CINEFORUM
ISSN: 0009-7039
Vol. 66. No. S6, 2026 (Special Issue 6)

EM Side-Channel Based Trojan Detection Using Deep SVDD For Unsupervised Anomaly Identification

Yasala Charishma Raghavi[1*], Mr. L. Chandra Shekhar[2*], Dr. Miriampally Venkata Raghavendra[3*]

[1]Student of MTech in VLSI System Design, Department of Electronics and Communication Engineering, Sree Dattha Institute of Engineering and Sciences, Hyderabad, Telangana, India.
[2]Assistant Professor, Department of Electronics and Communication Engineering, Sree Dattha Institute of Engineering and Sciences, Hyderabad, Telangana, India.
[3]Associate Professor, Department of Electronics and Communication Engineering, Sree Dattha Group of Institutions, Hyderabad, Telangana, India.

Corresponding Author
Yasala Charishma Raghavi
Email Id: ycraghavi@gmail.com

Abstract
Integrated-circuit design and fabrication worldwide have brought about greater chances of malicious manipulation of hardware functionalities that might not be detected through standard functional test. In this paper, an electromagnetic (EM) side-channel platform on Deep Support Vector Data Description (Deep SVDD) is used to detect hardware- Trojan. The technique measures the non-invasive EM emissions of a test device, conditions and digitalizes the signal, and processes the signal with Hann windowing and Fast Fourier Transformer analysis, and calculates the magnitude-squared spectrum. Strong sibling spectral peaks around the location of operating-clocks are picked up in order to create a small feature-vector. Deep SVDD uses exclusively golden, Trojan-free features with which it learns a concise latent-space representation of the normal behaviour. Online functionality When a new feature vector is mapped with the trained model their squared distance to the learned centre is contrasted against the threshold to yield a normal-or-Trojan decision. MATLAB facilitates the preparation of data and offline training of a model, whereas the hardware-oriented processing chain is modelled using the Verilog modules of FFT interfacing, power-spectrum generation, peak extraction, anomaly scoring, and top-level detection. The architecture is not reliant on labelled Trojan samples, and allows the detection of new structural changes that were never seen before. The feasibility of implementation, functional validation requirements, constraint and directions of quantitative FPGA and ASIC benchmarking are also discussed.
Keywords—Electromagnetic side channel; hardware Trojan; Deep SVDD; one-class classification; FFT; anomaly detection; Verilog.

1. Introduction

Current integrated circuit development is typically based on a distributed supply chain consisting of third party intellectual-property core, outsourced manufacturing, third party design and post-silicon assembly. This type of model is cheaper and less time consuming in development but it also leaves a chance of an undue circuit modification. Hardware Trojan can induce a modification of functionality, sensitive information disclosure, lowered reliability or a payload denial-of-service only by a trigger event meeting a infrequent condition [1]-[3]. Due to the fact that the logic, when inserted can be very small and intentionally dormant, normal production tests might not reveal it.

Detection Many hardware-Trojan-based solutions fall into the categories of pre-silicon and post-silicon hardware-Trojan detection. Formal verification, structural verification, logic testing, and scan-based analysis may be important prior to or immediately after fabrication, but require reliable specifications, and the control of internal nodes as well as access to appropriate test patterns. Destructive reverse engineering is very confidence and expensive, time-consuming and cannot be used to screen all the devices. These restrictions have inspired the facilities of physical side channels that can expose internal switching activity without necessarily having to access circuit nodes.

Signatures based on power-consumption and timing-delay have been extensively researched to this end [4]-[6]. Power analysis experiences aggregate current variation, but the activity of a big design can obliterate the contribution of a small Trojan. The changes of path can be detected through timing analysis, but process, voltage, temperature and aging changes can result in the same changes as a Trojan. Electromagnetic measurement Because a near-field probe can measure more local switching behaviour, and can be repositioned across areas of interest, it is an appealing technique [7], [8].

The main challenge is that EM measurements are high-dimensional noisy. Raw waveforms are dependent on the

position of the probe, clocking versus offset, work load, environmental interactions, and the acquisition hardware. The practical detector will consequently need a repeatable signal-processing chain, which prioritizes the stable spectral data and eliminates unwarranted variation. The frequency-domain process can easily be applied to clocked digital systems since periodic switching-activity will manifest itself as strong peaks around the operating clock and its harmonics.

The second challenge is that there is limited representative data on Trojans. When supervised classifiers train on the same families of Trojan observed in the testing phase, they can be effective, but they might not be in case a new trigger, payload, placement or activation pattern is observed. In one-class learning, the dependency of one-class learning is abated by changing the modelling to just trusted behaviour and then the deviations are viewed as anomalies. Autoencoders and deep one-class models have been proposed, thus emerging as significant alternatives to the one-class SVM [9]–[12].

It is a combination of EM spectral analysis and Deep SVDD which is a deep one-class model where the normal features are mapped to a low-dimensional hypersphere in the latent space [10]. The given flow divides computationally-heavy offline training steps with the lightweight online inference ones. The golden-data model is created with MATLAB and the hypersphere centre and decision threshold are derived; the online path is designed to run with Verilog. Its contributions include a non-invasive EM acquisition workflow, reduced spectral feature representation, normal-only Deep SVDD training, as well as a how to do hardware-based anomaly decision mechanism.

2. Related Work

A popular taxonomy of hardware Trojans was introduced by Tehranipoor and Koushanfar [1] and categorized the Trojans based on physical features, activation methods and payload behaviour. Karri et al. also delineated the issue of trust in contemporary hardware, and the challenge of detecting malicious logic that happens to be silent in likely tests [2]. The initial post-silicon studies were based on logic activation, scan access, delay fingerprints as well as current signatures [3]-[6].

Multiple-parameter side-channel analysis enhances robustness since it provides multiple physical effects with which to analyze the information [4]. Aggregate power and path delay are, however, dependent upon process and environmental variation. EM-based techniques enhance spatial sensitivity and have the capability of revealing changes in switching activities at the local level. EM fingerprints, comparison using golden chips, and localized leakage have been previously investigated in [7], [8], [18], [19], [21], [26], and in detecting suspicious hardware behaviour.

Conventional machine-learning detectors are implemented with Support Vector Machines, neural networks, extreme-learning machine, or other similar classifiers and engineered side-channel features. These methods can effectively isolate known classes, but define representative labels to clean and infected devices. Surveys of machine-learning-based Trojan detection point to the lack of data, domain shift, diversity in measurements, and inadequate cross-design generalization as common pitfalls [14], [15], [20], [22].

In the case where trusted devices only exist, novelty detection formulates a more appropriate formulation. One-Class SVM learns a boundary around normal data [9] whereas anomalies detected on auto associative networks and autoencoders with reconstruction error [11], [12], [28]-[30]. When over-parameterized or under-regularized, reconstruction-based models do however have the ability to give rise to anomalous inputs.

Deep SVDD explicitly reduces the size of a hypersphere which encloses deep represences of normal samples [10]. Its goal compliments that of hardware trust assessment since the training phase does not need to go through many instances of all possible malicious modifications. Other recently studied topics include transfer learning, LSTM autoencoders, and other deep models in detecting Trojans [13], [16], [17], [24], [25]. The current scheme uses Deep SVDD as its distance based score has a straight forward interpretation and this can be reduced to hardware inference.

3. Research Gap and System Overview

Raw EM analysis is also computationally and storage intensive and the analysis using all spectral bins is also prone to lower-generalization as it retains noise-contamination factors.

The proposed framework fails to close these gaps because it makes use of only golden EM signatures to learn, and the proposed retention of a small number of dominant spectral peaks around the operating-clock region. Identical definition of features is applied when training offline and detecting online to avoid an incompatibility between the processing chain that is described in Verilog and the MATLAB model. The declaration of an anomaly start with a sample being an outlier to the learned-space distance over the threshold value, which is determined by using normal training samples.

The traditional flow depicted below depicts how most of the previous systems were based on comparison of

reference and supervised classification. Conversely, the proposed detector adopts for one-class normality modelling instead of labelled normal-versus-Trojan training problem.
The current techniques are often based on a golden chip or a finished reference model, or labelled exhibits of Trojan-infected behaviour. Outsourced and changing supply chains find it hard to meet these assumptions

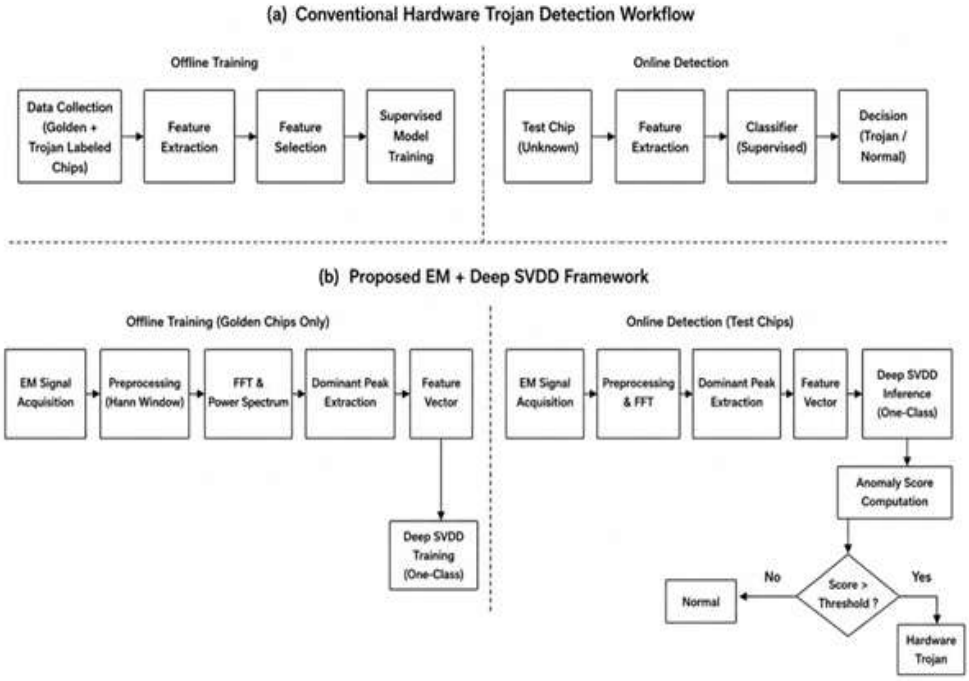

Figure 1. Conventional hardware-Trojan detection workflow based on feature extraction, supervised model training, and diagnostic reporting.

Table 1. Qualitative comparison of hardware-Trojan detection strategies.

Approach	Training/Reference Requirement	Major Limitation	Suitability for Unknown Trojans
Functional and scan testing	Expected responses and activation-oriented test vectors	Rare triggers may remain inactive; test-space growth is severe	Low
Power or timing side channel	Golden comparison or calibrated statistical model	Sensitive to process, voltage, temperature, workload, and noise	Moderate
Supervised machine learning	Labelled normal and Trojan samples	Generalization is limited when Trojan structures differ from training data	Moderate
Proposed EM + Deep SVDD	Golden EM features only	Requires controlled acquisition and threshold calibration	High, because deviations are detected without Trojan labels

4. Proposed Methodology

The attack suggestions adopt the use of the electromagnetic signal-based Hardware Trojan detection system, which is a mix of signal acquisition, digital pre-processing, feature computations, offline unsupervised model training, and online anomaly detecting. The entire system is programmed to be divided into five significant steps: EM signal acquisition, signal pre-processing, generation of feature vectors, Deep SVDD training model, and online detection. The core aim is to differentiate between a golden hardware instance, and a Trojan-planted hardware instance, by learning the normal electromagnetic behaviour of the target circuit, and then detecting anomalies in the learned target behaviour.

The current implementation adheres to an unsupervised scheme compared to traditional methods of supervision, which involve both normal and Trojan-labelled data in the course of training.
The reference model is built based on only the golden or normal electromagnetic signatures. Any alteration as to this trained regular distribution is considered a possible anomaly and thus, unknown or previously unidentified Hardware Trojans can be detected.

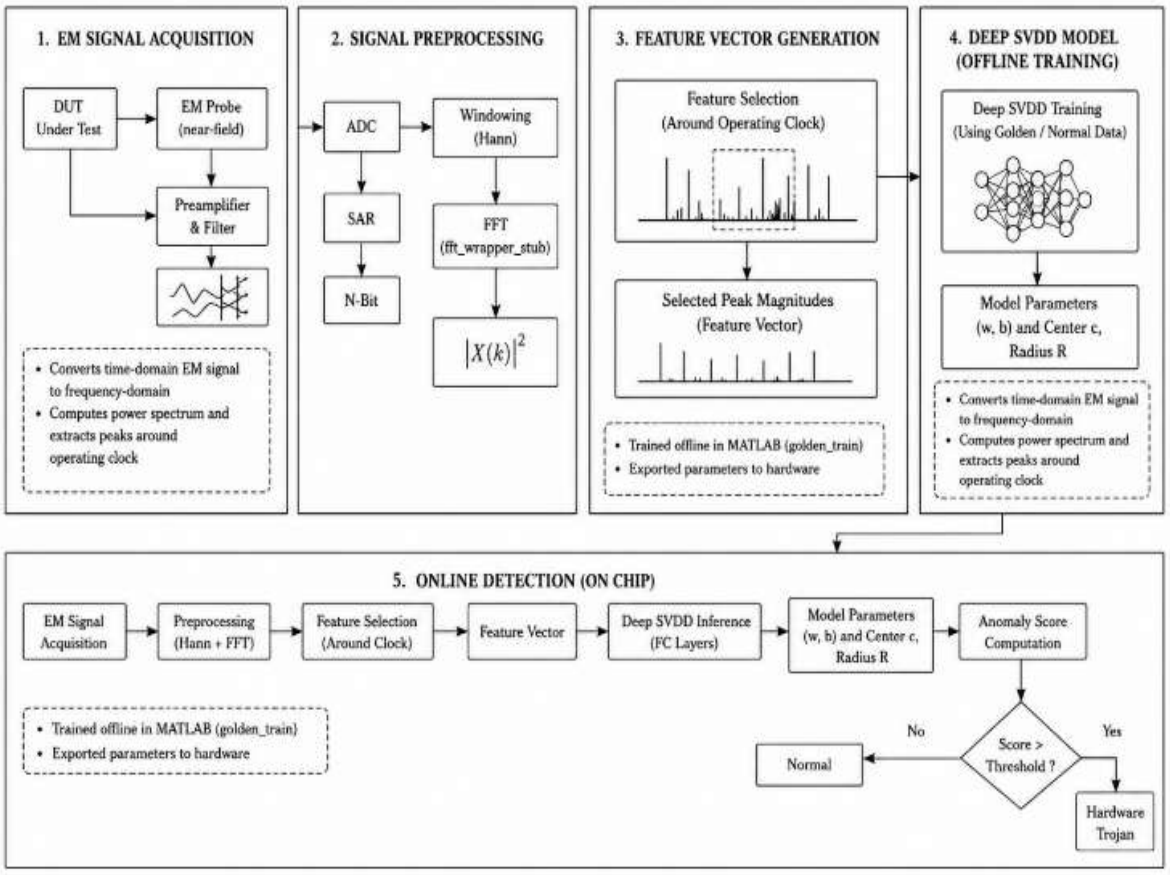

Figure 2: Proposed EM-Based Hardware Trojan Detection Framework (Deep SVDD-Based Implementation)

Stage 1: EM Signal Acquisition

The electromagnetic radiation of the Device Under Test (DUT) is observed in the first step when the device is operated under controlled conditions and via a near-field EM probe. The signal recorded at the point of observation is the side-channel behaviour of the hardware, and includes information of interest regarding internal switching behaviour, timing behaviour and structural alterations of Trojan circuitry.

EM emanation is beneficial as it is non-invasive and does not engage in direct electrical contact with internal nodes. Simple hardware changes can change switching transitions, current paths and density of activity and this change can be captured by the emitted electromagnetic signature. Thus, the EM signal is not a direct yet very informative measure of the internal state of the hardware.

The probe signal is generally good and corrupted with environmental noise. Therefore, it is amplified by preamplifier and filter. The amplifier enhances the signal strength that can be measured whereas the filter suppresses interference out-of-band and enhances the quality of the signal. This step of conditioning is necessary so that the following blocks of digital data are going to work with more informative and cleaner signal.
The result of this step is then an analog time-domain EM waveform, which is:

$$x(t)$$

x(t) is electromagnetic signal obtained with respect to time.

This step is the backbone of the overall detecting implementations. When the quality of the signal acquired is poor, then the following stages will be compromised. Thus, the correct positioning of the probes, clock stability, and consistency of measurements, and analog conditioning strongly contribute to achieving stable signatures on

the golden, as well as on the Trojan-edited circuit.

Stage 2: Signal Pre-processing

Once acquired, the analog EM signal is converted in an ADC/SAR based conversion path to obtain a representation in discrete time, which can be processed digitally. The signal is initially found in the time domain, so they need to convert it to a more interpretable form to be analysed. To this end, windowing then Fast Fourier Transform (FFT) are used.

Before performing FFT they use the Hann window so that the spectral leakage decreases. In real-world operations the sampled waveform will have a finite duration, the edges will have discontinuities and the spectral components, when abruptly truncated, may spread unwantedly. The frequency representation can be smoothed to make the representation more stable and enhance peak discrimination. The FFT block will then compute the frequency spectrum. Suppose sampled sequence is x [n], its frequency-domain representation is given as:

$$X[k] = \sum_{n=0}^{N-1} \quad x[n]e^{-j2\pi kn/N} \qquad (1)$$

where $N$is the number of samples, and $X[k]$is the complex-valued spectrum at the k-th frequency bin.

However, for hardware Trojan analysis, the phase component is not the primary concern. The implementation focuses mainly on the energy or strength of spectral components. Therefore, the FFT output is passed through the magnitude squared block, which computes the spectral power:

$$P[k] = | X[k] |^2$$

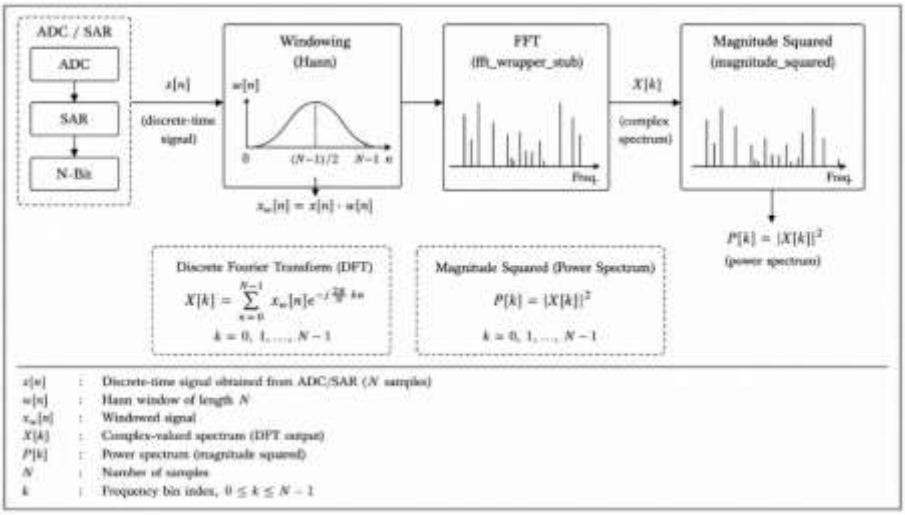

Figure 3: Signal Pre-processing Stage: ADC, windowing, FFT, and magnitude-squared operations used to convert raw EM waveforms into power spectral representations.

This change is significant as Hardware Trojans do not necessarily provide a new spectral line, but in many cases modify the load of the power around the frequencies of interest. In this way, the power spectrum would be a powerful feature space to differentiate between normal and anomalous hardware behaviour.

This step transforms the raw measurements into a form of the structured spectral form. The process of the pre-processing flow cancels time-domain anomalies as well as accentuates frequency-domain behaviours that correlate with switching activity of the DUT. The block related to the FFT in this implementation is projected onto the Verilog module fft-wrapper-stub, with magnitude squared doing the power calculation. The combination of these blocks prepares data to meaningfully extract features.

Stage 3: Feature Vector Generation

Such full power spectrum has a large number of frequency bins, yet only a few of them are fully applicable in detecting Trojans. Thus, the system does not feed the entire spectrum in the anomaly detector, but it does feature selection. The features chosen are the ones that align with the highest magnitudes around the operating clock and its important neighbouring components.

This method is motivated by the fact that Trojan insertion will impact internal switching behaviour and the cycles of repeated operations that will consequently impact distribution of EM energy around the clock-related spectral

region. The implementation minimizes the dimensionality as the mostly discriminatory information is retained by focusing on these prominent peaks.

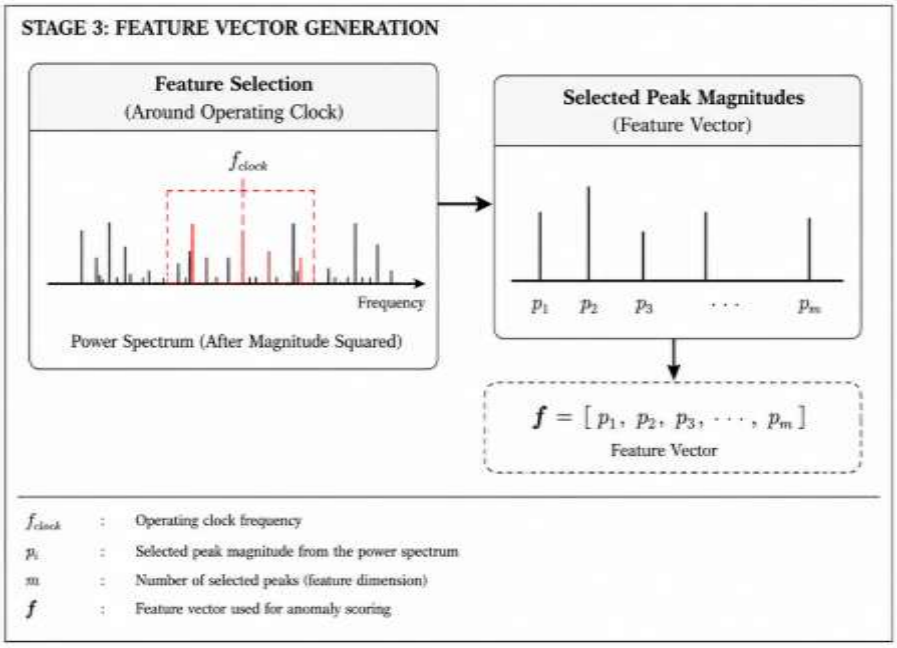

Figure 4: Feature Vector Generation Stage: selection of spectral peaks around the operating clock and formation of the reduced-dimensional feature vector.

Assume the spectral peaks chosen to be represented as:

$$f = [p_1, p_2, p_3, \dots, p_m]$$

where f is the feature vector and $p_1, p_2, \dots, p_m$ are the chosen peak magnitudes from the power spectrum.

The resulting feature vector is compact, stable, and suitable for both offline training and online classification. It also reduces memory and computation requirements, which is especially beneficial for hardware implementation. In your Verilog design, the block peak_extractor identifies important peaks, and the selected values are used to form the final vector that is passed to the scoring mechanism.

The interface between machine learning and signal processing is presented in this stage. The approach avoids direct use of raw signals, but constructs an informative vector to represent the EM signature of the hardware in a small mathematical description. More than just enhancing the performance of classification it also synchronizes the MATLAB training flow to that of the online implementation of the Verilog code.

Stage 4: Deep SVDD Model Training

The fourth stage is the offline training stage which is implemented in MATLAB with the golden_train data. In this case, feature vectors are only of golden or Trojan-free. This is not to learn the characteristics of Trojan circuits directly, but to learn a succinct description of normal hardware behaviour.

To this end, Deep Support Vector Data Description (Deep SVDD) as a methodology is used. Deep SVDD projects the input features onto a latent space, and tries to retain all normal samples that are concentrated around a centre. The learning goal is to build a small hypersphere which captures the normal data distribution. A sample that is near to this centre would be considered as a normal sample whereas a sample that is distant would be considered

as anomalous.

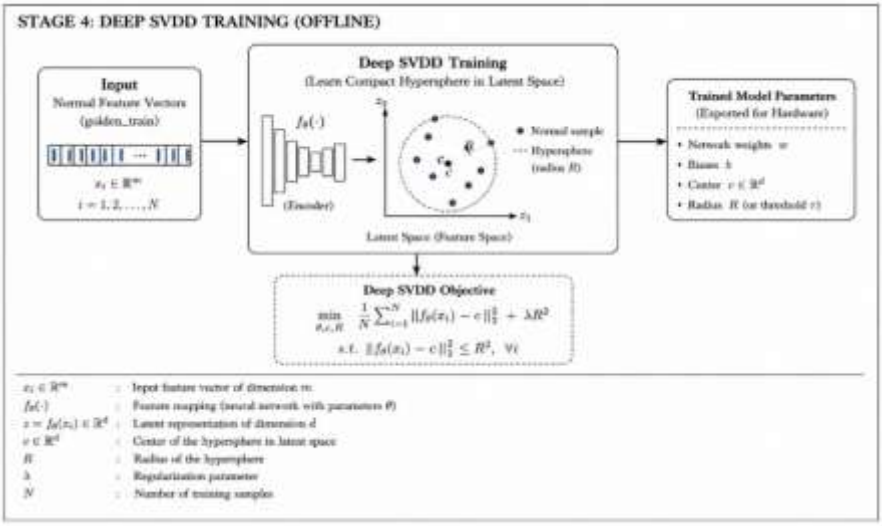

Figure 5: Deep SVDD Offline Training Stage: training the one-class model using only golden feature vectors to learn the centre and compact boundary of normal hardware behaviour.

The distance between the mapped feature representation and the hypersphere centre is squared to get the core anomaly score:

$$s(f) = \| \phi(f) - c \|^2$$

where $\phi(f)$is the learned mapping of the feature vector f, and $c$is the center of normal data in the latent space. Only normal data are used to get the model parameters, centre and decision threshold in the course of training. This represents the greatest system methodological benefit, since the framework is able to identify unknown, scaled down or unseen versions of Trojans without explicit Trojan labels during model formation.

MATLAB then exports the trained parameters which are then used in the hardware implementation. Data preparation, model fitting, validation, and demonstration are facilitated by the folders golden_train, golden_test, trojan_test as well as scripts like DATA.m, MAIN.m, and DEMO.m.

This is the heart of the methodology; the intelligence. It converts the detection problem into a traditional class-label problem to a normality modelling problem. Because real-world systems can encounter Trojan forms that were not enforced at the time of design-time experiments, such a one-class approach would be far more applicable to the hardened hardware security applications.

Stage 5: Online Detection

The last stage after the Deep SVDD model is trained consists of online detection. During this step a new observed EM signal of the DUT is processing through the same processing chain: acquisition, pre-processing, FFT, magnitude-squared calculation, and feature vector making. It then creates an extracted feature vector which is analyzed with the model parameters trained.

The anomaly score is calculated on the basis of the distance learnt offline. This score is compared against a threshold which is based on the normal training distribution in a decision rule. When the score is in the acceptable normal range, then the hardware can be considered normal/golden. A score higher than the limit value will be considered as an anomalous/Trojan sample.

This can be decided as:

$$Decision = \{Normal, s(f) \leq \tau \; Trojan, s(f) > \tau$$

where $\tau$is the anomaly threshold.

In the Verilog-oriented pipeline, this functionality is distributed across modules such as anomaly_scorer and ht_detector_top. The testbench tb_ht_detector_top verifies the digital flow. Thus, the final system achieves a practical bridge between MATLAB-based model development and Verilog-based implementation for hardware-

level deployment.

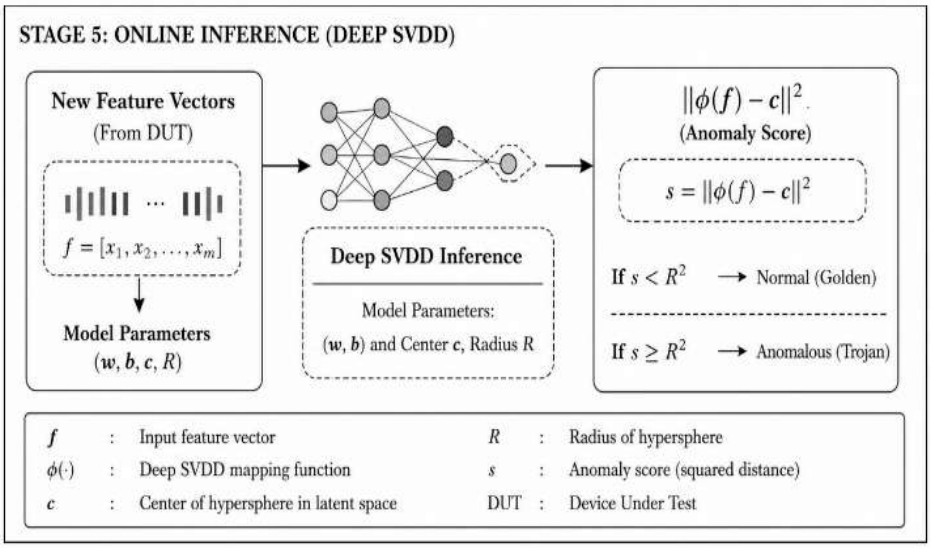

Figure 7: Online Detection Stage: inference of new feature vectors using trained Deep SVDD parameters and final decision as Normal or Trojan.

The phase transforms the theoretical model acquired into a working security mechanism. It allows real-time or near-real-time screening hardware instances, so the system is applicable in both runtime validation and production screening, as well as trust verification of integrated circuits.

The experiment starts with a near-field probe and analog front-end measuring the electromagnetic radiation of the hardware of interest. The signal is pre-processed in the frequency domain by converting the signal to digital form and transforming it into the frequency domain by FFT. Out of the obtained power spectrum, only the dominant peaks within the operating clock are picked to create a small feature vector. The training of a Deep SVDD model using only golden hardware data is based on these feature vectors. Lastly, the trained model is used in online mode whereby it provides a classification, either normal or Trojan, to the new features vectors depending on an anomaly score.

This entire methodology is important as it integrates:

- side-channel based observation
- spectral-domain analysis
- compact feature engineering
- unsupervised anomaly learning
- practical Verilog implementation

Consequently, the suggested system will be appropriate to detect not only the known Hardware Trojans, but also unknown as well as structurally altered threats.

5. Implementation Framework

The implementation is subdivided into offline and online. The offline aspect obtains feature vectors based on golden machines, standardizes the information, educates the Deep SVDD mapping, calculates the latent-space centre and decides the anomaly threshold using the normal score distribution. This step is done in MATLAB as it needs to optimize the process iteratively and easily analyze data.

Table 2. Software and hardware mapping of the proposed detector.

Stage	Primary Operation	Implementation Mapping	Output
Acquisition	Near-field EM measurement, amplification, filtering, and digitization	EM probe, analog front end, ADC/SAR	Discrete time-domain samples
Pre-processing	Hann window, FFT, magnitude squared	Window multiplier, fft_wrapper_stub, magnitude_squared	Power spectrum

Stage	Primary Operation	Implementation Mapping	Output
Feature generation	Clock-region peak selection	peak_extractor	Reduced feature vector
Offline learning	Deep SVDD optimization, centre and threshold estimation	MATLAB scripts and golden_train data	Weights, centre, and threshold
Online decision	Latent mapping, squared-distance score, threshold comparison	anomaly_scorer and ht_detector_top	Normal/Trojan flag

The online mode uses the same sequence of acquisition and feature-extraction with each new device or run-time monitoring. Processing chain is structured into re-usable Verilog blocks. The fft_wrapper stub is the interface of an FFT core; magnitude squared maps complex FFT output to spectral power; peak extractor stores the rejected clock-region peaks; anomaly scorer is the distance-based score; and ht interface detector top is the coordinating-point of the overall decision flow. The top testbench is the tb_ht_detector.
The online path should be represented using fixed-point representation as the operations it requires are mainly window multiplication, FFT interface operations, magnitude operations, feature selection operations, affine transformations and squared-distance accumulation. The word lengths should be chosen among the observed dynamic range in such a manner that quantization does not boost the distinction between normal and anomalous scores. In between stages, it is possible to add a pipeline register that will enhance throughput without altering the algorithm.

6. Results and Discussion

The material used to implement the project defines the entire processing and decision flow, but omits a tabular representation of results with numbers (sample counts, accuracy, precision, recall, false-alarm rate, latency, FPGA usage and power consumption). As a result of this, the argumentation of this paper presents the functional results aided by the given design and does not present unequal percentages and synthesis figures.
Hann windowing at the signal-processing level removes discontinuities at boundary of the observation and as a result restricts spectral leakage preceding FFT analysis. The magnitude-squared operation eliminates the dependence on spectral phase and a focus on the distribution of switching energy. It is possible to choose peaks near operating clock and neighbouring components to decrease the feature dimension and direct the detector towards the activity regions which are highly affected by repetitive synchronous activity.
The learning level strategy based on normal only training yields a compact representation which focuses on the golden feature distribution. The golden test vectors which is not far away as the learned centre generate a score falling within the accepted radius and a structurally distorted or Trojan-like vectors would give a larger score.
The threshold has the effect of a priori converting a continuous distance measurement into a hardware decision. This behaviour corresponds to the Deep SVDD objective and aid in the identification of the attack patterns, which are not explicitly represented when training.
At implementation level one of the most practical benefits is the separation between the optimization of the offline models and the online inference.
Repetition of training is possible when the probe position, work load, device family, or environmental conditions vary, with the trained detector just storing the model parameters that are exported, and doing forward arithmetic. The superstructure of the modular Verilog also allows individual verification of the interface to the FFT, spectral-power evaluation, feature extraction, score generation and final determination

Table 3. Functional validation criteria for the proposed processing and detection flow.

Validation Aspect	Golden/Normal Behaviour	Trojan/Anomalous Behaviour	Interpretation
Power-spectrum structure	Dominant peaks remain within the learned clock-region pattern	Peak amplitudes or neighbouring components deviate from the normal pattern	Physical switching behaviour has changed

Validation Aspect	Golden/Normal Behaviour	Trojan/Anomalous Behaviour	Interpretation
Feature vector	Values remain inside the normal feature distribution	One or more selected peaks move outside the accepted distribution	Compact representation preserves discriminative variation
Deep SVDD score	Squared distance is less than or equal to the calibrated threshold	Squared distance exceeds the threshold	Normal/Trojan decision is produced
Verilog decision path	Normal flag asserted and Trojan flag cleared	Trojan/anomaly flag asserted	Top-level logic implements the one-class rule
Robustness check	Repeated golden captures remain stable after normalization	Persistent deviation remains after noise suppression	Reduces the risk of classifying transient noise as a Trojan

The main limitation is calibration sensitivity. Probe displacement, clock drift, supply variation, and temperature can shift normal EM features and increase false alarms. Reliable deployment therefore requires consistent acquisition conditions, repeated golden measurements, normalization, and a threshold selected from a validation set rather than from the training set alone.
Quantitative claims should be reported only after testing across multiple chips, Trojan sizes, workloads, probe locations, and environmental conditions.

Table 4. Decision-level comparison between supervised classification and Deep SVDD.

Property	Conventional Supervised Detector	Proposed Deep SVDD Detector
Required training labels	Normal and representative Trojan classes	Normal/golden data only
Response to unseen Trojan structure	Depends on similarity to trained classes	Detected as deviation from learned normality
Decision quantity	Class probability or separating hyperplane	Distance from the normal latent-space center
Online complexity	Model dependent; may require multi-class output	Forward mapping, squared distance, and threshold comparison
Deployment concern	Dataset coverage and class imbalance	Threshold calibration and domain shift

The following figures summarize the simulation, training, threshold selection, and comparative anomaly-detection results of the EM side-channel-based Hardware Trojan detection framework.

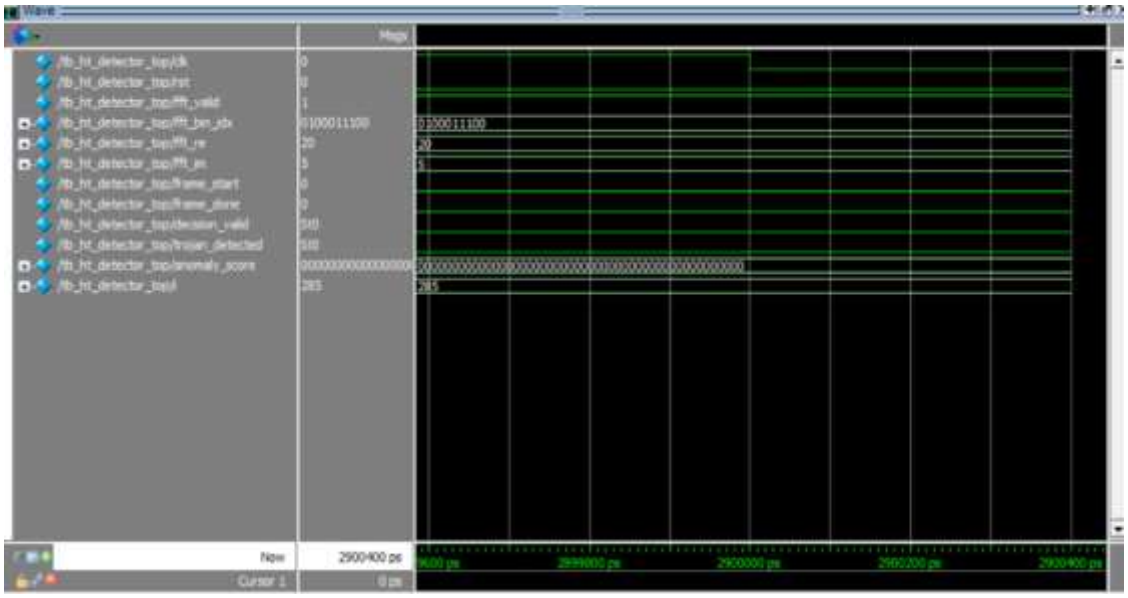

Figure 8. Verilog Simulation Waveform of the Hardware Trojan Detector

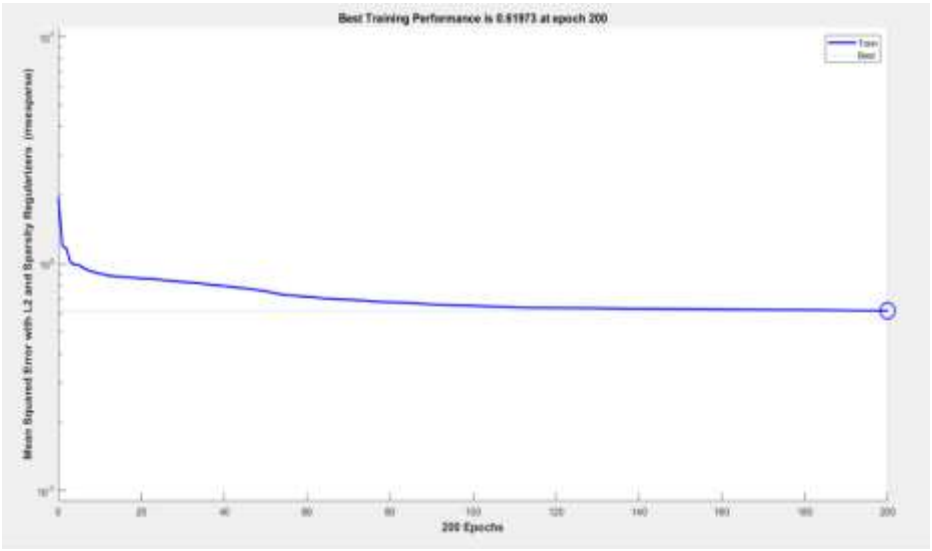

Figure 9. Training Performance of the Feature-Learning Network

The simulation waveform shows the operation of the top-level Hardware Trojan detection module. The clock and reset signals control the processing sequence, while the FFT input values are supplied to the detector. The outputs include frame completion, decision validity, Trojan status, and the computed anomaly score, confirming the functional integration of the Verilog detection pipeline.

The training curve presents the reduction in mean squared error with sparsity regularization over 200 epochs. The error decreases rapidly during the initial epochs and then converges gradually. The best training performance of 0.61973 at epoch 200 indicates stable learning of the normal electromagnetic feature representation.

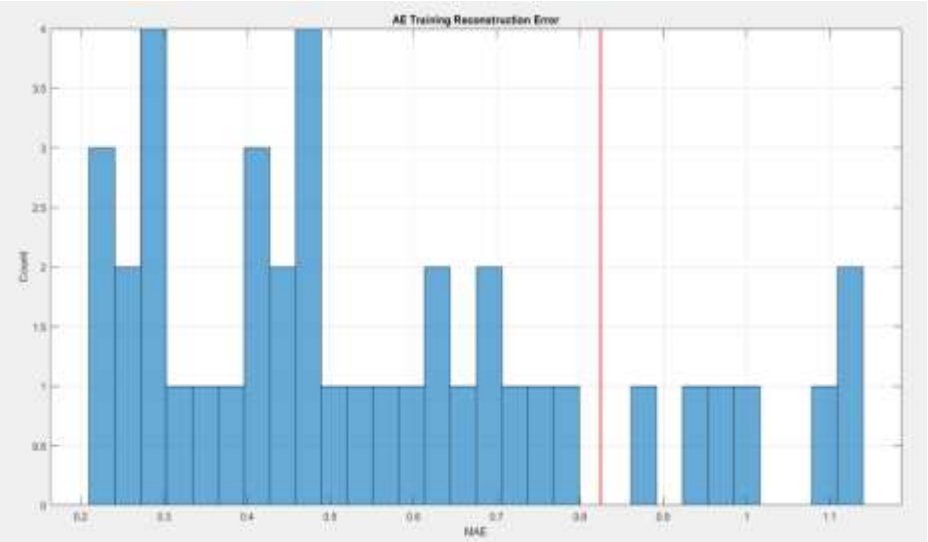

Figure 10. Distribution of Autoencoder Training Reconstruction Error

The histogram shows the mean absolute reconstruction error obtained from golden training samples. Most normal samples are concentrated at lower error values, while the red vertical line represents the selected anomaly threshold. Samples producing an error above this limit can be treated as suspicious or Trojan-affected observations

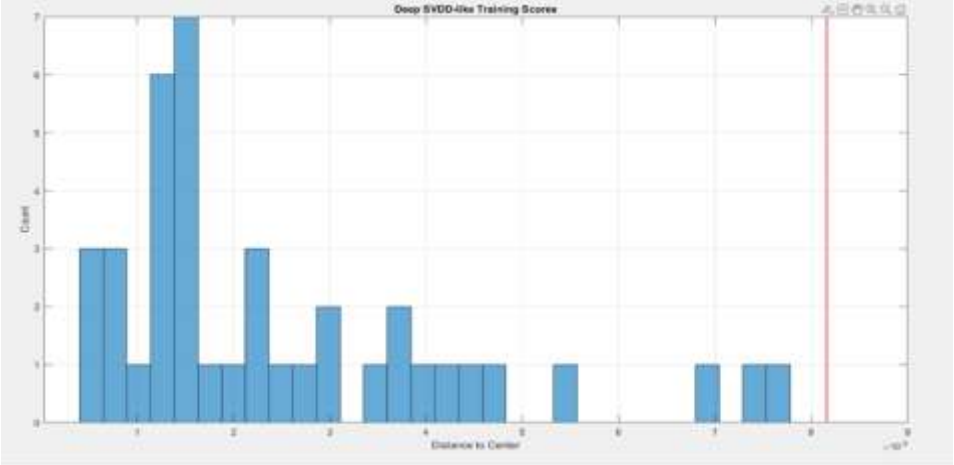

Figure 11. Distribution of Deep SVDD Training Anomaly Scores

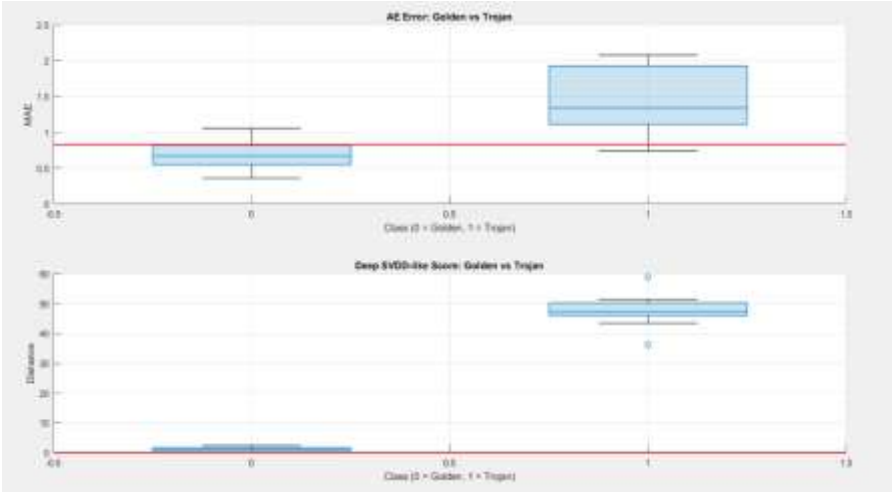

Figure 12. Comparison of Golden and Trojan Samples Using AE and Deep SVDD Scores

This histogram illustrates the distance of golden training samples from the learned Deep SVDD centre. The majority of normal samples remain close to the centre, demonstrating compact one-class representation. The red vertical line denotes the decision boundary used to identify samples that deviate significantly from normal hardware behaviour.

The upper box plot compares autoencoder reconstruction error for golden and Trojan samples, while the lower plot compares their Deep SVDD distance scores. Trojan samples exhibit substantially higher values in both cases, with clearer separation in the Deep SVDD score. This confirms that the learned anomaly model can effectively distinguish normal hardware signatures from Trojan-induced deviations.

7. Conclusion and Future Work

The hardware-Trojan detecting framework described in this paper consists of non-invasive EM side-channel measurement, frequency-domain feature extraction, Deep SVDD-modelled normality and an inference path implemented in hardware. The feature dimension is limited by the utilization of dominant spectral peaks about the operating clock and information concerning synchronous switching activity is retained. Deep SVDD does not require labelled examples of all types of Trojans and gives an explicit anomaly value as the distance between a new representation and the learned centre of the golden behaviour.

This partition of the MATLAB to Verilog is supported on practical deployment: offline model optimization and threshold choice, but online hardware path implementation: deterministic pre-processing, feature formation, score computation, and threshold comparison. The method is thus appropriate in production screening, design testing and run time trust checking on condition that the condition of acquisition is regulated and that the threshold should be tuned on independent golden data.

Additional experimental campaigns that are statistically complete with many devices, probe placement, workloads, and Trojan benchmarks should be added to future work. Actual MATLAB and FPGA/ASIC values should be given concerning accuracy, precision, recall, F1-score, false-positive rate, receiver-operating-characteristic area, latency, throughput, LUT/flip-flop/DSP/BRAM utilization, and power. Robustness can be further enhanced with multi-point EM acquisition, adaptive normalization, time-frequency features, multimodal fusion with power and timing channels as well as online domain adaptation.

References

[1] M. Tehranipoor and F. Koushanfar, “A survey of hardware Trojan taxonomy and detection,” *IEEE Design Test Comput.*, vol. 27, no. 1, pp. 10–25, 2010.
[2] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, “Trustworthy hardware: Identifying and classifying hardware Trojans,” *Computer*, vol. 43, no. 10, pp. 39–46, 2010.
[3] R. S. Chakraborty, S. Narasimhan, and S. Bhunia, “Hardware Trojan: Threats and emerging solutions,” in *Proc. IEEE High-Level Design Validation Test Workshop (HLDVT)*, 2009, pp. 166–171.
[4] S. Narasimhan, X. Wang, D. Du, R. S. Chakraborty, and S. Bhunia, “Multiple-parameter side-channel analysis for hardware Trojan detection,” *IEEE Trans. Comput.*, vol. 62, no. 11, pp. 2183–2195, 2013.
[5] J. Rajendran, O. Sinanoglu, and R. Karri, “Security analysis of logic obfuscation,” in *Proc. Design Autom. Conf. (DAC)*, 2012, pp. 83–89.
[6] Y. Jin and Y. Makris, “Hardware Trojan detection using path delay fingerprint,” in *Proc. IEEE Int. Symp. Hardware-Oriented Security Trust (HOST)*, 2008, pp. 51–57.
[7] J. He, Y. Zhao, X. Guo, and Y. Jin, “Hardware Trojan detection through electromagnetic side-channel analysis,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 10, pp. 2939–2948, 2017.
[8] Y. Zheng, S. Bhunia, and S. Yang, “Self-referencing-based Trojan detection without a golden chip,” *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 35, no. 1, pp. 37–48, 2016.
[9] B. Schölkopf *et al.*, “Support vector method for novelty detection,” in *Advances in Neural Information Processing Systems*, 1999.
[10] L. Ruff et al., “Deep one-class classification,” in Proc. Int. Conf. Mach. Learn. (ICML), 2018, pp. 4393–4402.
[11] M. A. Kramer, “Autoassociative neural networks,” Comput. Chem. Eng., vol. 16, no. 4, pp. 313–328, 1992.
[12] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” Nature, vol. 323, pp. 533–536, 1986.
[13] S. Sankaran et al., “Deep learning-based approach for hardware Trojan detection,” in Proc. IEEE Int. Symp. Smart Electronic Systems (iSES), 2021.
[14] K. G. Liakos et al., “Machine learning for hardware Trojan detection: A review,” in Proc. PACET, 2019.
[15] Z. Huang et al., “A survey on machine learning against hardware Trojan attacks,” IEEE Access, vol. 8, pp. 10796–10826, 2020.
[16] A. Sumarsono and Z. Masters, “Application of LSTM autoencoder in Trojan detection,” in Proc. IEEE

Comput. Commun. Workshop Conf. (CCWC), 2023.
[17] S. Faezi, R. Yasaei, and M. A. Al Faruque, "HTNet: Transfer learning for Trojan detection," in Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE), 2021.
[18] J. Takahashi et al., "Machine learning-based Trojan detection using electromagnetic emanation," IEICE Trans., 2022.
[19] L. N. Nguyen et al., "Backscattering side-channel for Trojan detection," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 27, no. 7, pp. 1561–1574, 2019.
[20] H. Li, Q. Liu, and J. Zhang, "A survey of hardware Trojan threat and defense," Integration, vol. 55, pp. 426–437, 2016.
[21] J. He et al., "Golden chip-free Trojan detection leveraging electromagnetic fingerprinting," IEICE Electron. Express, 2019.
[22] K. Xiao et al., "Hardware Trojans: Lessons learned after one decade of research," ACM Trans. Design Autom. Electron. Syst., vol. 22, no. 1, 2016.
[23] S. Wang et al., "Trojan detection based on an extreme learning machine neural network," in Proc. Int. Conf. Comput. Commun. Informatics (ICCCI), 2016.
[24] M. Priyatharishini and M. N. Devi, "Deep learning-based malicious module identification," Measurement, vol. 194, 2022.
[25] N. Nour et al., "Machine learning-based Trojan identification," J. Telecommun., 2017.
[26] L. Zhang et al., "Trojan detection using electromagnetic leakage," China Commun., vol. 16, no. 12, pp. 100–110, 2019.
[27] A. Fournaris et al., "FPGA Trojan detection using multiple parameters," Microprocess. Microsyst., 2019.
[28] J. U. Ko et al., "Autoencoder-based anomaly detection with a dynamic threshold," Expert Syst. Appl., 2022.
[29] P. Bergmann et al., "Unsupervised defect detection using autoencoders," arXiv preprint arXiv:1807.02011, 2018.
[30] F. Wan et al., "Outlier detection using a stacked autoencoder," *IEEE Access*, vol. 7, 2019.