Special Session – Machine Learning in Test: A Survey of Analog, Digital, Memory, and RF Integrated Circuits

Soham Roy, Spencer K. Millican, and Vishwani D. Agrawal
Department of Electrical and Computer Engineering
Auburn University, Auburn, AL 36849-5201
{*szr0075, millican, agrawvd*}*@auburn.edu*

***Abstract*—Integrated circuit (IC) testing presents complex problems that, when ICs become large, 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 machines to adopt these human skills is called machine learning (ML). This survey examines applications of ML to testing analog, digital, memory, radio frequency (RF), and other application-based ICs. This survey then highlights significant challenges and potential research directions.**

***Index Terms*—Machine intelligence (MI), machine learning (ML), analog testing, digital testing, memory test and repair, RF testing, hardware security.**

I. Introduction

IC defects behave differently depending on the type of circuit they implement, thus ICs have numerous test methodologies to detect these defects. Analog and RF tests are functional and are designed based on high-level specifications [1], digital tests are structural and target defect-representing faults, and memory tests also target faults but test them in a functional manner [2]. For any circuit type, increasing integration reduces cost, but tests must address increased circuit complexity and test for nuanced faults not seen in previous generations of circuits.

Unfortunately, some test problems, like digital test pattern generation, are computationally complex, while others, like IC yield enhancement, are not easily addressed with simple algorithms. Human intuition can address such problems, but the cost of employing teams of engineers to apply their intuition is too great. To address this, engineers can apply machine learning (ML), also known as machine intelligence (MI) to create novel solutions to test problems. Beyond creating high-power and novel test approaches, ML makes programming easier and reduces software development cycles and costs.

This article presents a survey of ML applied to test. Two recent surveys [3], [4] thoroughly studied specialized ML applications to testing, thus this survey addresses fields absent in previous surveys. Section II studies ML applied to testing analog and RF circuits. Section III explores new ML techniques for digital circuits, which is an additional contribution to recent surveys. Memory testing is the subject of Section IV. Going beyond the classical test domains, Section V discusses recent applications of ML to hardware security, IC counterfeiting, and devices based on emerging technologies. Section VI concludes this survey with open challenges yet addressed by ML in test.

II. Analog and RF Testing

A. Hardware implementation of analog ANN BIST

Complex RF devices are an integral part of modern consumer electronics, and testing them requires sophisticated equipment and methods to ensure compliance and design specifications. Such devices demand more time and indirectly increase manufacturing costs. Reducing test time leads to alternate test strategies: generating signatures that differentiate faulty and fault-free circuits [5], [6]; built-in test (BIT) or using an on-chip tester [7], [8] that switches the DUT into testable mode by fetching signals from sensors [9]–[14]; built-off test (BOT) or converting RF signals to DC signals using an interface (placed on a load board) between design-under-test (DUT) and tester [15], [16]; and implicit test, i.e., statistical model-based testing to make off-line PASS/FAIL decisions [17]–[21].

An ML approach [22] used on-chip artificial neural networks (ANNs) on RF devices under test (DUT) to analyze sensor measurements and indicate PASS/FAIL. The ANN compacts off-chip extraction and post-processing measurements to a single-bit output, making a standalone built-in self-test (BIST) circuit shown in Fig. 1. ANN training is performed off-line by measuring patterns of fabricated chips and mapping them to one-bit outputs (indicating PASS/FAIL) [22]. This training was further enriched [23] by the topology for the ANN in determining local memory; test mode downloads weights of the ANN and the DUT is excited by the on-chip generated stimuli, then measurement acquisition sensors monitor the outputs of DUT and provide measurement patterns to the classifier. A non-linear classifier tries to examine the DUT by comparing its actual and expected one-bit output, and it is shown that a hardware implementation of the ANN can learn optimally to reduce the probability of misclassification for a given measurement pattern.

978-1-6654-1949-9/21/$31.00 ©2021 IEEE

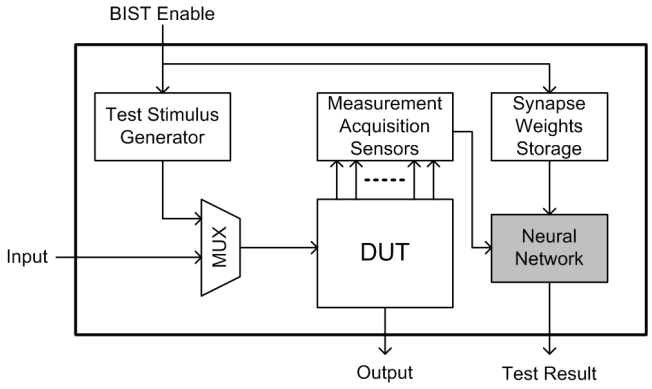

Fig. 1. Built-in self-test (BIST) of a radio frequency (RF) device under test (DUT) [22].

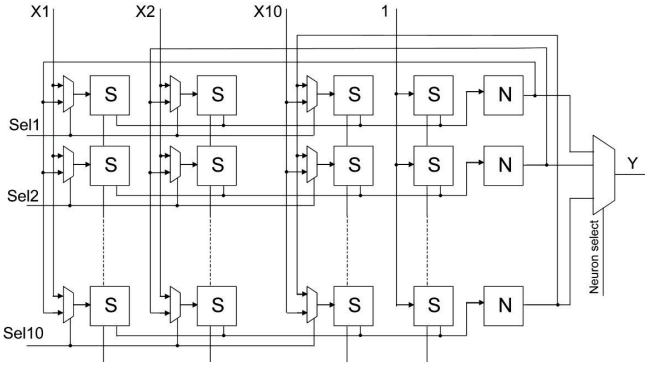

Fig. 2. Reconfigurable ANN [22].

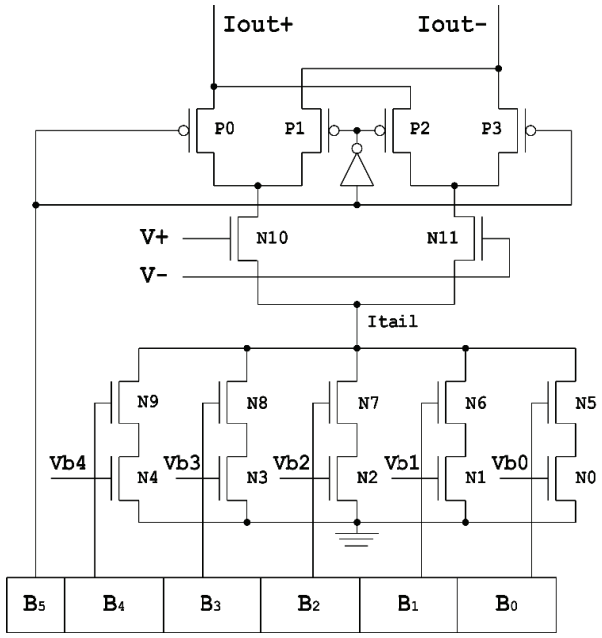

Fig. 3. Schematic diagram of synapse [22].

Analog ANNs in silicon densely pack synapses and computing elements for high parallel processing ability, robustness, and fault tolerance. Such ANNs are superior to digital implementations: they're faster, smaller, consume less power, are easy to reconfigure, and are easy to train. However, analog ANNs designs must consider 1) topology, 2) training algorithms, and 3) weight/bias storage. Fabrication technology makes implementing analog ANNs on silicon difficult since conventional CMOS technologies have significant parameter variation noise [24]–[27].

Fig. 2 illustrates the typical architecture of a reconfigurable, single hidden layer ANN: it comprises of synapses *(S)*, multiplexers, and neurons *(N)* in a matrix. Each synapse is mixed-signal hardware that it performs computation in analog while storing weights and biases in a digital RAM. A schematic of a typical synapse circuit, shown in Fig. 3, illustrates multiplication implemented through a digital-to-analog converter (DAC) [28], a combination of differential input voltages, and a programmable tail current. 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 current "N5-N9" and binary-weighted current sources "N0-N4". Therefore, 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 [24]. 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 the positive difference of "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 the following advantages:

1) Modular and expands up to any number of neurons and inputs within the chip area.
2) The output multiplexer reduces the number of pins and ADC devices.
3) All signals are differential and have broad input ranges and improve the noise resiliency.

Training algorithms for on-chip ANNs are limited and conventional (i.e., backpropagation algorithms) and suffer from low precision and high area overhead. Work in [29] usesperturbation-based, i.e., a parallel stochastic weight perturbation. This is preferred since it does not require on-chip support and provides a more compact solution. In this method, a random vector perturbs all weights of edges of the ANN. The mean squared error (MSE) is calculated over the entire training set to check the error status. If the error decreases,

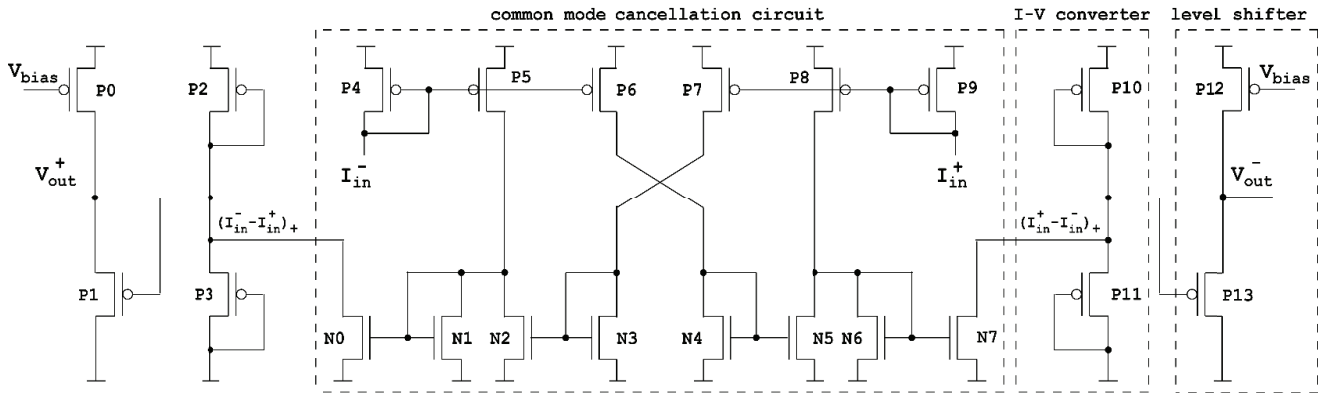

Fig. 4. Schematic diagram of neuron [22].

a new random vector with weights is accepted, otherwise it is discarded. This method is likely to get trapped in local minima, thus several rounds of training may be needed for the right solution. The local minima trap can be avoided by using a simulated annealing technique, which allows the state of the network to move "uphill."

Experiments on low noise amplifiers (LNAs) circuits with two RF amplitude detectors placed at the inputs and outputs of LNA produced DC signals comparable to RF power seen at the detector inputs [22]. These DC signals were fed to an analog ANN classifier. This classifier was trained in different configurations with 2, 4, and 8 neurons in a single hidden layer and 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 is 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 noted by the 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 becomes too low or saturate [30].
3) Weight resolution is problem-specific and depends on network architecture. However, it can be increased in the presence of high non-linearity with minimal size devices but may lead to mismatch and parameter variation in the manufacturing process [31].
4) The training algorithm demonstrated significant convergence properties with minimal variance of the final error, but this required an increase in training time.
5) Weight storage is large since it is implemented as digital memory. However, in BIST, these weights need to be stored permanently, which may require memories using floating gate transistors [32], [33]. Nevertheless, using floating gate memories to store weights of analog neural networks may raise further issues like handling high voltage, accurate programming schemes, and weight updates.
6) Further investigation is needed whether the ML-based approach considers the effects of DUT degradation during the device lifetime.

B. An adaptive RF front-end design

A recent study [34] reexamined the tuning area of low-frequency systems for yield improvement, which has also been studied earlier [35]–[38]. It identified a novel problem of tunable RF front ends and established a low-power channel compatible front end to guarantee high throughput and low power operation. The authors [34] applied their proposed theory to both transmitter and receiver hardware systems. This hardware can be either low power multiple-input-multiple-output (MIMO) or single-input-single-output (SISO) having certain features such as the ability to switch between multiple modulation techniques, different MIMO modes, and different code-rates of channels. The key contributions are as follows [34]:

1) Although prior work demonstrated channel adaption for SISO systems, the channel adaption technique applied to MIMO systems resulted in insignificant performance margins across the system. The proposed technique [34] applied a use-aware channel adaption technique to a prototype "2 x 2 MIMO transceiver" and observed seamless switching between multiple operational modes and modulation techniques depending upon the channel ability to adapt to low-power operations. This technique holds equally well for SISO systems.
2) ANN-based channel adaption while switching channel coding rates, modulation rates, and different MIMO modes has benefits in terms of throughput, power, and energy savings [34]. This ML-based front-end design is useful for low-power IC-based applications.

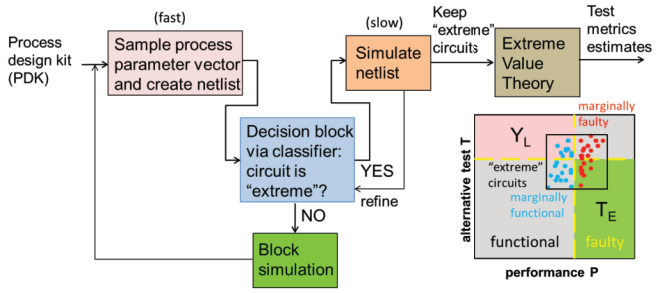

Fig. 5. Simulation flow for parametric test metrics estimation [3].

C. Estimation of parametric test metrics using machine learning algorithms

There is a need to replace standard specification-based testing to improve fault coverage of analog ICs. Test engineers have procedures to estimate analog parameters to reduce test costs, yield loss, and test escapes, but these test metrics should be accurately estimated at the simulation level ahead of silicon manufacturing to minimize the test development time and cost. A proposed test strategy [3], [39] that includes ML algorithms, shown in Fig. 5, illustrates the following:

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 the available process design kit (PDK) [3] from the IP vendors and simultaneously trains the ANN with the process metrics to classify the "extreme" circuits.
3) These "extreme" circuits define a statistical blockade [40].
4) The ANN is re-trained with the simulated circuit netlists so that the class of the "extreme" circuits is closer to the specification and the process is repeated.
5) The re-training of ANN progressively increases the probability of generating correct "extreme" instances.

"Extreme" instances can serve as fault models based on parameters that could examine high-performance by applying an alternative test scheme [41], [42]. This method speeds up Monte Carlo transistor-level simulations. Typically, fault models account for process parameters based on their joint distribution as given in their respective PDKs [41]. Finally, the fault model is verifiable after performing a transistor-level simulation. The algorithm [39] outputs a more refined parametric fault model compared to the generalized technique-specific fault model and helps estimate fault coverage and yield loss more precisely. This method is applied to a low noise amplifier (LNA) [41] and was shown to reduce simulation run-time by eliminating the redundant specification tests and replacing them with the proposed ML-based parametric measurements. This technique was applied to data-converters [42], but it has yet to be explored for other analog ICs whose simulation run-time is high, such as phase-locked-loops (PLLs).

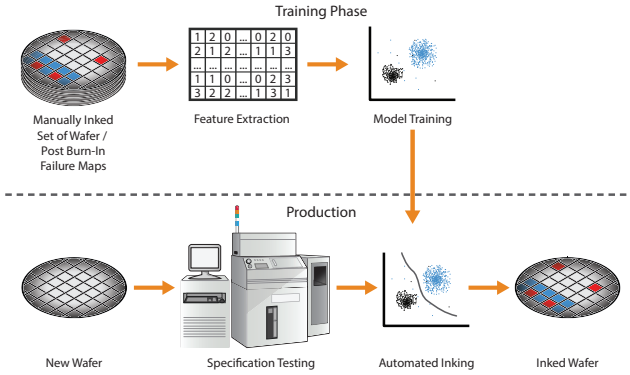

Fig. 6. ML-based die inking process [46].

Additionally, "extreme" circuits can estimate test escapes [43] to reduce test cost and develop analog IC test procedures. The statistical blockade algorithm [40] can accurately predict defective PPM and confidence intervals through circuit simulation.

III. Digital Testing

A. Wafer testing

Generally, logic defects occur in clusters on the wafers [44], thus clustering algorithms [45] can identify defect concentrations across the wafers. Such algorithms works in two steps: 1) cluster containment and 2) learning. The algorithm first identifies the wafers with cluster patterns and screens out passing dies having no defects inside these clusters, thereby marking them for high risk of failure. This process repeats based on cluster size, wafer location, and failure composition across multiple wafers to avoid additional yield loss and failure analysis. Work in [46] proposed a similar cluster-detecting ML algorithm using support vectors machines (SVMs). The SVM's kernel is a radial basis function, generally a Gaussian function, for distance computation to identify the die from the defective clusters during classification, as shown in Fig. 6.

B. Scan chain diagnosis

Defective scan latches can fail with permanent faults (which are easy to model) or intermittent faults (which are difficult to model). Various heuristics efficiently diagnose permanent faults in scan chains, but diagnosing intermittent faults requires alternative methods. [47] used Bayesian learning [48] to identify faulty scan cells in the presence of intermittent faults using an unsupervised learning approach. The method analyzes a test set and the corresponding failure log of the faulty scan chain. The product of this is twofold:

1) "Abit" counts the number of test vectors required to capture failure bits (sensitive bits) by respective scan cells.
2) "Bbit" counts the number of "Abit" that fails to capture the scan cell's sensitive bit. The calculation of "Bbit" is the probability that a scan cell captures faulty bits

based on binomial distribution followed by diagnosis using Bayes' formula.

Another work [49] proposed a different ML-based scan diagnosis technique based on supervised learning. It used a multi-level ANN (in this study: the coarse global neural network (CGNN)) to provide high-resolution scan diagnosis. 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 binary response vectors compressed into a single integer failure vector (IFV) computed by performing bit-wise addition of all response binary vectors. The number of scan latches in scan chains 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 method [49] also proposed a different solution to the compression of binary response vectors by concatenating it sequentially to form a single vector. The author proposed an affine group comprising those scan cells whose euclidean distance between their IFV and candidate scan cells' faulty scan cell is minimal. The length of such a modified IFV, also known as "reduced cascaded vector (RCV)," can be reduced by removing the bits at certain positions based on the affine group. This updated CGNN comprises 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. This novel scan diagnosis methodology achieved reasonably high diagnostic accuracy.

C. Classifying fault models

Defective ICs can be used to produce failure logs for fault diagnosis, but logging substantial data can be memory-expensive, and analysis of the entire dataset is time-consuming and may be infeasible. To address this [50] used ML to decide when data collection can be stopped without sacrificing the efficiency of fault diagnosis. This idea was demonstrated by using different types of ML: k-nearest neighbors (kNN), SVMs, and decision trees. Beyond this, a survey [51] of diagnosis using machine learning examines relevancy of failure log information for fault diagnosis, locating defects in scan chain or functional logic block, and diagnosis time.

Fault diagnosis plays a vital role as a prerequisite for the physical failure analysis (PFA), but too many candidate faults diminish diagnostic efficacy and leads to low diagnosis resolution. To alleviate the diagnosis process, ML-based techniques try to meet specific objectives such as 1) mapping of diagnosed faults onto corresponding defects based on the failure response of the circuit [52], [53] and 2) tuning the set of candidate faults to improve diagnostic resolution further [54]. The ANNs used in these studies get help from the layout and logical information of the circuit and failure response.

D. Board testing

Testing each component on a board is vital from the real-time testing perspective. Even when an in-circuit test [55] of individual components using automatic test equipment (ATE) passes, the board-level functional test can fail. This phenomenon is a foreboding and severe issue that needs a structured way of testing to guarantee the reliability of a 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 new boards' defective components. The syndromes for faulty boards serve as a set of features, and the diagnosed root-causes serves as labels in the training data set.

A reasoning-based approach model [56] is effective in functional debugging as it continuously learns during debugging and development. However, it is difficult to fix the model if reasoning-based learning is improper and incorrectly identifies a faulty component on the board. Replacement of the entire model is trivial, but it could adversely affect the correct detection of a failure syndrome. The authors kept the fixation of their model as an open problem for the future.

An ML-based method [57] proposed a technique to debug and repair board-level functional failures. The method uses a fundamental connection between failure syndromes and repair actions to train an ANN not to infer from visual inspection of log files and datasets.

An SVM-based technique [58], [59] can diagnose boards by learning incrementally to locate the root causes of the failures. The learning tunes a SVM kernel to achieve high accuracy in diagnosis. The overall system training time improves with incremental learning of SVM.

Weighted-majority voting [60] using both ANN and SVM can optimize repair [61], [62]. 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 to train the ML models to diagnose failures is a significant concern. Work in [63] proposed a technique of syndrome merging to reduce feature vector sizes. However, some syndromes are not computable and do not merge. A proposed technique [64] can still diagnose a system with a non-computable or missing syndrome using a label-imputation method and two-feature-selection methods.

E. Volume diagnosis

Conventional diagnostic tools claim to be highly accurate, but they fail to identify certain faults by not considering layout information. Such faults occur due to systematic defects, and EDA tools and yield learning methods such as PFA are incapable of handling them. This can be addressed by analyzing the fail-logs of multiple ICs, also known as volume

diagnosis. This involves analysis of large amounts of data, but this is time-consuming and expensive.

An ML-based technique [65] can be embedded into the yield-learning process to identify systematic defects and distinguish them from random defects. This method works by creating signatures of defective ICs using failure responses and clustering them using the farthest-neighbor method [66]. Work in [67] extended this technique to identify defect locations in fanout-free regions by observing how systematic faults affect the same set of outputs. To do this, the circuit is decomposed in fanout-free regions based on a specific kind of defect or defect class, which are then classified based on failure outputs using an 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. Two methods, namely a statistical-learning approach [68], and a Bayesian network-based approach [69] evaluate the failure feature probability.

The ML-based volume diagnosis technique [67] has been compared in terms of run time to traditional analysis. The authors assume that faults in fanout free regions in a circuit can be excited, propagated through common paths, and observed at common scan latches. The proposed technique has several advantages: 1) since the proposed technique relies on certain decision-based subroutines, computation complexity is much lower than traditional volume diagnosis methods; 2) the proposed technique provides high-resolution diagnosis and statistical data, which classifies defective chips based on the defect location; 3) the ML-based technique also works for scan compressed designs and locates defect locations in most defective ICs. Experimental results [67] confirm that the proposed technique could detect more than 90% defective chips in a 50x output compacted design, which is faster than the traditional diagnosis methods. Nevertheless, the proposed diagnosis technique also showed that it could detect 86% defective chips with 100x outputs compacted designs in a few milliseconds.

An ML-based method that assists PFA [70] provides high-resolution detection of defects. According to the authors, defects are grouped in various modes known as "defective modes". A statistical tool such as a χ^2 independence test is applied to the data obtained from layout-aware scan diagnosis. The χ^2 tests evaluate the amount of correlation between the defects and the "defective modes". The "defective 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).

F. Test compression

Recently, due to technology node width shrinkage, reduction in test cost of high-density ICs is a primary concern to be addressed by technology advances. Test cost is estimated in terms of test data volume and time to generate them.

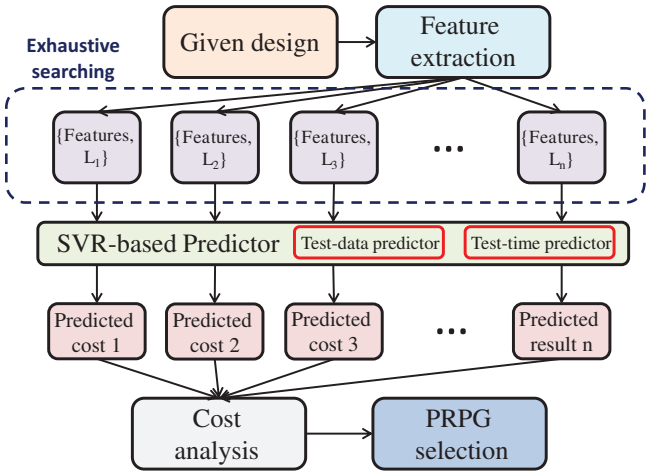

Fig. 7. PRPG selection methodology [71].

Traditionally, compressor/decompressor (CODEC) reduces the test cost by loading scan chains using decompressors and compacting test responses using multiple input signature registers (MISRs) [72]. However, the length of a pseudo-random pattern generator (PRPG) does impact the test time irrespective of various circuit parameters [71]. The problem of the length of PRPG may be solved 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 [71] also give a correlation-based feature selection method applied to industrial designs for reducing the test time with high prediction accuracy [73].

G. Testability analysis

1) ***X-sensitivity prediction****:* X-sources degrade the quality of fault detection and their sources can be many: these include uninitialized memory cells, bus contentions, anomalous analog-to-digital conversion, and manufacturing defects during post-silicon validation. Work in [73] used a SVR X-sensitivity predictor which predicts the sensitivity of the circuit inputs toward the value "X". The method ranked circuit inputs to prioritize X-sensitivity and tries to mask or eliminate inputs with the highest priorities or rectify the manufacturing defects by reconfiguring the responsible components.

2) ***Signal probability calculation****:* The IC test community has several testability measures such as COP [74], SCOAP [75], and CAMELOT [76], but these measures do not consider the effect of re-convergent fanouts. Savir [77] conjectured that it is impossible to calculate a simple testability measure based on controllability and observability of a circuit containing re-convergent fanouts, but almost all industrial circuits contain re-convergent fanouts.

A method from [78] can detect re-convergent fanouts, but pin-counts are limited and the method is computationally burdensome since it requires exhaustive circuit simulation. To

address this, [79] used an ANN to predict signal probabilities given minimal fanout information, and the result was increased circuit accuracy with minimal computation time overhead.

*3) **Test point insertion**:* To improve the fault coverage of logic built-in self-test (LBIST), designers can insert test points (TPs) to modify the circuit's internal signal values and observe signals to detect random pattern resistant (RPR) faults. Test point insertion (TPI) [80] techniques find high-quality TPs to improve fault coverage or reduce the test count. These techniques are classified based on how testability is measured: fault simulation, probabilistic testability measures, and multiple measurements [81].

A deep learning-based technique to tackle the TPI problem of a logic circuit was proposed [82]. The technique uses a graph convolutional network (GCN) to analyze circuit nodes and classify them as either as easy-to-observe or difficult-to-observe points. The ANN analyzes attributes of each node and the node's neighbors, most notably the nodes' SCOAP [75] values. Work in [83]–[86] used fully-connected neural networks to evaluate control-0, control-1, and observe testpoints' impact on fault coverage and found iterative TPI had improved fault coverage and massivly reduced TPI time. Work in [84] further trained the ANN-based on random circuits and found such an ANN yields similar performance compared to training on benchmark circuits.

H. Timing analysis

IR drop is a significant concern in the design and test of ICs [87], [88] regarding the control of power supply noise (PSN) since unrestrained PSN may lead to performance glitches and impact timing [89], [90] and excessive PSN may create false failures during test. Some test patterns are known to induce substantial PSN that exceeds the functional mode behavior [91]–[94]. Therefore, PSN-induced circuit simulation is not trivial and needs attention. Timing analysis is vital because it determines the clock frequency for the IC. Circuit timing depends on static and dynamic characteristics, because PSN impacts the input voltage reaching individual gates in the circuit, produces propagation delays, and slows down the switching.

A SVR [95] could predict voltage droop in a field-programmable gated array (FPGA) that dynamically adjusts the clock frequency of a processor. However, without feature extraction, the method is applicable only to small ICs. Another ML-based technique [96] includes feature extraction methods, such as ANNs [97], SVRs [95], [98], and least-square boosting (LSBoost) [98]. Here, ANNs are the best predictors in terms of predicting the circuit timing for all test patterns.

I. Built-in self-test (BIST)

The problem of improving fault coverage during pseudo-random testing is significant and techniques to achieve high fault coverage are many. This survey already discussed the TPI where the structure of the circuit under test (CUT) is modified, but another alternative is to modify the test patterns for high fault coverage.

Conventional pseudo-random do not include specific test patterns that detect RPR faults, thus [99] used an ANN to generate test patterns that detect RPR as well as easy-to-detect faults. A self-learning capability suitable for SoCs also deals with aging-induced degradation. The proposed flow uses existing LBIST and an ML-based software predictor to remedy the problems that arise from wear-out/aging of ICs in the field. The ANN is developed using LBIST patterns (converted from ATPG-generated transition delay faults), which activate critical/near-critical paths and corresponding responses. The results demonstrate [99] that a gate-overlap and path delay aware algorithm can select the optimum set of test vectors, and the proposed methodology is area and test-time efficient.

J. 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 complexity increases exponentially with circuit size.

Roth's D-ALG [100] first conceptualized ATPG by defining the D algebra and giving a complete search algorithm. The symbol D represents a composite state of a signal in 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-ALG has high complexity for large circuits as it manipulates all internal signals of the circuit. It is especially inefficient for circuits containing *XOR* gates and re-convergent fanouts. The PODEM [101] algorithm improves the search efficiency by focusing on PIs. It allows the use of various heuristics to speed up the search and checks to prevent fruitless searches through the following characteristics.

- The search space is reduced from 2^n, where n is the total number of signals (gates and PI) in the circuit, to $2^{\#PI}$.
- The concept of X-path-check is introduced. Here, X refers to an unknown or yet undetermined value of a signal. D-ALG 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. D-frontier is the set of all gates that have a D or $\overline{D}$ at their input but the output is still unknown.
- PODEM originally proposed a distance-based heuristic to identify easy or hard to control inputs of logic gates while backtracing, as opposed to D-ALG which chose a random gate input. Several other heuristics based on the circuit topology have been proposed.

The FAN algorithm [102] proposes improvements over PODEM by introducing additional heuristics: immediate implications of signal assignments, unique sensitization, headlines, and multiple backtraces to restrict the search space. Their main

contribution is a breadth-first backtrace as opposed to depth-first strategy in PODEM.

Dominator ATPG programs [103] provide further improvement. A *dominator* is a signal through which a fault's effect must necessarily pass to reach a PO. Signals controlling dominators within the fault effect cone must be assigned non-controlling values to allow the fault effect propagation. These assignments are compulsory and can be determined by circuit topology, without search. Therefore, some faults may prove to be redundant without search.

The SOCRATES [104]–[106] program comprises of static and dynamic learning. Static learning, a form of preprocessing, assigns all signals with 0/1 and saves implications. The same procedure is used dynamically at each step in the search algorithm to find a test. Dynamically learned ATPG is costly but provides more scope to identify the implied signals that further help to find a test quickly.

EST [107]–[109] uses a form of dynamic programming. This is the only ATPG algorithm that finds a test for a target fault taking help from the tests of previously detected faults. EST introduced E-frontier, which signifies a sub-set between the circuit lines being already assigned and not assigned i.e., X (also includes D-Frontier). E-frontiers are generated at each decision step of ATPG and stored. ATPG continues by comparing the current E-frontiers and prior-learned E-frontiers by a circuit decomposition process. EST also uses multiple parallel backtraces of the ATPG, which eventually speeds up the process.

A recursive learning [110] program was introduced to improve the FAN algorithm by applying SOCRATES style learning recursively to signals as a part of implications. It has an advantage over SOCRATES due to the recursive, or repetitive, nature. The test generation time for recursive learning may grow exponentially, but the memory grows linearly with recursion depth.

TRAN [111], [112] formulates ATPG as a Boolean satisfiability (SAT) problem. It uses implication graphs and transitive closure for faster signal assignments and fewer backtracks than other ATPG algorithms. Several other authors have used satisfiability (SAT) [113]–[115]. Larrabee [113], [114] used path variables to find the solution efficiently. Other SAT based ATPG programs include GRASP [116], NEMESIS [114], TEGUS [117], and those reported by Henftling *et al.* [118] and Tafertshofer *et al.* [119].

Although some of the above algorithms have flavor of ML, in the first explicit application, ANN is used to model a digital circuit where a bi-directional binary neuron represents the state of a signal [120]. When the neural network is modified with a fault, the stable state of the network is also the minimum energy state thus finding it gives a test in the form of the states of PI neurons. This application to 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 has many local minima, which can make the search for a test (absolute minimum) for some faults rather difficult.

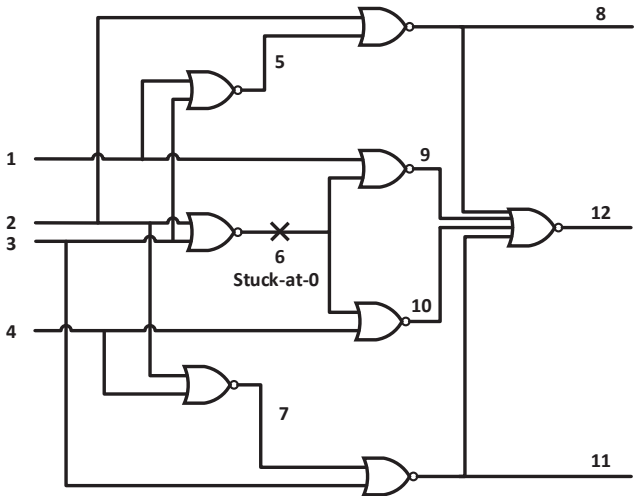

Fig. 8. Schneider circuit [100].

The second application of ML is related to the heuristic part of ATPG algorithms. All algorithms have been using different heuristics to speed-up the ATPG process. These heuristics can be the distance between the PIs and outputs to signal sites, testability measures, voting of digital logic values on fanout stem depending on its branches, other learning techniques using implication graphs, etc [121], [122]. ANN-based ML [123], [124] or unsupervised ML [125] can learn from ATPG with any or no fixed set of rules to guide tracing backward, and develop an intelligent system to use circuit information (in this case it is distance and COP) during ATPG. The measure of success will be fewer backtracks compared to conventional heuristics.

This section ends with an interesting illustration. In 1967 paper, Roth et al. [100] used the Schneider circuit of Figure 8 as an example to demonstrate the exhaustiveness of D-algorithm: it would always find a test if one existed. The fault *line 6 stuck-at-0* has a test 0000, which can be found if paths through lines 9 and 10 are simultaneously sensitized to output 12. Some of the older single path sensitization algorithms would fail to find this test. The Schneider circuit can show a benefit of ML, which is to reduce backtracks. Three PODEM-based ATPG programs were used to find the test for *line 6 stuck-at-0* fault. These programs are described in [123], [124] on the use of ML in ATPG. The first program, PODEM(D), is a straightforward implementation of PODEM with a backtrace heuristic of logic distance [101]. The second program, PODEM(C), is similar but backtracing is guided by COP testability measures [74]. The third program, PODEM(M), has an ANN-guiding the backtrace. The ANN was trained from histories of successful and failed backtraces when conventional PODEM was applied to typical logic circuits, as well as logic distance and COP data of circuit nodes. For the target fault in Figure 8, all three programs found the test, which

is not surprising because PODEM is an exhaustive algorithm. However, PODEM(D) and PODEM(C) each had to backtrack once, while PODEM(M) required no backtrack. Also, when the ANN was trained with just one feature (either distance or COP), it did require one backtrack. For complex circuits, one may not be so lucky to have zero backtracks but experience shows that ML by combining multiple heuristics, produces tests with fewer backtracks [123].

IV. Memory testing and repair

A. ML based BISR DRAM test and repair

The width of technology nodes of VLSI circuits decreases rapidly, and therefore the yield is also decreasing due 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 bypasses the faulty area and repairs them. DRAMs are densely packed, and redundant rows and columns are added to reconfigure faulty cells of rows and columns of memory sub-arrays using electronically programmable latches. The problem of optimal reconfiguration and redundant component allocation is an old problem and widely studied by many researchers [126]. 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 BIST that comprehensively tests these memory arrays and discards them if they fail the tests. This 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, [127], but increasing memory sizes makes the search space too large and the types of faults become complicated. Therefore, conventional repair algorithms, both greedy [128] and exhaustive [129], became ineffective. The memory repair problem was then formulated as NP-complete [130], and heuristic algorithms were introduced that included branch and bound [131], approximation [131], best-first search [132], and others [133], [134]. These algorithms have worst-case complexities that are nearly exponential and they are not easily implementable in BISR. Focus then shifted [135] to: 1) devising an efficient algorithm so that overall throughput improves with the chip yield and 2) hardware implementable algorithms. The paper [135] focuses on a self-repair scheme using BISR to repair memory subarrays by reconfiguring the redundant rows/columns. As "Repair Most (RM)" is simple and easily implementable hardware, the performance of ANN-based memory repair algorithm was compared against RM [135].

ANNs have been used to tackle optimization problems, e.g., the famous *Travelling Salesman Problem* [136], as proposed by Hopfield [137]. Lyapunov's energy function can represent an optimization cost function, and the convergence property of the NN from a random initial state to a local minimum state can reduce this cost by using a gradient descent algorithm. However, this kind of ANN formulation has low-quality, and therefore a proposed algorithm [135] 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. In other words, conventional repair algorithms run slow on digital computers, whereas ANN's collective computational property provides a faster solution. A gradient descent algorithm [135] 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 [135] 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 a best candidate for the self-repair circuit. 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 NN.

B. ML based software-assisted in-chip self-test flash memory test and repair

Automotive IC testing must ensure chips function correctly after calibration, test, and repair of flash memories [138], and doing so requires redundant memory cells (i.e., spare word lines "WLs" and bit lines "BLs") and activation of these redundant structures. Redundant component analysis can be done on-line in software-assisted in-chip self-test (SIST) [138], but a major bottleneck is reconfiguring the redundant components efficiently (i.e., quickly and accurately). A bitmap scheme was used to reconfigure faulty memory cells by downloading the cell coordinates, but later it proved to be ineffective and time-consuming and therefore was not regular industrial 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 due to the repair algorithm's inability.

One must deal with false fail identification and the prohibiting unnecessary repair cases [138]. The vital step before using an ML-based predictor to identify false fails is to extract training features. Training features were extracted using a coloring algorithm [139], where every fault is assigned a

unique color, and their occurrences are evaluated statistically. This algorithm combines different faults with unique colors to provide a chunk of datasets to an ANN.

An ML-based technique [138] works by 1) in the development phase, bitmaps are collected for selected devices that compose training/test datasets and 2) during production, training features are extracted using the coloring algorithm [139] and the discarded devices are labeled as false failures. A detailed analysis was done on the extracted training features and the results were fed back to the color algorithm designers. Supervised and unsupervised training techniques were deployed to assess the false fails and whether or not they are correctly discriminated against in training features. Artificial bitmaps were added to original bitmaps to keep unaltered fail signature characteristics: bitwise and, or, xor, noise, and many more. These additional bitmaps provided more comprehensive training datasets including false fails and significantly better prediction accuracy. It is also found [138] that the training data sets are highly unbalanced and therefore a confusion matrix is used, which 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 must not affect the overall test time of the IC production flow.

Experimental results [138] showed that the ML-based predictor of a model "decision tree" compared to other models such as "random forest" and "feed-forward" has a better score and minimal variance with the fastest and easy-to-implement sub-routine. 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.

C. Improving SRAM yield using statistical blockade

As transistor sizes shrink, the statistical blockade technique [40] is found to improve the yield of SRAM ICs since they contain high repeatable components. This technique has proved to be more efficient and novel than the conventional Monte Carlo methods and significantly provides accuracy and speedup of approximately two orders of magnitude across circuits despite of parametric variations.

V. Hardware security and emerging technologies

Recently, ML has been used for hardware defense and attack. Hardware defense techniques use ML-based algorithms against hardware Trojans and IC counterfeiting. Countermeasures with and without a golden chip are two broad categories to defend against hardware Trojans. Methods extract training data from golden chips to classify on-chip sensor data [140]–[143], gate-level design nodes [144], and traffic congestion prevails on-chip [145]–[147].

IC counterfeiting is an alarming and prominent threat to the IC manufacturing industry. Manual inspection and detection of counterfeit ICs is accurate but time-consuming, thus an ML approach [148] inspected ICs using ANN-based image classification. SVM analysis [143] of on-chip sensor sensor data has been used to identify recycled ICs. FPGAs contain many ring oscillators (ROs), and their frequencies may degrade due to the intrusion of some defects [149]; SVM (supervised learning) [143] or K-means clustering (unsupervised learning) [150] were used on operating frequencies of IC to detect recycled ICs [151]. Although ML-based algorithms have not been directly applied to reverse engineering, they have been used to identify golden chips [152], [153]. ML-based algorithms were also used to apply profiling- and non-profiling-based side-channel analysis attacks, typically used in cryptographic secret extraction [154]–[160]. ML-based algorithms also applied profiling-based side-channel analysis for instruction-based assembly [161]–[163].

VI. Conclusions

This survey highlighted key aspects of ML-based testing of analog, digital, memory, RF, and counterfeit devices, and motivates integrating ML into testing ICs of the future. However, certain properties of ML may become a menace for ICs used in the defense, healthcare, space, and automotive industries:

- Recently, ML has been on 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 hardware security can prove dangerous if the attackers use ML-based model to attack either good (false positive) or bad (false negative) ICs.
- Emerging technology design and their conventional testing is in a nascent stage and full of imperfections and variations, and therefore supplying products based on these technologies using ML may not provide confidence to IC suppliers and customers.

ML-based testing requires either a repair or reconfiguration mechanism, which may provide redundancy in terms of area overhead, but may provide a fast prediction of anomalies in a system with robustness and resiliency. There is ample research scope in intelligent lithographic hotspot detection [164], 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 [165], monolithic 3D (M3D) devices (specifically resistive RAMs (ReRAMs)) [166] and many more.

References

[1] J. Kelly and M. Engelhardt, *Advanced Production Testing of RF, SoC, and SiP Devices*. Boston: Artech House, Inc., 2007.

[2] R. D. Adams, *High Performance Memory Testing*. Frontiers in Electronic Testing Book Series, Springer, 2003.

[3] H. Stratigopoulos, "Machine learning applications in IC testing," in *Proc. IEEE 23rd European Test Symposium (ETS)*, 2018, pp. 1–10.

[4] M. Pradhan and B. B. Bhattacharya, "A Survey of Digital Circuit Testing in the Light of Machine Learning," *WIREs Data Mining Knowl. Discov.*, pp. 1–18, 2020.

[5] E. Silva, J. Pineda de Gyvez, and G. Gronthoud, "Functional vs. multi-VDD testing of RF circuits," in *Proc. IEEE International Test Conference*, 2005, pp. 9–420.

[6] E. Acar and S. Ozev, "Defect-Oriented Testing of RF Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 5, pp. 920–931, 2008.

[7] M. M. Hafed, N. Abaskharoun, and G. W. Roberts, "A 4-GHz effective sample rate integrated test core for analog and mixed-signal circuits," *IEEE Journal of Solid-State Circuits*, vol. 37, no. 4, pp. 499–514, 2002.

[8] J.-Y. Rye and B. C. Kim, "Low-Cost Testing of 5 GHz Low Noise Amplifiers Using New RF BIST Circuit," *J. Electronic Testing*, vol. 21, pp. 571–581, 2005.

[9] A. Gopalan, M. Margala, and P. R. Mukund, "A current based self-test methodology for RF front-end circuits," *Microelectronics Journal*, vol. 36, no. 12, pp. 1091–1102, 2005.

[10] M. Cimino, H. Lapuyade, M. De Matos, T. Taris, Y. Deval, and J. B. Begueret, "A Robust 130nm-CMOS Built-In Current Sensor Dedicated to RF Applications," in *Proc. Eleventh IEEE European Test Symposium (ETS'06)*, 2006, pp. 151–158.

[11] Y. Huang, H. Hsieh, and L. Lu, "A Low-Noise Amplifier with Integrated Current and Power Sensors for RF BIST Applications," in *Proc. 25th IEEE VLSI Test Symposium (VTS'07)*, 2007, pp. 401–408.

[12] D. Mateo, J. Altet, and E. Aldrete-Vidrio, "Electrical characterization of analogue and RF integrated circuits by thermal measurements," *Microelectronics Journal*, vol. 38, no. 2, pp. 151–156, 2007, 2005 Workshop on Thermal Investigations of ICs and Systems (THERMINIC).

[13] A. Valdes-Garcia, R. Venkatasubramanian, J. Silva-Martinez, and E. Sanchez-Sinencio, "A Broadband CMOS Amplitude Detector for On-Chip RF Measurements," *IEEE Transactions on Instrumentation and Measurement*, vol. 57, no. 7, pp. 1470–1477, 2008.

[14] L. Abdallah, H. Stratigopoulos, C. Kelma, and S. Mir, "Sensors for built-in alternate RF test," in *Proc. 15th IEEE European Test Symposium (ETS)*. Los Alamitos, CA, USA: IEEE Computer Society, 2010, pp. 49–54.

[15] J. Ferrario, R. Wolf, S. Moss, and M. Slamani, "A low-cost test solution for wireless phone RFICs," *IEEE Communications Magazine*, vol. 41, no. 9, pp. 82–88, 2003.

[16] S. Bhattacharya and A. Chatterjee, "A DFT Approach for Testing Embedded Systems Using DC Sensors," *IEEE Design Test of Computers*, vol. 23, no. 6, pp. 464–475, 2006.

[17] S. S. Akbay, J. L. Torres, J. M. Rumer, A. Chatterjee, and J. Amtsfield, "Alternate Test of RF Front Ends with IP Constraints: Frequency Domain Test Generation and Validation," in *Proc. IEEE International Test Conference*, 2006, pp. 1–10.

[18] S. Ellouz, P. Gamand, C. Kelma, B. Vandewiele, and B. Allard, "Combining Internal Probing with Artificial Neural Networks for Optimal RFIC Testing," in *Proc. IEEE International Test Conference*, 2006, pp. 1–9.

[19] R. Voorakaranam, S. S. Akbay, S. Bhattacharya, S. Cherubal, and A. Chatterjee, "Signature Testing of Analog and RF Circuits: Algorithms and Methodology," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 54, no. 5, pp. 1018–1031, 2007.

[20] H. Stratigopoulos, S. Mir, E. Acar, and S. Ozev, "Defect Filter for Alternate RF Test," in *Proc. 14th IEEE European Test Symposium*, 2009, pp. 101–106.

[21] H. Stratigopoulos and Y. Makris, "Error Moderation in Low-Cost Machine-Learning-Based Analog/RF Testing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 2, pp. 339–351, 2008.

[22] D. Maliuk, H.-G. Stratigopoulos, H. Huang, and Y. Makris, "Analog Neural Network Design for RF Built-In Self-Test," in *Proc. International Test Conference (ITC)*, 2010, pp. 23.2.1–23.2.10.

[23] H. Stratigopoulos, S. Mir, and Y. Makris, "Enrichment of limited training sets in machine-learning-based analog/RF test," in *Proc. Design, Automation & Test in Europe Conference & Exhibition*, 2009, pp. 1668–1673.

[24] J. B. Lont and W. Guggenbuhl, "Analog CMOS Implementation of a Multilayer Perceptron with Nonlinear Synapses," *IEEE Press*, vol. 3, no. 3, p. 457–465, 1992.

[25] M. Milev and M. Hristov, "Analog implementation of ANN with inherent quadratic nonlinearity of the synapses," *IEEE Transactions on Neural Networks*, vol. 14, no. 5, pp. 1187–1200, 2003.

[26] Holler, Tam, Castro, and Benson, "An electrically trainable artificial neural network (ETANN) with 10240 'floating gate' synapses," in *Proc. International 1989 Joint Conference on Neural Networks*, 1989, pp. 191–196.

[27] A. J. Montalvo, R. S. Gyurcsik, and J. J. Paulos, "Toward a general-purpose analog VLSI neural network with on-chip learning," *IEEE Transactions on Neural Networks*, vol. 8, no. 2, pp. 413–423, 1997.

[28] V. F. Koosh and R. M. Goodman, "Analog VLSI neural network with digital perturbative learning," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 49, no. 5, pp. 359–368, 2002.

[29] M. Jabri and B. Flower, "Weight Perturbation: An Optimal Architecture and Learning Technique for Analog VLSI Feedforward and Recurrent Multilayer Networks," *Neural Computation*, vol. 3, no. 4, pp. 546–565, 1991.

[30] M. Hohfeld and S. E. Fahlman, "Probabilistic rounding in neural network learning with limited precision," *Neurocomputing*, vol. 4, no. 6, pp. 291 – 299, 1992.

[31] B. Linares-Barranco, T. Serrano-Gotarredona, and R. Serrano-Gotarredona, "Compact low-power calibration mini-DACs for neural arrays with programmable weights," *IEEE Transactions on Neural Networks*, vol. 14, no. 5, pp. 1207–1216, 2003.

[32] R. R. Harrison, J. A. Bragg, P. Hasler, B. A. Minch, and S. P. Deweerth, "A CMOS programmable analog memory-cell array using floating-gate circuits," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 48, no. 1, pp. 4–11, 2001.

[33] P. Hasler and T. S. Lande, "Overview of floating-gate devices, circuits, and systems," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 48, no. 1, pp. 1–3, 2001.

[34] D. Banerjee, S. K. Devarakond, X. Wang, S. Sen, and A. Chatterjee, "Real-Time Use-Aware Adaptive RF Transceiver Systems for Energy Efficiency Under BER Constraints," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 8, pp. 1209–1222, 2015.

[35] X. Wang, B. Kenfack, E. Silva, and A. Chatterjee, "Built-In Test of Switched-Mode Power Converters: Avoiding DUT Damage Using Alternative Safe Measurements," in *Proc. 22nd Asian Test Symposium*, 2013, pp. 56–61.

[36] X. Wang, K. Blanchard, S. Estella, and A. Chatterjee, "Alternative safe test of hysteretic power converters," in *Proc. IEEE 32nd VLSI Test Symposium (VTS)*. Los Alamitos, CA, USA: IEEE Computer Society, apr 2014, pp. 1–6.

[37] X. Wang, K. Blanchard, S. Estella, and A. Chatterjee, "A self-tuning architecture for buck converters based on alternative test," in *Proc. International Test Conference*, 2014, pp. 1–10.

[38] S. Hsiao, X. Wang, and A. Chatterjee, "Analog Sensor Based Testing of Phase-Locked Loop Dynamic Performance Parameters," in *Proc. 22nd Asian Test Symposium*, 2013, pp. 50–55.

[39] H. Stratigopoulos and S. Mir, "Adaptive Alternate Analog Test," *IEEE Design Test of Computers*, vol. 29, no. 4, pp. 71–79, 2012.

[40] A. Singhee and R. A. Rutenbar, "Statistical Blockade: Very Fast Statistical Simulation and Modeling of Rare Circuit Events and Its Application to Memory Design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 8, pp. 1176–1189, 2009.

[41] H. Stratigopoulos and S. Sunter, "Fast Monte Carlo-Based Estimation of Analog Parametric Test Metrics," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 12, pp. 1977–1990, 2014.

[42] M. J. Barragan, H. Stratigopoulos, S. Mir, H. Le-Gall, N. Bhargava, and A. Bal, "Practical Simulation Flow for Evaluating Analog/Mixed-Signal Test Techniques," *IEEE Design Test*, vol. 33, no. 6, pp. 46–54, 2016.

[43] H. Stratigopoulos, "Test Metrics Model for Analog Test Development," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 7, pp. 1116–1128, 2012.

[44] M. P. Ooi, E. Kwang Joo Sim, Y. C. Kuang, L. Kleeman, C. Chan, and S. Demidenko, "Automatic Defect Cluster Extraction for Semiconductor Wafers," in *Proc. IEEE Instrumentation Measurement Technology Conference Proceedings*, 2010, pp. 1024–1029.

[45] N. Sumikawa, M. Nero, and L. Wang, "Kernel based clustering for quality improvement and excursion detection," in *Proc. IEEE International Test Conference (ITC)*, 2017, pp. 1–10.

[46] C. Xanthopoulos, P. Sarson, H. Reiter, and Y. Makris, "Automated die inking: A pattern recognition-based approach," in *Proc. IEEE International Test Conference (ITC)*, 2017, pp. 1–6.

[47] Y. Huang, R. Guo, W. Cheng, and J. C. Li, "Survey of Scan Chain Diagnosis," *IEEE Design Test of Computers*, vol. 25, no. 3, pp. 240–248, 2008.

[48] M. E. Tipping, *Bayesian Inference: An Introduction to Principles and Practice in Machine Learning*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 41–62.

[49] M. Chern, S.-W. Lee, S.-Y. Huang, Y. Huang, G. Veda, K.-H. H. Tsai, and W.-T. Cheng, "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)*, 2019, pp. 341–346.

[50] H. Wang, O. Poku, X. Yu, S. Liu, I. Komara, and R. D. Blanton, "Test-data volume optimization for diagnosis," in *Proc. Design Automation Conference*, 2012, pp. 567–572.

[51] Q. Huang, C. Fang, S. Mittal, and R. D. Blanton, "Improving Diagnosis Efficiency via Machine Learning," in *Proc. IEEE International Test Conference (ITC)*, 2018, pp. 1–10.

[52] L. R. Gómez and H. Wunderlich, "A Neural-Network-Based Fault Classifier," in *Proc. IEEE 25th Asian Test Symposium (ATS)*, 2016, pp. 144–149.

[53] L. R. Gómez, A. Cook, T. Indlekofer, S. Hellebrand, and H.-J. Wunderlich, "Adaptive Bayesian Diagnosis of Intermittent Faults," *J. Electron. Test.*, vol. 30, no. 5, p. 527–540, Oct. 2014.

[54] Y. Xue, O. Poku, X. Li, and R. D. Blanton, "PADRE: Physically-Aware Diagnostic Resolution Enhancement," in *Proc. IEEE International Test Conference (ITC)*, 2013, pp. 1–10.

[55] J. Bateson, *In-Circuit Testing*. New York: Van Nostrand Reinhold Company, 1985.

[56] C. O'Farrill, M. Moakil-Chbany, and B. Eklow, "Optimized reasoning-based diagnosis for non-random, board-level, production defects," in *Proc. IEEE International Test Conference*, 2005, pp. 7 pp.–179.

[57] Z. Zhang, K. Chakrabarty, Z. Wang, Z. Wang, and X. Gu, "Smart diagnosis: Efficient board-level diagnosis and repair using artificial neural networks," in *Proc. International Test Conference*, 2011, pp. 1–9.

[58] F. Ye, Z. Zhang, K. Chakrabarty, and X. Gu, "Board-Level Functional Fault Diagnosis Using Multikernel Support Vector Machines and Incremental Learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 2, pp. 279–290, 2014.

[59] Z. Zhang, X. Gu, Y. Xie, Z. Wang, Z. Wang, and K. Chakrabarty, "Diagnostic system based on support-vector machines for board-level functional diagnosis," in *Proc. 17th IEEE European Test Symposium (ETS)*, 2012, pp. 1–6.

[60] N. Littlestone and M. K. Warmuth, "The weighted majority algorithm," in *Proc. 30th Annual Symposium on Foundations of Computer Science*, 1989, pp. 256–261.

[61] Z. Sun, L. Jiang, Q. Xu, Z. Zhang, Z. Wang, and X. Gu, "AgentDiag: An agent-assisted diagnostic framework for board-level functional failures," in *Proc. IEEE International Test Conference (ITC)*, 2013, pp. 1–8.

[62] F. Ye, Z. Zhang, K. Chakrabarty, and X. Gu, "Board-Level Functional Fault Diagnosis Using Artificial Neural Networks, Support-Vector Machines, and Weighted-Majority Voting," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 5, pp. 723–736, 2013.

[63] Z. Sun, L. Jiang, Q. Xu, Z. Zhang, Z. Wang, and X. Gu, "On test syndrome merging for reasoning-based board-level functional fault diagnosis," in *Proc. 20th Asia and South Pacific Design Automation Conference*, 2015, pp. 737–742.

[64] S. Jin, F. Ye, Z. Zhang, K. Chakrabarty, and X. Gu, "Efficient Board-Level Functional Fault Diagnosis With Missing Syndromes," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 6, pp. 985–998, 2016.

[65] L. M. Huisman, M. Kassab, and L. Pastel, "Data mining integrated circuit fails with fail commonalities," in *Proc. International Test Conference*, 2004, pp. 661–668.

[66] W. R. Dillon and M. Goldstein, *Multivariate Analysis: Methods and Applications*. Wiley Publishing Company, Incorporated, 1984.

[67] S. Wang and W. Wei, "Machine learning-based volume diagnosis," in *Proc. Design, Automation & Test in Europe Conference & Exhibition*, 2009, pp. 902–905.

[68] H. Tang, S. Manish, J. Rajski, M. Keim, and B. Benware, "Analyzing Volume Diagnosis Results with Statistical Learning for Yield Improvement," in *Proc. 12th IEEE European Test Symposium (ETS'07)*, 2007, pp. 145–150.

[69] W. Cheng, Yue Tian, and S. M. Reddy, "Volume diagnosis data mining," in *Proc. 22nd IEEE European Test Symposium (ETS)*, 2017, pp. 1–10.

[70] C. Shan, P. Babighian, Y. Pan, J. Carulli, and L. Wang, "Systematic defect detection methodology for volume diagnosis: A data mining perspective," in *Proc. IEEE International Test Conference (ITC)*, 2017, pp. 1–10.

[71] Z. Li, J. E. Colburn, V. Pagalone, K. Narayanun, and K. Chakrabarty, "Test-cost optimization in a scan-compression architecture using support-vector regression," in *Proc. IEEE 35th VLSI Test Symposium (VTS)*, 2017, pp. 1–6.

[72] M. L. Bushnell and V. D. Agrawal, *Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits*. Springer Publishing Company, Incorporated, 2013.

[73] M. Pradhan, B. B. Bhattacharya, K. Chakrabarty, and B. B. Bhattacharya, "Predicting X -Sensitivity of Circuit-Inputs on Test-Coverage: A Machine-Learning Approach," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 12, pp. 2343–2356, 2019.

[74] F. Brglez, "On Testability Analysis of Combinational Circuits," *Proc. International Symp. Circuits and Systems*, pp. 221–225, 1984.

[75] L. Goldstein, "Controllability/Observability Analysis of Digital Circuits," *IEEE Transactions on Circuits and Systems*, vol. 26, pp. 685–693, 1979.

[76] R. G. Bennetts, C. M. Maunder, and G. D. Robinson, "COMELOT: a computer-aided measure for logic testability," *IEE Proceedings E - Computers and Digital Techniques*, vol. 128, no. 5, pp. 177–189, 1981.

[77] Savir, "Good Controllability and Observability Do Not Guarantee Good Testability," *IEEE Transactions on Computers*, vol. C-32, no. 12, pp. 1198–1200, 1983.

[78] M. W. Roberts and P. K. Lala, "Algorithm to detect reconvergent fanouts in logic circuits," *IEE Proceedings E - Computers and Digital Techniques*, vol. 134, no. 2, pp. 105–111, 1987.

[79] J. Immanuel and S. K. Millican, "Calculating signal controllability using neural networks: Improvements to testability analysis and test point insertion," in *Proc. IEEE 29th North Atlantic Test Workshop (NATW)*, 2020, pp. 1–6.

[80] J. P. Hayes and A. D. Friedman, "Test Point Placement to Simplify Fault Detection," *IEEE Transactions on Computers*, vol. C-23, no. 7, pp. 727–735, July 1974.

[81] Y. Sun, S. K. Millican, and V. D. Agrawal, "Special session: Survey of test point insertion for logic built-in self-test," in *Proc. IEEE 38th VLSI Test Symposium (VTS)*, 2020, pp. 1–6.

[82] Y. Ma, H. Ren, B. Khailany, H. Sikka, L. Luo, K. Natarajan, and B. Yu, "High Performance Graph Convolutional Networks with Applications in Testability Analysis," in *Proc. 56th ACM/IEEE Design Automation Conference (DAC)*, 2019, pp. 1–6.

[83] Y. Sun and S. K. Millican, "Test Point Insertion Using Artificial Neural Networks," in *Proc. IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2019, pp. 253–258.

[84] S. K. Millican, Y. Sun, S. Roy, and V. D. Agrawal, "Applying Neural Networks to Delay Fault Testing: Test Point Insertion and Random Circuit Training," in *Proc. IEEE 28th Asian Test Symposium (ATS)*, 2019, pp. 13–18.

[85] S. Roy, B. Stiene, S. K. Millican, and V. D. Agrawal, "Improved Random Pattern Delay Fault Coverage Using Inversion Test Points," in *Proc. IEEE 28th North Atlantic Test Workshop (NATW)*, 2019, pp. 206–211.

[86] S. Roy, B. Stiene, S. K. Millican, and V. D. Agrawal, "Improved Pseudo-Random Fault Coverage Through Inversions: a Study on Test Point Architectures," *J. Electron. Test*, vol. 36, no. 1, p. 123–133, Feb. 2020.

[87] K. L. Shepard and V. Narayanan, "Noise in deep submicron digital design," in *Proceedings of International Conference on Computer Aided Design*, 1996, pp. 524–531.

[88] M. Tehranipoor and K. M. Butler, "Power Supply Noise: A Survey on Effects and Research," *IEEE Design Test of Computers*, vol. 27, no. 2, pp. 51–67, 2010.

[89] H. H. Chen and D. D. Ling, "Power Supply Noise Analysis Methodology for Deep-Submicron VLSI Chip Design," in *Proceedings of the 34th Annual Design Automation Conference*, ser. DAC '97. New York, NY, USA: Association for Computing Machinery, 1997, p. 638–643.

[90] Yi-Min Jiang and Kwang-Ting Cheng, "Analysis of performance impact caused by power supply noise in deep submicron devices," in *Proceedings Design Automation Conference (Cat. No. 99CH36361)*, 1999, pp. 760–765.

[91] J. Wang, D. M. H. Walker, A. Majhi, B. Kruseman, G. Gronthoud, L. E. Villagra, P. van de Wiel, and S. Eichenberger, "Power Supply Noise in Delay Testing," in *Proc. IEEE International Test Conference*, 2006, pp. 1–10.

[92] Y. Li, W. Lien, I. Lin, and K. Lee, "Capture-Power-Safe Test Pattern Determination for At-Speed Scan-Based Testing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 1, pp. 127–138, 2014.

[93] Xiaoqing Wen, Y. Yamashita, S. Kajihara, Laung-Terng Wang, K. K. Saluja, and K. Kinoshita, "On low-capture-power test generation for scan testing," in *Proc. 23rd IEEE VLSI Test Symposium (VTS'05)*, 2005, pp. 265–270.

[94] D. Gizopoulos, K. Roy, P. Girard, N. Nicolici, and X. Wen, "Power-Aware Testing and Test Strategies for Low Power Devices," in *Proc. Design, Automation and Test in Europe*, 2008, pp. xliv–xliv.

[95] F. Ye, F. Firouzi, Y. Yang, K. Chakrabarty, and M. B. Tahoori, "On-chip voltage-droop prediction using support-vector machines," in *Proc. IEEE 32nd VLSI Test Symposium (VTS)*, 2014, pp. 1–6.

[96] Y. Liu, C. Han, S. Lin, and J. C. Li, "PSN-aware circuit test timing prediction using machine learning," *IET Computers Digital Techniques*, vol. 11, no. 2, pp. 60–67, 2017.

[97] W. R. Daasch and R. Madge, "Data-driven models for statistical testing: measurements, estimates and residuals," in *Proc. IEEE International Test Conference*, 2005, pp. 10 pp.–322.

[98] C. Bishop, *Pattern Recognition and Machine Learning*. Springer Publishing Company, Incorporated, 2006.

[99] C. FAGOT, P. GIRARD, and C. LANDRAULT, "On using machine learning for logic bist," in *Proc. IEEE International Test Conference*. USA: IEEE Computer Society, 1997, p. 338.

[100] J. P. Roth, W. G. Bouricius, and P. R. Schneider, "Programmed Algorithms to Compute Tests to Detect and Distinguish Between Failures in Logic Circuits," *IEEE Transactions on Electronic Computers*, vol. EC-16, no. 5, pp. 567–580, Oct. 1967.

[101] P. Goel, "An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits," *IEEE Transactions on Computers*, vol. C-30, pp. 215–222, 1981.

[102] Fujiwara and Shimono, "On the Acceleration of Test Generation Algorithms," *IEEE Transactions on Computers*, vol. C-32, pp. 1137–1144, 1983.

[103] T. Kirkland and M. R. Mercer, "A Topological Search Algorithm for ATPG," *Proceedings of the 24th ACM/IEEE Design Automation Conference*, pp. 502–508, 1987.

[104] M. H. Schulz and E. Auth, "Advanced Automatic Test Pattern Generation and Redundancy Identification Techniques," *[1988] The Eighteenth International Symposium on Fault-Tolerant Computing. Digest of Papers*, pp. 30–35, 1988.

[105] M. H. Schulz and E. Auth, "Improved Deterministic Test Pattern Generation With Applications to Redundancy Identification," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 8, pp. 811–816, 1989.

[106] M. H. Schulz, E. Trischler, and T. M. Sarfert, "SOCRATES: A Highly Efficient Automatic Test Pattern Generation System," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 7, pp. 126–137, 1988.

[107] M. L. Bushnell and J. Giraldi, "A Functional Decomposition Method for Redundancy Identification and Test Generation," *J. Electronic Testing*, vol. 10, pp. 175–195, 1997.

[108] K.-T. Cheng, "On Removing Redundancy in Sequential Circuits," in *Proceedings of the 28th ACM/IEEE Design Automation Conference (DAC)*, 1991, pp. 164–169.

[109] K.-T. Cheng and V. D. Agrawal, *Unified Methods for VLSI Simulation and Test Generation*. Springer, 1989.

[110] W. Kunz and D. K. Pradhan, "Recursive Learning: An Attractive Alternative to the Decision Tree for Test Generation in Digital Circuits," in *Proceedings of the IEEE International Test Conference*, 1992, pp. 816–825.

[111] S. T. Chakradhar and V. D. Agrawal, "A Transitive Closure Based Algorithm for Test Generation," in *Proceedings of the 28th ACM/IEEE Design Automation Conference*, ser. DAC '91. New York, NY, USA: Association for Computing Machinery, 1991, pp. 353–358.

[112] S. T. Chakradhar, V. D. Agrawal, and S. G. Rothweiler, "A Transitive Closure Algorithm for Test Generation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 7, pp. 1015–1028, Jul. 1993.

[113] T. Larrabee, "Efficient Generation of Test Patterns Using Boolean Difference," in *Proceedings International Test Conference*, 1989, pp. 795–801.

[114] T. Larrabee, "Test Pattern Generation Using Boolean Satisfiability," *IEEE Trans. on CAD*, vol. 11, no. 1, pp. 4–15, Jan. 1992.

[115] S. T. Chakradhar, "Neural network models and optimization methods for digital testing," Ph.D. dissertation, Rutgers University, USA, 1991.

[116] J. P. Marques Silva and K. A. Sakallah, "GRASP - A New Search Algorithm for Satisfiability," in *Proceedings of International Conference on Computer Aided Design*, 1996, pp. 220–227.

[117] P. Stephan, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, "Combinational Test Generation Using Satisfiability," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, pp. 1167–1176, 1996.

[118] M. Henftling, H. Wittmann, and K. J. Antreich, "A Formal Non-Heuristic ATPG Approach," *Proceedings of the Conference on European Design Automation*, pp. 248–253, 1995.

[119] P. Tafertshofer, A. Ganz, and M. Henftling, "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, 1997.

[120] S. T. Chakradhar, V. D. Agrawal, and M. L. Bushnell, *Neural Models and Algorithms for Digital Testing*. Springer, 1991.

[121] J. Patel and S. Patel, "WHAT HEURISTICS ARE BEST FOR PODEM?" in *Proc. First International Workshop on VLSI Design*, 1985, pp. 1–20.

[122] S. Patel and J. Patel, "Effectiveness of Heuristics Measures for Automatic Test Pattern Generation," in *Proceedings of the 23rd ACM/IEEE Design Automation Conference*, ser. DAC '86. IEEE Press, 1986, p. 547–552.

[123] S. Roy, S. K. Millican, and V. D. Agrawal, "Machine Intelligence for Efficient Test Pattern Generation," in *Proceedings of the IEEE International Test Conference*, Washington D.C, Nov. 2020.

[124] S. Roy, S. K. Millican, and V. D. Agrawal, "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*, 2021.

[125] S. Roy, S. K. Millican, and V. D. Agrawal, "Unsupervised Learning in Test Generation for Digital Integrated Circuits," in *Proceedings of the IEEE European Test Symposium*, 2021.

[126] W. K. Fuchs and M.-F. Chang, *Diagnosis and Repair of Large Memories: A Critical Review and Recent Results*. Boston, MA: Springer US, 1989, pp. 213–225.

[127] C. H. Stapper, A. N. McLaren, and M. Dreckmann, "Yield Model for Productivity Optimization of VLSI Memory Chips with Redundancy and Partially Good Product," *IBM J. Res. Dev.*, vol. 24, no. 3, p. 398–409, May 1980.

[128] R. C. Evans, "Testing Repairable RAMs and Mostly Good Memories," in *Proceedings International Test Conference*. IEEE Computer Society, 1981, pp. 49–55.

[129] J. R. Day, "A Fault-Driven, Comprehensive Redundancy Algorithm," *IEEE Design Test of Computers*, vol. 2, no. 3, pp. 35–44, 1985.

[130] R. W. Haddad, A. T. Dahbura, and A. B. Sharma, "Increased throughput for the testing and repair of RAMs with redundancy," *IEEE Transactions on Computers*, vol. 40, no. 2, pp. 154–166, 1991.

[131] S. Kuo and W. K. Fuchs, "Efficient Spare Allocation for Reconfigurable Arrays," *IEEE Design Test of Computers*, vol. 4, no. 1, pp. 24–31, 1987.

[132] N. Hasan and C. L. Liu, "Minimum fault coverage in reconfigurable arrays," in *Eighteenth International Symposium on Fault-Tolerant Computing. Digest of Papers*, 1988, pp. 348–353.

[133] F. Lombardi and W. K. Huang, "Approaches for the repair of VLSI/WSI RRAMs by row/column deletion," in *Eighteenth International Symposium on Fault-Tolerant Computing. Digest of Papers*, 1988, pp. 342–347.

[134] Chin-Long Wey and F. Lombardi, "On the Repair of Redundant RAM's," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 6, no. 2, pp. 222–231, 1987.

[135] P. Mazumder and Y. . Jih, "A new built-in self-repair approach to VLSI memory yield enhancement by using neural-type circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 1, pp. 124–136, 1993.

[136] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. USA: W. H. Freeman amp; Co., 1990.

[137] J. Hopfield and D. Tank, ""Neural" computation of decisions in optimization problems," *Biological Cybernetics*, vol. 52, pp. 141–152, 2004.

[138] A. Manzini, P. Inglese, L. Caldi, R. Cantero, G. Carnevale, M. Coppetta, M. Giltrelli, N. Mautone, F. Irrera, R. Ullmann, and P. Bernardi, "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)*, 2019, pp. 1–6.

[139] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," *J. Mach. Learn. Res.*, vol. 3, no. null, p. 1157–1182, Mar. 2003.

[140] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their Applications*, vol. 13, no. 4, pp. 18–28, 1998.

[141] A. Heuser and M. Zohner, "Intelligent Machine Homicide," in *Proceedings of the Third International Conference on Constructive Side-Channel Analysis and Secure Design*, ser. COSADE'12. Berlin, Heidelberg: Springer-Verlag, 2012, p. 249–264.

[142] G. Hospodar, B. Gierlichs, E. Mulder, I. Verbauwhede, and J. Vandewalle, "Machine learning in side-channel analysis: A first study," *J. Cryptographic Engineering*, vol. 1, pp. 293–302, 12 2011.

[143] K. Huang, J. M. Carulli, and Y. Makris, "Parametric counterfeit IC detection via Support Vector Machines," in *Proc. IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, 2012, pp. 7–12.

[144] T. Iwase, Y. Nozaki, M. Yoshikawa, and T. Kumaki, "Detection technique for hardware Trojans using machine learning in frequency domain," in *Proc. IEEE 4th Global Conference on Consumer Electronics (GCCE)*, 2015, pp. 185–186.

[145] D. Jap, M. Stöttinger, and S. Bhasin, "Support Vector Regression: Exploiting Machine Learning Techniques for Leakage Modeling," in *Proceedings of the Fourth Workshop on Hardware and Architectural Support for Security and Privacy*, ser. HASP '15. New York, NY, USA: Association for Computing Machinery, 2015.

[146] Y. Jin, D. Maliuk, and Y. Makris, "Post-deployment trust evaluation in wireless cryptographic ICs," in *Proc. Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2012, pp. 965–970.

[147] R. Kohavi, "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, ser. IJCAI'95. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995, p. 1137–1143.

[148] N. Asadizanjani, M. Tehranipoor, and D. Forte, "Counterfeit Electronics Detection Using Image Processing and Machine Learning," *Journal of Physics: Conference Series*, vol. 787, p. 012023, jan 2017.

[149] H. Dogan, D. Forte, and M. M. Tehranipoor, "Aging analysis for recycled FPGA detection," in *Proc. IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, 2014, pp. 171–176.

[150] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A K-Means Clustering Algorithm." *Journal of the Royal Statistical Society*, vol. 28, pp. 100–108, Nov 1979.

[151] M. M. Alam, M. Tehranipoor, and D. Forte, "Recycled fpga detection using exhaustive lut path delay characterization," in *Proc. IEEE International Test Conference (ITC)*, 2016, pp. 1–10.

[152] C. Bao, D. Forte, and A. Srivastava, "On application of one-class SVM to reverse engineering-based hardware Trojan detection," in *Proc. 15th International Symp. Quality Electronic Design*, 2014, pp. 47–54.

[153] C. Bao, D. Forte, and A. Srivastava, "On Reverse Engineering-Based Hardware Trojan Detection," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 1, pp. 49–57, 2016.

[154] J. Li, J. hang Cheng, J. Shi, and F. Huang, "Brief Introduction of Back Propagation (BP) Neural Network Algorithm and Its Improvement," in *Advances in Computer Science and Information Engineering*, 2012, pp. 553–558.

[155] Jun Li, Lin Ni, Jihua Chen, and E. Zhou, "A novel hardware Trojan detection based on BP neural network," in *Proc. 2nd IEEE International Conference on Computer and Communications (ICCC)*, 2016, pp. 2790–2794.

[156] Y. Liu, Y. Jin, A. Nosratinia, and Y. Makris, "Silicon Demonstration of Hardware Trojan Design and Detection in Wireless Cryptographic ICs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 4, pp. 1506–1519, 2017.

[157] H. Maghrebi, T. Portigliatti, and E. Prouff, "Breaking Cryptographic Implementations Using Deep Learning Techniques," *IACR Cryptol. ePrint Arch.*, vol. 2016, p. 921, 2016.

[158] R. L. Rivest, "Learning Decision Lists," *Mach. Learn.*, vol. 2, no. 3, p. 229–246, Nov. 1987.

[159] Hanchuan Peng, Fuhui Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.

[160] H. Salmani, "COTD: Reference-Free Hardware Trojan Detection and Recovery Based on Controllability and Observability in Gate-Level Netlist," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 2, pp. 338–350, 2017.

[161] D. Mandic and J. Chambers, *Recurrent Neural Networks for Prediction: Learning Algorithms,Architectures and Stability*. Wiley Publishing Company, Incorporated, 01 2001.

[162] Z. Martinasek and V. Zeman, "Innovative Method of the Power Analysis," *Radioengineering*, vol. 22, pp. 586–594, 06 2013.

[163] Schölkopf, Bernhard and Smola, Alex J. and Williamson, Robert C. and Bartlett, Peter L., "New support vector algorithms," *Neural Comput.*, vol. 12, no. 5, p. 1207–1245, May 2000.

[164] Y. Xioa, X. Huang, and K. Liu, "Model Transferability from ImageNet to Lithography Hotspot Detection," *J. Electronic Testing*, 2021.

[165] S. Banerjee, A. Chaudhuri, and K. Chakrabarty, "Analysis of the Impact of Process Variations and Manufacturing Defects on the Performance of Carbon-Nanotube FETs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 6, pp. 1513–1526, 2020.

[166] A. Chaudhuri, S. Banerjee, H. Park, J. Kim, G. Murali, E. Lee, D. Kim, S. K. Lim, S. Mukhopadhyay, and K. Chakrabarty, "Advances in Design and Test of Monolithic 3-D ICs," *IEEE Design Test*, vol. 37, no. 4, pp. 92–100, 2020.