli M, Mautone N, Irrera F, Ullmann R, Bernardi P (2019) A machine learning-based approach to optimize repair and increase yield of embedded flash memories in automotive systems-on-chip. In Proc. IEEE European Test Symposium (ETS), pp. 1–6
103. Marques Silva JP, Sakallah KA (1996) GRASP - a new search algorithm for satisfiability. In Proceedings of International Conference on Computer Aided Design, pp. 220–227
104. Mateo D, Altet J, Aldrete-Vidrio E (2007) Electrical characterization of analogue and RF integrated circuits by thermal measurements. Microelectron J 38(2):151–156
105. Mazumder P, Jih YS (1993) A new built-in self-repair approach to VLSI memory yield enhancement by using neural-type circuits. IEEE Trans Comput Aided Des Integr Circuits Syst 12(1):124–136
106. Milev M, Hristov M (2003) Analog implementation of ANN with inherent quadratic nonlinearity of the synapses. IEEE Trans Neural Netw 14(5):1187–1200
107. Millican S, Sun Y, Roy S, Agrawal V (2021) System and method for optimizing fault coverage based on optimized test point insertion determinations for logical circuits. U.S. Patent 17226950
108. Millican SK, Sun Y, Roy S, Agrawal VD (2019) Applying neural networks to delay fault testing: test point insertion and random circuit training. In Proc. IEEE 28th Asian Test Symposium (ATS), pp. 13–18
109. Moness M, Gabor L, Hussein AI, Ali HM (2022) Automated design error debugging of digital VLSI circuits. J Electron Test Theory Appl 38(4):395–417
110. Montalvo AJ, Gyurcsik RS, Paulos JJ (1997) Toward a general-purpose analog VLSI neural network with on-chip learning. IEEE Trans Neural Netw 8(2):413–423
111. Montgomery DC, Peck EA, Vining GG (2012) Introduction to linear regression analysis. Wiley Publishing Company, Incorporated
112. Najafi-Haghi ZP, Wunderlich H (2023) Identifying resistive open defects in embedded cells under variations. J Electron Test Theory Appl 39(1):27–40
113. O'Farrill C, Moakil-Chbany M, Eklow B (2005) Optimized reasoning-based diagnosis for non-random, board-level, production defects. in Proc. IEEE International Test Conference, pp. 1–7 (Paper 8.2)
114. Ooi MP, Kwang Joo Sim E, Kuang YC, Kleeman L, Chan C, Demidenko S (2010) Automatic defect cluster extraction for semiconductor wafers. In Proc. IEEE Instrumentation Measurement Technology Conference Proceedings, pp. 1024–1029
115. Patel J, Patel S (1985) What heuristics are best for PODEM?. In Proc. First International Workshop on VLSI Design, pp. 1–20
116. Patel S, Patel J (1986) Effectiveness of heuristics measures for automatic test pattern generation. In Proc. 23rd ACM/IEEE Design Automation Conference (DAC), pp. 547–552
117. Pearson K (1901) On lines and planes of closest fit to systems of points in space. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2(11):559–572
118. Peng H, Long F, Ding C (2005) Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Trans Pattern Anal Mach Intell 27(8):1226–1238
119. Pradhan M, Bhattacharya BB (2020) A survey of digital circuit testing in the light of machine learning. WIREs Data Mining Knowl Discov 1–18
120. Pradhan M, Bhattacharya BB, Chakrabarty K, Bhattacharya BB (2019) Predicting *X*-sensitivity of circuit-inputs on test-coverage: A machine-learning approach. IEEE Trans Comput Aided Des Integr Circuits Syst 38(12):2343–2356
121. Quinlan JR (1986) Induction of decision trees. Mach Learn 1(1):81–106
122. Roberts MW, Lala PK (1987) Algorithm to detect reconvergent fanouts in logic circuits. IEE Proceedings E - Computers and Digital Techniques 134(2):105–111
123. Roth JP, Bouricius WG, Schneider PR (1967) Programmed algorithms to compute tests to detect and distinguish between failures in logic circuits. IEEE Transactions on Electronic Computers EC–16(5):567–580
124. Roy S (2021) Toward zero backtracks in test pattern search algorithms with machine learning. PhD thesis, Auburn University, USA
125. Roy S, Agrawal VD (2024) An amalgamated testability measure derived from machine intelligence. in Proceedings of 37th International Conference on VLSI Design & 23rd International Conference on Embedded Systems
126. Roy S, Millican SK, Agrawal VD (2020) Machine intelligence for efficient test pattern generation. In Proceedings of the IEEE International Test Conference, (Washington D.C), pp. 1–5
127. Roy S, Millican SK, Agrawal VD (2021) Principal component analysis in machine intelligence-based test generation. In Proc. IEEE Microelectronics Design and Test Symp. (MDTS'21), (USA), pp. 1–6
128. Roy S, Millican SK, Agrawal VD (2021) Special session - machine learning in test: A survey of analog, digital, memory, and RF integrated circuits. In Proc. IEEE VLSI Test Symp. (VTS'21), (USA), Apr. 2021, pp. 1–10
129. Roy S, Millican SK, Agrawal VD (2021) Training neural network for machine intelligence in automatic test pattern generator. In Proceedings of 34th International Conference on VLSI Design & 20th International Conference on Embedded Systems, pp. 316–321
130. Roy S, Millican SK, Agrawal VD (2021) Unsupervised learning in test generation for digital integrated circuits. In Proceedings of the IEEE European Test Symposium, pp. 1–4
131. Roy S, Millican SK, Agrawal VD (2022) Multi-heuristic machine intelligence guidance in automatic test pattern generation. In Proc. 31st Microelectronics Design and Test Symposium (MDTS), pp. 1–6
132. Roy S, Stiene B, Millican SK, Agrawal VD (2019) Improved random pattern delay fault coverage using inversion test points. In Proc. IEEE 28th North Atlantic Test Workshop (NATW), pp. 206–211

133. Roy S, Stiene B, Millican SK, Agrawal VD (2020) Improved pseudo-random fault coverage through inversions: A study on test point architectures. J Electron Testing Theory Applic 36(1):123–133
134. Ryu JY, Kim BC (2005) Low-cost testing of 5 GHz low noise amplifiers using new RF BIST circuit. J Electron Test Theory Appl 21(6):571–581
135. Savir (1983) Good controllability and observability do not guarantee good testability. IEEE Transactions on Computers C-32(12)
136. Schölkopf B, Smola AJ, editors (2001) Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT Press
137. Schulz MH, Auth E (1988) Advanced automatic test pattern generation and redundancy identification techniques. In Digest of Papers, 18th International Symposium on Fault-Tolerant Computing, pp. 30–35
138. Schulz MH, Auth E (1989) Improved deterministic test pattern generation with applications to redundancy identification. IEEE Trans Comput Aided Des Integr Circuits Syst 8(7):811–816
139. Schulz MH, Trischler E, Sarfert TM (1988) SOCRATES: A highly efficient automatic test pattern generation system. IEEE Trans Comput Aided Des Integr Circuits Syst 7(1):126–137
140. Seth SC, Agrawal VD (1989) A new model for computation of probabilistic testability in combinational circuits. Integr VLSI J 7:49–75
141. Seth SC, Agrawal VD, Farhat H (1990) A statistical theory of digital circuit testability. IEEE Trans Comput 39(4):582–586
142. Shan C, Babighian P, Pan Y, Carulli J, Wang L (2017) Systematic defect detection methodology for volume diagnosis: A data mining perspective. In Proc. IEEE International Test Conference (ITC), pp. 1–10
143. Shepard KL, Narayanan V (1996) Noise in deep submicron digital design. In Proceedings of International Conference on Computer Aided Design, pp. 524–531
144. Silva E, Pineda de Gyvez J, Gronthoud G (2005) Functional vs. multi-VDD testing of RF circuits. In Proc. IEEE International Test Conference, pp. 9–420
145. Singh A, Bharadwaj LM, Harpreet S (2005) DNA and quantum based algorithms for VLSI circuits testing. Nat Comput 4:53–72
146. Singh S, Singh A (2003) Applying quantum search to automated test pattern generation for VLSI circuits. In Proc. 4th International Conf. on Parallel and Distributed Computing, Applications and Technologies, (Chengdu, China), pp. 648–651
147. Singhee A, Rutenbar RA (2009) Statistical blockade: Very fast statistical simulation and modeling of rare circuit events and its application to memory design. IEEE Trans Comput Aided Des Integr Circuits Syst 28(8):1176–1189
148. Skabar A (2003) Single-class classifier learning using neural networks: An application to the prediction of mineral deposits. In Proceedings of the 2003 International Conference on Machine Learning and Cybernetics, volume 4, pp. 2127–2132 Vol.4
149. Stapper CH, McLaren AN, Dreckmann M (1980) Yield model for productivity optimization of VLSI memory chips with redundancy and partially good product. IBM J Res Dev 24(3):398–409
150. Stephan P, Brayton RK, Sangiovanni-Vincentelli AL (1996) Combinational test generation using satisfiability. IEEE Trans Comput Aided Des Integr Circuits Syst 15:1167–1176
151. Stephenson JE, Grason J (1976) A testability measure for register transfer level digital circuits. In Proc. 6th International Fault Tolerant Computing Symp, pp. 101–107
152. Stratigopoulos H (2018) Machine learning applications in IC testing. In Proc. IEEE 23rd European Test Symposium (ETS), pp. 1–10
153. Stratigopoulos H, Makris Y (2008) Error moderation in low-cost machine-learning-based analog/RF testing. IEEE Trans Comput Aided Des Integr Circuits Syst 27(2):339–351
154. Stratigopoulos H, Mir S (2012) Adaptive alternate analog test. IEEE Des Test Comput 29(4):71–79
155. Stratigopoulos H, Mir S, Acar E, Ozev S (2009) Defect filter for alternate RF test. In Proc. 14th IEEE European Test Symposium, pp. 101–106
156. Stratigopoulos H, Mir S, Makris Y (2009) Enrichment of limited training sets in machine-learning-based analog/RF test. In Proc. Design, Automation & Test in Europe Conference & Exhibition, pp. 1668–1673
157. Stratigopoulos H, Sunter S (2014) Fast Monte Carlo-based estimation of analog parametric test metrics. IEEE Trans Comput Aided Des Integr Circuits Syst 33(12):1977–1990
158. Sumikawa N, Nero M, Wang L (2017) Kernel based clustering for quality improvement and excursion detection. In Proc. IEEE International Test Conference (ITC), pp. 1–10
159. Sun Y, Millican SK (2019) Test point insertion using artificial neural networks. In Proc. IEEE Computer Society Annual Symposium on VLSI (ISVLSI), pp. 253–258
160. Sun Y, Millican SK (2022) Applying artificial neural networks to logic built-in self-test: Improving test point insertion. J Electron Test Theory Appl 38(4):339–352
161. Sun Y, Millican SK, Agrawal VD (2020) Special session: Survey of test point insertion for logic built-in self-test. In Proc. IEEE 38th VLSI Test Symposium (VTS), pp. 1–6
162. Sun Z, Jiang L, Xu Q, Zhang Z, Wang Z, Gu X (2013) AgentDiag: An agent-assisted diagnostic framework for board-level functional failures. In Proc. IEEE International Test Conference (ITC), pp. 1–8
163. Sun Z, Jiang L, Xu Q, Zhang Z, Wang Z, Gu X (2015) On test syndrome merging for reasoning-based board-level functional fault diagnosis. In Proc. 20th Asia and South Pacific Design Automation Conference, pp. 737–742
164. Tafertshofer P, Ganz A, Henftling M (1997) A SAT-based implication engine for efficient ATPG, equivalence checking, and optimization of netlists. 1997 Proceedings of IEEE International Conference on Computer Aided Design (ICCAD), pp. 648–655
165. Tan SC, Ting KM, Liu TF (2011) Fast anomaly detection for streaming data. IJCAI'11, AAAI Press, p. 1511-1516
166. Tang H, Manish S, Rajski J, Keim M, Benware B (2007) Analyzing volume diagnosis results with statistical learning for yield improvement. In Proc. 12th IEEE European Test Symposium (ETS'07), pp. 145–150
167. Tehranipoor M, Butler KM (2010) Power supply noise: A survey on effects and research. IEEE Des Test Comput 27(2):51–67
168. Tipping ME (2004) Bayesian inference: An introduction to principles and practice in machine learning, pp. 41–62. Berlin, Heidelberg: Springer Berlin Heidelberg
169. Valdes-Garcia A, Venkatasubramanian R, Silva-Martinez J, Sanchez-Sinencio E (2008) A broadband CMOS amplitude detector for on-chip RF measurements. IEEE Trans Instrum Meas 57(7):1470–1477
170. Voorakaranam R, Akbay SS, Bhattacharya S, Cherubal S, Chatterjee A (2007) Signature testing of analog and RF circuits: Algorithms and methodology. IEEE Trans Circuits Syst I Regul Pap 54(5):1018–1031
171. Wang H, Poku O, Yu X, Liu S, Komara I, Blanton RD (2012) Test-data volume optimization for diagnosis. In Proc. Design Automation Conference, pp. 567–572
172. Wang J, Walker DMH, Majhi A, Kruseman B, Gronthoud G, Villagra LE, van de Wiel P, Eichenberger S (2006) Power supply noise in delay testing. In Proc. IEEE International Test Conference, pp. 1–10
173. Wang S, Wei W (2009) Machine learning-based volume diagnosis. In Proc. Design, Automation & Test in Europe Conference & Exhibition, pp. 902–905

174. Wen X, Yamashita Y, Kajihara S, Wang LT, Saluja KK, Kinoshita K (2005) On low-capture-power test generation for scan testing. In Proc. 23rd IEEE VLSI Test Symposium (VTS), pp. 265–270
175. Wey CL, Lombardi F (1987) On the repair of redundant RAM's. IEEE Trans Comput Aided Des Integr Circuits Syst 6(2):222–231
176. Xanthopoulos C, Sarson P, Reiter H, Makris Y (2017) Automated die inking: A pattern recognition-based approach. In Proc. IEEE International Test Conference (ITC), pp. 1–6
177. Xiao Y, Huang X, Liu K (2021) Model transferability from imagenet to lithography hotspot detection. J Electronic Testing Theory Applications 37(1):141–149
178. Xue Y, Poku O, Li X, Blanton RD (2013) PADRE: Physically-aware diagnostic resolution enhancement. In Proc. IEEE International Test Conference (ITC), pp. 1–10
179. Ye F, Firouzi F, Yang Y, Chakrabarty K, Tahoori MB (2014) On-chip voltage-droop prediction using support-vector machines. In Proc. IEEE 32nd VLSI Test Symposium (VTS), pp. 1–6
180. Ye F, Zhang Z, Chakrabarty K, Gu X (2013) Board-level functional fault diagnosis using artificial neural networks, support-vector machines, and weighted-majority voting. IEEE Trans Comput Aided Des Integr Circuits Syst 32(5):723–736
181. Ye F, Zhang Z, Chakrabarty K, Gu X (2014) Board-level functional fault diagnosis using multikernel support vector machines and incremental learning. IEEE Trans Comput Aided Des Integr Circuits Syst 33(2):279–290
182. Zhang W, Goh AT (2016) Multivariate adaptive regression splines and neural network models for prediction of pile drivability. Geoscience Frontiers 7(1)45–52. Special Issue: Progress of Machine Learning in Geosciences
183. Zhang Y, Agrawal VD (2010) A diagnostic test generation system. In Proc. IEEE International Test Conference (ITC), pp. 12.3.1–12.3.9
184. Zhang Z, Chakrabarty K, Wang Z, Wang Z, Gu X (2011) Smart diagnosis: Efficient board-level diagnosis and repair using artificial neural networks. In Proc. International Test Conference, pp. 1–9
185. Zhang Z, Gu X, Xie Y, Wang Z, Wang Z, Chakrabarty K (2012) Diagnostic system based on support-vector machines for board-level functional diagnosis. In Proc. 17th IEEE European Test Symposium (ETS), pp. 1–6
186. Zhou Z, Guin U, Li P, Agrawal VD (2021) Defect characterization and testing of skyrmion-based logic circuits. In Proc. IEEE VLSI Test Symp. (VTS'21), (USA), pp. 1–7
187. Zhou Z, Guin U, Li P, Agrawal VD (2022) Fault modeling and test generation for technology-specific defects of skyrmion logic circuits. In Proc. IEEE VLSI Test Symp. (VTS'22), (USA), pp. 1–7

Soham Roy is a DFT CAD Engineer in the department of Design for Test Engineering Group at Intel Corporation, Santa Clara, CA, USA. He received his Bachelor of Technology (BTech) Degree in Electronics and Instrumentation from West Bengal University of Technology, Kolkata, India, in 2011. He was with Wipro Ltd., VLSI Division, Bangalore, India, as a design for test engineer from 2011-2015. He received his Master of Science (MS) degree from the Department of Electrical and Computer Engineering, Technical University of Dresden, Dresden, Germany, in 2018. He received his Doctor of Philosophy (PhD) in Electrical and Computer Engineering from the Auburn University, USA, in 2021. He has published several articles and has filed patents in applying machine learning in test point insertion (TPI) and automatic test pattern generation (ATPG). He worked as Technology Development Module and Integration Yield Engineer at Intel Corporation, Hillsboro, OR, USA. His research interest includes VLSI design and test and artificial intelligence.

Spencer K. Millican received his PhD, MS, and BS degrees from the University of Wisconsin - Madison in 2015, 2013, and 2011, respectively. He was with IBM in Rochester, MN, USA, as a design for test engineer for two years, and afterwards was an assistant professor at Auburn University. He is currently a Chief Microelectronics Hardware Engineer at Dynetics, Inc. where he applies his knowledge of circuit testing to the defense of safety-critical circuits. He has published several articles, including receiving the best paper award at the 2014 IEEE International Conference on VLSI Design, and he has received patents in the field of encrypted circuit simulation. His research interests include logic built-in self-test, obfuscated circuits, and secure microelectronics.

Vishwani D. Agrawal is an Emeritus Professor in the Department of Electrical and Computer Engineering at Auburn University, Alabama, USA. Prior to retirement in 2016, he was the James J. Danaher Professor in the same department. He has over 45 years of industry and university experience, working at Auburn University, Bell Labs, Murray Hill, NJ, USA; Rutgers University, New Brunswick, NJ, USA; TRW, Redondo Beach, CA, USA; Indian Institute of Technology Delhi (IITD), New Delhi, India; EG &G, Albuquerque, NM, USA; and ATI, Champaign, IL, USA. His areas of expertise include VLSI testing, low-power design, and microwave antennas. He obtained a BE (1964) degree from the Indian Institute of Technology Roorkee (IITR), Roorkee, India; ME (1966) from the Indian Institute of Science, Bangalore, India; and PhD (1971) from the University of Illinois at Urbana-Champaign (UIUC), IL, USA. He has coauthored over 400 papers and 5 books, and holds 13 United States patents. He is the Editor-in-Chief of the Journal of Electronic Testing: Theory and Applications, and a past Editor-in-Chief (1985-87) of the IEEE Design & Test of Computers magazine. His invited talks include the plenary (1998) at the International Test Conference, Washington, DC, USA and the keynote (2012) at the 25th International Conference on VLSI Design, Hyderabad, India. He served on the Board of Governors (1989-90) of the IEEE Computer Society, and in 1994 chaired the Fellow Selection Committee of that Society. He has received ten Best Paper Awards, two Lifetime Achievement Awards, and two Distinguished Alumnus Awards. Agrawal is a Fellow of the ACM, IEEE and IETE-India. He has served on the Advisory Boards of the ECE Departments at UIUC, New Jersey Institute of Technology (NJIT), and the City College of the City University of New York (CCNY). See his website: http://www.eng.auburn.edu/~vagrawal.