Universal Library of Engineering Technology Research Article

ISSN: 3064-996X

Open Access | PP: 53-61

DOI: https://doi.org/10.70315/uloap.ulete.2023.008

Universal Library Open Access Publications LLC

# Machine Learning for Defect Detection in Design-for-Testability Processes: Techniques and Trends

**Sri Harsha Panchali[1], Usha Mohani kavirayani[2], Krishna Bhardwaj Mylavarapu[3], Jenitha Pilli[4], Prathik Kumar Jannu[5], Javed Ali Mohammad[6]**

[1]Information Systems Engineer, CrowdStrike Inc.

[2]Kent State University, MS in Computer Science.

[3]MS in Computer Science, University of Illinois Springfield.

[4]MS in Computer Science, University of Louisiana at Lafayette.

[5]Computer Science Engineering, JNTU Hyderabad.

[6]Masters in Telecommunications, Middlesex University.

## Abstract

*In the current fast changing technology, accuracy, reliability and efficiency in defect detection have risen to be key to sustain the quality and performance of current semiconductor and electronic system. In this paper, the inadequacy of traditional methods of inspection in the ever-growing complexity, density, and variability of defects in modern manufacturing contexts has been reviewed as being the foundation of the approach. Combining machine learning methods with proven Design-for-Testability (DFT) methods enable organizations to discover defects much more efficiently, to eliminate testing costs, and to obtain diagnostic accuracy at a higher level. The strengths of the supervised, unsupervised and reinforcement learning methods were discussed as well as practical DFT mechanisms, including scan chains, memory BIST and serial scan architecture. Anomaly detection systems, convolutional neural networks, and autoencoders are a few examples of modern deep learning architectures that show promise for improving pattern identification and training to accommodate changing fault patterns. Regardless of these developments, there are still 4 areas that have challenges, namely, data quality, interpretability, cost of computation and resource-constrained deployment. On the whole, the review helps to realize that the combination of ML-based detection and DFT techniques opens a future direction in creating a more robust, scalable, and intelligent testing architecture that can be utilized to serve next-generation VLSI design and defect-sensitive industrial applications.*

**Keywords:** *Design-for-Testability (DFT), Machine Learning (ML), Defect Detection, VLSI Testing, Built-In Self-Test (BIST), Scan Design, Supervised Learning.*

## INTRODUCTION

The extensive and complicated picture data acquired necessitates a variety of methods for automatic flaw detection. The techniques used might vary from simple edge detection and thresholding to complex machine learning (ML) methods [1]. Notably, ML techniques provide numerous benefits over traditional methods in infrastructure fault identification, including automation, speed, accuracy, customizability, and scalability. This has led to a proliferation of studies that use ML algorithms for automated fault detection, such as methods based on picture categorization, object detection, and semantic segmentation. As an example, it was shown that Convolutional Neural Networks (CNNs) combined with heuristic post-processing techniques could achieve excellent accuracy in crack detection in tunnels [2]. The research does focus on tunnel-specific infrastructure, but it does show how ML techniques can be applied to different kinds of infrastructure.

An ongoing area of study is the use of AI and DL to enhance defect inspection methods. used a defect-free curve to forecast gas porosity in laser metal deposition (LMD) instead of the melt-pool gradient temperature. Another study used center-surround operation, low-resolution processing, and Gaussian pyramid decomposition to find flaws in steel strips. However, crucial picture data can be lost as a result. Developed the Lent model for number recognition using artificial neural networks; it used convolution and pooling layers.

Design for Test (DFT) has become an inescapable discipline in semiconductor engineering, owing to the increased complexity and scale of integrated circuits (ICs) with billions of transistors. In HPC and GPU contexts, DFT lies at the cutting edge of the functional correctness of chips, fault tolerance, and production viability for chips that operate at the edge of speed, power, and concurrency [3]. Tests of these modern chips (which commonly appear in data centers, AI training, and rendering engines) require robust test strategies that can reveal subtle defects without sacrificing their time to market and yield targets

The development of complicated hardware and software systems that consistently perform throughout their operational lives depends on creating testable designs. Design defects may be undetected until the product is in use if testability is not implemented, and operational faults may also be hard to identify and fix. External black-box behavior testing becomes increasingly impractical for comprehensive system integrity assessment as system complexity increases [4][5]. System complexity further complicates testing equipment and procedures. Designing a system to be more testable should improve quality, decrease test costs, and shorten time to market.

The circuit under test (CUT) is subjected to test patterns, and its response is compared to the response of the good circuit, which is determined by simulation. Enhanced controllability and observability of digital circuit internal nodes are achieved through the application of design for testability (DFT) approaches. Scanning forward approaches are extremely popular among DFT methods. The ability to toggle between the circuit's normal operating mode and a test mode is crucial for the operation of scan-path techniques [6]. Within the test mode, the bedtables are linked together to form a shift register. By switching to test mode, and move the bedtables to follow whatever random pattern like. The bedtables acquire the circuit's reaction to the test pattern by returning to functional mode for one clock pulse. Switching back to test mode allows to simultaneously change the pattern in the chain and remove the answer. Integrated self-testing refers to the incorporation of circuitry within a chip that may either generate test vectors or evaluate the results of the chip's output. Regularly, cellular automata or linear feedback shift registers (LFSRs) are used to generate patterns in BIST.

## Structure of the Paper

The outline of this paper is as follows In Section II, go over the basics of Design-for-Testability (DFT) and some important approaches, like scan design and Memory BIST. In Section III, go over the many ML paradigms—supervised, unsupervised, and reinforcement learning—and how they apply to DFT. Traditional algorithms and deep learning models are provided in Section IV, which covers ML approaches for defect detection. Section V presents a comprehensive analysis of the current research literature, while Section VI wraps up by outlining important conclusions and potential avenues for farther study.

## FUNDAMENTALS OF DESIGN-FOR-TESTABILITY (DFT) AND TECHNIQUES

Test engineers, worried about how difficult it will be to test a poorly planned design, often drive and influence the design-for-test (DFT) process. Many companies still haven't answered the two main questions: when DFT be done and who execute it? In response to first question, yes, design engineers are the ones who should do DFT in the end; test engineers should be considered beneficiaries (or at most, consultants) rather than implementers [7]. Implementing DFT early in the design process is essential to avoid redesigning for testability, as it is a design activity. Due to the complexity of the various variables, the exact cost of DFT remains unknown. For instance, design engineers incur a learning cost when they initially use boundary scan, but this expense not be necessary for future designs that utilize this technology. The designers might also become proficient in various DFT approaches. The incorporation of DFT also ensures that designs of subsequent generations DFT-ready, since many circuits are based on earlier designs.

### Design for Testability in VLSI

Integration of testability into the design and production of ICs (integrated circuits) has been a growing trend. Various physical failures can occur for many different reasons, including manufacturing flaws or exposure to vibration, heat, or cosmic rays, which is why chip resting is so important. Design verification is an essential step in engineering development as it guarantees the functionality is right. Full functional accuracy is assumed during chip testing. Hardware testing, on the other hand, presupposes functional faults and flawless replication [8]. For the purpose of testing chips, input bit strings are applied and the resulting output bit strings are reviewed. In a basic AND gate, for instance, a logical 1 only be output if and only if both input lines are positive. To design an integrated circuit for testing, and can take one of three primary approaches. First, component-specific ad hoc solutions. Capabilities such as decoupling elements, increasing the number of test points, and breaking down networks into smaller, more manageable components are outlined here. 2) In sequential circuits, the structured approach denotes design rules that can be used to decrease or make manageable the states of eleven variables. When compared to evaluating a combinational circuit, testing a sequential circuit (a state machine) is far more tough. New to Built-in Test (BIT) are the industry-standard test circuits. Despite a relatively stable number of pins, the density of logic gates on circuits has been rising.

### Memory BIST

The memory test is where DFT technologies would like to start because it is more difficult and can be done at the design level of register transfer. When problems arise with embedded memory tests, users often turn to Memory BIST (Built-In Self-Test) because of its proven track record of success [9]. With Memory BIST, pattern generation is made easier, algorithmic patterns provide good test quality, and test logics greatly minimize the impact on timing and area. Memory BIST basically encapsulates the RAM cores in an extra layer of testing circuitry, acting as a go-between for other logics and the RAM.

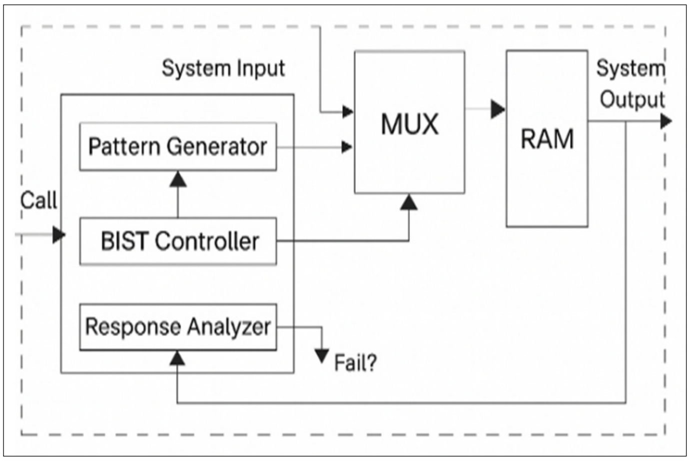

**Fig. 1.** Principle of Memory BIST

The memory BIST principle is illustrated in Figure 1. On one side, and see the test logic that is employed to create test patterns, apply them to the Mused RAM, and then analyze the results. Wrapped in mux collars is the RAM that is being tested on the right side: The pattern generator, which is an FSM, implements several test methods, including MARCH, to generate the control and test signals. By comparing the predicted values with the output data using the response analyzer—a comparator or an XOR compressor—the RAM is determined to be OK. To simplify test logic and save space for applications with a lot of RAMs, several RAMs can be evaluated under a single BIST controller. It is also possible to decrease test time using pipelines if the RAM is quite huge.

### Serial Scan with Random Test Patterns

Several factions have proposed switching to BET mode and employing the scan chain technique. LeBlanc implemented an LSSD scan using pseudorandom test patterns. A comparable approach is demonstrated here, which augments the BEST method with a generic scan chain. By combining BEST with serial scan, a BIST method may be created, which improves the internal node's controllability and observability compared to BEST alone. shows that for the main inputs, an LFSR applies test patterns, and for the main outputs, a MISR compresses them [10]. The scan chain is also driven by patterns that are scanned in from the LFSR. This is the order in which the circuit tests: Step 1: Get the circuit back to its original, known condition by using the scan chain or a master reset signal. 2. Enter scan-test mode and operate the input LFSR for a fixed number of pulses (let's call it m) to randomly set the internal flip-flops. 3. To test the combinational logic of the circuit, return to the usual working mode for n clock pulses. 4. Put another random pattern into the internal registers and go back to step 2. Keep going until gets a satisfactory level of fault coverage in the test patterns. It is possible to find the parameters m and n, as well as the optimal number of iterations (say p), through the use of trial-and-error methods and failure simulation.

## MACHINE LEARNING IN DEFECT DETECTION

The main goal of defect detection technology is to examine test materials for surface flaws such spots, scratches, color variations, and inadequacies. Acquiring data regarding the flaws, including their location, classification, dimensions, shapes, and other distinguishing characteristics, is the objective [11]. Traditional defect detection techniques were mostly based on human intervention, which had several limitations, including inefficiency, poor precision, high costs, and the difficulty to do inspections on a large scale. The advancement of information perception sensor technologies, particularly visual imaging approaches, has, nevertheless, pushed ML-based fault detection systems forward.

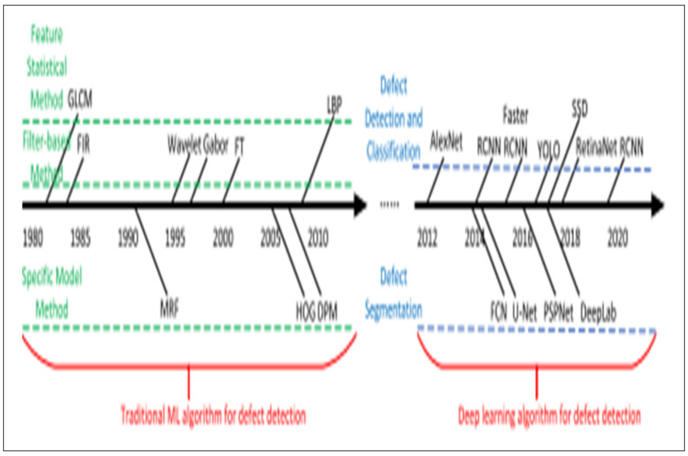

**Fig. 2.** Significant Technical Milestones of Defect Detection Based on ML

The present study traces the historical progression of ML-based defect detection technology, as shown in Figure 2. The comeback of neural networks in 2012 serves as the fulcrum around which the distinction revolves, carving out two distinct eras in the history of defect detection technology: the conventional ML period and the DL era.

### Overview of Machine Learning Concepts

ML is any method that is data-driven to enable a system to learn correlations in data and base predictive or decision-making process on the data correlations with no degree of effort by the user of performing the prediction or decision process directly through written code. ML has three dominant categories, namely supervised, unsupervised and reinforcement learning and each of the categories applied to different classes of problems.

- **Supervised Learning:** Training a model with supervised learning requires paired input and output labels, which are achieved through labelled data. Over time, the model is trained to anticipate the outcome given the input.
- **Unsupervised Learning:** Unsupervised learning seeks to discover inherent structures or hidden patterns in unlabelled data.
- **Reinforcement Learning:** Training a model to create decision sequences using reinforcement learning entails

rewarding desired behaviours and penalizing undesired ones.

### Benefits of ML in DFT Context

The benefits related to DFT offered by ML can assist to address the issues of the available traditional methods of testing and diagnosing. This benefit of ML is, first of all, the increase in defect localization when ML models are trained on large amounts of data in order to discover complex and even non-linear interrelations that are inaccessible to rule-based systems [12]. Besides, ML lessens the need to manually interfere with test development and error categorization, accelerating the entire process of the design verification. Moreover, ML is flexible with the trends that are present in fresher defect types which are realized in progressive technologies resulting in the detection of latent defects earlier. ML is also data-based, and engineers can trade off the test coverage quantitatively, cut short test time and cut short the test expense. Moreover, ML-based methods employ incomplete labelling methods like anomaly detection that can capture rare burdensome pattern in the labelling process. Incorporation of ML provides intelligent, scalable and robust process of DFT to maintain the rate with respect to complexity of the VLSI design.

### Challenges of ML Integration in Hardware Design

Accuracy, energy efficiency, throughput, latency, and cost are the main metrics for embedded machine learning. It is important to use a big enough dataset to evaluate the machine learning algorithm's accuracy [13]. Researchers can make use of numerous popular publicly available datasets, such as ImageNet. The requirement for programmability is emphasized by the fact that the weights must be adjusted in reaction to changes in the environment or application. For DNNs to work, the processor needs to be flexible enough to accommodate networks with variable topologies, filter sizes, channels, and layer counts. Data transfer and computing are both accelerated by the high dimensionality and the requirement for programmability. Reading and storing the weights is also required by programmability, and the amount of data generated rises with more dimensionality. Since transporting data is more expensive than computing it, this presents a problem for energy efficiency. It goes over a number of ways to lessen data transmission in order to cut down on power usage. The computational load, which grows in proportion to the data's dimensionality, determines the throughput. In order to decrease the number of processes needed, it goes over different ways the data might be changed. The quantity of storage space needed on the chip determines the cost. It include a range of techniques for lowering storage costs, shrinking chip size, and keeping off-chip memory bandwidth low.

## ML TECHNIQUES FOR DFT DEFECT DETECTION

The problem of defect detection has been addressed through increasing employing of ML techniques to enhance defect detection during DFT processes. In supervised learning algorithms (the SVM, DT and NNs methods, etc.), high accuracy of labeling defective and non-defective patterns is achieved by using vertical-labelled data. The purpose of such models is to train to identify the associations that exist between the responses to the test and the diagnosed errors and so that it becomes simpler to expose the errors during the tests swiftly and more effectively. In the case that labelled data are not easily obtainable, unsupervised learning techniques (i.e., clustering, and anomaly detection, etc.) may be utilized to identify patterns, or identify anomalous rows, indicating fault. Defect detection There are numerous applications of DL to defect detection (using CNNs to detect defects in pictures of circuits (i.e., layout-level data), and using autoencoders to compress data and distill features). Tests are also being deployed to the use of Reinforcement Learning (RL) methods in order to optimize the generation of patterns. Although these ML approaches hold so much potential in automating and optimizing defect detection, other issues relating to model interpretability, quality of data, and practical application in the real life.

### Supervised Learning Method

Supervised techniques make decisions on the raw, unlabelled data from the testing phase using the precisely annotated data from the training phase. In the supervised approach, there are two steps: training and testing [14]. A model is generated by mapping the features of the data points to the classes. At this point, we put the model through its paces by having it sort the unlabelled data. To showcase the variable outcomes, human interaction is required throughout the training process. Supervised classification outperforms unsupervised classification in every metric that matters. Here go over some of the classifiers.

#### *K-Nearest Model*

The k-NN algorithm, a memory-based supervised learning method, compares newly-introduced issue instances without labels to a set of labelled examples in the training set [15]. The k-NN classification model consists of two steps.

- "Determining the closest neighbours for an instance without labels
- Using neighbours to determine the instance class

#### *Support Vector Machine*

Using a hyper-plane, SVM divides the product image into two categories, with the best classifier being determined. Nearest data point is located on the hyper-plane, which has the maximum distance.

### Unsupervised Learning Method

Unsupervised segmentation with homogeneous characteristics does not necessitate the training data [16]. The algorithm takes care of determining the number of classes automatically. Active Contour Models and Clustering are two examples of unsupervised approaches.

### *Clustering*

This method for segmenting product images does not rely on labels and instead divides the unlabelled data into clusters. When pixels are clustered, they share the same features. Two distinct kinds of clustering exist: rigid and soft. When using hard clustering, each data point can only be classified as either belonging to a cluster or not. In contrast to k-means clustering, which uses cluster membership probabilities, soft clustering instead uses dissimilarities. Fuzzy C-Means clustering allows for data points to simultaneously be part of multiple clusters.

### *Fuzzy C-means Clustering*

The FCM clustering approach is based on the idea that for every feature data point xi in the set ℘d, membership values uij are assigned to it, where i = 1, 2…, n. Each cluster's centre cj, where j ranges from 1 to k. In this case, the value of k is less than n, which is determined by the data point's distance from the base of the pyramid. In this case, the membership values meet all of the criteria. Standard FCM algorithm's main drawbacks are its sensitivity to noise and its use of the non-robust Euclidean distance. The second constraint is that several academics have attempted to address it by developing algorithms that take spatial information into consideration.

### Deep Learning Approaches

There are two different ways to describe the goals of mistake detection judgment. Determining the volume of the contents in the transparent container should be the first step. Second, may determine if there was an error by looking at the label sticker on the container and how damaged it is. Both of these types of defects are illustrated by the following figure, Figure 3.

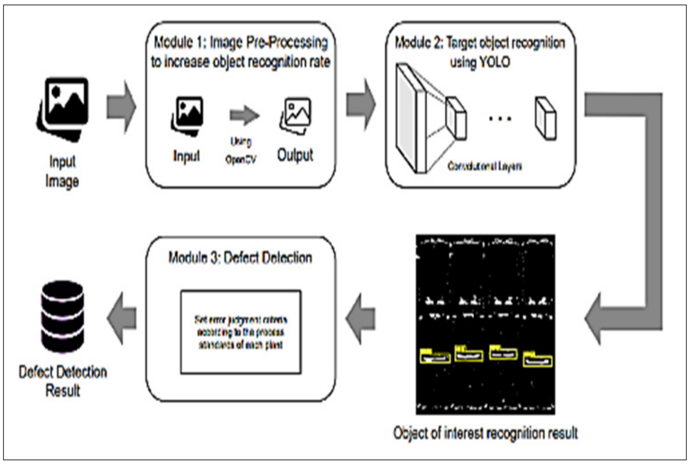
Input
Image
Module 1: Image Pre-Processing to increase object recognition rate
Input
Using OpenCV
Output
Module 2: Target object recognition using YOLO
Convolutional Layers
Object of interest recognition result
Module 3: Defect Detection
Set error judgment criteria according to the process standards of each plant
Defect Detection Result

**Fig. 3.** A Proposed Deep Learning-Based Defect Detection Model

The image that was taken is (1) pre-processed with OpenCV, and (2) YOLO that has already been learned is used to find the object that was captured. There are three steps to detecting errors in a process: (1) extracting the ROI, or area of interest of the object; (2) using the information from the extracted region; and (3) comparing the results to the error determination standard that each manufacturer has defined in advance. Minimizing the financial and technological strain on a small plant is achievable with this model, which may be utilized to identify flaws that would otherwise require a scale or supplementary tools to discover, all from outside photos.

## LITERATURE REVIEW

This literature Summary summarizes recent advancements in defect detection and testing methodologies, emphasizing the integration of ML, DL, and DFT strategies across various domains, highlighting their effectiveness, comparative evaluations, application-specific innovations, and future research potential.

H.S. et al. (2019) proposes a new method that uses ML algorithms to classify and locate issues with centrifugal water pumps used in agriculture that are powered by induction motors. All forms of life rely on water as an essential component. Water is used by humans for many different things, including cleaning, power generating, cooking, bathing, washing, drinking, and farming [17].

Sun et al. (2019) analyzes the various problems with agricultural induction-motor-driven centrifugal water pumps and proposes a new method to detect and classify these problems using ML. Water is the most vital component of all living things. Water is used by humans for many things, including cooking, cleaning, bathing, drinking, growing, and producing electricity [18].

Hosseinzadeh, Masoodzadeh and Roshandel (2019) Using PCA and LDA is a benefit of the proposed strategy. The dataset matrices are shrunk by PDA. Principal component analysis (PCA) is used to minimize dataset sizes and eliminate prospective singularities. When the LDA method is applied to the data produced by the PCA, the distance between classes increases and the dataset's within-class distance decreases. Lastly, the problem is identified and its classes are determined using the popular K-nearest neighbour technique. The study's findings show how reliable and efficient the suggested algorithm is for identifying the problem class in smart grids [19].

Amruthnath and Gupta (2018) Various unsupervised learning approaches were evaluated for performance, accuracy, and stability using a basic set of vibration data from an exhaust fan. The principal component analysis is one such method. The T2 statistic, K-Means, fuzzy C-Means, and model-based clustering are all part of architectural clustering. Ultimately, suggested a process to evaluate many algorithms and select the best model [20].

Zidi, Moulahi and Alaya (2018) The SVM classification method is used for this purpose. SLT forms the basis of SVM, and use it to create a decision function. Due to the minimal resource requirement of this judgment function, cluster heads can easily employ it to detect abnormal sensors. Through an experimental study, the efficacy of SVM for fault detection in WSNs is demonstrated, contrasting it with the most recent for the same application [21].

Heo and Lee (2018) The fault detection and classification challenge is solved using DNNs to demonstrate their capabilities. Step one is to reframe the issue of defect identification and classification as a neural network-based classification problem. The next step is to train neural networks to detect errors. Investigate next the effects of various hyperparameters on these networks' performance, such as the total number of hidden layers and the neuron count in the last hidden layer. Additionally, using neural networks with data augmentation helps with the problem of defect categorization. Finally, the results are compared to other data-driven approaches to show how deep neural networks are beneficial [22].

Table I presents a comparative overview of recent studies employing ML and DL techniques for defect detection in DFT, highlighting methodologies, key findings, challenges, and future research directions

**Table 1.** Comparative Table of Reviewed Studies Machine Learning for Defect Detection in Design-for-Testability Processes

| Author | Study On | Approach | Key Findings | Challenges | Future Directions |
|---|---|---|---|---|---|
| H.S. et al. (2019) | Fault analysis in centrifugal water pumps driven by induction motors (Agricultural use) | Algorithm for machine learning - based defect detection and categorization | Proposed an efficient and accurate algorithm to detect and classify multiple pump faults; improved reliability in agricultural pump systems | Limited to agricultural pump datasets; performance may vary for industrial or large-scale motors | Extend to real-time monitoring, multi-sensor fusion, and broader mechanical systems |
| Sun et al. (2019) | Fault detection in chemical processes (TEP benchmark & real chemical dataset) | The Variational Dropout Bayesian Recurrent Neural Network (BRNN) | BRNN effectively models nonlinear dynamics; provides uncertainty estimates; outperforms PCA-based methods in fault detection and propagation analysis | BRNN training is computationally heavy; performance depends on time-series data quality | Apply BRNN to more industrial processes, integrate with predictive maintenance, explore hybrid ML-model-based systems |
| Hosseinzadeh, Masoodzadeh & Roshandel (2019) | Fault classification in Smart Grids | PCA for dimensionality reduction + LDA for class separation + K-NN for classification | Combined PCA–LDA–KNN framework is robust and effective in identifying fault classes; reduces data singularity issues | Performance depends on feature quality; K-NN may be slow for large datasets | Explore deep learning, ensemble methods, real-time grid monitoring, and improved feature extraction |
| Amruthnath & Gupta (2018) | Fault detection using vibration data from exhaust fan systems | Hierarchical clustering, PCA $T^2$ Statistic, K-Means, Fuzzy C-Means, and Model-Based Clustering are all examples of unsupervised ML approaches. | Benchmarking framework developed; demonstrated variability of performance across clustering techniques; unsupervised methods useful when labels are unavailable | Limited to small dataset; unsupervised models struggle with ambiguous clusters | Apply to larger industrial datasets, develop automatic clustering selection mechanism, and integrate deep unsupervised learning |
| Zidi, Moulahi & Alaya (2018) | Fault detection in Wireless Sensor Networks (WSNs) | SVM for anomaly detection | SVM provides lightweight and accurate detection suitable for resource-constrained sensor nodes; performs well compared to state-of-the-art | Limited to specific WSN configurations; may not scale well in dynamic environments | Extend to adaptive SVM, hybrid ML models, energy-efficient distributed fault detection |
| Heo & Lee (2018) | Fault detection & classification across engineering systems | Deep Neural Networks (DNNs) with hyperparameter tuning and data augmentation | DNNs show superior performance over traditional approaches; data augmentation improves model robustness; hyperparameters significantly influence accuracy | Requires large training data; computationally expensive; risk of overfitting | Apply CNN/LSTM architectures, real-time fault diagnosis, develop lightweight DNNs for embedded systems |

## CONCLUSION AND FUTURE WORK

In the current fast changing world of technology, accuracy, reliability, and efficiency in defects detection has been crucial to the quality and the performance of the modern semiconductor and electronic systems. In this paper, the review has been conducted on the inadequacy of the traditional inspection methods as being too basic to manage the rising complexity, density and variability of faults within the modernized manufacturing settings. With the combination of ml technologies and proven DFT practices, organizations able to tremendously improve their level of defect discovery, decrease testing resources, and attain greater levels of diagnostic accuracy. The strengths of supervised, unsupervised and reinforcement learning methods and the practical mechanisms of DFT that have been used like scan chains, memory BIST and serial scan architectures as discussed here. Moreover, new DL systems (such as CNN, autoencoders, and ADS) provide a bright way forward on enhancing pattern recognition and changing response to evolving defect patterns. Although these strides have been made, there are still problems in areas like quality of data, interpretability, cost of computation, and resource constrained deployment. On the whole, the review illustrates that the combination of the ML-based detection and DFT technologies offers a transformative direction of creating more resistant, scalable, and intelligent test frameworks that may be used to exploit next-generation VLSI design and defect-sensitive industrial solutions.

## REFERENCES

1. M. Felderer, B. Russo, and F. Auer, "On Testing Data-Intensive Software Systems," in Security and Quality in Cyber-Physical Systems Engineering, 2019, pp. 129–148. doi: 10.1007/978-3-030-25312-7_6.
2. P. V. Shantagiri and R. Kapur, "Aggressive Exclusion of Scan Flip-Flops from Compression Architecture for Better Coverage and Reduced TDV: A Hybrid Approach," J. Low Power Electron. Appl., vol. 9, no. 2, p. 18, May 2019, doi: 10.3390/jlpea9020018.
3. K. Tahera, D. C. Wynn, C. Earl, and C. M. Eckert, "Testing in the incremental design and development of complex products," Res. Eng. Des., vol. 30, no. 2, pp. 291–316, Apr. 2019, doi: 10.1007/s00163-018-0295-6.
4. E. Khalil, M. El-Mahlawy, F. Ibrahim, and M. Abdel-Azeem, "Design for Testability of Circuits and Systems; An overview," Int. Conf. Electr. Eng., vol. 5, no. 5, pp. 1–24, 2006, doi: 10.21608/iceeng.2006.33550.
5. A. Vijay, S. Arora, S. Gupta, and M. Chhabra, "Halophilic starch degrading bacteria isolated from Sambhar Lake, India, as potential anode catalyst in microbial fuel cell: A promising process for saline water treatment," Bioresour. Technol., vol. 256, pp. 391–398, May 2018, doi: 10.1016/j.biortech.2018.02.044.
6. S. Asvini and C. Nirmala, "Design for Testability in Timely Testing of Vlsi Circuits," J. Eng. Res. Appl., vol. 5, no. 3, pp. 10–13, 2015.
7. L. Y. Ungar, "Design for Testability (DFT) to Overcome Functional Board Test Complexities in Manufacturing Test," Proc. IPC APEX, 2017.
8. R. V. Binder, "Design for testability in object-oriented systems," Commun. ACM, vol. 37, no. 9, pp. 87–101, Sep. 1994, doi: 10.1145/182987.184077.
9. Y. X. Y. Xu, T. L. T. Lv, W. L. W. Lu, X. Y. X. Yang, H. L. H. Li, and X. L. X. Li, "Advanced topics of DFT technologies in a general purposed CPU chip," in 2003 5th International Conference on ASIC Proceedings (IEEE Cat No 03TH8690) ICASIC-03, IEEE, 2003, pp. 1179-1182 Vol.2. doi: 10.1109/ICASIC.2003.1277424.
10. H. T. Nagle, S. C. Roy, C. F. Hawkins, M. G. McNamer, and R. R. Fritzemeier, "Design for testability and built-in self test: a review," IEEE Trans. Ind. Electron., vol. 36, no. 2, pp. 129–140, 1989, doi: 10.1109/41.19062.
11. S. Garg, "Predictive Analytics and Auto Remediation using Artificial Inteligence and Machine learning in Cloud Computing Operations," Int. J. Innov. Res. Eng. Multidiscip. Phys. Sci., vol. 7, no. 2, 2019, doi: 10.5281/zenodo.15362327.
12. D. Hely, K. Rosenfeld, and R. Karri, "Security challenges during VLSI test," in 2011 IEEE 9th International New Circuits and systems conference, IEEE, Jun. 2011, pp. 486–489. doi: 10.1109/NEWCAS.2011.5981325.
13. V. Sze, Y. H. Chen, J. Einer, A. Suleiman, and Z. Zhang, "Hardware for machine learning: Challenges and opportunities," Proc. Cust. Integr. Circuits Conf., vol. 2017-April, no. December, 2017, doi: 10.1109/CICC.2017.7993626.
14. A. A. Nair, T. D. Tran, A. Reiter, and M. A. Lediju Bell, "One-Step Deep Learning Approach to Ultrasound Image Formation and Image Segmentation with a Fully Convolutional Neural Network," IEEE Int. Ultrason. Symp. IUS, vol. 2019-Octob, pp. 1481–1484, 2019, doi: 10.1109/ULTSYM.2019.8925836.
15. S. Liu and W. Luk, "Towards an efficient accelerator for DNN-based remote sensing image segmentation on FPGAs," Proc. - 29th Int. Conf. Field-Programmable Log. Appl. FPL 2019, pp. 187–193, 2019, doi: 10.1109/FPL.2019.00037.
16. P. Qin, J. Zhang, J. Zeng, H. Liu, and Y. Cui, "A framework combining DNN and level-set method to segment brain tumor in multi-modalities MR image," Soft Comput., vol. 23, no. 19, pp. 9237–9251, 2019, doi: 10.1007/s00500-019-03778-x.
17. R. C. H.S., S. C. Bharadwaj, U. S., S. K. Padmanaban, N. Dutta, and J. B. Holm-Nielsen, "Electrical Fault Detection Using Machine Learning Algorithm For Centrifugal Water Pumps," in 2019 IEEE International Conference on Environment and Electrical Engineering and 2019 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I&CPS Europe), IEEE, Jun. 2019, pp. 1–6. doi: 10.1109/EEEIC.2019.8783841.
18. W. Sun, A. R. C. Paiva, A. Peng Xu, and R. D. B. Sundaram, "Fault Detection and Identification using Bayesian Recurrent Neural Networks," arXiv, vol. 141, 2019, doi: 10.48550/arXiv.1911.04386.

19. J. Hosseinzadeh, F. Masoodzadeh, and E. Roshandel, "Fault detection and classification in smart grids using augmented K-NN algorithm," SN Appl. Sci., vol. 1, no. 12, p. 1627, Dec. 2019, doi: 10.1007/s42452-019-1672-0.

20. N. Amruthnath and T. Gupta, "A research study on unsupervised machine learning algorithms for early fault detection in predictive maintenance," in 2018 5th International Conference on Industrial Engineering and Applications, ICIEA 2018, 2018. doi: 10.1109/IEA.2018.8387124.

21. S. Zidi, T. Moulahi, and B. Alaya, "Fault Detection in Wireless Sensor Networks Through SVM Classifier," IEEE Sens. J., vol. 18, no. 1, pp. 340–347, 2018, doi: 10.1109/JSEN.2017.2771226.

22. S. Heo and J. H. Lee, "Fault detection and classification using artificial neural networks," IFAC-PapersOnLine, vol. 51, no. 18, pp. 470–475, 2018, doi: 10.1016/j.ifacol.2018.09.380.

23. Routhu, K. K. (2022). From Case Management to Conversational HR: Redefining Help Desks with Oracle's AI and NLP Framework. International Journal of Science, Engineering and Technology, 10(6).

24. Vattikonda, N., Gupta, A. K., Polu, A. R., Narra, B., Buddula, D. V. K. R., & Patchipulusu, H. H. S. (2022). Blockchain Technology in Supply Chain and Logistics: A Comprehensive Review of Applications, Challenges, and Innovations. International Journal of Emerging Trends in Computer Science and Information Technology, 3(3), 72-80.

25. Attipalli, A., BITKURI, V., Mamidala, J. V., Kendyala, R., & KURMA, J. (2022). Empowering Cloud Security with Artificial Intelligence: Detecting Threats Using Advanced Machine learning Technologies. Available at SSRN 5741263.

26. Padur, S. K. R. (2022). Intelligent resource management: AI methods for predictive workload forecasting in cloud data centers. J. Artif. Intell. Mach. Learn. & Data Sci, 1(1), 2936-2941.

27. Routhu, K. K. (2022). From RFID to Geofencing: IoT-Enabled Smart Time Tracking in Oracle HCM Cloud. International Journal of Science, Engineering and Technology, 10(4).

28. Polam, R. M., Kamarthapu, B., Kakani, A. B., Nandiraju, S. K. K., Chundru, S. K., & Vangala, S. R. (2022). Data Security in Cloud Computing: Encryption, Zero Trust, and Homomorphic Encryption. International Journal of Emerging Trends in Computer Science and Information Technology, 3(4), 31-41.

29. Padur, S. K. R. (2022). AI augmented platform engineering, transforming developer experience through intelligent automation and self optimizing internal platforms. International Journal of Science, Engineering and Technology, 10(5), 10-5281.

30. Polu, A. R., Buddula, D. V. K. R., Narra, B., Gupta, A., Vattikonda, N., & Patchipulusu, H. (2021). Evolution of AI in Software Development and Cybersecurity: Unifying Automation, Innovation, and Protection in the Digital Age. Available at SSRN 5266517.

31. Padur, S. K. R. (2020). From centralized control to democratized insights: Migrating enterprise reporting from IBM Cognos to Microsoft Power BI. Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol, 6(1), 218-225.

32. Bitkuri, V., Kendyala, R., Kurma, J., Mamidala, V., Enokkaren, S. J., & Attipalli, A. (2021). Systematic Review of Artificial Intelligence Techniques for Enhancing Financial Reporting and Regulatory Compliance. International Journal of Emerging Trends in Computer Science and Information Technology, 2(4), 73-80.

33. Padur, S. K. R. (2019). Machine learning for predictive capacity planning: Evolution from analytical modeling to autonomous infrastructure. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 5(5), 285-293.

34. Attipalli, A., Enokkaren, S., BITKURI, V., Kendyala, R., KURMA, J., & Mamidala, J. V. (2021). Enhancing Cloud Infrastructure Security Through AI-Powered Big Data Anomaly Detection. Available at SSRN 5741305.

35. Singh, A. A. S., Tamilmani, V., Maniar, V., Kothamaram, R. R., Rajendran, D., & Namburi, V. D. (2021). Predictive Modeling for Classification of SMS Spam Using NLP and ML Techniques. International Journal of Artificial Intelligence, Data Science, and Machine Learning, 2(4), 60-69.

36. Padur, S. K. R. (2020). AI augmented disaster recovery simulations: From chaos engineering to autonomous resilience orchestration. International Journal of Scientific Research in Science, Engineering and Technology, 7(6), 367-378.

37. Reddy Padur, S. K. (2021). From Scripts to Platforms-as-Code: The Role of Terraform and Ansible in Declarative Infrastructure Rollouts. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 621-628.

38. Kothamaram, R. R., Rajendran, D., Namburi, V. D., Singh, A. A. S., Tamilmani, V., & Maniar, V. (2021). A Survey of Adoption Challenges and Barriers in Implementing Digital Payroll Management Systems in Across Organizations. International Journal of Emerging Research in Engineering and Technology, 2(2), 64-72.

39. Padur, S. K. R. (2018). Autonomous cloud economics: AI driven right sizing and cost optimization in hybrid infrastructures. International Journal of Scientific Research in Science and Technology, 4(5), 2090-2097.

40. Rajendran, D., Namburi, V. D., Singh, A. A. S., Tamilmani, V., Maniar, V., & Kothamaram, R. R. (2021). Anomaly Identification in IoT-Networks Using Artificial Intelligence-Based Data-Driven Techniques in Cloud Environmen. International Journal of Emerging Trends in Computer Science and Information Technology, 2(2), 83-91.

41. Padur, S. K. R. (2021). Bridging Human, System, and Cloud Integration through RESTful Automation and Governance. the International Journal of Science, Engineering and Technology, 9(6).

42. Attipalli, A., BITKURI, V., KURMA, J., Enokkaren, S., Kendyala, R., & Mamidala, J. V. (2021). A Survey of Artificial Intelligence Methods in Liquidity Risk Management: Challenges and Future Directions. Available at SSRN 5741342.

43. Padur, S. K. R. (2021). From Control to Code: Governance Models for Multi-Cloud ERP Modernization. International Journal of Scientific Research & Engineering Trends, 7(3).

44. Routhu, K. K. (2021). Harnessing AI Dashboards in Oracle Cloud HCM: Advancing Predictive Workforce Intelligence and Managerial Agility. International Journal of Scientific Research & Engineering Trends, 7(6).

45. Padur, S. K. R. (2021). Deep learning and process mining for ERP anomaly detection: Toward predictive and self-monitoring enterprise platforms. Available at SSRN 5605531.

**Citation:** Sri Harsha Panchali, Usha Mohani kavirayani, et al., “Machine Learning for Defect Detection in Design-for-Testability Processes: Techniques and Trends”, Universal Library of Engineering Technology, 2023; 53-61. DOI: https://doi.org/10.70315/uloap.ulete.2023.008.