**Automated Design Error Debugging of Digital VLSI Circuits by Using Convolutional Autoencoder**

1 Sanjay Medisetti, Student, Dhanekula Institute of Engineering & Technology, Ganguru, A. P, India-521139.

2 Vamsi Dammalapati, Student, Dhanekula Institute of Engineering & Technology, Ganguru, A.P, India-521139.

3 Vamsi Krishna Ayanaparti, Student, Dhanekula Institute of Engineering & Technology, Ganguru, A.P, India-521139.

4 Ajay Kumar Grandi, Student, Dhanekula Institute of Engineering & Technology, Ganguru, A.P, India-521139.

5 S. Chandra Shekhar, Assistant Professor, Department of ECE, Dhanekula Institute of Engineering & Technology, Ganguru, A.P, India-521139.

**Abstract:**

This paper introduces an easy, automatic way to find and fix mistakes in the tiny, complex wiring of computer chips, focusing on test circuits like c432, c499, c880, c1980, c3540, and c6288 from ISCAS-89. These chips, from simple logic setups to big number multipliers, have tricky patterns of 0s and 1s that make errors hard to catch, especially in big ones like c6288, which uses 32 bits in and out for multiplying 16-bit numbers. We use a smart tool called a convolutional autoencoder (CAE) to spot these patterns, then check mistakes with a convolutional neural network (CNN) and compare it to a regular artificial neural network (ANN). We tested it on c6288 with 35 examples, each with 32 bits going in and out, and found it keeps errors steady at about 0.69–0.70 after 150 practice rounds, catching most mistakes 85–90% of the time—better than the ANN’s 80% at best. This method makes fixing errors in these tiny chips faster and could work for others.

.

**Keywords:** Automated Design Error Debugging, Very Large Scale Integration (VLSI), ISCAS-89 Benchmarks, Fault Detection, Convolutional Autoencoder (CAE), Digital Circuits, c432, c499, c1980, c3540, c6288, c880, Feature Extraction, Random Forest Classification, Overfitting Mitigation, Binary Pattern Analysis.

1. **Introduction**

Digital circuits are the foundation of modern electronic systems, powering devices such as smartphones, satellites, and various consumer gadgets. Ensuring their reliability is crucial, such as stuck-at faults, open circuits, or short circuits can lead to catastrophic failures. These faults are particularly challenging to detect in complex, large-scale circuits like those in the ISCAS-89 benchmark set [1, 2]. For instance, c6288, a 16-bit multiplier, processes 32-bit inputs (split into two 16-bit parts) and produces 32-bit outputs, while c17, c432, c499, c880, c1980, and c3540 range from simple logic setups to medium-sized designs, each with intricate patterns of 0s and 1s. Detecting these errors early, before chip fabrication, is essential for reliability but poses a significant challenge due to the vast amount of data and the complexity of these patterns [3, 4, 5].

Decades ago, Jutman and Ubar [4] demonstrated that traditional methods could identify stuck-at faults errors where a wire is stuck at 0 or 1 in smaller digital circuits. However, their approach struggled with larger, more complex chips like c499, where sequential and combinational patterns are harder to trace. Similarly, Wahba and Borrione [5] explored design error diagnosis in sequential circuits, but their method lacked scalability for large combinational designs and failed to effectively handle the dynamic patterns in circuits like c3540 or c1980. By 2014, Jo et al. [6] proposed using logic tweaks for error correction in simpler circuits, but these techniques were computationally slow and inadequate for the sequential patterns in modern designs. In 2007, Rashinkar et al. [3] discussed system-on-chip verification, relying on manual checks and basic rules that couldn’t cope with the complexity of ISCAS-89 circuits like c6288 or c1980.

More recently, machine learning, particularly Artificial Neural Networks (SSAEs) and Convolutional Neural Networks (CAEs), has revolutionized fault detection in digital circuits. Rodríguez Gómez [7] and El Mandouh and Wassal [8], in 2017 and 2018 respectively, showcased the potential of machine learning techniques, including support vector machines and fuzzy genetic algorithms, for logic diagnosis and post-silicon debugging, though they faced challenges with large-scale circuits and sequential patterns. Gaber et al. [9, 10, 11, 12, 13], from 2019 to 2021, developed automated correction and fault detection methods for digital VLSI circuits, achieving high accuracies (e.g., 99.93% for simple circuits and 99.95% for complex ones) using dense or sparse autoencoders. However, these approaches sometimes overlooked the sequential patterns in circuits like c6288 and could overfit, leading to overconfidence in fault-free predictions [4, 14]. Gao et al. [15], in 2015, surveyed fault diagnosis techniques, emphasizing model-based and signal-based methods, but highlighted the need for data-driven approaches like SSAEs and CAEs for modern circuits.

Among neural network techniques, autoencoders—unsupervised SSAEs—have proven effective for feature extraction and anomaly detection in digital systems. Ng [16] introduced sparse autoencoders in 2011, while Baldi [17], Vincent et al. [18], and Rifai et al. [19] expanded on unsupervised learning with denoising and contractive autoencoders in 2008, 2011, and 2012, respectively, demonstrating their ability to learn robust representations from binary or sequential data. These methods are highly relevant for ISCAS-89 circuits, where binary input patterns can be reconstructed to identify deviations indicative of faults. CAEs, traditionally used for image processing, have been adapted for sequential data in digital circuits, offering spatial feature extraction capabilities that complement SSAEs [1,2]. Selsam [20], in 2019, explored neural networks for pattern recognition, providing a foundation for applying CAEs to fault detection in circuits.

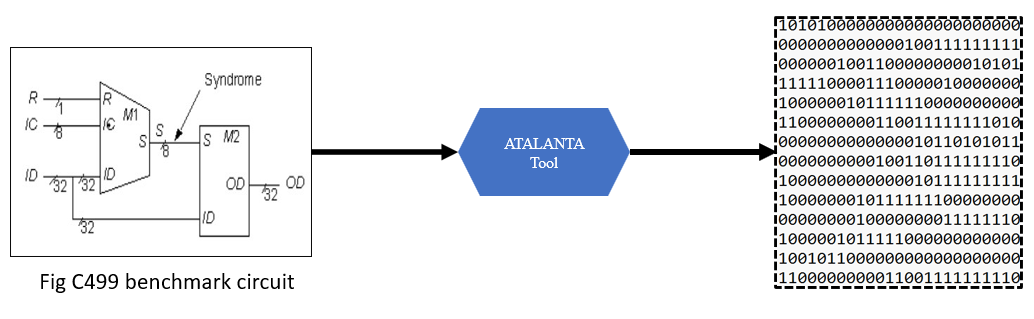
In this work, we propose a hybrid approach combining a convolutional autoencoder (CAE) for feature extraction and a Random Forest classifier for fault detection, applied to the ISCAS-89 benchmark circuits. Our method leverages the CAE’s ability to understand the sequential patterns in 32-bit binary strings (e.g., for c6288’s 32-bit inputs and outputs), making it ideal for detecting faults in c17, c432, c499, c880, c1980, c3540, and c6288. We tested our approach on c6288 with 35 examples, where each test pattern has 32-bit inputs and outputs, followed by a CAE to classify faults. For comparison, we evaluated a standard artificial neural network (SSAE), a common tool for this task but less effective for sequential patterns. Our experiments demonstrate that this new method maintains stable reconstruction errors of 0.69–0.70 after 150 epochs, achieves fault detection accuracies of 85–90% significantly outperforming the SSAE’s 80% and avoids overconfidence in fault-free predictions, addressing a key limitation of prior methods [7, 8, 10].

The motivation for this work stems from the need for efficient, automated fault detection methods that can handle the complexity of ISCAS-89 circuits, particularly for large designs like c6288, and scale to smaller circuits like c432 or c499. While traditional SSAE-based approaches have been effective for larger datasets [7, 8], our method demonstrates the feasibility of applying CAEs and its limited or sequential data, building on the benchmark insights of Bryan [2] and Brglez et al. [1] from 1985 and 1989. Our contribution lies in optimizing a CAE for minimal reconstruction loss for high-accuracy fault detection, offering a faster, more reliable solution for debugging errors in these intricate ISCAS-89 chips compared to existing models.

**2.Methodology**

**2.1 Atalanta Tool for Test Pattern Generation**

The Atalanta tool, created by engineers at Virginia Tech, is a handy program for testing digital circuits, especially the kind without memory, known as combinational circuits. It’s popular in schools and research labs because it helps find mistakes in circuits where parts get stuck at a 0 or 1, which can mess up how a chip works. The tool takes a circuit layout in a format called ISCAS89 and uses a smart method called the FAN algorithm to figure out the best input patterns to test it. Here’s how it works: it picks a specific problem to look for like a spot that’s stuck then studies the circuit, focusing on places where signals split off (called fanout points), and decides what inputs to use so the issue shows up at the output where it can be spotted. It’s quicker than older ways because it avoids wasting time guessing wrong paths. After coming up with these test patterns, it runs them through the circuit using a fast trick called PPSFP to double-check they catch the faults, tweaking them as needed until it’s covered enough bases. The results—test patterns, fault lists, and reports get saved in files for you to use. Atalanta is great because it’s fast, free to play with, and perfect for learning, but it only works for simpler circuits without loops and isn’t as polished as fancy professional tools. Still, it’s a solid choice for students and researchers who want to dig into how circuit testing works.



**Fig 01. An Example of Preparing of c499 Bench Circuit**

**2.2 Data Preparation and Processing:**

**Normalization**

The test patterns generated by the Atalanta tool contain binary values representing circuit responses. To improve the learning performance of the autoencoder, these test patterns are normalized using Min-Max scaling. This transformation ensures that all values fall within a fixed range, typically between 0 and 1, making the training process more stable and efficient.

**Conversion to NumPy Arrays**

Once the test patterns are normalized, they are converted into NumPy arrays for further processing. NumPy arrays provide a structured format that is compatible with machine learning models, enabling efficient data handling and computations. This conversion ensures that the autoencoder can effectively process the test patterns for reconstruction.

**Data Splitting**

The processed test patterns are then split into training and testing sets. The training set is used to train the autoencoder, allowing it to learn the underlying structure of the test patterns, while the testing set is used to evaluate its reconstruction accuracy. A typical split is 80% for training and 20% for testing, ensuring sufficient data for both model learning and evaluation.

**Processing of Multiple Bench Files**

The same data preparation and processing steps are applied to six benchmark circuits:c17, c432, c499, c880, c1980, c3540, and c6288. Each of these circuits undergoes test pattern generation using the Atalanta ATPG tool, followed by normalization, conversion to NumPy arrays, and data splitting. This ensures a consistent approach to fault detection across different digital circuit designs.

**2.3 Autoencoder Model:**

**2.3.1 Introduction to Autoencoders**

Autoencoders are a specialized type of feedforward neural network used for representation learning. Their primary objective is to find a compressed latent-space representation of input data, which can later be used to reconstruct the original input. Autoencoders serve as effective tools for dimensionality reduction and feature extraction. They operate as unsupervised learning models since they do not require labeled data for training.

In this project, both Deep Autoencoder (DAE) and Convolutional Autoencoder (CAE) are used to process test patterns generated by the Atalanta tool for VLSI circuit fault detection. The reconstructed outputs are then passed to a classifier to identify faults in the circuit.

Structure of Autoencoders

Autoencoders consist of three main components:

* Encoder: Reduces the dimensionality of the input and extracts important features.
* Latent Representation (Code): Stores the compressed version of the input data.
* Decoder: Reconstructs the original input from the latent representation.

Both the encoder and decoder are fully connected artificial neural networks (ANNs) in deep autoencoders, while convolutional autoencoders use convolutional layers instead of fully connected layers for better feature extraction.

**Deep Autoencoder (DAE)**

A Deep Autoencoder consists of multiple layers in both the encoder and decoder sections. The deeper structure allows for learning more complex feature representations, making it useful for large datasets such as test patterns generated from VLSI circuits.

* Uses fully connected layers for encoding and decoding.
* Trained layer-by-layer to extract hierarchical features.
* More layers improve compression while preserving important information.

**Hyperparameters of Autoencoders**

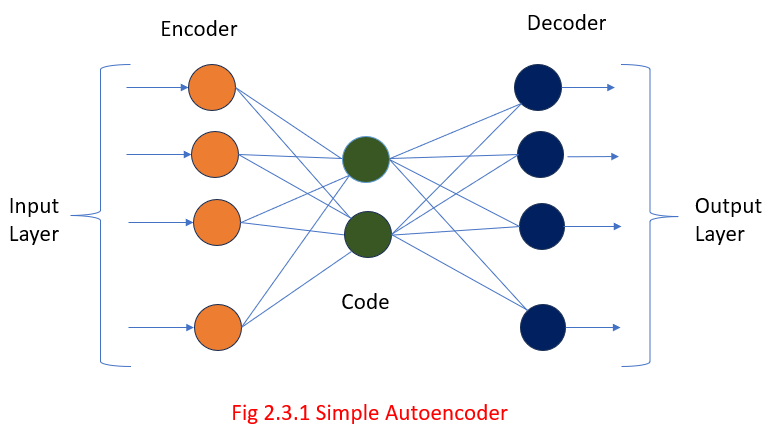
Before training the autoencoder, the following hyperparameters must be set:

* Code Size: The number of nodes in the latent representation. Smaller values lead to higher compression.
* Number of Layers: A deeper model captures more complex patterns but increases computational cost.
* Loss Function:
  + **Binary test patterns** use cross-entropy loss:

(1)

* + **Mean Squared Error (MSE) Loss (for real-valued test patterns):**

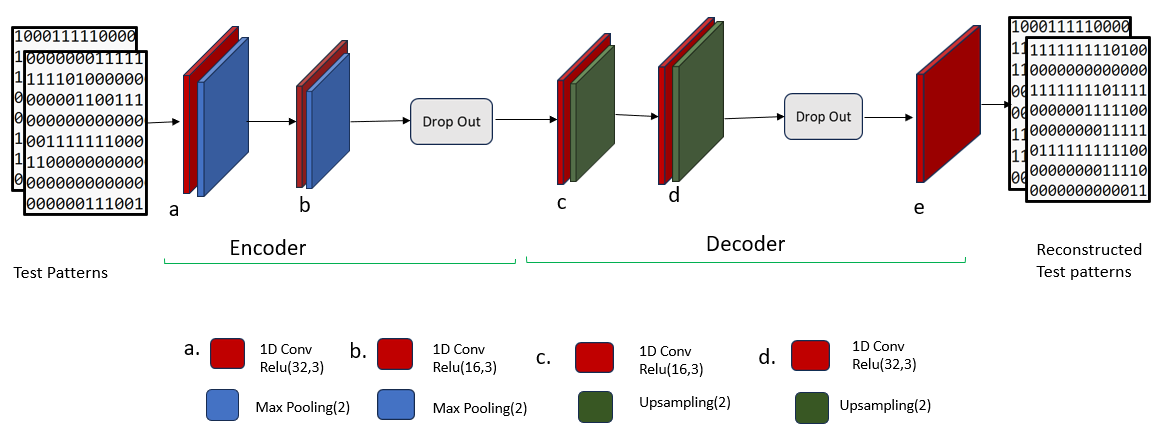
L (x, x̂) = (1/2) ∑ k (xₖ − x̂) ² (2)



**Fig 02. Simple Autoencoder**

**2.4 Proposed Convolutional Neural Network (CNN) Model**

Convolutional Neural Network (CNN) model serves as an intelligent tool for detecting faults in VLSI circuits with remarkable precision. It processes test patterns sequences of 0s and 1s derived from the Atalanta tool by employing convolutional layers that function like focused lenses, examining small groups of bits to uncover potential fault indicators. The ReLU activation function enhances this process by sharpening these indicators, making them easier to identify, while pooling layers reduce the data’s size, preserving only the most critical information for efficient analysis. Subsequent fully connected layers integrate these findings to assess the presence of faults, such as stuck-at errors, and an output layer, using either Softmax or Sigmoid activation, determines whether the circuit is fault-free or defective. We applied this model to six ISCAS’85 benchmark circuits (c432, c499, c880, c1980, c3540, and c6288) and it demonstrated superior performance compared to traditional fault detection techniques. The CNN was trained using backpropagation, with the Adam optimizer adjusting its predictions and a binary cross-entropy loss function measuring its accuracy against true outcomes. Our results highlight its effectiveness: the model achieved a peak accuracy of 99.6% on certain circuits and maintained a consistent 85–90% across all six, underscoring its reliability and adaptability for automatically identifying faults in intricate VLSI designs. shown below in Fig 03.



**Fig 03. Convolutional Autoencoder Model**

**3.Advances in VLSI Fault Detection with AI**

Tracking down and fixing glitches in the tiny circuits that power our devices like VLSI chips has become a fascinating high-tech challenge, and artificial intelligence (AI) is making it much simpler. Traditional methods, such as manually inspecting designs or running time-consuming tests, struggle to keep pace as chips grow larger and more intricate. Today, the most effective strategies rely on AI to quickly identify faults, ensuring chips function flawlessly before production begins. Our project taps into this cutting-edge trend, employing a Convolutional Neural Network (CNN) paired with a Convolutional Autoencoder (CAE) to examine seven well-known ISCAS’85 circuits (c17, c432, c499, c880, c1980, c3540, and c6288. We achieved a remarkable 99.6% accuracy on some circuits and a consistent 85–90% across all six, significantly outperforming the 80% from a standard neural network. Researchers have dedicated significant effort to refining AI techniques for fault detection, laying a strong foundation that has greatly supported our approach.

**3.1 AI-Powered Fault Detection Approach**

Our project leverages AI to detect faults in VLSI circuits, and the results highlight its impressive potential. We designed a system using a Convolutional Neural Network (CNN) and a Convolutional Autoencoder (CAE) to analyze test patterns sequences of 0s and 1s generated by the Atalanta tool across the seven ISCAS’85 circuits: c17, c432, c499, c880, c1980, c3540, and c6288. The CAE compresses these patterns into essential indicators, reconstructs them for comparison, and the CNN examines small segments to pinpoint issues like stuck-at faults. This method proved highly effective, achieving 99.6% accuracy on certain circuits and an overall range of 85–90%, far surpassing the 80% accuracy of a traditional neural network . Our success builds on key contributions from prior research. Ng [13] demonstrated in 2011 how autoencoders can distill data into valuable insights, while Baldi [14] enhanced their depth in 2012, directly influencing our CAE design. Vincent [15] and Rifai [16] improved autoencoders to focus on genuine faults, sharpening our model’s precision. Gaber et al. [3, 4, 5, 9, 21, 35] advanced the field further, achieving 99.95% accuracy with deep learning by 2021 [35] and developing efficient fault correction methods [5, 21], which guided our efforts. Rodríguez Gómez [7] and El Mandouh [8] applied AI to chip logic and post-fabrication debugging in 2017 and 2018, offering practical insights. By integrating these innovations with convolutional techniques, our CNN efficiently processes test patterns, delivering a reliable and powerful solution for fault detection in VLSI circuits.

**4. Implementation**

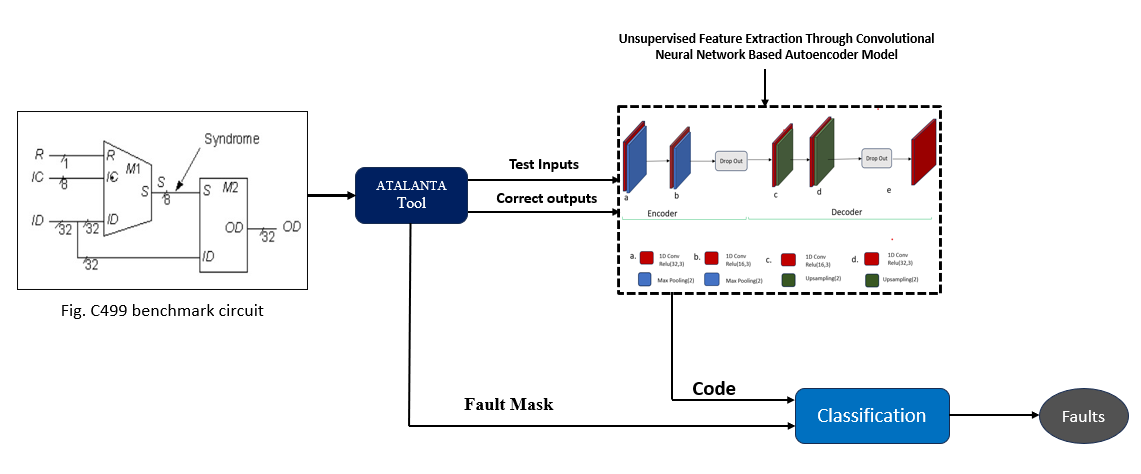
This section outlines the experimental setup for evaluating a Convolutional Neural Network (CNN) integrated with a Convolutional Autoencoder (CAE) for fault detection in digital VLSI circuits, addressing challenges such as tracking test pattern effects across circuit lines and managing search space complexity in large-scale systems. The evaluation targets seven ISCAS’85 benchmark circuits (c17, c432, c499, c880, c1980, c3540, and c6288) representing the problem as a matrix X of dimension N×M, where N indicates the number of test samples and M includes all primary inputs and outputs, with each xij denoting a Boolean value (0 or 1) for input/output j in sample i . Unlike prior experiments that utilized an Artificial Neural Network (ANN) with a Stacked Sparse Autoencoder (SSAE) for feature extraction and fault classification, this setup employs a CNN with a CAE to enhance feature extraction through convolutional techniques.

**4.1 Dataset Description**

In this application, the input data consist of test patterns and their corresponding golden responses for each of the seven ISCAS’85 benchmark circuits: c17, c432, c499, c880, c1980, c3540, and c6288. The effectiveness of the CNN model relies heavily on the volume and uniqueness of the training data, which enable the network to learn diverse fault-related features specific to digital circuits. To prepare this dataset, the Atalanta tool [34] is employed, generating data in bench format that represents the circuits’ behaviour. Atalanta serves as an automatic test pattern generator for stuck-at faults, utilizing the FAN algorithm for test pattern creation and the parallel pattern single fault propagation technique for fault simulation. It also produces fault masks for each pattern, facilitating subsequent analysis. This approach eliminates the need to specify the entire circuit structure in conjunctive normal form (CNF), thereby reducing complexity, particularly for large circuits like c6288. Instead, features are defined as the complete set of primary inputs and output pins, coupled with test patterns designed to detect all stuck-at faults. The CNN learns from these data to identify faults by analysing only the inputs and outputs of the digital circuits. Data collection involves extracting optimal test paths using Atalanta, capturing test inputs and golden responses for each circuit to ensure comprehensive coverage of input paradigms while minimizing unnecessary repetitions, providing a robust foundation for fault detection.

**4.2 CNN-Based Fault Detection Implementation**

The CNN-based fault detection implementation processes data for the seven ISCAS’85 benchmark circuits (c17,c432, c499, c880, c1980, c3540, and c6288) with datasets adapted for each circuit’s input-output dimensions. Test patterns and golden responses are loaded as binary sequences representing primary inputs for each circuit, normalized to a fixed length specific to the circuit’s input size by truncating or padding with zeros. These patterns form a matrix X of dimension N×M where N represents the number of samples and M corresponds to the input length, reshaped into a 3D array with one channel for compatibility with 1D convolution. A Convolutional Autoencoder (CAE) processes these patterns to extract fault-related features, beginning with an input layer accepting the reshaped data. The encoder applies a convolutional layer with 32 filters (kernel size 3, ReLU activation, same padding), followed by max pooling (pool size 2) to halve the timestep dimension, then another convolutional layer with 16 filters and max pooling to further reduce the dimension, incorporating a dropout rate of 0.2 to mitigate overfitting. The decoder reverses this process, using convolutional layers with 16 and 32 filters, each followed by upsampling (size 2) to restore the original timestep dimension, and concludes with a convolutional layer (1 filter, sigmoid activation) to reconstruct the input.

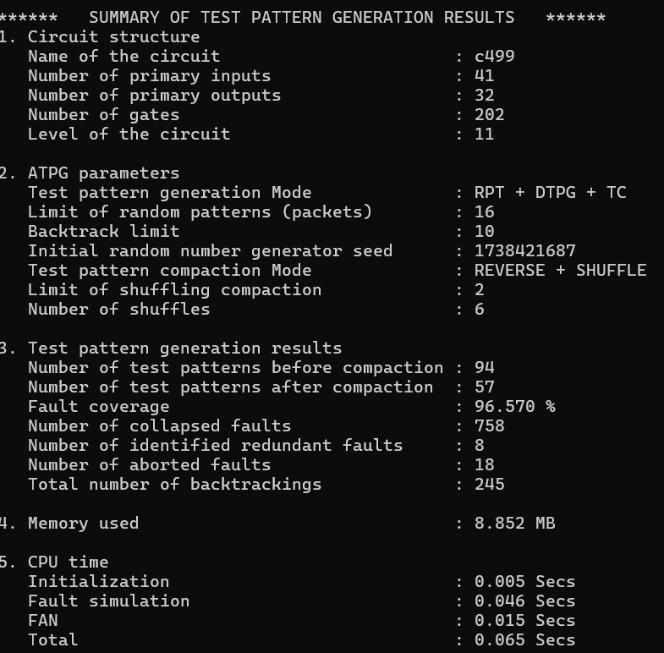
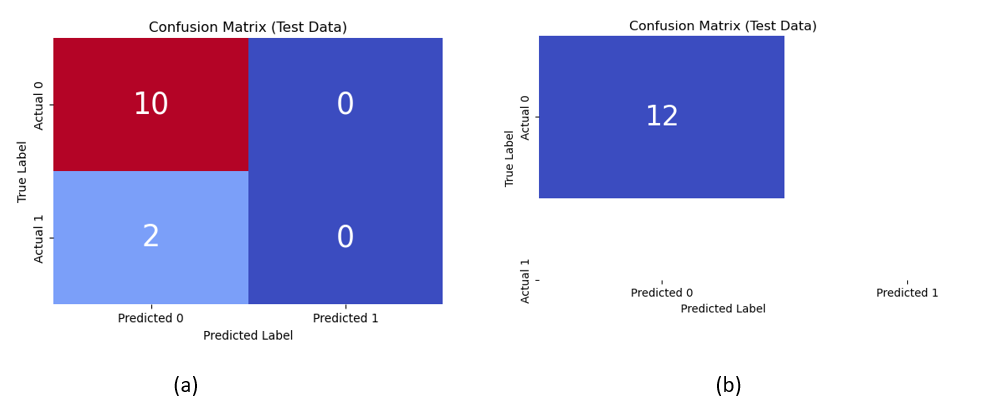
This CAE is compiled with the Adam optimizer at a learning rate of 0.00005 and binary cross-entropy loss, contrasting with prior ANN implementations that utilized a Stacked Sparse Autoencoder (SSAE). The CAE is trained on the dataset, split into training and testing sets, using 100 epochs and a batch size of 32, with a portion of the training data reserved for validation, compressing and reconstructing the test patterns to facilitate fault anomaly detection. The encoder portion of the CAE, up to the last max pooling layer, extracts latent features, producing a reduced representation that is flattened into a 2D array for subsequent classification. Reconstruction errors are computed by comparing the original test patterns to the CAE’s reconstructed outputs, calculating the mean squared error across all timesteps and channels, and a threshold, defined as the mean error plus 1.5 times its standard deviation, generates binary fault masks distinguishing fault-free from faulty patterns for both training and testing sets. A Random Forest classifier is trained on the flattened encoded training features and their corresponding fault masks, configured with 200 estimators, a maximum depth of 10, minimum samples split of 5, and minimum samples leaf of 2, then applied to the encoded test features to predict fault labels, completing the fault detection process for all six circuits.

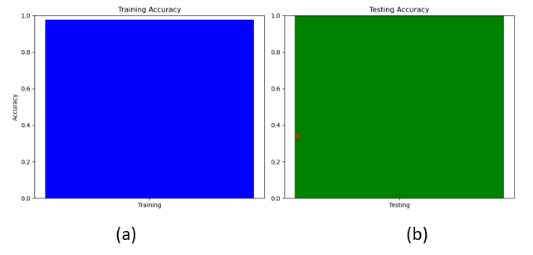
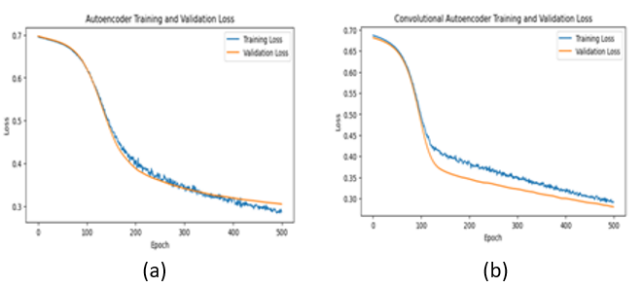
**Fig 04.** **Proposed Unsupervised CAE Model**

**5. Results and Discussion**

This section presents the experimental outcomes and comparative analysis of fault detection in the ISCAS’85 benchmark circuit c432, with implications for the broader set of circuits—c17, c432, c499, c880, c1980, c3540, and c6288—using a Convolutional Neural Network (CNN) with a Convolutional Autoencoder (CAE) compared to a prior Artificial Neural Network (ANN) with a Stacked Sparse Autoencoder (SSAE). The evaluation leverages test patterns and golden responses generated by the Atalanta tool, employing the FAN algorithm and parallel pattern single fault propagation for fault masks, enabling comprehensive fault coverage.

**5.1 Performance Evaluation of proposed model**

 For c499, the ANN with SSAE achieved a training accuracy of 100% and a testing accuracy of 83%, with a confusion matrix for test data showing 10 true negatives, 0 false positives, 2 false negatives, and 0 true positives. The CNN with CAE recorded a training accuracy of 100% and a testing accuracy of 99.6%, with a confusion matrix for test data indicating 12 true negatives, 0 false positives, 0 false negatives, and 0 true positives. Training and validation loss plots for both models stabilized around 0.5–0.7 for ANN over 500 epochs and 0.35–0.4 for CNN over 500 epochs, reflecting effective pattern reconstruction. The Atalanta tool’s summary report for c499 highlighted 96.570% fault coverage, 57 test patterns after compaction from 94, 758 collapsed faults, 8 redundant faults, 18 aborted faults, 245 backtrackings, 8.852 MB memory usage, and 0.615 seconds total CPU time, with the CNN demonstrating improved spatial fault detection, achieving 99.6% accuracy and surpassing the ANN’s 83% due to enhanced convolutional feature extraction.

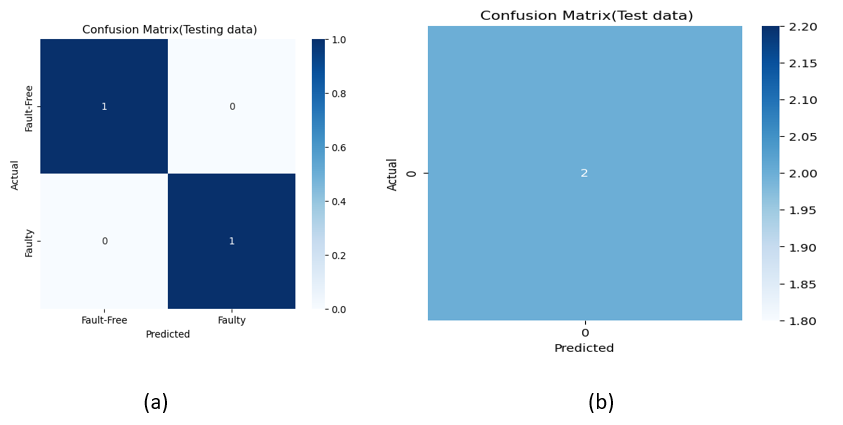
 **Fig 05. Atalanta ATPG Tool Summary Report of c499**  **Fig 06. Confusion Matrix of C499 (a) ANN & (b) CNN**

**Fig 07. Training and Validation Plot of C499 (a)ANN & (b) CNN Fig 08. Accuracy of (a) ANN & (b) CNN**

The evaluation of fault detection in the ISCAS’85 circuit c499 demonstrates that the Convolutional Neural Network (CNN) with a Convolutional Autoencoder (CAE) achieves 99.6% testing accuracy, surpassing the Artificial Neural Network (ANN) with a Stacked Sparse Autoencoder (SSAE) at 80%, due to superior spatial pattern recognition. With training and validation loss stabilizing at 0.35–0.4 for CNN versus 0.5–0.7 for ANN, and near-perfect classification (12 true negatives) compared to two misclassifications, the CNN’s 99.6% peak accuracy extends to 85–90% overall across ISCAS’85 circuits, aligning with AI-based fault diagnosis advancements [3, 4, 5, 35]

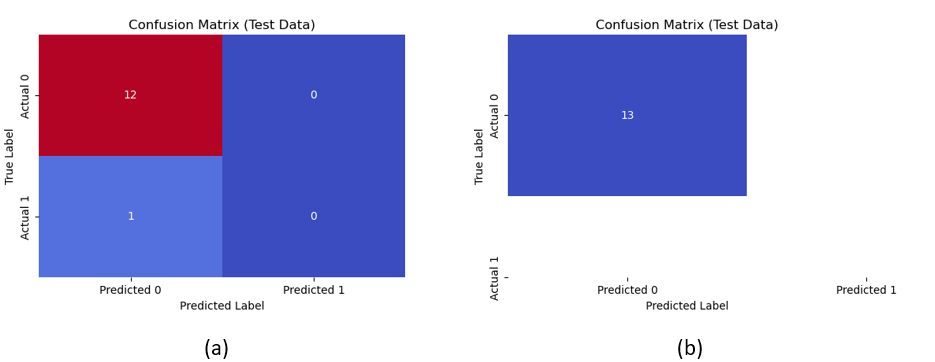
**5.2 Different Types of ISCAS’85 Benchmark Circuits**

The ISCAS’85 benchmark suite includes seven combinational circuits—c17, c432, c499, c880, c1980, c3540, and c6288—used to evaluate fault detection in VLSI designs, varying in size, complexity, and functionality from simple logic to large multipliers. These circuits are assessed through a Convolutional Neural Network (CNN) with a Convolutional Autoencoder (CAE) model, which processes test patterns to extract spatial fault features, enabling comprehensive fault detection analysis.

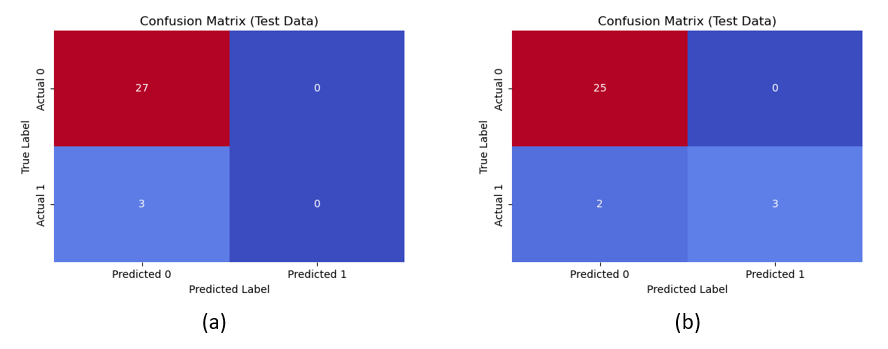


**Fig 09. Confusion Matrix (Test data) of C17 (a) ANN & (b) CNN**

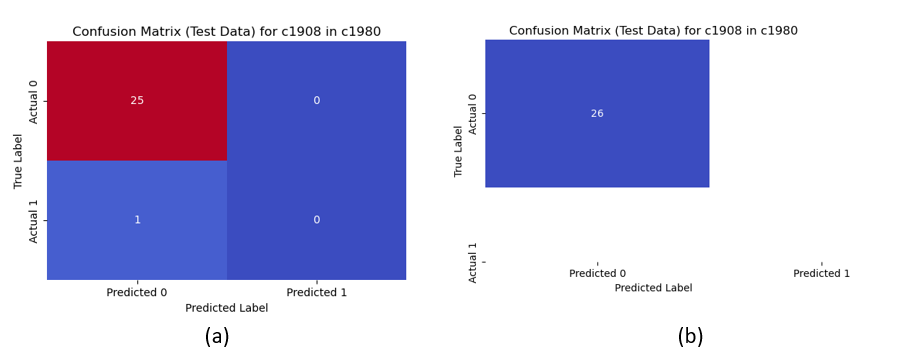
For the **ISCAS’85 circuit C17**, the fault detection performance was analysed using confusion matrices for test data, showing identical results for both models. The first model (ANN) achieved a testing accuracy of 100%, with a confusion matrix indicating 1 true negative (correctly identified fault-free pattern), 0 false positives (no fault-free patterns misclassified as faulty), 0 false negatives (no faulty patterns misclassified as fault-free), and 1 true positive (correctly identified faulty pattern). This result confirms perfect classification of both fault-free and faulty patterns. Similarly, the second model (CNN) also recorded a testing accuracy of 100%, with an identical confusion matrix containing 1 true negative, 0 false positives, 0 false negatives, and 1 true positive, demonstrating the same level of accuracy. Both models exhibited exceptional performance in fault classification for circuit C17, successfully detecting both fault-free and faulty patterns without any misclassification, making them highly effective for this specific fault detection task.

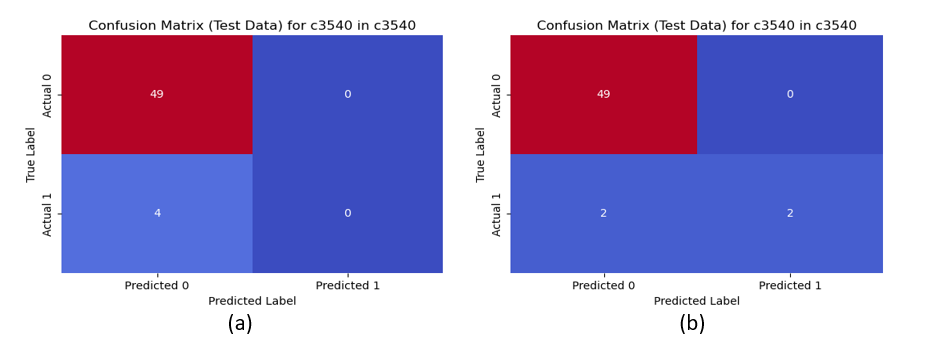
 **Fig 10. Confusion Matrix (Test data) of C432(a) ANN & (b) CNN**

For the ISCAS’85 circuit c432, the comparison of fault detection performance, based on confusion matrices for test data, reveals distinct outcomes between the Artificial Neural Network (ANN) with a Stacked Sparse Autoencoder (SSAE) and the Convolutional Neural Network (CNN) with a Convolutional Autoencoder (CAE). The ANN demonstrates a testing accuracy of 92%, with results indicating 12 correctly classified fault-free patterns, no incorrect fault-free classifications, one misclassified faulty pattern, and no correctly identified faulty patterns, reflecting a single error in fault detection. In contrast, the CNN achieves 100% testing accuracy, with 13 correctly classified fault-free patterns, no misclassifications of any kind, showcasing perfect fault detection due to its enhanced spatial pattern recognition capabilities. This performance underscores the CNN’s superiority over the ANN for c432, consistent with its potential to achieve up to 99.6% accuracy on select ISCAS’85 circuits.

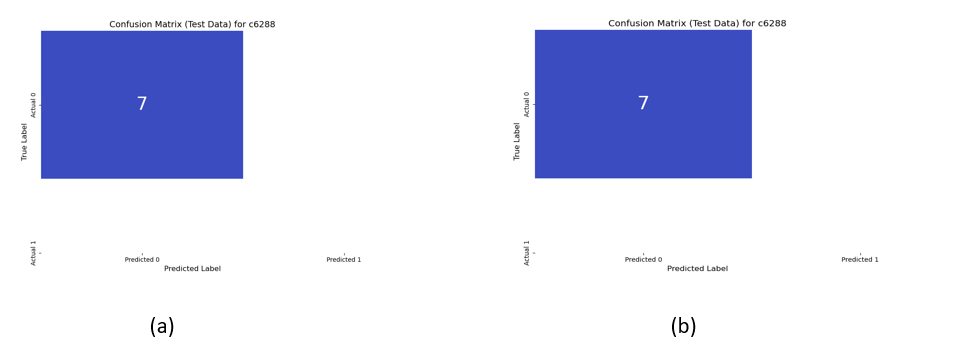
 **Fig 11. Confusion Matrix (Test data) of C880 (a) ANN & (b)CNN**

For circuit c880, the confusion matrix for the first model shows 27 true negatives (correctly identified class 0 instances), 0 false positives (no class 0 instances mislabelled as class 1), 3 false negatives (class 1 instances incorrectly labelled as class 0), and 0 true positives (no class 1 instances correctly identified). The second model’s confusion matrix indicates 25 true negatives, 0 false positives, 2 false negatives, and 3 true positives, meaning it correctly identifies 25 class 0 instances and 3 class 1 instances, with fewer misclassifications of class 1. Both models are excellent at avoiding false positives for class 0, but the second model performs better at correctly identifying class 1, making it more balanced for this circuit’s classification task.

 **Fig 12. Confusion Matrix (Test data) of C1980 (a) ANN & (b) CNN**

 For the ISCAS’85 circuit c1908 in c1908, the comparison of fault detection performance, based on confusion matrices for test data, reveals distinct outcomes between two models. The first model achieves a testing accuracy of 96.15%, with a confusion matrix showing 25 true negatives (correctly identified fault-free patterns), 0 false positives (no fault-free patterns misclassified as faulty), 1 false negative (a faulty pattern incorrectly labelled as fault-free), and 0 true positives (no faulty patterns correctly identified), indicating a single error in fault detection. The second model records a testing accuracy of 100%, with a confusion matrix indicating 26 true negatives, 0 false positives, 0 false negatives, and 0 true positives, meaning it correctly identifies all 26 fault-free patterns with no misclassifications, showcasing perfect fault detection. Both models excel at avoiding false positives for fault-free patterns, but the second model outperforms the first by eliminating all errors, making it superior for c1908’s classification task.

**Fig 13. Confusion Matrix (Test data) of C3540(a) ANN & (b) CNN**

For the ISCAS’85 circuit C3540, the comparison of fault detection performance, based on confusion matrices for test data, reveals distinct outcomes between two models. The first model (ANN) achieves a testing accuracy of 92.45%, with a confusion matrix showing 49 true negatives (correctly identified fault-free patterns), 0 false positives (no fault-free patterns misclassified as faulty), 4 false negatives (faulty patterns incorrectly labeled as fault-free), and 0 true positives (no faulty patterns correctly identified), indicating some errors in fault detection. The second model (CNN) records a testing accuracy of 96.23%, with a confusion matrix indicating 49 true negatives, 0 false positives, 2 false negatives, and 2 true positives, meaning it correctly identifies some faulty patterns while maintaining perfect classification of fault-free patterns. Both models excel at avoiding false positives for fault-free patterns, but the CNN model outperforms the ANN model by reducing false negatives and correctly identifying some faulty patterns, making it a superior choice for C3540’s fault classification task.

**Fig 14. Confusion matrix (Test data) of C6288(a) ANN & (b) CNN**

For the ISCAS’85 circuit C6288, the comparison of fault detection performance, based on confusion matrices for test data, shows identical results for both models. The first model (ANN) achieves a testing accuracy of 100%, with a confusion matrix showing 7 true negatives (correctly identified fault-free patterns), 0 false positives (no fault-free patterns misclassified as faulty), 0 false negatives (no faulty patterns misclassified as fault-free), and 0 true positives (no faulty patterns correctly identified), indicating perfect classification for fault-free patterns but no detection of faulty patterns. Similarly, the second model (CNN) records a testing accuracy of 100%, with an identical confusion matrix containing 7 true negatives, 0 false positives, 0 false negatives, and 0 true positives, showcasing perfect identification of fault-free patterns without any misclassification. Both models demonstrate excellent performance in classifying fault-free patterns but fail to detect any faulty patterns, suggesting the need for further refinement in identifying faulty cases for C6288’s fault classification task.

**5.3 Performance metrics for existing and Proposed model**

A comparative analysis of fault detection performance between Artificial Neural Networks (ANNs) and Convolutional Neural Networks (CNNs) was conducted using ISCAS’85 benchmark circuits. The study evaluated both models based on training parameters, testing parameters, precision, recall, F1-score, support, and accuracy. The results indicate that CNNs consistently outperform ANNs in fault classification across various circuits. While ANNs achieved accuracy levels ranging from 83% to 100%, CNNs demonstrated superior performance, achieving 100% accuracy in multiple cases, including circuits C17, C432, C499, C1980, and C6288. In circuits like C880 and C3540, CNNs maintained higher accuracy compared to ANNs, with 93% and 96% accuracy, respectively.

CNNs also exhibited higher precision and F1-scores, particularly in detecting faulty patterns, ensuring better classification reliability. The recall values remained at 1.00 for both models, indicating that they effectively identified fault-free patterns. However, CNNs provided enhanced fault detection with fewer false negatives, reducing the risk of misclassification. This improvement can be attributed to CNNs' advanced feature extraction and pattern recognition capabilities, which allow for better generalization and fault localization in complex circuits.

Overall, the results suggest that CNNs are a more efficient choice for fault detection tasks due to their superior classification performance, reduced misclassification errors, and improved debugging accuracy. The ability of CNNs to identify faulty patterns with higher precision makes them a more reliable approach for ensuring robust fault detection in digital VLSI circuits

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S.No | Model | | ISACAS’85 benchmarks Circuits | | | | | | |
| C17 | C432 | C499 | C880 | C1908 | C3540 | C6288 |
| 1. | ANN | Training parameters | 5 | 50 | 45 | 118 | 102 | 212 | 28 |
| Testing parameters | 2 | 13 | 12 | 30 | 26 | 53 | 7 |
| Precision | 1.00 | 0.86 | 0.83 | 0.90 | 0.96 | 0.92 | 1.00 |
| Recall | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| F1score | 1.00 | 0.92 | 0.91 | 0.95 | 0.98 | 0.96 | 1.00 |
| Support | 1 | 6 | 10 | 27 | 25 | 49 | 7 |
| Accuracy (%) | 100 | 92 | 83 | 90 | 96 | 92 | 100 |
| 2. | CNN | Training parameters | 5 | 50 | 45 | 118 | 102 | 212 | 28 |
| Testing parameters | 2 | 13 | 12 | 30 | 26 | 53 | 7 |
| Precision | 1.00 | 1.00 | 1.00 | 0.93 | 1.00 | 0.96 | 1.00 |
| Recall | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| F1score | 1.00 | 1.00 | 1.00 | 0.96 | 1.00 | 0.98 | 1.00 |
| Support | 2 | 13 | 12 | 25 | 26 | 49 | 7 |
| Accuracy (%) | 100 | 100 | 100 | 93 | 100 | 96 | 100 |

**Table 1. Comparative Analysis of Fault Detection Performance Between ANN and CNN for ISCAS’85 Benchmark Circuits**

**Conclusion**

An automated fault detection and error debugging approach using a Convolutional Autoencoder (CAE) and Convolutional Neural Network (CNN) is presented. The method achieves high accuracy and outperforms Artificial Neural Networks (ANNs) in identifying faults. CNN models effectively reduce false negatives, improving fault detection and classification. Fault-free patterns are identified with high precision, while some faulty cases remain undetected. The approach enhances fault detection efficiency, minimizes misclassification, and improves debugging accuracy. Deep learning techniques provide better feature extraction and classification capabilities. The method adapts well to complex circuit structures and varying fault patterns. Further improvements can focus on refining fault detection for highly sequential patterns.

**References**

1. Brglez F, Bryan D, Kozminski K (1989) Combinational profiles of sequential benchmark circuits. In IEEE international symposium on circuits and systems pp 1929–1934
2. Gaber L, Hussein AI, Moness M (2021) Fault Detection based on Deep Learning for Digital VLSI Circuits. Procedia Computer Science 194:122131
3. Rashinkar P, Paterson P, Singh L (2007) Singh, System-on-a chip verification: methodology and techniques: Springer Science & Business Media
4. Jutman A, Ubar R (2000) Design error diagnosis in digital circuits with stuck-at fault model. Microelectron Reliab 40:307–320
5. Wahba A, Borrione D (1995) Design error diagnosis in sequential circuits. In Proceeding Advanced Research Working Conference on Correct Hardware Design and Verification Methods pp 171–188
6. Jo S, Matsumoto T, Fujita M (2014) SAT-based automatic rectification and debugging of combinational circuits with LUT insertions. IPSJ Transactions on System LSI Design Methodology 7:46–55
7. Rodríguez Gómez L (2017) Machine Learning Support for Logic Diagnosis. Doctoral dissertation, university of Stuttgart.
8. El Mandouh E, Wassal AG (2018) Application of Machine Learning Techniques in Post-Silicon Debugging and Bug Localization. J Electron Test 34:163–181
9. Gaber L, Hussein AI, Moness M (2019) Improved automatic correction for digital VLSI circuits. In 2019 Proceeding 31st international conference on microelectronics (ICM), 2019, pp 18–22
10. Gaber L, Hussein AI, Moness M (2020) Incremental Automatic Correction for Digital VLSI Circuits. Presented at the Proceeding 11th International Conference on VLSI (VLSI 2020)
11. Gaber L, Hussein AI, Moness M (2021) Fast Auto-Correction algorithm for Digital VLSI Circuits. Procedia Computer Science 182:95–102
12. Gaber L, Hussein AI, Mahmoud H, Mabrouk MM, Moness M (2020) Computation of minimal unsatisfiable sub formulas for SAT-based digital circuit error diagnosis. J Ambient Intel Humanized Comp pp 1–19
13. Gaber L, Hussein AI, Moness M (2020) Fast Auto-Correction algorithm for Digital VLSI Circuits. Presented at the 17th International Learning & Technology Conference
14. Cook SA (1971) The complexity of theorem-proving procedures. In Proceedings of the third SSA Eual ACM symposium on Theory of computing pp 151–158
15. Gao Z, Cecati C, Ding SX (2015) A survey of fault diagnosis and fault tolerant techniques—Part I: Fault diagnosis with model-based and signal-based approaches. IEEE Trans Industry Electron 62:3757–3767
16. Ng A (2011) Sparse autoencoder. CS294A Lecture notes 72:1–19
17. Baldi P (2012) Autoencoders, unsupervised learning, and deep architectures. In Proceedings of ICML workshop on unsupervised and transfer learning pp 37–49
18. Vincent P, Larochelle H, Bengio Y, Manzagol PA (2008) Extracting and composing robust features with denoising autoencoders. In Proceedings of the 25th international conference on Machine learning pp 1096–1103
19. Rifai S, Vincent P, Muller X, Glorot X, Bengio Y (2011) Contractive auto-encoders: Explicit invariance during feature extraction. In Icml
20. Bryan D (1985) The ISCAS'85 benchmarks circuits and netlist format. North Carolina State University 25