The Editor

Journal of Electronic Testing: Theory and Applications

Subject: Revision of previously reviewed manuscript JETT-D-22-00129.

Dear Sir/Madam,

This new manuscript is a revision of our previous manuscript [JETT-D-22-00129] titled “***A Survey and Recent Advances: Machine Intelligence in Electronic Testing***” that was submitted last year. Professor Haralampos Stratigopoulos was the Coordinating Editor. It was examined by two reviewers and the editor asked for a revision based on the review comments.

Our revision took rather long owing to unavoidable circumstances. Two of the three authors, all of whom were at Auburn University, have since moved to Santa Clara, CA, and to Huntsville, AL, respectively. Because of the long delay, the original paper had to be withdrawn, and the revision is now being submitted as a new paper.

We request you to kindly regard this submission as a revision of a previously reviewed manuscript, JETT-D-22-00129. We have included a rebuttal as a summary of our responses to reviewers’ comments and the corresponding changes to the manuscript. Our hope is to make the examination straightforward for the editor.

Following the advice of the editor, Professor Haralampos Stratigopoulos, we are submitting it as a new manuscript, with a request that this new manuscript be assigned to him. Thank you.

Regards,

Soham Roy

Intel Corporation, Santa Clara, CA

November 18, 2023

Authors Note: The comments from the two reviewers allowed us to look at the paper from a knowledgeable reader's viewpoint and helped us revise meaningfully. We hope the changes that are spread all across the manuscript make it a much improved survey.

Reviewer #1:

Reviewer's Comment - AI in electronic testing is clearly a hot topic and for this reason a survey about it is, in general, needed. I also support the idea of covering memories, analog/RF and digital circuits in one text. However, I do see quite a large number of issues with this survey, both on a conceptual level and when covering individual techniques. I think that for being acceptable, this paper has to undergo a rather large-scale revision.

Scope: - Parts of the paper lack focus. For no obvious reason, II.A, after one paragraph of discussing test-related issues, turns to design of analog ANNs with no obvious relationship to testing and spends two pages with this topic (which might be important but out of scope).

Authors' Response: Since reference [116, or 112 in the old version] is a paper primarily based upon analog ANN, we thought necessary to cover detailed outlining of this topic containing, RF testing issues, design of ANN, training of ANN, how ANN is finally applied to RF BIST followed by detailed future directions.

Reviewer: - II.B describes a design flow that does not have an apparent connection with testing. And even the connection with ML is only mentioned in the end, along the lines of saying "it uses ML and it brings the following benefits", with no explanation or intuition what ML is used for. I recommend to completely remove II.B, or explain its connection to testing.

Authors: We agree, and we have removed Section II.B of the original paper.

Reviewer: - V "Hardware security" is, in my opinion, unfortunate. It does not cover a comprehensive list of topics on ML in hardware security: 2-5 out of its paragraphs discuss counterfeiting detection. Alone the topic of ML for power side-channel attacks would easily fill a survey of similar length to the submission! In the remaining half paragraph, side-channel attacks are mentioned, but they seem to relate to detection of hardware Trojans by side-channel measurements. "Detecting profiling and non-profiling-based side channel analysis" is confusing and misleading at least, as I don't think the mentioned papers are detecting side-channel attacks. I would either remove V completely or rename it into counterfeiting detection and argue that this is a task related to testing.

Authors: Reviewer's remarks are relevant. However, we retain Section V, which now begins with two new paragraphs narrowing the focus down to two areas within the broader field of hardware security.

Reviewer: Structure - The sections are extremely unevenly distributed. This is best seen in Section III, where most of no less than 11 subsections are 1 or 2 paragraphs long and do not go far beyond citing the

discussed papers, whereas the last section III.K is over 2 pages long and includes a comprehensive introduction into D-algorithm and PODEM. When reading the 11 subsections, III.G, III.H and III.K essentially consider the same question: how to compute testability metrics using ML. And III.C and III.E both discuss diagnosis.

Authors: We give a justification for Section III.I (old III.K) being so long in the "Introduction" section. The reason being Section III.I outlines authors' recent research, which has not been widely available until recently. The idea is to give highlights in the best possible manner without disturbing the goals of other subsections. Sections III.F, III.G, and III.I (old III.G, III.H and III.I) now are distinctly different; F discusses ML usage in computation of testability measures, and G and I, applications of ML and testability measures to problems of test point insertion and automatic test pattern generation, respectively.

Reviewer: - Several subsections start with introduction-style sentences, which are either repetitive (saying over and over that due to scaling we need to test more logic) or suddenly shifting the focus. E.g. IV.B without any prior warning argues over automotive ICs - are the techniques exclusive for this application?

Authors: We agree with the remark and have changed the start of sections without repetitive lines. The beginning sentence of Section IV.B now says that the automotive industry is not an exclusive domain and flash memories may have other applications.

Reviewer: - Detailed comments:

Introduction - It is not true that the survey "covers most of the details"; most descriptions are superficial and cannot be understood without reading some of the cited papers. This is not a drawback per se, as it can be a valid task of a survey to make readers aware of the problem and point them towards detailed descriptions, but such claim then should not be made.

- The authors make a point about being biased. If they see they are biased, they should please de-bias their text until at least themselves do not see the bias.

Authors: We have removed explicit statements showing bias. This survey is written as an update of previously published surveys. It was motivated by some very recent research on MI applications to test pattern generation that included our work.

Reviewer: Section II - Comma after "elusive goal".

Authors: Fixed. See the paragraph just before Section II.A.

Reviewer: Section II.A - It is not explained what a "valid/invalid codeword" means in the context of analog/RF testing. Are the authors considering digital signals and propose to use some redundancy? Does "invalid codeword" mean a "non-codeword"?

Authors: Parts of Section II.A have been rewritten to eliminate this concern.

Reviewer: - The statement "As a result of a training phase, suitable topology is formed" is wrong. The topology is the result of architecture search (even though in many cases a previously known topology is simply selected), and training only sets the weights and biases.

Authors: Addressed through rewording of the first paragraph in Section II.A.

Reviewer - What is "pattern of measurements"? Are these patterns applied to a device (inputs) or measured on that device (outputs)?

Authors: Also addressed through rewording of the first paragraph in Section II.A.

Reviewer: - Apart from analog ANN design discussion being questionable in such (see above), "analog ANN design must consider…" includes criteria that apply to any ANN, not analog ones. I would expect here a discussion what is specific to analog ANN.

Authors: Addressed through rewording of the second paragraph in Section II.A.

Reviewer: - "This method is likely to get trapped in local minima, thus several rounds of training may be needed" - this makes no sense. If the method is trapped in a local minimum, running it several times will lead it to being trapped in (possibly different) local minima each time. If the problem is non-convex (as is the case in ML), there must be some technique to get from one local minimum to the other.

Authors: Addressed through rewording of the last few lines of fourth paragraph in Section II.A.

Reviewer: - The only loosely test-related context in all of the ANN discussion that I see is the mentioning of "effects of DUT degradation". And here it is unclear whether DUT is the ANN itself or the analog ANN is a separate block somehow in charge of testing a DUT that is a different block.

Authors: Addressed through rewording of the sixth point of the ANN discussion in the 6-point list at the end Section II.A.

Reviewer: - Section II.C (now Section II.B): It is not defined what "estimated closer to the specification" means. Is such an "extreme instance" a good or a bad thing to have?

- Under 3), "The ANN is re-trained … so that the class of the "extreme" circuits is closer to the specification" - what does that mean? Do the circuits become better because of training some ANN?

"showed reduce simulation run-time" -> "reduced".

Authors: Addressed by changes and corrections throughout Section II.B (previously II. C), and eliminating the last paragraph.

Reviewer: - Section III.A: - Fig. 6, different from claims in text, does not show any radial basis functions or anything else related to SVMs.

Authors: Addressed through rewording of the text in Section III.A.

Reviewer: - Section III.B: - "Affine group" is mentioned several times without being defined or explained. If it is so important that scan cells form a group and that this group is affine, more details should be included.

Authors: The second paragraph of Section III.B is now split into two paragraphs. Relevant details of Affine group are now given in the last paragraph of the section.

Reviewer: - Section III.C: - I fail to see what diagnosis methods discussed here have to do with the section title "classifying fault models"

Authors: Indeed, the title of Section III.C, Classifying Fault Models," was inappropriate. Therefore, the contents of old Section III.C are merged into the present Section III.D. The are other changes too. The present organization of Section III is as follows:

Section III.A Wafer Testing	**Old Section III.A**
Section III.B Scan Chain Diagnosis	**Old Section III.B**
Section III.C Printed-Circuit Board (PCB) Testing	**Old Section III.D**
Section III.D Fault Diagnosis	**Old Sections III.C and III.E**
Section III.E Test Compression	**Old Section III.F**
Section III.F Testability Analysis	**Old Section III.G**
Section III.G Built-In Self-Test (BIST) and Test Point Insertion (TPI)	**Old Sections III.H and III.J**
Section III.H Power Supply Noise (PSN) and Signal Integrity	**Old Section III.I**
Section III.I Machine Intelligence Applied to ATPG	**Old Section III.K**

Reviewer: - Section III.D (now Section III.C): - The way how it is written up, I do not understand why (weighted) majority voting needs ANNs or SVMs. It is a simple addition followed by a comparison; what role do ANNs play here?

Authors: Addressed through rewording of the second last paragraph in Section III.C.

Reviewer: - Section III.F (now Section III.E): - I did not understand what the role of an PRPG in test compression is supposed to be and length of what is being optimized. If it is the TPG/decompression block, then it was called CODEC just 4 lines above; if it is something else, then what is it?

Authors: Addressed through rewording of the first paragraph and adding necessary details in Section III.E.

Reviewer: - Section III.G (now Section III.F): - Restricting "testability analysis" to "linear-complexity procedures" not only is quite unusual but also effectively excludes all ML methods, which would not have linear complexity in anything.

Authors: In Section III.F. The first sentence is modified as, "Testability analysis generally refers to linear, or at most polynomial but not exponential, complexity procedures that can identify test bottlenecks in a circuit [4]."

Reviewer: - Section III.K (now Section III.I): - "search space consists of 2^#PI … Thus ATPG is a search algorithm whose complexity increase exponentially" - this is a wrong implication. E.g., sorting algorithms have an exponential search space of n! possibilities but run in O(n log n).

Authors: Addressed through rewording of the first paragraph in Section III.I (previously Section III.K).

Reviewer: - Mentioning quantum computing is unexpected; these methods have nothing to do with ML.

- "whith backtrace".

Authors: Addressed through rewording of the sixth paragraph, "Application of quantum . . .," in Section III.I. Typo fixed in the paragraph starting as, "Statistical analysis . . ."

Reviewer: - "We find that…"; "The results showed the effectiveness…" - is this published data or a new result? In the latter case, showing the data would be useful.

Authors: It is published data. Readers are strongly encouraged to read the cited papers.

Section IV.A:

Reviewer: - Over 1/2 of this section discusses reference [118] (reference [122] in the revised version) from 1993. How relevant is this line of research? Has there been any follow-up by anybody?

Authors: This survey is not about criticizing or evaluating the relevancy of the line of already published research. The goal of this survey is to consolidate the ML-related work in testing. In this, subsection, it's ML-based memory testing.

Reviewer: - The discussion of "stuck-at firing or nonfiring" neurons is confusing. Are these spiking NNs? Some discussion, including some actual numbers how many neurons out of how many in total were defective, would be useful. In the recent years, there is a large body of literature on behavior of ANNs under errors; it does not help the readers to point them to that paper from 1993.

Authors: Addressed through rewording of the last paragraph in the reorganized Section IV.A.

Reviewer: - Section IV.B: - *The section continuously talks about "false fails". What are they? Coverage holes? Overtesting issues? Why do they show up?*

Authors: Also addressed through rewording of the second paragraph in Section IV.B.

Reviewer: - Section IV.C: - These 6 lines essentially cite one paper from year 2009, which seems to be on statistics rather than on testing. Either more explanations are needed what this work does and what it has to do with ML, or it should be removed.

Authors: Addressed through rewording of Section IV.C and improvised Monte Carlo based yield improvement method for SRAM ICs.

Reviewer: - Section V:

- I don't think the claim "ML algorithms have not been directly applied to reverse engineering" is true. RE involves image processing, which for sure will rely on ML.

- "instruction-based assembly" is unclear; if the authors mean "assembly language" then it is "instruction-based" by definition.

Authors: Each comment is addressed through rewording and clarifications in Section V

Reviewer - Conclusions:

- This is a problematic part of the submission. Bullet 1 has nothing to do with the content of the paper and considers an important but different issue: use of ANNs in safety-critical systems (such as pedestrian recognition) - what does it have to do with IOC test? Bullet 2 focuses on security, which is again not the main topic of the paper, and it is undefined what is meant "attack either good or bad ICs". Bullet 3 refers to variations in emerging technologies and argues about "supplying products based these technologies using ML". What do variations on emerging device level have to do with ML, and where in the paper did we read how testing helps (or could help) here?

Authors: We do not completely agree with reviewer's argument against bullet 1, and as it is an opinion, we believe readers will go either way. We have addressed comments on other bullets through rewording. Also, the section is enlarged to include suggestions for future work.

Reviewer: - The remainder of conclusions raises two new issues that have not been described above. First, repair/reconfiguration, where I don't see the argument why ML-based testing requires repair or reconfiguration. I could see that repair is needed for memories, but I do not think it is widely used before, and I also do not see how the fact that testing is ML-based influences the need for it. Then, fast detection of anomalies is mentioned - for the first time in the paper. Finally, another completely new topic is mentioned: lithographic hotspot detection. If the authors see a connection to testing, they should add a subsection in Section III (or IV) and describe it there, rather than bring it as a last sentence of the paper.

Authors: These are mentioned as a part of future in the revised section.

Reviewer - Overall, I think that the story of this survey must be re-thought completely, the paper's sections need a different structure, irrelevant parts must be removed, technical mistakes must be corrected, and lengthy discussions of papers that appeared 30 years ago should be reconsidered. There should be also some uniform way of introducing a problem to the reader: either consistently superficial or in detail, but not sometimes this and sometimes that way. In its present form, I would not recommend this paper to a student who wishes to get acclimatized in this research area.

Authors: We have given a new shape to this paper by removing irrelevant parts, rectifying technical mistakes, shortening the published line-of-research and keeping coherent parts only.

Reviewer #3:

Reviewer: - This is a survey covers a large area on testing using machine learning techniques.

It is interesting that the survey covers not only digital circuit testing but also analog circuit testing or hardware security.

Most part are well written and easily understandable.

However, some of parts are difficult to read.

The reviewer is concerned about the imbalance as a whole.

Some of parts should be totally written.

The following parts are difficult to understand.

Section II-A

This part is deeply focused on [112] ([116] in the revised version)

especially the implementation of analog ANN circuit.

The effectiveness of [112] as BIST should be mentioned.

Authors: Addressed through rewording of Section II.A.

Reviewer: - In addition, though [112] is a work in 2010 in which several future research directions were reported as mentioned, there is no description on related works following [112]. Reference [112] is now [116].

Authors: No action was necessary because 2-3 lines description is already provided in our survey for each future work. More descriptions are available in the cited papers.

Reviewer: - The reviewer wonders why [112] is so highlighted. Reference [112] is now [116].

At the knowledge of the reviewer, there are a lot of works on analog ANN using NVM, though it is not sure that they are used as a part of BIST.

If [112] has some special features to be used as a part of BIST, it can be a reason why it is highlighted.

Authors: A strong reason to use that reference ([116] in the revised version) is that the cited paper is amongst the initial group of papers that mentioned analog neural network design for RF BIST.

Reviewer: - Section III-B - 1st paragraph: There is no description of the method proposed in [78] whose targets are intermittent faults. Reference [78] is now [79].

Authors: Addressed by mentioning that the proposed method (unsupervised learning-based method), initially proposed for permanent faults, has positive impact in diagnosing intermittent faults too (mentioned in Section IV of paper [78], or [79] in the revised version). We gave the initial crux of the proposed method, further details can be obtained from the paper itself.

Reviewer: - 2nd paragraph: The description is very confusing. From line 4, it says "The ANN has the following input features: fault type, faulty cell's identification number, and the probability of a test pattern activating the fault." However, from line 8, it says "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."

Actually, [31], now [32], proposes multi-stage ANNs consisting of CGNN and RLNN. However, there is no such explanation in the survey.

Authors: Addressed by adding/editing few lines in the second paragraph and also removing the confusing elements in the paragraph. We have touched base with the core concept and application of the proposed methodology in our survey. However, to study more, one needs to visit the referenced paper [32] itself to further deep dive into the proposed methodology.

Reviewer - Minor comments:

page 5, left, line 52: response binary vector -> binary response vector page 7, left, line 44: To improve ..., designers inserts ... to detect .... One sentence has "to ..." twice.

Authors: Addressed through fixing of the typos in Section III.B

Reviewer - page 9, left. 43 - 46:

controllability and observability program (COP)

Sandia Controllability/Observability Analysis Program (SCOAP)

COP and SCOAP already appeared in page 8.

SCOAP [50] and SCOAP [49] are mixed.

Authors: The confusion of duplicated references [49] and [50] has been fixed. The SCOAP reference now appears everywhere as [51]. We appreciate the reviewer for bringing this to our notice.

A Survey and Recent Advances: Machine Intelligence in Electronic Testing

Soham Roy
Intel Corp., Santa Clara, CA 95054
soham.roy@intel.com

Spencer K. Millican
Dynetics Inc., Huntsville, AL 35806
spencer.k.millican@dynetics.com

Vishwani D. Agrawal
Auburn Univ., Auburn, AL 36849
agrawvd@auburn.edu

***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).**

***Index Terms*—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).**

I. 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 [93], digital tests are structural and target modeled faults [21], 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 [137], [173] have discussed ML applications to testing. Our recent article at the VLSI Test Symposium (VTS'21) [147] 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 III-I. These are,

- Establish the feasibility of training artificial neural network (ANN) to guide an ATPG algorithm [145].
- Optimize the training of ANN for ATPG [148].
- Use principal component analysis (PCA) to combine multiple heuristics in ATPG [149].
- Impact of ML guidance on the performance of ATPG [146].
- Use PCA to combine multiple heuristics for backtracing and D-drive [50] in a practical ATPG system (i.e., random patterns followed by algorithmic vectors) [150].

Section III-I is derived from authors' recent research in which they try to follow the elusive goal of zero backtracks in ATPG [143]. 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 II discusses ML applications in testing of analog and RF circuits. Section III explores new ML techniques for digital circuits, which is an additional contribution beyond the previous surveys. Memory testing is the subject of Section IV. Moving beyond classical testing, Section V outlines recent applications of ML to hardware security, IC counterfeiting, and issues related to emerging technologies. Section VI concludes the survey by listing some open test-related challenges yet to be addressed by ML.

II. 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], [165]; built-in test (BIT) or the use of an on-chip tester [59], [153] that switches the device under

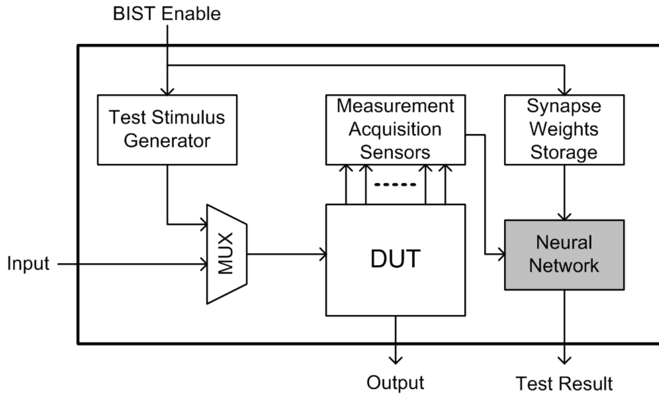

Fig. 1. Built-in self-test (BIST) of a radio frequency (RF) device under test (DUT) [116].

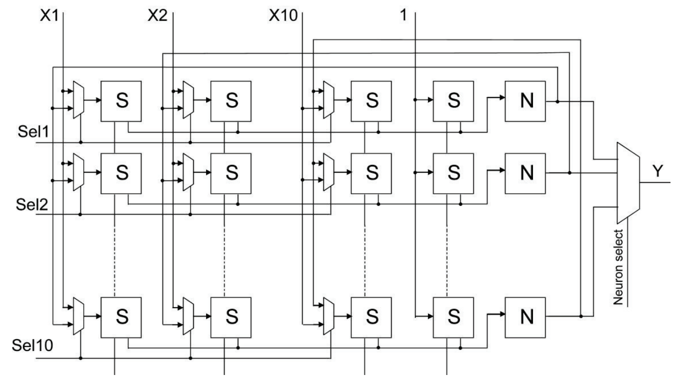

Fig. 2. Reconfigurable ANN [116].

test (DUT) into test mode by fetching signals from sensors [1], [33], [53], [81], [121], [190]; built-off test (BOT) or converting RF signals to DC signals using an interface (placed on a load board) between the DUT and tester [16], [43]; and implicit test, i.e., statistical model-based test that can make an off-line PASS/FAIL decision [5], [176], [192].

Complete automation in this area has been an elusive goal, and that is where machine learning has begun to play a role [40], [174].

A. 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 devise under test (DUT), such as a low noise amplifier (LNA), for illustration. A proposed architecture [116] 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 Figure 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 [177].

For an effective implementation of the BIST circuit shown in Figure 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 [73], [113], [123], [127].

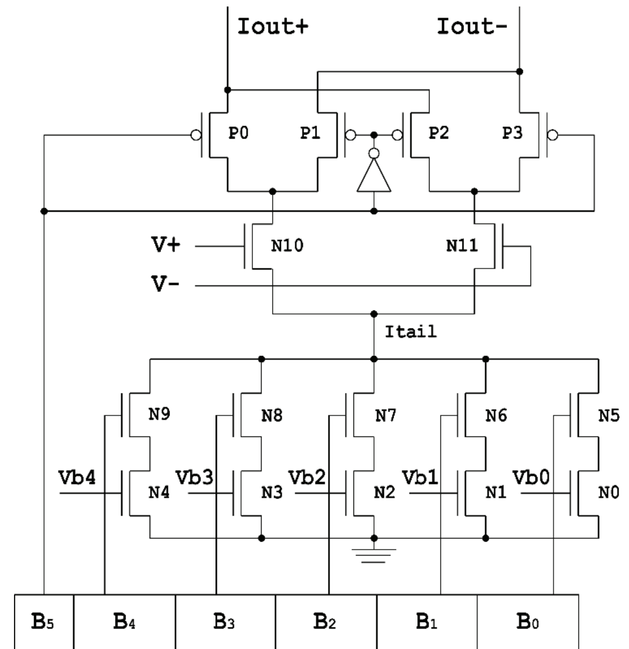

Fig. 3. Schematic diagram of synapse [116].

Figure 2 illustrates an architecture of a reconfigurable, single hidden layer ANN [116]. 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 Figure 3, illustrates multiplication implemented through a digital-to-analog converter (DAC) [97], a combination of differential input voltages, and programmable tail currents. The upper half of Figure 3 is a differential pair "N10-N11" performing multiplication while switching transistors "P0-P3" controlled by bit "B5", steer the current

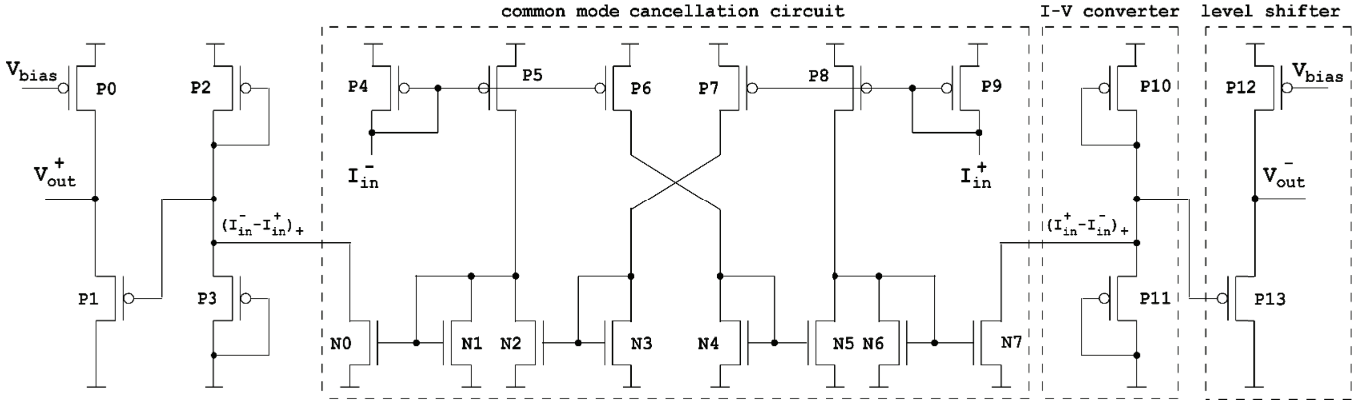

Fig. 4. Schematic diagram of neuron [116].

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 [113]. Multiplexers select input sources from previous layers, and the summation of synapses is fed into a neural circuit, as illustrated in Figure 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 [85] 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.”

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 [116]. 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 [72].
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 [108].
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 [60], [63]. 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.

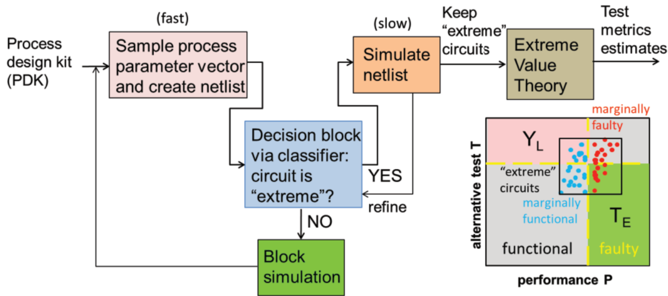

Fig. 5. Simulation flow for parametric test metrics estimation [173].

B. 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 [173], [175] that includes ML algorithms is shown in Figure 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) [173] 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 [168].
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 [13], [178]. This method speeds up the Monte Carlo transistor-level simulation. Typically, fault models account for process parameters based on a joint distribution as given in their respective PDK [178]. Finally, the fault model is verifiable after performing transistor-level simulation. The algorithm [175] 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) [178] 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 [13], but is yet to be explored for other analog ICs whose simulation run-time is high, such as phase-locked-loops (PLLs).

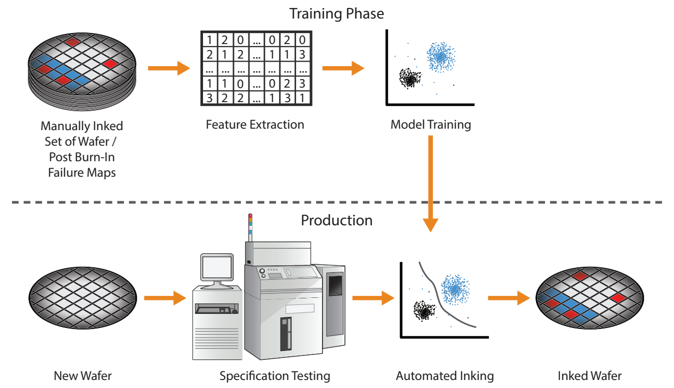

Fig. 6. ML-based die inking process [198].

III. Digital Testing

In a modern electronic system, digital parts cover most area. Similarly, digital testing occupies most pages in a book on electronic test [21]. 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.

A. Wafer Testing

In general, logic defects occur on wafers in physical clusters [131]. Thus, clustering algorithms [179] 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 [198] proposes a similar cluster-detecting ML algorithm using support vector machine (SVM) [66], [156]. 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 Figure 6.

B. 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 [80] points to Bayesian learning [189] for identifying 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 scan chain [79]. 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 [32] 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 [32] 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.

C. Printed-Circuit Board (PCB) Testing

Testing each component on a board is vital from the real-time testing perspective. Even when an in-circuit test [14] 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 [130] 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 [130] kept the fixing of their approach as an open problem for the future.

Another ML application [206] 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 [203], [207] 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) [109]. 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 [183], [202]. 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 [184] to reduce feature vector size. However, some syndromes that are not easily computable do not allow merging. Another technique [90] can still diagnose a system with a non-computable or missing syndrome using label-imputation and the so-called two-feature-selection methods.

D. 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 [193]. The idea has been demonstrated by using different types of ML approaches, namely, k-nearest neighbor (kNN) [101], support-vector machine (SVM) [66], [156], and decision trees [70]. Both, unsupervised and supervised learning methods can cooperate in identifying design bugs [126]. A survey [78] 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) [205]. 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 [52], [57], and 2) tuning the set of candidate faults to further improve the diagnostic resolution [200]. 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 [82] 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 [38]. Later work [195] 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 [187] and Bayesian network approach [31], can evaluate the failure feature probability.

An ML-based volume diagnosis technique [195] 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 [195] 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.

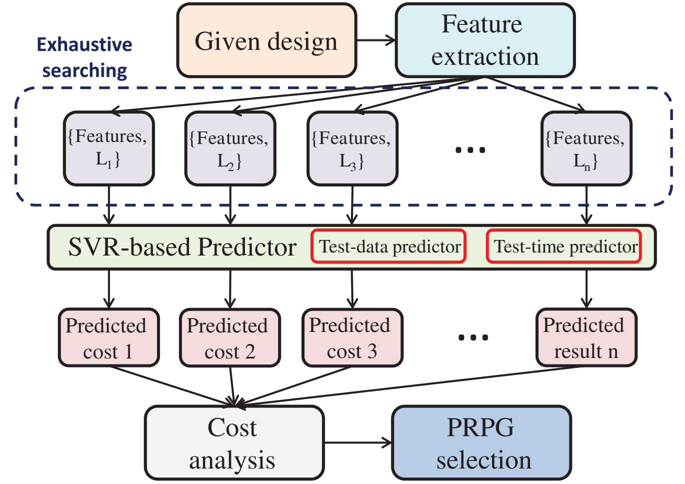

Fig. 7. Pseudo-random pattern generation (PRPG) methodology [107].

An ML-based method that assists PFA [163] 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).

E. 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) [21]. However, the length of a PRPG does impact the test time irrespective of various circuit parameters [107]. 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 Figure 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 [107] give a correlation-based feature selection method applied to industrial designs for reducing the test time with high prediction accuracy [138].

F. 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 [50]. 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 [172], Sandia Controllability/Observability Analysis Program (SCOAP) [51], CAMELOT [15], and controllability and observability program (COP) [19]. 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.

1) Combining Testability Measures: Several of the testability measures listed above have been combined into a composite measure using unsupervised learning [144]. For every signal node, four testability measures have been defined, 0-controllability, 1-controllability, 0-observability, and 1-observability [86]. The last two measures are often replaced by a single measure, observability, leading to three measures per node. The combination process has following steps:

- For testability measures, e.g., distances, SCOAP [51], 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 [19] 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) [76]. 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.

The PCA combined testability measure has been used to guide the ATPG with notable performance improvement as shown in Figures 8 and 9, and discussed in Section III-I. 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.

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 [138] 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) Signal Probability: Savir [155] 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 [141], [161] can detect re-convergent fanouts, but they can be computationally burdensome. Toward application of ML, recent work [83] has used ANN to predict signal probabilities from minimal fanout information, resulting in increased accuracy with reasonably small computation time.

G. 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) [21]. However, an LFSR may not generate specific patterns to detect random pattern resistant (RPR) faults. As an ML solution to this problem, ANN has been used [42] to generate test patterns to detect RPR faults as well as easy-to-detect faults.

The self-learning capability, suitable for an system-on-chip (SoC), also deals with aging-induced degradation. This proposed flow [42] 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. 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) [64] 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 [124], [182].

A deep learning technique to solve the TPI problem of logic circuits has been proposed [114]. 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 [51]. Further work [151], [152], [180] 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 [125], 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 [181] 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.

H. 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 [88]. 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) [164], [188]. Unrestrained PSN can lead to performance glitches and impact timing [28], [89]. Also, excessive PSN during test can cause false failure if a test pattern induces PSN that substantially exceeds the functional mode behavior [49], [106], [194], [196]. 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) [201] 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 [110] includes feature extraction methods, such as ANN [36], SVM [18], [201], and least-square boosting (LSBoost) [18]. Here, ANNs are found to be the best predictors of circuit timing for test patterns.

A recent paper [129] 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 [66], [156], k-nearest neighbors [101], and random decision forests [70]. The results show that the learning scheme based on random decision forest classifies the embedded faulty cells with higher accuracy.

I. 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 [142] 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 re-convergent fanouts. The path oriented decision making (PODEM) [50] 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 [46], TOPS [94], SOCRATES [158]–[160], EST [22], [29], [30], recursive learning [99], TRAN [26], GRASP [119], NEMESIS [103], TEGUS [171], and Boolean satisfiability (SAT) [23], [25], [102], [103], have been reported. Although the search space size remains $2^{\#PI}$, researchers [68], [185] 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 [23], [25]. 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 [24], [26], 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 [166], [167]. 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 [98].

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 [132], [133] 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 [132], [133].

Recent work [143] uses ANN and principal component analysis (PCA) 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 [50] 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 [149], [150], or by a trained ANN [145], [146], [148]. The former is referred to as unsupervised learning, while the latter is called supervised learning.

In the unsupervised learning model training data has no specified correct output value (referred to as labels). 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 [61], partitioning around medoids (PAMs) [92], ordering points to identify the clustering structure (OPTICS) [7], principal component analysis (PCA) [76], [134], minimum redundancy maximum relevance (mRMR) [135], and self-organizing maps (SOMs) [96]. These methods are typically used to segment text topics, classify items, and identify data outliers.

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 [19] and SCOAP [51] values into a set of principal components (PCs). Then the largest (major) PC would guide the PODEM ATPG backtraces [149], 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" [146]. 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 [162] 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 [143], [145], [146], [148], [149], or distance (logic depth) heuristic [50], or controllability and observability program (COP) [19], or SCOAP [51]. We find that MI-guided ATPG shows significantly improved performance over others.

Unsupervised learning or PCA was applied only in the backtrace step [149] in the early work, while the D-drive used the conventional distance (logic depth) heuristic [50]. The ATPG system we will examine now [150] 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 Figures 8 and 9 circuits are arranged left to right in the order of increasing number of nodes. Figure 8 shows combined backtracks and Figure 9 gives total CPU time for all stuck-at faults left over from the random phase. Number of backtracks (four bars in Figure 8) and CPU milliseconds (ms) (four data points in Figure 9) for each circuit correspond to the four versions of PODEM. These are PODEM programs where backtraces and D-drives are directed, respectively, by

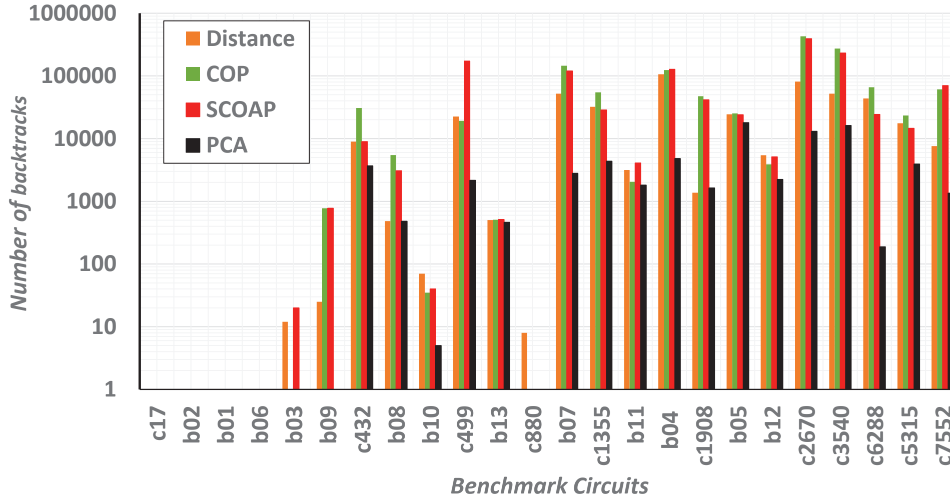

Fig. 8. Total backtracks used while finding a test or proving redundancy for the checkpoint faults left after the random ATPG phase applied to ISCAS'85 [20] and ITC'99 [34] benchmark circuits [150].

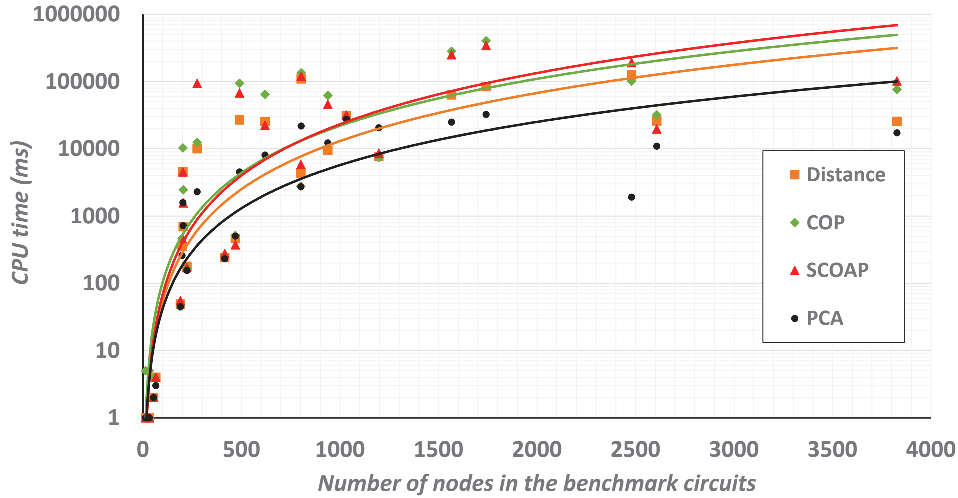

Fig. 9. Total CPU times on Intel i7-8700 based workstation with 8-GB RAM to find a test or prove redundancy for the checkpoint faults left after random ATPG phase applied to ISCAS'85 [20] and ITC'99 [34] benchmark circuits. Four types of data points and trend curves are for PODEM ATPG programs guided, 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) [150].

distance (logic depth), COP, SCOAP, and the major principal component from PCA combining distance, COP and SCOAP measures. In Figure 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.

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 Figures 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 Figure 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

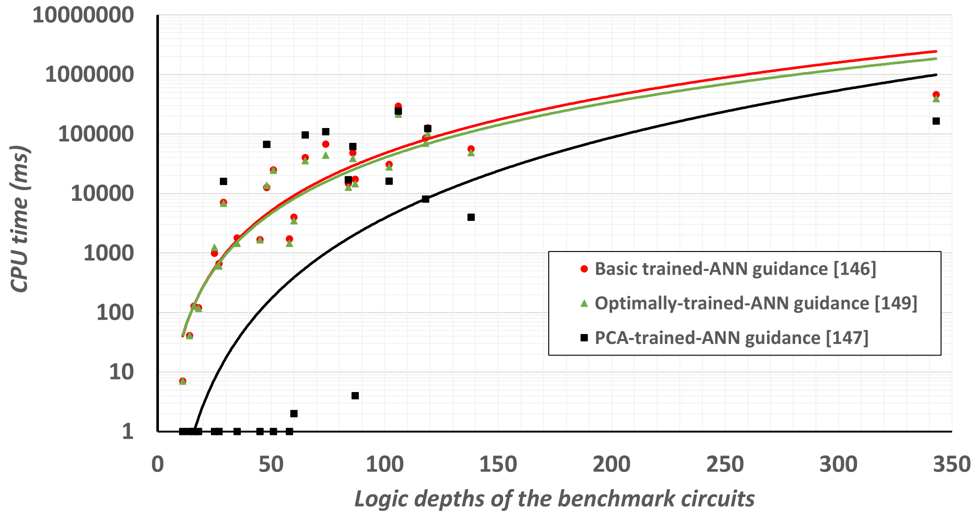

Fig. 10. Total CPU times on Intel 8700 processor based workstation with 8-GB RAM to find a test or prove redundancy for *all* checkpoint faults in ISCAS'85 [20] and ITC'99 [34] benchmark circuits. Three types of data points and trend curves are for PODEM ATPG guided by, respectively, basic-trained ANN (round points, top curve), optimally-trained ANN (triangular points, middle curve), and PCA-trained ANN (square points, bottom curve) [146].

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) [66], [156], one-class SVM [66] or one-class neural network [169], decision trees (DT) [139], random forest (RF) [71], linear regression (LR) [128], multivariate adaptive regression splines (MARS) [204], logistic regression [35], adaboosts [44], ANNs [65], convolutional neural network (CNNs) [65], autoencoders [9], recurrent neural network (RNNs) [65], long short-term memories (LSTMs) [48], and half-space trees (HS-Trees) [186].

A recent study examined MI's supervised learning ability to enhance ATPG by reducing backtracks [145], 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 [148], 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 [146] 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 Figure 9 [150], 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 Figure 9 and logic depth in Figure 10. Noticeable difference is seen, however, for several small and medium size circuits in Figure 10 where the ATPG CPU time with the PCA trained ANN guidance is negligibly small.

IV. Memory Test and Repair

A. 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 [45]. 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 [170]. 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 [41] and exhaustive [37], became ineffective. Since memory repair problem is NP-complete [58], heuristic algorithms were introduced. These included branch and bound [100], approximation [100], best-first search [62], and

others [112], [197]. 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 [122] 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 [122] 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 [122].

ANNs have been used to tackle optimization problems, e.g., the famous *traveling salesman problem* [47], for which a solution was proposed by Hopfield [74]. 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 of ANN formulation has low-quality, and therefore another proposed algorithm [122] 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 [122] 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 [122] 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.

B. 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 [118]. 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) [118], 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 [118]. 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 [56], 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 [118] 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 [56], 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 [118] 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 [118] 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.

C. SRAM Yield Improvement using Statistical Blockade

As transistor size shrinks, the statistical blockade technique [168], discussed in Section II-B, 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.

V. Hardware Security

Over the years, hardware security has evolved as a serious topic of study [17], [191]. Many system-on-chip (SoC) design companies today outsource their production across the world to semiconductor foundries. This creates the threat of hardware Trojan (HT) or malicious modification to a design to modify its functionality such that an adversary gains control of the system, interrupts the normal operation, or steals information. Another form of risk known as counterfeiting of integrated circuits involves overproduction and marketing by unauthorized individuals without the knowledge of the customer [54], [55].

A full-scale discussion of hardware security will be too long, requiring a digression from the testing focus of this paper. We will, therefore, restrict to introductory remarks only.

Recently, machine learning (ML) has been used for hardware protection against malicious attacks. Hardware defense techniques use ML algorithms to detect hardware Trojans and IC counterfeiting. Countermeasures with and without a golden chip are two broad categories of defense against hardware Trojans. Methods extract training data from golden chips to classify on-chip sensor data [67], [69], [75], [77], gate-level design nodes [84], and traffic congestion sites on-chip [87], [91], [95].

IC counterfeiting is an alarming threat in the IC manufacturing industry. Manual inspection and detection of counterfeit ICs is accurate but time-consuming. An ML approach [8] inspects ICs using ANN-based image classification. Support vector machine (SVM) analysis [77] of on-chip sensor data has been used to identify recycled ICs. Consider an FPGA containing several ring oscillators (ROs), whose frequencies may degrade due to aging [39]; supervised learning in the form of SVM [77] and unsupervised learning such as k-means clustering [61] have been used to examine frequencies of the circuit to detect recycled ICs [6].

ML algorithms are used to identify hardware Trojans [11], [12]. ML algorithms can also detect profiling and non-profiling-based side-channel attacks, typically in cryptographic secret extraction [104], [105], [111], [115], [136], [140], [154]. Further, ML-algorithms can provide profiling-based side-channel analysis for assembly language [117], [120], [157].

VI. 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 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 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 II 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 can be automated by ML. Subsection III-D and several other subsections on digital circuit diagnosis and Section IV 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 have a complexity that grows 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 Subsection III-D point to some very effective applications of ML in the supervised and unsupervised learning modes.

For the future, there is ample research scope in new areas like intelligent lithographic hotspot detection [199], 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 [10], monolithic 3D (M3D) devices (specifically resistive RAMs (ReRAMs)) [27] and many more.

Data Availability Statement

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.

References

[1] L. Abdallah, H. Stratigopoulos, C. Kelma, and S. Mir, "Sensors for Built-In Alternate RF Test," in *Proc. 15th IEEE European Test Symposium (ETS)*, 2010, pp. 49–54.

[2] 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.

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

[4] V. D. Agrawal and M. R. Mercer, "Testability Measures – What Do They Tell Us?," in *Proc. International Test Conf.*, (Philadelphia, PA), Nov. 1982, pp. 391–396.

[5] 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.

[6] 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.

[7] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "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, 1999, p. 49–60.

[8] 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.

[9] P. Baldi, "Autoencoders, Unsupervised Learning, and Deep Architectures," in I. Guyon, G. Dror, V. Lemaire, G. Taylor, and D. Silver, editors, *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, volume 27 of *Proceedings of Machine Learning Research*, (Bellevue, Washington, USA), PMLR, July 2012, pp. 37–49.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

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

[15] R. G. Bennetts, C. M. Maunder, and G. D. Robinson, "CAMELOT: A Computer-Aided Measure for Logic Testability," *IEE Proceedings E - Computers and Digital Techniques*, vol. 128, no. 5, pp. 177–189, 1981.

[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. Bhunia and M. Tehranipoor, *Hardware Security: A Hands-on Learning Approach, 1st Edition*. Morgan Kaufmann, 2018.

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

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

[20] F. Brglez and H. Fujiwara, "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, June 1985.

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

[22] 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.

[23] S. T. Chakradhar, *Neural Network Models and Optimization Methods for Digital Testing*. PhD thesis, Rutgers University, USA, 1991.

[24] 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*, DAC '91, 1991, pp. 353–358.

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

[26] 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, July 1993.

[27] 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.

[28] 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*, 1997, p. 638–643.

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

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

[31] W. Cheng, Yue Tian, and S. M. Reddy, "Volume Diagnosis Data Mining," in *Proc. 22nd IEEE European Test Symposium (ETS)*, 2017, pp. 1–10.

[32] 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.

[33] 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.

[34] F. Corno, M. S. Reorda, and G. Squillero, “RT-Level ITC’99 Benchmarks and First ATPG Results,” *IEEE Design & Test of Computers*, vol. 17, pp. 44–53, July 2000.

[35] D. R. Cox, “The Regression Analysis of Binary Sequences,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 20, no. 2, pp. 215–242, 1958.

[36] 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.

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

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

[39] 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.

[40] 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.

[41] R. C. Evans, “Testing Repairable RAMs and Mostly Good Memories,” in *Proceedings International Test Conference*, 1981, pp. 49–55.

[42] C. Fagot, P. Girard, and C. Landrault, “On Using Machine Learning for Logic BIST,” in *Proc. IEEE International Test Conference*, 1997, pp. 338–346.

[43] 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.

[44] Y. Freund and R. E. Schapire, “A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting,” *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

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

[46] H. Fujiwara and T. Shimono, “On the Acceleration of Test Generation Algorithms,” *IEEE Transactions on Computers*, vol. C-32, no. 12, pp. 1137–1144, Dec. 1983.

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

[48] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, “Learning Precise Timing with Lstm Recurrent Networks,” *J. Mach. Learn. Res.*, vol. 3, no. null, p. 115–143, Mar. 2003.

[49] P. Girard, N. Nicolici, and X. Wen, editors, *Power-Aware Testing and Test Strategies for Low Power Devices*. Springer, 2010.

[50] P. Goel, “An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits,” *IEEE Transactions on Computers*, vol. C-30, no. 3, pp. 215–222, Mar. 1981.

[51] L. Goldstein, “Controllability/Observability Analysis of Digital Circuits,” *IEEE Transactions on Circuits and Systems*, vol. CAS-26, no. 9, pp. 685–693, Sept. 1979.

[52] 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.

[53] 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.

[54] U. Guin, D. DiMase, and M. Tehranipoor, “A Comprehensive Framework for Counterfeit Defect Coverage Analysis and Detection Assessment,” *Journal of Electronic Testing: Theory and Applications*, vol. 30, no. 1, pp. 25–40, Feb. 2014.

[55] U. Guin, D. DiMase, and M. Tehranipoor, “Counterfeit Integrated Circuits: Detection, Avoidance, and the Challenges Ahead,” *Journal of Electronic Testing: Theory and Applications*, vol. 30, no. 1, pp. 9–23, Feb. 2014.

[56] 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.

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

[58] 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, Feb. 1991.

[59] 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.

[60] 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.

[61] 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.

[62] N. Hasan and C. L. Liu, “Minimum Fault Coverage in Reconfigurable Arrays,” in *Digest of Papers 18th International Symposium on Fault-Tolerant Computing*, 1988, pp. 348–353.

[63] 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.

[64] 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.

[65] S. S. Haykin, *Neural Networks and Learning Machines*. Upper Saddle River, NJ: Pearson Education, third edition, 2009.

[66] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Schölkopf, “Support Vector Machines,” volume 13, 1998, pp. 18–28.

[67] 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, July/August 1998.

[68] 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.

[69] A. Heuser and M. Zohner, “Intelligent Machine Homicide,” in *Proceedings of the Third International Conference on Constructive Side-Channel Analysis and Secure Design*, COSADE’12, (Berlin, Heidelberg), Springer-Verlag, 2012, p. 249–264.

[70] T. K. Ho, “Random Decision Forests,” in *Proc. International Conference on Document Analysis and Recognition (ICDAR)*, 1995, pp. 278–282.

[71] T. K. Ho, “Random Decision Forests,” in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, volume 1, 1995, pp. 278–282 vol.1.

[72] M. Hoehfeld and S. E. Fahlman, “Probabilistic Rounding in Neural Network Learning with Limited Precision,” *Neurocomputing*, vol. 4, no. 6, pp. 291–299, 1992.

[73] M. A. Holler, S. M. Tam, H. A. Castro, and R. Benson, “An Electrically Trainable Artificial Neural Network (ETANN) with 10240 ’Floating Gate’ Synapses,” in *Proc. International Joint*

Conference on Neural Networks, 1989, pp. 191–196.

[74] J. Hopfield and D. Tank, "Neural Computation of Decisions in Optimization Problems," *Biological Cybernetics*, vol. 52, pp. 141–152, 2004.

[75] 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, Dec. 2011.

[76] H. Hotelling, "Analysis of a Complex of Statistical Variables into Principal Components," *Journal of Educational Psychology*, vol. 24, no. 6, pp. 417–441, 1933.

[77] 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.

[78] 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.

[79] Y. Huang, B. Benware, R. Klingenberg, H. Tang, J. Dsouza, and W.-T. Cheng, "Scan Chain Diagnosis Based on Unsupervised Machine Learning," in *2017 IEEE 26th Asian Test Symposium (ATS)*, 2017, pp. 225–230.

[80] 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.

[81] 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.

[82] 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.

[83] 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.

[84] 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.

[85] 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.

[86] S. K. Jain and V. D. Agrawal, "Statistical Fault Analysis," *IEEE Design & Test of Computers*, vol. 2, pp. 38–44, Feb. 1985.

[87] 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*, HASP '15, (New York, NY, USA), Association for Computing Machinery, 2015.

[88] F. Jensen and N. E. Petersen, *Burn-In*. Chichester, UK: John Wiley & Sons, Inc., 1982.

[89] Y.-M. Jiang and K.-T. Cheng, "Analysis of Performance Impact Caused by Power Supply Noise in Deep Submicron Devices," in *Proceedings Design Automation Conference*, 1999, pp. 760–765.

[90] 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.

[91] 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.

[92] L. Kaufman and P. J. Rousseeuw, *Partitioning Around Medoids (Program PAM)*, pp. 68–125. John Wiley Sons, Inc., 2008.

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

[94] 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.

[95] 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*, IJCAI'95, (San Francisco, CA, USA), Morgan Kaufmann Publishers Inc., 1995, p. 1137–1143.

[96] T. Kohonen, "The Self-Organizing Map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[97] 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.

[98] B. Krishnamurthy, "Hierarchical Test Generation: Can AI Help?," in *Proc. International Test Conf.*, 1987.

[99] 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.

[100] 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.

[101] J. Laaksonen and E. Oja, "Classification with Learning K-Nearest Neighbors," in *Proc. International Conference on Neural Networks (ICNN)*, volume 3, 1996, pp. 1480–1483.

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

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

[104] J. Li, J.-H. 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.

[105] J. Li, L. Ni, J. 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.

[106] Y.-H. Li, W.-C. Lien, I.-C. Lin, and K.-J. 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, Jan. 2014.

[107] 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.

[108] 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.

[109] N. Littlestone and M. K. Warmuth, "The Weighted Majority Algorithm," in *Proc. 30th Annual Symposium on Foundations of Computer Science*, 1989, pp. 256–261.

[110] 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.

[111] Y. Liu, Y. Jin, A. Nosratinia, and Y. Makris, "Silicon Demon-

stration 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.

[112] F. Lombardi and W. K. Huang, "Approaches for the Repair of VLSI/WSI RRAMs by Row/Column Deletion," in *Digest of Papers, 18th International Symposium on Fault-Tolerant Computing*, 1988, pp. 342–347.

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

[114] 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.

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

[116] 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.

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

[118] 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.

[119] 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.

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

[121] 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.

[122] P. Mazumder and Y. S. 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, Jan. 1993.

[123] 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.

[124] S. Millican, Y. Sun, S. Roy, and V. Agrawal, "System and Method for Optimizing Fault Coverage Based on Optimized Test Point Insertion Determinations for Logical Circuits," U.S. Patent 17226950, Oct. 2021.

[125] 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.

[126] M. Moness, L. Gabor, A. I. Hussein, and H. M. Ali, "Automated Design Error Debugging of Digital VLSI Circuits," *Journal of Electronic Testing: Theory and Applications*, vol. 38, no. 4, p. 395–417, Oct.

[127] 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.

[128] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to Linear Regression Analysis*. Wiley Publishing Company, Incorporated, 2012.

[129] Z. P. Najafi-Haghi1 and H. Wunderlich1, "Identifying Resistive Open Defects in Embedded Cells under Variations," *J. Electronic Testing: Theory and Applications*, vol. 39, no. 1, pp. 27–40, Feb. 2023.

[130] 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. 1–7 (Paper 8.2).

[131] 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.

[132] J. Patel and S. Patel, "What Heuristics are Best for PODEM?," in *Proc. First International Workshop on VLSI Design*, 1985, pp. 1–20.

[133] S. Patel and J. Patel, "Effectiveness of Heuristics Measures for Automatic Test Pattern Generation," in *Proc. 23rd ACM/IEEE Design Automation Conference (DAC)*, 1986, pp. 547–552.

[134] K. Pearson, "On Lines and Planes of Closest Fit to Systems of Points in Space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[135] H. Peng, F. 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.

[136] H. Peng, F. 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.

[137] 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.

[138] 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, Dec. 2019.

[139] J. R. Quinlan, "Induction of Decision Trees," *Mach. Learn.*, vol. 1, no. 1, p. 81–106, Mar. 1986.

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

[141] 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.

[142] 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.

[143] S. Roy, *Toward Zero Backtracks in Test Pattern Search Algorithms with Machine Learning*. PhD thesis, Auburn University, USA, 2021.

[144] S. Roy and V. D. Agrawal, "An Amalgamated Testability Measure Derived from Machine Intelligence," in *Proceedings of 37th International Conference on VLSI Design & 23rd International Conference on Embedded Systems*, Jan. 2024.

[145] 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, pp. 1–5.
[146] S. Roy, S. K. Millican, and V. D. Agrawal, "Principal Component Analysis in Machine Intelligence-Based Test Generation," in *Proc. IEEE Microelectronics Design and Test Symp. (MDTS'21)*, (USA), May 2021, pp. 1–6.
[147] S. Roy, S. K. Millican, and V. D. Agrawal, "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.
[148] 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, pp. 316–321.
[149] 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, pp. 1–4.
[150] S. Roy, S. K. Millican, and V. D. Agrawal, "Multi-Heuristic Machine Intelligence Guidance in Automatic Test Pattern Generation," in *Proc. 31st Microelectronics Design and Test Symposium (MDTS)*, 2022, pp. 1–6.
[151] 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.
[152] 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. Testing: Theory and Applic.*, vol. 36, no. 1, p. 123–133, Feb. 2020.
[153] J.-Y. Ryu and B. C. Kim, "Low-Cost Testing of 5 GHz Low Noise Amplifiers Using New RF BIST Circuit," *J. Electronic Testing: Theory and Applications*, vol. 21, no. 6, pp. 571–581, Dec. 2005.
[154] 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.
[155] Savir, "Good Controllability and Observability Do Not Guarantee Good Testability," *IEEE Transactions on Computers*, vol. C-32, no. 12, Dec. 1983.
[156] B. Schölkopf and A. J. Smola, editors, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.
[157] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett, "New Support Vector Algorithms," *Neural Comput.*, vol. 12, no. 5, p. 1207–1245, May 2000.
[158] M. H. Schulz and E. Auth, "Advanced Automatic Test Pattern Generation and Redundancy Identification Techniques," in *Digest of Papers, 18th International Symposium on Fault-Tolerant Computing*, 1988, pp. 30–35.
[159] 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, no. 7, pp. 811–816, July 1989.
[160] 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, no. 1, pp. 126–137, Jan. 1988.
[161] S. C. Seth and V. D. Agrawal, "A New Model for Computation of Probabilistic Testability in Combinational Circuits," *INTEGRATION, The VLSI Journal*, vol. 7, pp. 49–75, 1989.
[162] S. C. Seth, V. D. Agrawal, and H. Farhat, "A Statistical Theory of Digital Circuit Testability," *IEEE Transactions on Computers*, vol. 39, no. 4, pp. 582–586, 1990.
[163] 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.
[164] 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.
[165] 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.
[166] A. Singh, L. M. Bharadwaj, and S. Harpreet, "DNA and Quantum Based Algorithms for VLSI Circuits Testing," *Natural Computing*, vol. 4, pp. 53–72, 2005.
[167] S. Singh and A. Singh, "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), Aug. 2003, pp. 648–651.
[168] 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, Aug. 2009.
[169] A. Skabar, "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, 2003, pp. 2127–2132 Vol.4.
[170] 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.
[171] 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.
[172] J. E. Stephenson and J. Grason, "A Testability Measure for Register Transfer Level Digital Circuits," in *Proc. 6th International Fault Tolerant Computing Symp.*, June 1976, pp. 101–107.
[173] H. Stratigopoulos, "Machine Learning Applications in IC Testing," in *Proc. IEEE 23rd European Test Symposium (ETS)*, 2018, pp. 1–10.
[174] 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.
[175] H. Stratigopoulos and S. Mir, "Adaptive Alternate Analog Test," *IEEE Design & Test of Computers*, vol. 29, no. 4, pp. 71–79, 2012.
[176] 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.
[177] 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.
[178] 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.
[179] 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.
[180] Y. Sun and S. K. Millican, "Test Point Insertion Using Artifi-

cial Neural Networks," in *Proc. IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2019, pp. 253–258.

[181] Y. Sun and S. K. Millican, "Applying Artificial Neural Networks to Logic Built-in Self-test: Improving Test Point Insertion," *Journal of Electronic Testing: Theory and Applications*, vol. 38, no. 4, pp. 339–352, Oct. 2022.

[182] 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.

[183] 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.

[184] 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.

[185] 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.

[186] S. C. Tan, K. M. Ting, and T. F. Liu, "Fast Anomaly Detection for Streaming Data," IJCAI'11, AAAI Press, 2011, p. 1511–1516.

[187] 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.

[188] 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.

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

[190] 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.

[191] J.-G. Valle, *Practical Hardware Pentesting: A Guide to Attacking Embedded Systems and Protecting Them Against the Most Common Hardware Attacks*. Packt Publishing, 2021.

[192] 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.

[193] 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.

[194] 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.

[195] S. Wang and W. Wei, "Machine Learning-Based Volume Diagnosis," in *Proc. Design, Automation & Test in Europe Conference & Exhibition*, 2009, pp. 902–905.

[196] X. Wen, Y. Yamashita, S. Kajihara, L.-T. Wang, K. K. Saluja, and K. Kinoshita, "On Low-Capture-Power Test Generation for Scan Testing," in *Proc. 23rd IEEE VLSI Test Symposium (VTS)*, 2005, pp. 265–270.

[197] C.-L. 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.

[198] 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.

[199] Y. Xiao, X. Huang, and K. Liu, "Model Transferability from ImageNet to Lithography Hotspot Detection," *J. Electronic Testing: Theory and Applications*, vol. 37, no. 1, pp. 141–149, Feb. 2021.

[200] 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.

[201] 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.

[202] 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.

[203] 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.

[204] W. Zhang and A. T. Goh, "Multivariate Adaptive Regression Splines and Neural Network Models for Prediction of Pile Drivability," *Geoscience Frontiers*, vol. 7, no. 1, pp. 45–52, 2016. Special Issue: Progress of Machine Learning in Geosciences.

[205] Y. Zhang and V. D. Agrawal, "A Diagnostic Test Generation System," in *Proc. IEEE International Test Conference (ITC)*, 2010, pp. 12.3.1–12.3.9.

[206] 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.

[207] 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.

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.