Journal of Global Research in Electronics and Communication
Volume 1, No. 10, October 2025
Available Online at: www.jgrec.info
RESEARCH PAPER

Lightweight Machine Learning Techniques for Hardware Trojan Detection in IoT Devices

Dr. Chintal Kumar Patel
Associate Professor, CSE
Geetanjali Institute of Technical Studies
chintal.patel@gits.ac.in

***Abstract*—Hardware trojans are malicious pieces of software that attempt to prevent the normal operation of a chip, and are carefully engineered not to be detected during the silicon design and verification phase before it is actually sold to a consumer. The military, business, and academics are all looking into this new threat. Consequently, as a defense during chip deployment, run-time hardware Trojan identification is vitally needed. This work focuses on hardware Trojans that affect processor performance. This study presents a machine learning-based approach to detecting hardware Trojans in IoT devices by exploiting the Hardware Trojan Dataset. To preserve the most relevant features, the dataset was subjected to a thorough preparation procedure that included data cleaning, augmentation, label encoding, normalization, and feature selection using PCA. A number of models were evaluated, including Logistic Regression, ResNet, Decision Tree, and Long Short-Term Memory (LSTM). With the highest accuracy, precision, recall, and F1-score of 95.25%, 95.25%, 95.27%, and 95.25%, respectively, the LSTM model fared better than the others. The outcomes demonstrate how well feature selection and sequential deep learning architectures work together to capture temporal relationships in power trace data. Overall, the suggested approach shows a strong and trustworthy foundation for improving IoT hardware security against Trojan assaults.**

Keywords*—Integrated Circuit Security, Hardware Trojan, Deep Learning, Long short-term memory (LSTM), Hardware Trojan dataset, Power Trace Analysis.

I. INTRODUCTION

A huge network known as the Internet of Things (IoT) links every object in the world to the web through a variety of data-gathering devices, such as RFID tags, infrared detectors, and GPS trackers[1][2][3]. The Internet and its associated applications, such Smart homes, smart medical devices, and the Internet of Vehicles, have grown increasingly varied, improving the convenience and intelligence of people's life. These devices continuously sense, process, and exchange data, enabling real-time decision-making and intelligent automation[4]. However, this massive integration of IoT nodes into critical infrastructures has simultaneously expanded the security threat landscape make them particularly susceptible to both software and hardware-level attacks. Consequently, ensuring trust, confidentiality, and resilience at the hardware layer has become a cornerstone for the long-term sustainability of IoT ecosystems. [5][6].

To guarantee the safe deployment of these devices during their lifetime, hardware security is crucial, hardware-level vulnerabilities, *hardware Trojans (HTs)* represent one of the most severe and stealthy threats[7][8]. A harmful change that is purposefully introduced into an integrated circuit (IC) at any point in the hardware supply chain—during design, verification, manufacture, or testing—is known as a hardware Trojan[9]. These modifications can remain dormant under normal operating conditions and activate only under rare triggers, leading to functionality degradation, data leakage, or even complete system failure[10].These days, almost every network, from the home network to the national military and medical sectors, has at least one IoT device. Thus, for both personal and national security, these gadgets' security is crucial[11]. Conventional detection methods, including functional testing, side-channel signal analysis, and formal verification, rely heavily on golden reference models or high-precision test setups—resources typically unavailable in low-cost IoT environments. Traditional security mechanisms, software stack[12][13][14]. such as static analysis, rule-based heuristics, and signature-based detection, frequently fall short in adapting to the dynamic nature of modern assaults and generalizing across changing threat vectors.

Machine learning (ML) has become a revolutionary method in the detection of hardware Trojan in IoT devices. ML methods have the ability to learn patterns of discrimination with respect to side-channel measurements, power variations, or performance data automatically, without user intervention to distinguish between legitimate and infected circuits. In laboratory settings, it has been demonstrated that superior models' convolutional neural networks (CNNs), random forests, and support vector machines (SVMs) are among the most effective detection algorithms. Nevertheless, the conventional ML algorithms can be computationally infeasible with data of IoT edge hardware[15]. algorithms for AI-based detection. It goes into the effectiveness of ML defense models for tasks like protecting hardware or satellite broadcasts against side-channel attacks. Trojan horses that spy on equipment can be the most successful way to combat intellectual property theft by pirates.

A. Motivation and Contribution

This is driven by the fact that the security threats posed by IoT devices have been increasing, especially after the introduction of hardware trojans during manufacturing. These types of trojans may cause sensitive information leakages, provide an unprotected backdoor, shut down important functions, or support botnet attacks on a large scale. The computational capabilities of an IoT device are rather limited, and the activation of a trojan is often done stealthily, making it impossible to detect it using conventional methods. This challenge drives the development of an intelligent LSTM-based detection framework that can identify subtle temporal anomalies in hardware behavior while ensuring computational efficiency for resource-constrained IoT environments. The framework is designed to effectively distinguish between

benign circuits, dormant trojans, and active trojans with high accuracy. This research offers several key contributions as listed below:

- The Hardware Trojan Dataset was used to develop a ML-based framework to identify hardware Trojans in IoT devices.
- Implemented a comprehensive data preprocessing pipeline, including noise removal, augmentation, label encoding, normalization, and data balancing.
- Applied PCA-based feature selection to reduce dimensionality, eliminate redundancy, and enhance model efficiency.
- Provided a scalable and reliable solution for securing IoT hardware systems against Trojan attacks, addressing gaps in existing detection methods.
- Presented and analyzed a Long Short-Term Memory (LSTM) model that can learn the power traces data's sequential dependencies.
- Several performance metrics, such as F1-score, recall, accuracy, and precision, were employed to assess the model's efficacy in detecting hardware Trojans.

B. Justification and novelty

The work is rationale by the fact that security challenges in hardware Trojans in IoT devices are on the rise, and this is a significant threat to both data integrity, system reliability, and continuity of operation. Traditional methods of detecting lack the ability to detect Trojans because their activation mode is stealthy, and because of the limited resources available to the IoT environment. The originality of this study is based on the construction of an intelligent pattern of the LSTM-based detection framework, which is an effective way to capture the time-dependent dependency and missing behavioral trends of hardware performances. The suggested approach is applicable to the safe implementation of IoT hardware in the real world by taking advantage of the sequential modeling Deep learning (DL) capabilities to attain high accuracy, resilience, and computing efficiency of the framework, making it a workable and expandable solution to the issue.

C. Organization of the Paper

The structure of this paper is as follows: Section II synthesizes pertinent studies on the hardware of IoT devices, Malware identification, Section III explains the model's implementation, preprocessing procedures, and dataset, Section IV compares and contrasts the experimental findings, and Section V summarizes the study's main conclusions and suggests areas for further investigation.

II. LITERATURE REVIEW

A comprehensive review and analysis of key research studies on Hardware Trojan detection in IoT were undertaken to inform and enhance the design of this work.

Yoshimi *et al.* (2025), the design and manufacturing stages of IoT devices, there is a risk of Hardware Trojans (HTs) being inserted into circuits due to the intervention of outside companies. One method for effectively detecting HTs from gate-level netlists is to use an ensemble learning model four ensemble learning models: Random Forest, XGBoost, and evaluate the accuracy of HT detection by adding a new Trojan circuit generated using an automatic HT generation framework as a netlist for training and evaluation. it's also use SMOTE, ADASYN, and Borderline-SMOTE as oversampling methods used in training, and evaluate the HT detection accuracy 88.48% when the hyperparameters of each method are optimized[16].

Moussa and Rafla (2024) offers a better method for identifying hardware trojans by employing ML models to decrease to prevent over-fitting, the characteristics should be linear. The true positive and true negative rates for the supervised model were 99.2%, along with an F-measure, but the unsupervised model relied on random projection to get a true positive rate, providing a more robust ML-based approach for HT detection[17].

S and E (2024) suggested an unsupervised ML model that uses the controllability and observability Trojan detection (COTD) approach to identify and classify signals in the gate-level netlist as either valid or suspicious. The controllability and observability of each net from several ISCAS-85 and ISCAS-89 benchmark circuits are grouped together in a single round using this COTD approach. It then employs density-based clustering algorithms and K-means clustering to find suspicious or hardware trojan signals that have traits similar to those of HT-free signals. In spite of excellent precision, the false positive rate (FPR) was low because these signals were misshapen. The K-means clustering algorithm's experimental findings on the ISCAS-85 and ISCAS-89 benchmark circuits indicate that 98 and 0.8871[18].

Gourousis *et al.* (2023) combine a proposed approach to anomaly identification using a non-invasive means of measuring on-chip temperature, combined with an autoencoder-based ML system for hardware Trojan detection. Even when the hardware Trojan consumes only 2.5% of the circuit being tested for power, the developed algorithm in a case study detects it with above 90% accuracy. The program is capable of not only detecting the Trojan but also determining its precise position on the chip. In order to strengthen the security of current electronic systems' hardware, especially for IoT uses, an ML–based anomaly detection approach is now under development[19].

Sankar, Nirmala Devi. and Jayakumar (2022), The data handled by IoT devices is extensive and includes sensitive information pertaining to the app in use. IoT devices are vulnerable to a variety of assaults in such a situation. ML–based Trojan detection in RS232 significantly facilitates secure communication between IoT devices that are enabled by the edge. To effectively detect Trojans, most supervised algorithms for Trojan detection depend on high-quality labeled datasets. It is clear that semi-supervised hardware is both effective and practicable, with a true negative rate of 95.77% and an average true positive rate of Trojan horse identification[20].

Wang *et al.* (2021) provide a concept and technique for hardware Gate-level Trojan detection that may be used to look for trigger networks throughout the whole chip. Specifically, each net's trigger-net properties are taken from known netlists, and a variety of detection models are built using, ML depending on the trigger modes. The netlist of the integrated circuit being detected is searched for suspicious trigger nets using the detection models, which then assign a suspiciousness value to each net. Their average accuracy rate is 96%, and they able to identify the bulk of hardware Trojans

by identifying the suspiciousness ratings of the top 2% of suspicious nets[21].

Gayatri *et al.* (2020) proposes the Atmel XMega Controller (Target Board)'s AES-256 decryption method combined with side-channel power analysis and ML to identify hardware Trojans at the system level. It uses the ChipWhisperer-Lite board to analyze power. Utilizing the 80/20 rule, the ML model is trained utilizing the power traces of the hardware Trojan-infected and golden algorithm (Hardware Trojan-free) methods. The accuracy of the suggested ML model for every Trojan that was introduced ranged from 97% to 100%[22].

Table I presents an overview of recent research on Hardware Trojan detection, highlighting the proposed models, datasets used, key findings, and the challenges encountered.

TABLE I. RECENT STUDIES ON HARDWARE TROJAN DETECTION IN IOT DEVICES USING MACHINE LEARNING

Author	Key Dataset	Methodology	Key Findings	Limitations	Future Work / Scope
Yoshimi et al., 2025	Gate-level netlists; Automatically generated Trojan circuits	Ensemble Learning Models (RF, XGBoost, Oversampling with SMOTE, ADASYN,	Achieved 88.48% HT detection accuracy after hyperparameter optimization	Limited to gate-level netlists; does not address real-time or side-channel Trojan scenarios	Extend ensemble methods to RTL and FPGA-level designs; integrate with side-channel data for hybrid detection
Moussa & Rafla, 2024	Custom gate-level datasets; Random projection features	Supervised and Unsupervised ML models using random	Supervised: 99.2% TP/TN, F-measure	Focused only on reducing feature linearity; lacks scalability to large IoT designs	Develop scalable ML pipelines for complex SoC architectures integrate hybrid feature sets
S. & E., 2024	ISCAS-85 and ISCAS-89 benchmark circuits	Unsupervised ML model (K-means, Density-Based Clustering) using Controllability	K-means accuracy 98%, FPR 0.8871; effectively identifies suspicious signals	Misclustering occurs for Trojan-free signals; lacks robustness under noisy datasets	Improve clustering robustness; extend to dynamic Trojan behavior analysis
Gourousis et al., 2023	On-chip temperature data (experimental case study)	Anomaly detection via Autoencoder-based ML model coupled with non-invasive temperature sensing	Detected Trojans with >90% accuracy even at 2.5% power consumption; localized Trojan position	Only tested on limited power variation scenarios; not validated for diverse chip architectures	Expand anomaly detection to other physical parameters; real-time IoT deployment testing
Sankar, Nirmala Devi & Jayakumar, 2022	RS232 communication interface (IoT devices)	Semi-supervised ML Trojan detection for edge-assisted IoT devices	Achieved 95.77% true negative rate; effective for secured IoT communication	Relies on labeled datasets; limited coverage of multi-protocol IoT systems	Develop fully unsupervised or self-learning detection models for heterogeneous IoT systems
Wang et al., 2021	Known gate-level netlists datasets	ML-based trigger-net feature extraction and scoring for suspiciousness	Achieved 96% average detection accuracy by flagging top suspicious nets	Performance depends on accurate trigger-mode modeling; potential false positives	Improve adaptive learning of trigger patterns; integrate explainable AI for interpretability
Gayatri et al., 2020	AES-256 implementation on Atmel	ML-based side-channel power analysis	Detection accuracy between 97% across inserted Trojans	Limited to AES algorithm; dependent on side-channel setup	Extend to multi-algorithm detection; integrate cross-device generalization using transfer learning

Research gaps: Several research gaps still exist despite notable progress in hardware Trojan detection using ML and DL approaches. Most existing approaches focus on specific benchmark circuits or limited datasets, limiting their generalizability to diverse IoT devices and complex integrated circuits. Many methods rely heavily on side-channel analysis or gate-level features, which may be vulnerable to sophisticated Trojan designs or environmental variations. Additionally, while high accuracy has been achieved in controlled experiments, real-time detection in limited resources IoT situations continues to be difficult. Furthermore, there is a lack of defined frameworks for evaluating various detection models, and few studies address the trade-offs between detection accuracy, computational overhead, and scalability. Future research should aim to develop more robust, generalized, and lightweight detection techniques capable of handling emerging hardware threats across varied platforms.

III. RESEARCH METHODOLOGY

The proposed methodology for the Hardware Trojan Dataset is used to detect hardware Trojans in IoT devices, where selected power traces with and without HT circuits are analyzed using a contrastive learning framework. The data undergoes preprocessing, including missing value handling, noise removal, data augmentation, label encoding, and min-max normalization, followed by feature selection via PCA to retain the most informative features. It is thereafter divided into testing sets of 20% and training sets of 80%. An LSTM model that detects Trojan horses and logs temporal trends is trained using the processed data. The F1-score, recall, precision, and accuracy are used to evaluate the model's performance in order to guarantee accurate and robust detection. How well the model can predict and efficiently categorize all Trojan groups is demonstrated in Figure 1, which shows its efficacy in detecting several types of Trojans (No Trojan, Dormant Trojan, Active Trojan).

The next section provides a thorough description of every stage in the suggested approach:

A. Data Gathering and Analysis

This study utilizes the Hardware Trojan Dataset. It is designed to generate probability distributions for input data by utilizing the contrastive learning architecture for every category. In order to do this, 100 random data sets—50 with and 50 without HT circuits—are chosen from the validation dataset of different HT kinds. Data visualizations such as bar plots and heatmaps were used to examine attack distribution, feature correlations, etc., and are given below:

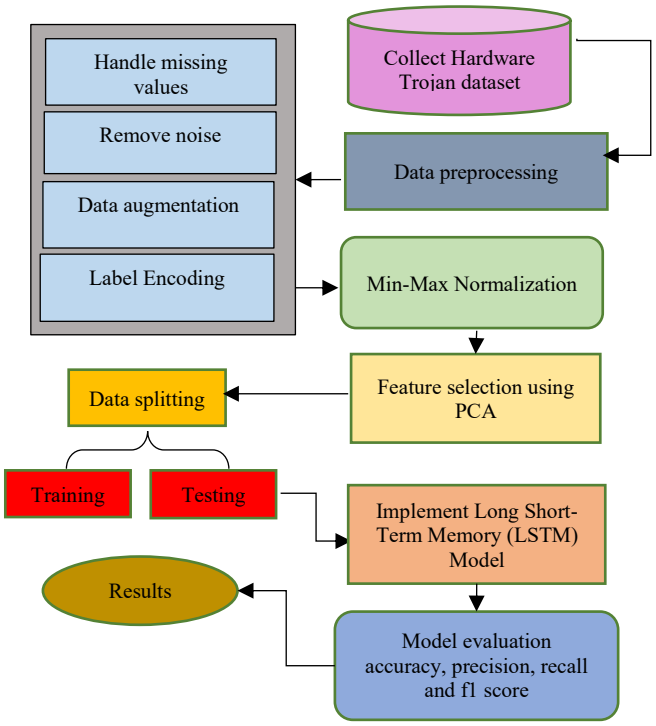

Fig. 1. Proposed flowchart for Hardware Trojan Detection in IoT Devices Using Machine Learning

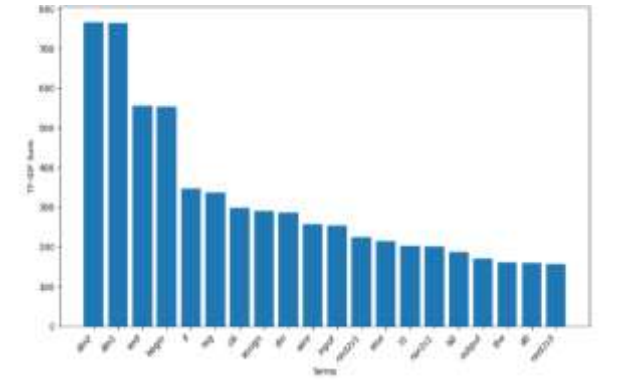

Fig. 2. Feature from the full hardware trojan dataset

The bar chart displays the sum of TF-IDF scores for the top terms, likely extracted from a corpus of text, potentially related to hardware description languages or programming in figure 2. The most significant terms, with scores approaching 800, are "dln2" and "dln1", followed by terms like "end" and "begin" which have scores around 550. Other notable terms, with scores generally decreasing from around 350 to just over 150, include control flow and structural words "if", "reg", "ok", "assign", "dln", "wire", "input", "else", and terms that appear to be variable names or labels like "nnd2s1", "j1", "ner2s1", "b0", "output", and "nnd2s3".

A heatmap evaluating classification performance across six data augmentation techniques (Logistic, Interpolate, Shift, Noise, Filter, Scale) is shown in Figure 3. Colors range from dark purple (96-120) to light yellow (60-72). Diagonal elements display higher values (95-108), indicating superior performance when training-testing augmentations match, while off-diagonal elements show reduced accuracy (66-70) with mismatched augmentation types.

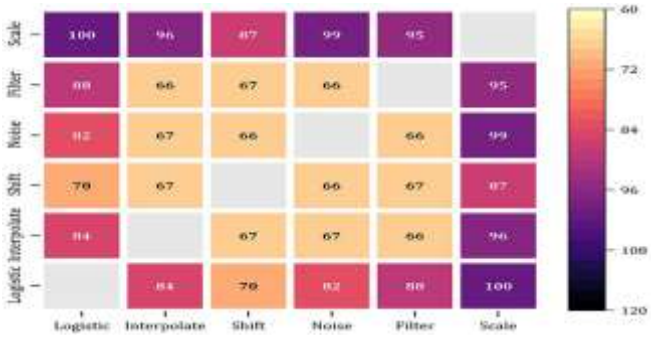

Fig. 3. Correlation Matrix Heatmap on Hardware Trojan Dataset

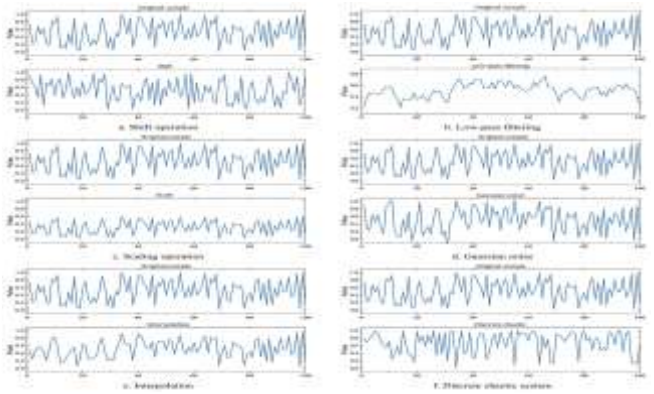

Fig. 4. Comparison of the effects of data augmentation

Time-series plots showing signal variations over 140 samples. Each subplot represents different signal processing or augmentation techniques in Figure 4. The left column depicts signals of greater frequency oscillations and amplitudes between about ±2, whereas the right column presents diversified characteristics in terms of smooth tendencies and varying frequency elements. There are blue lines in all plots, and the axes have a similar scale to facilitate comparison.

B. Data Pre-processing

The Hardware Trojan Dataset was used to prepare the data, which involved the concatenation, data cleansing, and feature engineering. The preprocessing stage included the processing of missing values, noise removal, and the use of some of the methods, like data augmentation, label encoding, and normalization. These important preprocessing procedures are summarized below:

- **Handle missing values:** The management of missing values is vital in improving the model's accuracy and preventing bias in data analysis. The statistic power of the data has been preserved by filling in unobserved data points with the use of imputation or removal, enabling sound findings.
- **Remove noise:** To increase the data's quality and ML models' effectiveness, it is crucial to eliminate noise during the data preparation stage. A variety of techniques are employed, depending on the type of data and noise.
- **Data augmentation:** Data augmentation is an ML technique that creates artificially bigger datasets by combining modified copies of preexisting data, increasing the size and diversity of training data.

- **Label Encoding:** In ML, categorical input is converted into a numerical representation using a data preparation technique known as label encoding. This kind of change is required since the majority of ML algorithms require numerical input for both training and prediction.

C. Min-Max Normalization

The normalization of records was done through the min-max technique to ensure that records lie within a range of 0 to 1. This was done in an attempt to optimize the performance of the employed classifiers and to lessen the influence of outliers. Normalization was performed with respect to the following mathematical equation (1):

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (1)$$

In which, X is the initial value of the feature, X' is the standardized value, X_{min} is the lowest value of the feature and X_{max} is its highest value.

D. Feature selection using PCA

The process of selecting and choosing feature selection is the process of choosing applying a dataset's most relevant subset of input attributes to an ML model[23]. Feature selection using Principal Component Analysis (PCA) involves selecting a subset of original features by identifying which ones contribute most to primary components that account for the maximum variance in the data.

E. Data Splitting

The dataset was divided into sets for testing and training, with 20% set aside for testing and performance evaluation and the remaining 80% used for model construction and parameter estimation.

F. Proposed Long Short-Term Memory (LSTM) Model

A Long Short-Term Memory (LSTM) model based on DL is suggested for detecting hardware Trojans in Internet of Things devices. Text categorization is a key area of expertise for LSTM, as it can identify long-term relationships between texts. The LSTM classifier is a type of multilayer network, known as an RNN, which employs the preceding layer's outputs as inputs for the subsequent layer. LSTM can handle data sequences rather than individual data points due to its feedback connections. All four of these gates—an input, an output, and a forget—make up an LSTM node.

The three gates regulate the information flow inside the cell, while the cell itself is in charge of storing data throughout time. The LSTM layers are composed of memory blocks that are connected recurrently and include three multiplicative gates each. The following updates have been made to the unit's input x_t, h_{t-1}, c_{t-1} and output $h_t,\ c_t$.

Gates:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (2)$$

$$f_t = \sigma\big(W_f x_t + U_i h_{t-1} + b_f\big) \quad (3)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (4)$$

Input transform:

$$g_t = tanh\big(W_g x_t + U_g h_{t-1} + b_g\big) \quad (5)$$

State update:

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad (6)$$

$$h_t = o_t \odot \tanh(c_t) \quad (7)$$

The preceding equations use σ to represent using the logistic sigmoid function and the symbol $\odot$ to indicate addition by elements. A memory cell ct at each time step t, an input gate i_t , a forget gate f_t, an output gate o_t, and a hidden unit htt are all components of the LSTM unit. Whereas W and U are the learnt parameters, B is the added bias. Consequently, the input gate controls the amount of updating for every unit, the forget gate controls the amount of erasing of memory cells, and the output gate controls.

G. Evaluation metrics

To assess the effectiveness of the suggested architecture, several performance indicators were utilized. The trained model's metrics were calculated by comparing the actual values with the projected ones: Number of True Negatives (TN), Number of False Negatives (FN), and Number of True Positives (TP). The next sections detail the important measurements that were generated using these: recall, accuracy, precision, and F1-score:

Accuracy: The proportion of cases that the trained model accurately predicted out of all the occurrences in the dataset (input samples), it is given as (8)-

$$Accuracy = \frac{\text{TP+TN}}{\text{TP+FP+TN+FN}} \quad (8)$$

Precision: The precision measures the ratio of the number of correctly predicted positive instances to the total number of positive occurrences anticipated by the model. Precision indicates how good the classifier is in predicting the positive classes and is expressed as (9)-

$$Precision = \frac{\text{TP}}{\text{TP+FP}} \quad (9)$$

Recall: This measure is the proportion of occurrences where positive outcomes were correctly anticipated relative to all cases where positive outcomes were expected. In mathematical form, it is given as (10)-

$$Recall = \frac{\text{TP}}{TP+FN} \quad (10)$$

F1 score: It integrates precision and memory in a harmonic manner, that is, it helps to balance recall and precision. Its range is [0, 1]. Mathematically, it is given as (10)-

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (11)$$

IV. Results And Discussion

This section describes the experimental configuration and performance of the suggested model in the training and testing stages, emphasizing its assessment and computational effectiveness. The experiments were conducted on the Linux virtual machine, which has an Ubuntu 20.04 operating system, 250 GB of disk, and 4GB of RAM. It was necessary to add 5 GB of swap memory for system design and analysis in Table II. The Hardware Trojan Dataset was used for training and evaluation of the proposed LSTM model. The following metrics were used to assess performance: F1-score, recall, accuracy, and precision. A 95.25% F1-score, 95.25% recall, 95.25% precision, and 95.25% accuracy were all achieved by the model. These findings validate the model's efficacy, dependability, and computing efficiency by showing how well it can identify and categorize hardware Trojans in IoT devices.

TABLE II. CLASSIFICATION RESULTS OF THE PROPOSED MODEL, HARDWARE TROJAN DETECTION IN IOT DEVICES USING HARDWARE TROJAN DATASET

Matrix	LSTM
Accuracy	95.25
Precision	95.27
Recall	95.25
F1-score	95.25

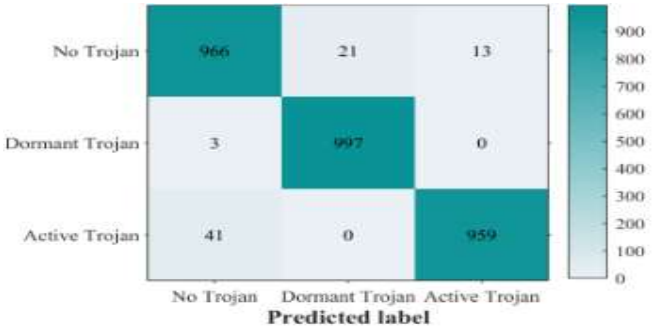

Fig. 5. confusion matrix of hardware trojan dataset

A confusion matrix displaying classification performance for Trojan detection with three categories: No Trojan, Dormant Trojan, and Active Trojan. The matrix uses a teal-to-white color gradient to represent prediction frequencies. Diagonal elements show high accuracy with values of 966, 997, and 959 for correct classifications, while off-diagonal elements indicate minimal misclassifications, demonstrating robust model performance in distinguishing between trojan states.

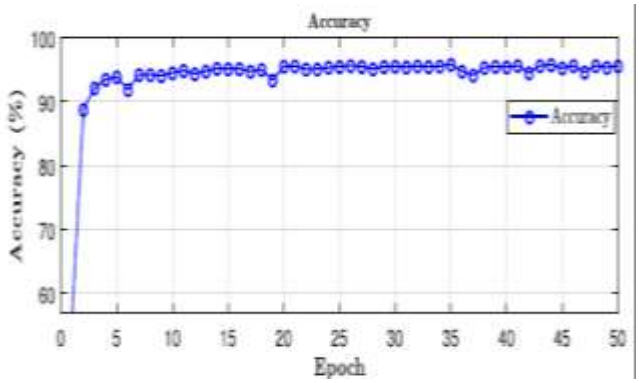

Fig. 6. Accuracy Curve for LSTM model

A line graph depicting LSTM model training accuracy for hardware trojan detection across 50 epochs in figure 6. The training epochs (0–50) are represented by the x-axis, while the accuracy percentage (0–100%) is displayed on the y-axis. The blue curve with circular markers demonstrates rapid convergence in trojan classification, achieving approximately 90% accuracy within the first 5 epochs, then stabilizing around 95-97% accuracy throughout remaining epochs, indicating effective learning of trojan detection patterns with minimal fluctuation and robust performance.

A hardware trojan detection model that uses LSTMs is shown in Figure 7, with the loss that was shown across 50 epochs, both for training and validation. The y-axis shows loss values (0-0.8), while the x-axis represents epochs (0-50). Both curves exhibit a rapid decline from initial values around 0.7-0.4, converging below 0.2 after 5 epochs, then stabilizing around 0.1-0.15, demonstrating effective model convergence in learning Trojan detection patterns with minimal overfitting.

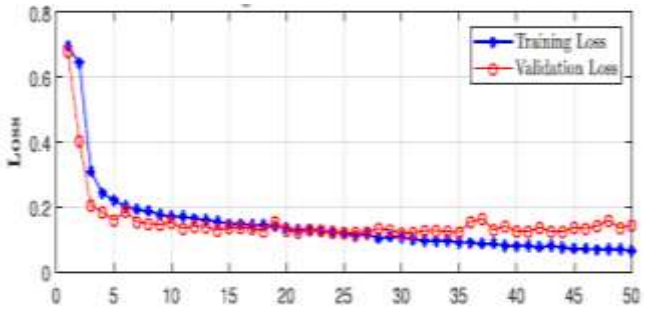

Fig. 7. Loss Curve for LSTM model

A. Comparative analysis

The accuracy of the suggested LSTM model is compared to other models that are currently in use in Table III in order to assess its efficacy. This assessment showcases the efficacy of several ML models in identifying hardware Trojans in Internet of Things devices by utilizing the Hardware Trojan dataset. Although precision was not specified, the Decision Tree (DT) model had an accuracy of 62.76%, a recall of 52.04%, and an F1-score of 58.29%. ResNet obtained an F1-score of 63%, accuracy of 67 percent, and precision of 71.7%; however, recall was not specified. The F1-score of 94.4%, recall of 93.1%, and accuracy of 94.49% demonstrated the high performance of the Logistic Regression (LR). This LSTM model scored the highest across the board with 95.25% accuracy, 95.27% precision, 95.25% recall, and 95.25 per cent F1-score. It is clear that identifying hardware Trojans in IoT devices was much easier using the proposed methodology.

TABLE III. COMPARISON OF DIFFERENT MACHINE LEARNING MODELS FOR HARDWARE TROJAN DETECTION IN IOT DEVICES ON HARDWARE TROJAN DATASET

Model	Accuracy	Precision	Recall	F1-score
DT[24]	62.76	-	52.04	58.29
ResNet[25]	67	71.7	-	63
LR	94.49	-	93.1	94.4
LSTM	95.25	**95.27**	**95.25**	**95.25**

The suggested LSTM network in Lightweight ML Techniques in Hardware Trojan Detection in IoT Devices shows an impressive benefit, as the accuracy of the proposed approach is 95.25% and indicates that the model has the potential to recognize sequential relationships and multifaceted patterns of the feature in the dataset. This high performance is an indication of how this model can capture small variations that were brought about by Hardware Trojans with greater accuracy than the traditional ML methods, hence making it a stable and effective solution to protect IoT devices against hardware threats.

V. CONCLUSION AND FUTURE WORK

Hardware security is now essential to guaranteeing overall system dependability due to the extensive use of IoT devices. Since integrated circuits are the foundation of IoT hardware, it is crucial to secure them against harmful alterations like hardware Trojans. Pre-silicon detection approaches are highly helpful since they don't require golden chips, aren't affected by process noise, and can be utilized for large-scale secure design verification. The study's experimental research reveals that the suggested LSTM model outperformed Logistic Regression (94.49%), ResNet (67%), and Decision Tree (62.76%) with a maximum accuracy of 95.25%. By decreasing dimensionality, filtering noise, and

improving model efficiency and generalization, PCA-based feature selection significantly enhanced detection performance for trustworthy Hardware Trojan detection in IoT devices. In order to better detect hardware Trojans, future studies will concentrate on developing ensemble DL models that include LSTM, CNN, and GRU to identify spatial-temporal patterns. Efforts will include developing lightweight, energy-efficient models via pruning and quantization, applying federated and transfer learning for scalable, privacy-preserving detection, and integrating attention mechanisms for component-level vulnerability analysis. Additional directions involve multi-modal detection using power, electromagnetic, and thermal data; GAN-based dataset expansion; explainable AI for interpretable auditing; hardware-in-the-loop validation; and automated response mechanisms for Trojan mitigation and recovery in critical IoT infrastructures.

REFERENCES

[1] A. M. Azab *et al.*, "SKEE: A Lightweight Secure Kernel-level Execution Environment for ARM," in *23rd Annual Network and Distributed System Security Symposium, NDSS 2016*, 2016. doi: 10.14722/ndss.2016.23009.

[2] J. Thomas, "The Effect and Challenges of the Internet of Things (IoT) on the Management of Supply Chains," vol. 8, no. 3, pp. 874–878, 2021.

[3] S. A. Pahune, P. Matapurkar, S. Mathur, and H. Sinha, "Generative Adversarial Networks for Improving Detection of Network Intrusions in IoT Environments," in *4th International Conference on Distributed Computing and Electrical Circuits and Electronics*, 2025, pp. 1–6. doi: 10.1109/icdcece65353.2025.11035844.

[4] N. Patel, "Sustainable Smart Cities: Leveraging Iot And Data Analytics For Energy Efficiency And Urban Development," *J. Emerg. Technol. Innov. Res.*, vol. 8, no. 3, pp. 313–219, 2021.

[5] H. Tao, M. Z. A. Bhuiyan, A. N. Abdalla, M. M. Hassan, J. M. Zain, and T. Hayajneh, "Secured Data Collection with Hardware-Based Ciphers for IoT-Based Healthcare," *IEEE Internet Things J.*, 2019, doi: 10.1109/JIOT.2018.2854714.

[6] I. Keshta and A. Odeh, "Security and privacy of electronic health records: Concerns and challenges," *Egypt. Informatics J.*, vol. 22, no. 2, pp. 177–183, 2021, doi: https://doi.org/10.1016/j.eij.2020.07.003.

[7] K. Seetharaman, "Incorporating the Internet of Things (IoT) for Smart Cities: Applications, Challenges, and Emerging Trends," *Asian J. Comput. Sci. Eng.*, vol. 08, no. 01, pp. 8–14, Mar. 2023, doi: 10.22377/ajcse.v8i01.199.

[8] G. Modalavalasa, "Strengthening Threat Detection and Mitigation Strategies in Cybersecurity with Artificial Intelligence," in *2025 5th International Conference on Intelligent Technologies (CONIT)*, 2025, pp. 1–6. doi: 10.1109/CONIT65521.2025.11166691.

[9] V. Prajapati, "Enhancing Threat Intelligence and Cyber Defense through Big Data Analytics: A Review Study," *J. Glob. Res. Math. Arch.*, vol. 12, no. 4, pp. 1–10, 2025.

[10] R. Patel, "Optimizing Communication Protocols in Industrial IoT Edge Networks : A Review of State-of-the-Art Techniques," *Int. J. Adv. Res. Sci. Commun. Technol.*, vol. 4, no. 19, 2023, doi: 10.48175/IJARSCT-11979B.

[11] B. J. Mohd, T. Hayajneh, and A. V. Vasilakos, "A survey on lightweight block ciphers for low-resource devices: Comparative study and open issues," *J. Netw. Comput. Appl.*, 2015, doi: 10.1016/j.jnca.2015.09.001.

[12] P. Pal, P. Chattopadhyay, and M. Swarnkar, "Temporal feature aggregation with attention for insider threat detection from activity logs," *Expert Syst. Appl.*, 2023, doi: 10.1016/j.eswa.2023.119925.

[13] Z. Pan, J. Sheldon, and P. Mishra, "Hardware-Assisted Malware Detection and Localization Using Explainable Machine Learning," *IEEE Trans. Comput.*, 2022, doi: 10.1109/TC.2022.3150573.

[14] S. Seneviratne, R. Shariffdeen, S. Rasnayaka, and N. Kasthuriarachchi, "Self-Supervised Vision Transformers for Malware Detection," *IEEE Access*, 2022, doi: 10.1109/ACCESS.2022.3206445.

[15] H. Sayadi *et al.*, "2SMaRT: A Two-Stage Machine Learning-Based Approach for Run-Time Specialized Hardware-Assisted Malware Detection," in *Proceedings of the 2019 Design, Automation and Test in Europe Conference and Exhibition, DATE 2019*, 2019. doi: 10.23919/DATE.2019.8715080.

[16] S. Yoshimi, Y. Ikegami, R. Negishi, K. Hisafuru, and N. Togawa, "Evaluation of Hardware-Trojan Detection by Ensemble Learning Model for Circuits Inserted with a Trojan by an Automated Framework," in *2025 IEEE International Conference on Consumer Electronics (ICCE)*, 2025, pp. 1–4. doi: 10.1109/ICCE63647.2025.10930147.

[17] A. Moussa and N. Rafla, "Enhanced Hardware Trojan Detection in Chips By Reducing Linearity Between Features," in *2024 IEEE 67th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2024, pp. 985–989. doi: 10.1109/MWSCAS60917.2024.10658816.

[18] K. S and P. E, "Hardware Trojan Detection using Unsupervised Machine Learning Algorithms in the Gate-level Netlist," in *2024 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, 2024, pp. 1–6. doi: 10.1109/CONECCT62155.2024.10677111.

[19] T. Gourousis *et al.*, "Identification of Stealthy Hardware Trojans through On-Chip Temperature Sensing and an Autoencoder-Based Machine Learning Algorithm," in *Midwest Symposium on Circuits and Systems*, 2023. doi: 10.1109/MWSCAS57524.2023.10405958.

[20] V. Sankar, M. Nirmala Devi., and M. Jayakumar., "Data Augmented Hardware Trojan Detection Using Label Spreading Algorithm Based Transductive Learning for Edge Computing-Assisted IoT Devices," *IEEE Access*, vol. 10, pp. 102789–102803, 2022, doi: 10.1109/ACCESS.2022.3209705.

[21] Y. Wang, P. Liu, X. Han, and Y. Jiang, "Hardware trojan detection method for inspecting integrated circuits based on machine learning," in *Proceedings - International Symposium on Quality Electronic Design, ISQED*, 2021. doi: 10.1109/ISQED51717.2021.9424314.

[22] R. Gayatri, Y. Gayatri, C. Mitra, S. Mekala, and M. Priyatharishini, "System Level Hardware Trojan Detection Using Side-Channel Power Analysis and Machine Learning," 2020. doi: 10.1109/icces48766.2020.9137882.

[23] U. A. Korat and A. Alimohammad, "A Reconfigurable Hardware Architecture for Principal Component Analysis," *Circuits, Syst. Signal Process.*, vol. 38, no. 5, pp. 2097–2113, 2019, doi: 10.1007/s00034-018-0953-y.

[24] V. T. Hayashi and W. Vicente Ruggiero, "Hardware Trojan Detection in Open-Source Hardware Designs Using Machine Learning," *IEEE Access*, vol. 13, pp. 37771–37788, 2025, doi: 10.1109/ACCESS.2025.3546156.

[25] Z. Jiang and Q. Ding, "A framework for hardware trojan detection based on contrastive learning," *Sci. Rep.*, vol. 14, no. 1, pp. 1–22, 2024, doi: 10.1038/s41598-024-81473-0.

