SEMARAK ILMU PUBLISHING

Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage: https://jaraset.com/
ISSN: 2462-1943

ENHANCING HARDWARE SECURITY IN INTEGRATED CIRCUIT DESIGN THROUGH PRE-SILICON DETECTION OF HARDWARE TROJANS USING COMPRESSIVE SENSING AND DEEP LEARNING TECHNIQUES

Ritu Sharma [1*], Prashant Ranjan [2]

[1] Research Scholar, Department of Electronics and Communication, University of Engineering and Management, Rajasthan, India; reetusharma310@gmail.com

[2] Associate Professor, Department of Electronics and Communication, University of Engineering and Management, Rajasthan, India; prashant.ranjan@uem.edu.in

ARTICLE INFO

Article history:
Received: 22-04-2024
Received in revised form: 12-05-2024
Accepted: 16-06-2024
Available online: 28-09-2024

Keywords:

Hardware Trojans, Side-channel analysis, Machine learning; IC security

ABSTRACT

The rapid advancement in semiconductor technology has led to an increase in the number of designers creating intellectual property cores, introducing significant hardware security risks in integrated circuit (IC) design. Outsourcing to unreliable manufacturing facilities enables attackers to incorporate harmful logic, known as hardware Trojans (HT), into the original design, posing threats to mission-critical systems. This research proposes developing pre-silicon detection systems using compressive sensing and deep learning algorithms to identify and eliminate HTs before ICs are deployed in crucial applications. By analyzing transition probabilities and employing adaptive learning, the detection strategy aims to improve hardware security, ensuring the integrity and reliability of ICs in various sectors, including medical, military, and automotive industries. This optimizes the chance of detection metric in the first phase. Additionally, the power profiles are tracked and measured at different times in order to rule out the golden circuit for anomaly detection. Recovery-based technique is used at the output power profiles for examining the Trojan at shorter time periods in order to demonstrate the efficacy of the CS algorithm. Using a decoding technique, the compressed power signals are recovered, and the correlation coefficient is examined. In the pre-silicon validation stage, it is found that compressed input patterns and power profiles at shorter time stamps make it easier to distinguish between the presence of a danger module and a regular circuit, which lowers the computational cost of vector production.

This model can identify Trojans in complicated circuits and handle a variety of Trojan kinds during training. In the third phase, deep learning models are used in an effort to more effectively automate feature extraction and modeling procedures in the training phase. With a higher prediction rate and maximum detection accuracy for complicated circuits, this method efficiently extracts a compact collection of characteristics. In order to handle the high dimensional data set in deep architectures, the extracted feature sets are dimensionally enhanced in the pre-processing step by modifying the deep convolutional generative adversarial network model.

INTRODUCTION

The need for high-profile and high-quality integrated circuits (ICs) is growing due to the rapid growth of system on a chip (SoC) hardware unit design flow. By contracting with many outside vendors to handle the IC design and chip fabrication, it has become more efficient and possible. By adding harmful alteration known as hardware Trojan (HT) (Bhasin et al., 2015) into the IC devices, such third-party intellectual property providers that are engaged in

complicated IC supply chains introduce security risks. Apart from disrupting the logic's operation, it might also reveal a private signal, cause a denial of service attack, or lower the system's performance (Chakraborty et al., 2009). These hardware Trojans are designed to be silent during regular operations and to activate only in response to uncommon circumstances, allowing them to avoid detection during the formal design testing and verification process. It is difficult to identify these hardware Trojans and to prevent the Trojans from being inserted into the IC design. Therefore, the fundamental premise of our approach is the identification of these malignant logics by the adaptation of appropriate detection systems.

Design time (Yang et al., 2020), test time (Mukherjee et al., 2022), and run time methods (Pavlidis et al., 2022) are the three categories into which the current HT detection systems are divided. The procedure of design time detection is solely pre-silicon based and is implemented prior to the manufacturing stage. Post-silicon methods are the detection method used after fabrication, and it is carried out during testing. During the design phase, the run-time methods are modified so they may be used in real-time applications after manufacturing. In order to preserve the integrity and reliability of IC, it is thus preferable to detect potential risks in netlist files early in the manufacturing process, even if the circuit design may also be modeled as built IC. Our study proposal focuses on pre-silicon ideas, which may be categorized into two phases: pre-silicon validation and pre-silicon detection.

In the suggested study, functional testing principle-based validation is combined with a pre-silicon validation approach. Generated compressed patterns are sent to netlist files in this security validation procedure in order to activate the Trojan modules; these patterns are only activated for uncommon patterns when functional authentication is verified prior to silicon development. As a result, it takes a lot of work to extract the appropriate test patterns for triggering the triggering circuit for complex integrated circuits. Reference circuits are also needed for this procedure in order to identify design anomalies. The pre-silicon detection approaches in the proposal, which are more adaptive to changing threats on gate level netlists and scalable to huge feature sets, are motivated by this circumstance. In the second phase, machine learning algorithms are encouraged to identify HT based on structural and functional properties, in order to manage a range of minute Trojans in large circuits. The deep learning model used in the pre-silicon based detection procedure is very effective at processing and extracting useful characteristics from large, complicated data sets. By lowering the amount of work that must be done by hand for analysis, it may also automate the features extraction process. As shown in the third section of this thesis proposal, the automated pre-silicon detection technique may reduce resource consumption while enhancing hardware Trojan detection efficacy and efficiency. Entrusted merchants are involved in the design and production of complicated circuits and integrated circuits (IC) because to the need for quick and flexible creation of intellectual property (IP) cores (Bhunia et al., 2013). The hardware Trojan lurks in the shadows and tampers with the IC process at different stages. As seen in Figure 1.1, the design and manufacture stages are particularly susceptible to these malevolent alterations. These design-stage vulnerabilities might lead to denial of service, alter the internal network, or let confidential data to leak (Bhunia, et al., 2013). The three key stages of design abstraction are the system, behavioral, and logical levels. During these phases, an attacker may introduce malicious logic or alter the internal logic of the hardware design. Additionally, Trojan modules are introduced into gate level and register level netlists by third-party electronic design automation (3P-EDA) tool suppliers.

These third-party IP (3P-IP) invasions occur on hard, firm, or soft IP. Hard IP is defined as the physical layout design, whereas soft IP deals with Verilog or VHDL. Firm IP is associated with gate level netlists. Trojan assaults are less common in the manufacturing stages of integrated circuit life cycles. Consequently, only a certain kind of HT is placed into the original design during the wafer probe and production process in the fabrication stage, which are referred to as semi-trusted stages.

REVIEW OF RELATED LITERATURE

The Trojan detection method based on side channels examines anomalous test circuit characteristics. Transient currents, power profiles, and route latency are among the circuit characteristics that are tracked in order to identify any anomalies in the integrated circuit design. (Maniriho, et al., 2022) proposes a static current analysis that tracks current measurements at various points along the test circuit's surface. A partitioning based functional testing technique was provided by (Chen, et al., 2023). In this approach, the circuits are divided into areas, and the instantaneous currents of the regions are examined by offering appropriate test vectors for HT detection. Hence, it is harder to find tiny Trojans in the original design and post-silicon based detection methods are more vulnerable to process noise. This drives the pre-silicon method, where the two main categories are seen to be validation and detection.

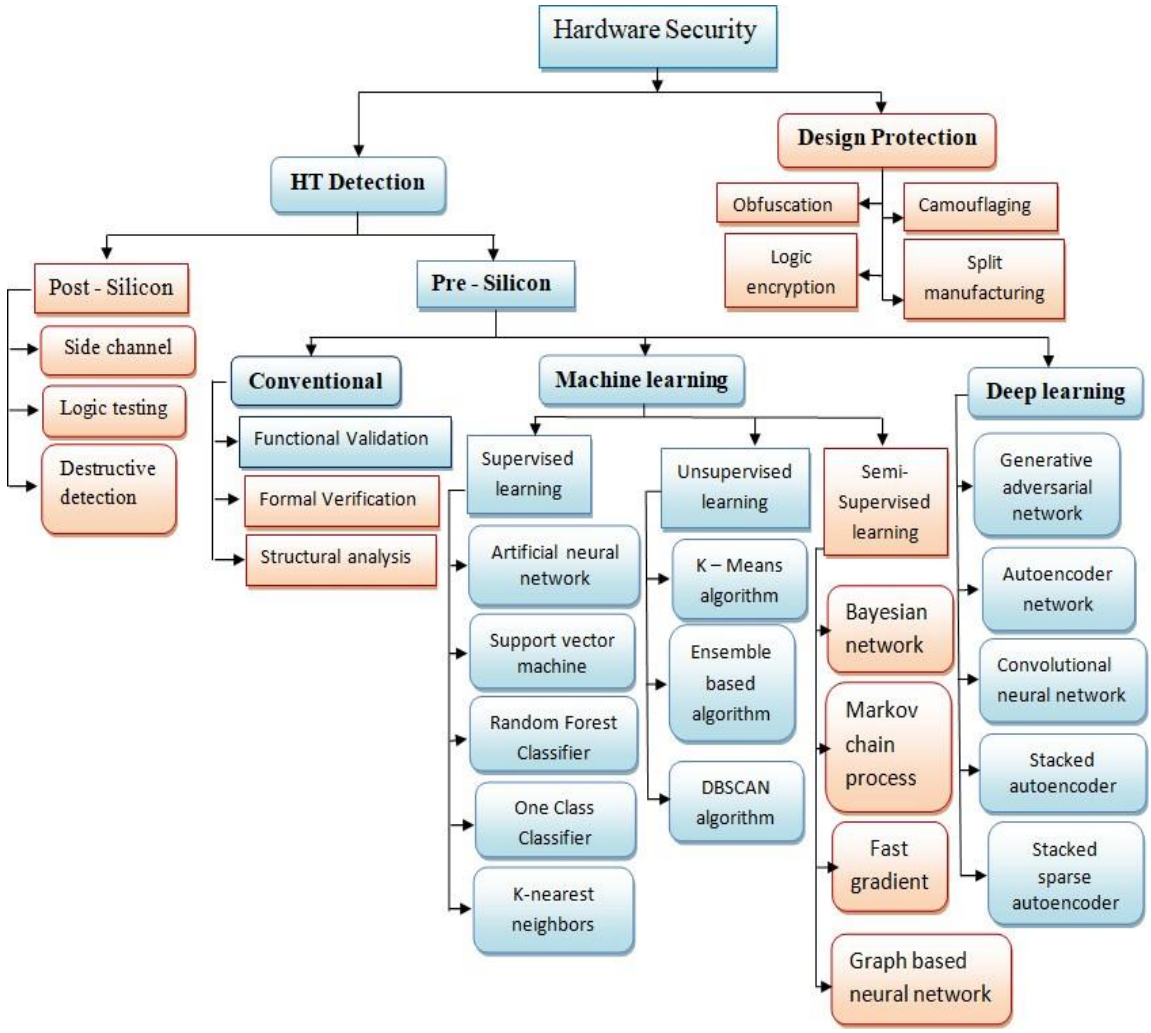


Figure 01: HT detection and design for security using hardware security schemes

In order to identify the harmful nets, the important elements from the input feature set are taken from the netlist and used to create a scalable learning algorithm for HT identification. Formal

logic verification and functional logic simulation are used in the pre-silicon conventional detection approach to find abnormalities in the design. Because the circuit designs are intricate, it is more challenging to find the malicious logics using the traditional method. Furthermore, Trojan module types and their effects are constantly changing. Therefore, at the pre-silicon stage, adaptive detection approaches such as those based on machine learning and deep learning are explored to identify hardware Trojans. Therefore, in order to produce a reliable integrated circuit (IC), it is necessary to integrate HT detection techniques and guarantee reliability at the design phase of the IC life cycle.

A pre-silicon learning model with improved classification accuracy and resistance to process noise, supervised learning is useful for small data sets. Thus, a machine learning strategy for identifying HTs is put out in (Jap, et al., 2016), whereby a number of supervised learning algorithms are included in machine learning schemes. The findings demonstrate that the algorithm achieves maximum accuracy while taking into account substantial noise, and they are verified for both designs with and without the golden circuit. The model that is supervised is divided into K-nearest neighbors, decision tree models, one-class classifiers, artificial neural networks, and support vector machine models are some methods for HT detection.

An artificial neural network (ANN) that monitors the power use of nonlinear data characteristics for Trojan identification was suggested by Liu et al. (2019). The model is built by the authors using a variety of networks depending on the connection pattern, and back propagation is used to update the parameter weights. The performance of ANN models for different strategies is assessed using the statistical indicator metric, which is presented. This detection technique works better with numerical issues and demands parallel computing capacity. An artificial neural network for identifying the malicious logic in the circuit under test is described by Wang et al. (2016). The power profiles for the test circuits are obtained and examined using a feature extraction method developed by the authors that is based on a mathematical model. The suggested plan is put into practice using FPGA to confirm the neural network's efficacy. As a result, each issue statement must be translated into a numerical value. This situation causes the data set used to train the model to be limited.

According to the authors, the support vector machine (SVM) model is used to discriminate between malicious and legitimate nets. The binary class classification model described by (Inoue, et al., 2018) uses the SVM learning approach to identify Trojans. In order to solve the mathematical optimization issues associated with artificial neural networks, this model maximizes the interval of the feature set. As the circuit net becomes bigger, the detection accuracy decreases and clusters start to overlap. Consequently, feature samples may overlap when the feature dimension is greater than the amount of training, which limits this model. An adaptive optimization approach was presented by (Xue, et al., 2017) to categorize Trojan nets from the gate level net list. By using matched algorithm pairs to increase the detection rate of Trojan nets, the authors extract the transient power profiles and analyze the erroneous classification rate of different algorithms. In order to identify Trojan nets utilizing a cost-sensitive detection methodology, this method additionally incorporates a cost matrix. A boundary net-based classification approach for HT detection in the gate level net-list is described by Hasegawa et al. (2018). During the training phase, the authors take the new set of boundary nets out of the circuit that is being tested. Performance degrades when classifying outliers using the restricted Trojan features that were retrieved during the training phase of this

technique.

By reverse engineering the IC, (Bao, et al., 2014) presented a detection strategy using a one-class SVM classification method. The one-class classifier model is trained using the appropriate characteristics that the authors extract from the IC's physical configuration. By identifying the unknown IC as a Trojan-infected circuit, it is deduced that the model is trained using the best data that was taken from the feature selection procedure and offers a higher level of accuracy. A unique detection method based on a one-class SVM classification strategy is described by (Jap, et al., 2016). In addition, the authors develop an unsupervised learning scheme that meets the golden chip condition of the supervised method. This scheme is more resilient when compared to the template-based Trojan detection strategy. According to Li et al. (2016), test data may be distinguished between a Trojan or golden net using a one class categorization.

Methodology for machine learning model

Finding harmful logic in gate level design is the primary objective of the suggested technique. The implementation is divided into three stages: the creation and extraction of sparse features, the HT detection stage, and the validation stage. The suggested model accepts the gate level netlist file as its input. In addition to the manually designed technique used for feature extraction, side channel features are produced by synthesizing the Trojan-inserted gate level netlist and the golden gate list. In order to extract the best features, a stacked sparse autoencoder learning model is suggested during the pre-processing phase.

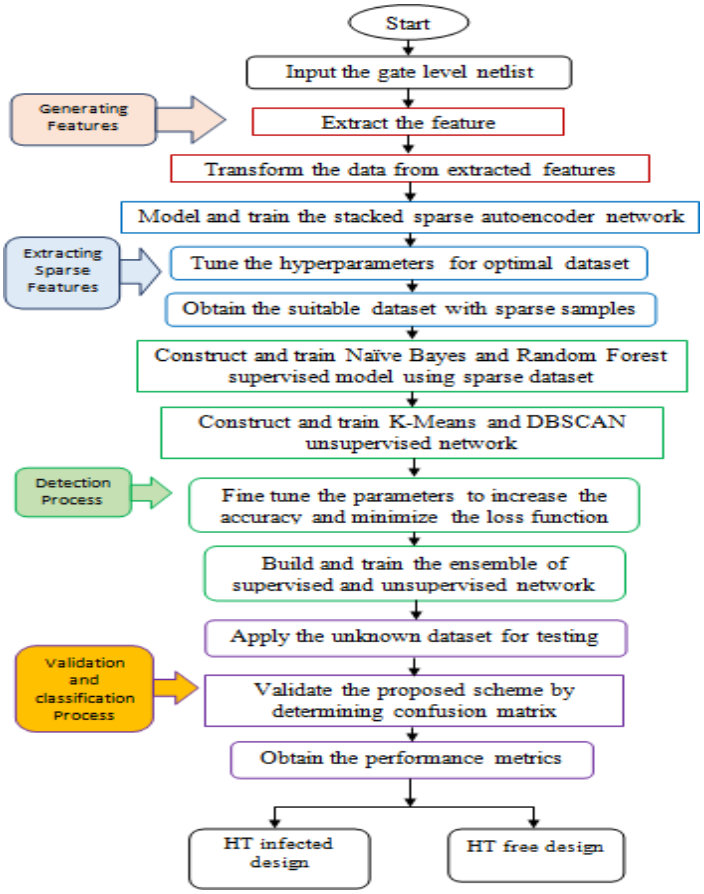


Figure 02: The general flow of the suggested plan

Therefore, unsupervised models are more helpful for identifying the hardware Trojan at the pre-silicon stage because they can identify abnormalities in the complex design without requiring previous knowledge of the harmful logics. Additionally, the malicious logic is made to display intricate patterns that might be challenging to extract using a single unsupervised model. As a result, an ensemble-based learning approach is used in the pre-silicon detection stage to increase accuracy. It consists of many models, with the ensemble model's ultimate choice being based on the average response of these models. Consequently, the integration of unsupervised models is used to both forecast Trojans and get superior precision.

Generating gate level net list

The first step in the suggested method is the creation of a gate level netlist. To create the Trojan netlist, the Trojan modules are created and added to the original blueprint. The Synopsys tool, used in the feature extraction process, synthesizes both the original and the Trojan netlist. In order to create the whole dataset, the characteristics are retrieved for both the malicious and the golden netlists at a certain moment in time known as the target node, as shown in Table 1.

Table 1: Features generated from the gate level netlist

Feature		Description
Hand crafted Features	Primary input	The lowest level considering the primary Variable input node to the target node *m*
	Primary output	The lowest level of the circuit from the Output variable net to the target node *m*
	Digital gate fan-in	The maximum value of inputs of the digital gate fara way from the target node *M*
	Digital gate fan-out	The maximum value of outputs of the digital gate departed from the target node *m*.
	Level	The quantity of digital gates that are dropped between primary variable in puts To the target node *m.*
	Connectivity	The total number of net to the Target node *m*

Data base generation

By extracting the necessary number of features, the data base for training the machine learning algorithms is created for the Trojan netlist and golden model. To improve the distinction between Trojan features and regular nets, the side channel parameters are further taken out of the synthesis tool. A useful dataset is created by combining all of the manually created features with the side channel parameters. This dataset is then converted and saved as a CSV file. The final data set that is given to the testing phase includes the side channel parameters from the synthesis tool, such as transition probability, toggle rate, switching power, net load, pins, and resistance, as well as the hand-crafted features like primary input, primary output, level, connectivity, fan-in, and fan-out.

Pre-processing unit

Excessive dataset modeling for the machine learning network might result in overfitting, which deteriorates algorithm performance. Furthermore, in a real-time environment, extracting important characteristics from the collection without labels is a challenging task. In order to solve this issue, the pre-processing stage of the implementation of the stacked sparse autoencoder (SSAE) network successfully extracts appropriate features that represent the Trojan properties. Additionally, the clustering algorithm's accuracy is increased by the SSAE network.

Stacked Sparse Auto encoder (SSAE)

The sparse auto encoder network (SSAE) architecture seen in below Figure is made up of many hidden layers that propagate via different input feature neurons. The last sparse neurons extract the effective features and minimize the error function.

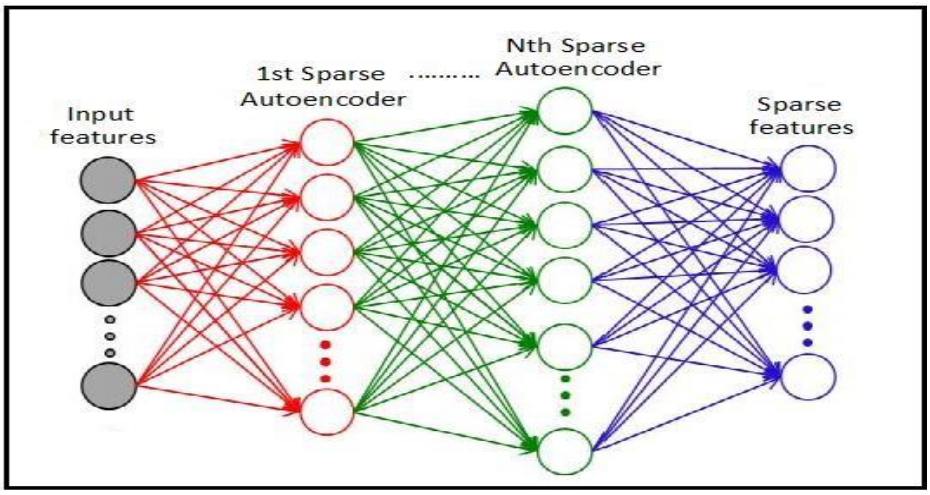


Figure 3: Architectural view of stacked sparse auto encoder (Bouchou,*etal.*,2017)

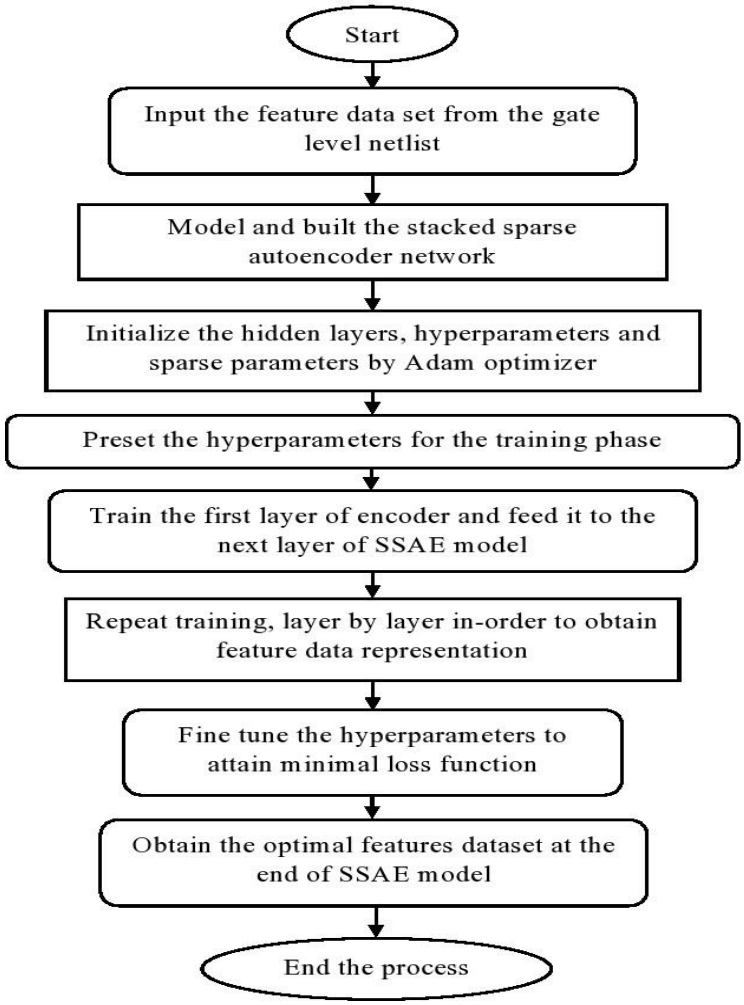


Figure 4: Flow diagram for the pre-processing stage's optimum feature extraction

HT Detection process based on machine learning approach

In the suggested flow, the supervised and unsupervised learning algorithms are built in two stages: model establishment and pre-training to distinguish between the real gate-level netlist nets and suspected Trojan nets. The pre-processing unit's sparse feature set is used as the machine learning models' data set. In the first step, machine learning models are constructed based on the data set, and in the pre-training phase, these models are trained using the data set. During this training phase, the related models' hyperparameters are also adjusted and fixed. In the validation step, the suggested model is examined and verified using a range of performance indicators for the unidentified gate level netlist.

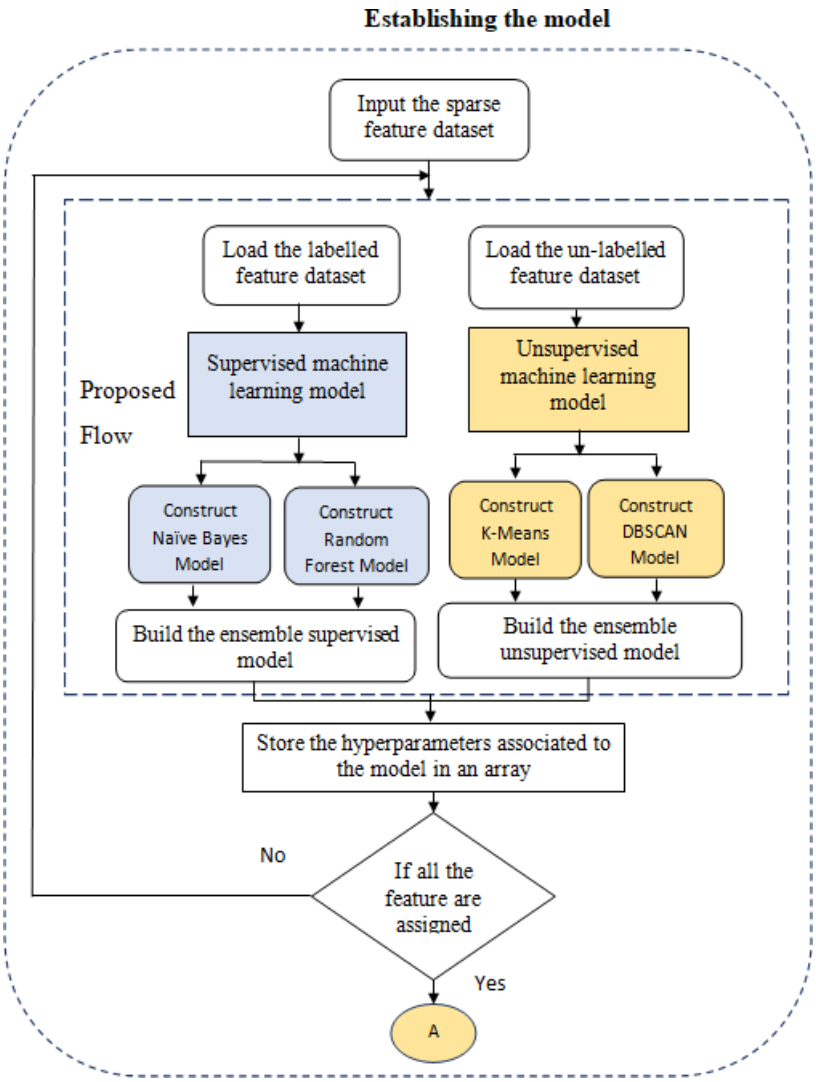
Supervised model

The characteristics of the data set with labeled information of each network indicating whether or not it is infected with Trojans are given to the supervised networks. In the dataset, the Trojan free nets in the circuits are marked with label 0 while the malicious nets in the gate level netlist are tagged with label 1. The training model receives this labeled data set, and using the features it has learnt, it forecasts the output for the unknown data that is supplied during the testing stage. In order to find the abnormality in the netlist, the most popular supervised models, including Naïve Bayes (Huang et al., 2020) and Random Forest algorithms (Hasegawa et al., 2017), are used in this step. In comparison to a single learning model, the presented work aims

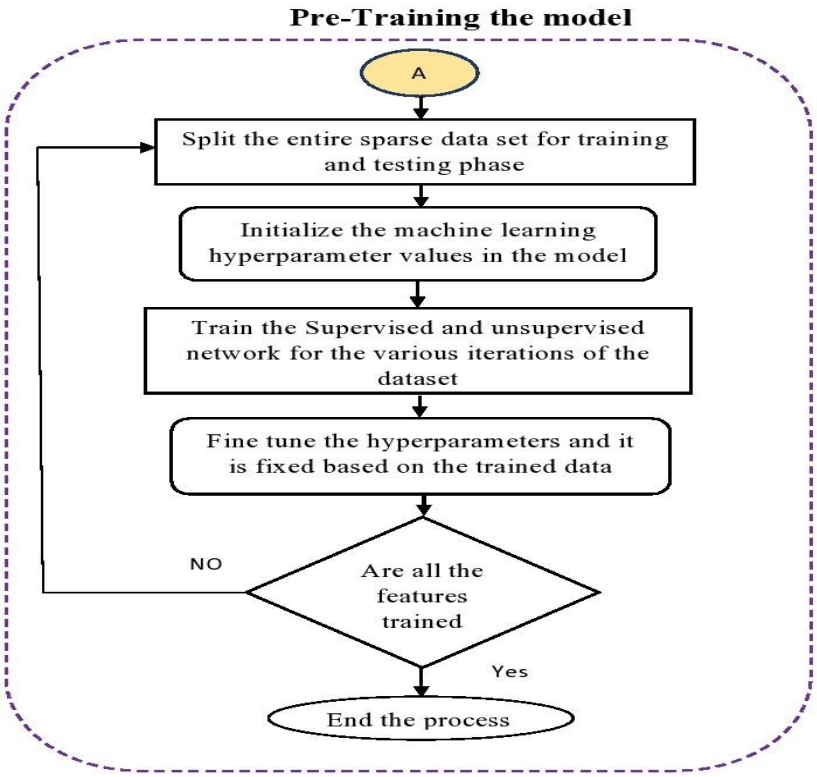
to create an ensemble of supervised models, which improves accuracy and facilitates better anomaly detection.

Naïve Bayes approach

Using linear properties in rth measurement for prediction, the supervised learning model Naïve Bayes makes predictions. In addition to updating the probability, this method analyzes the conditional property of independence amongst sparse feature datasets. Pseudocode 5.1 describes the structural flow of this technique and how it relies on the probability parameter. The Naïve Bayes method predicts the cluster of unknown feature datasets using the Bayes theorem's probability factor. Additionally, real-time data that is immune to extraneous data characteristics is predicted using this approach. According to Huang et al. (2020), it is also used to forecast the likelihood of multi-class issues based on several features and a high computation rate.



(a)



(b)

Figure 5: The general procedure for hardware Trojan detection: (a) Model establishment (b) Model pre-training

Pseudo code 5.1 Naïve Bayes Algorithm

1. Access the sparse feature dataset for training, Y = {f1, f2, f3,...,fr}.
2. Determine the feature variables' mean parameter and standard deviation metric.
3. For each cluster, use the gauss density function equation to get the probability of the feature data set Fi.
4. Step 3 should be repeated until all feature variable probabilities have been determined.
5. Determine the probability of every cluster.
6. Find the highest probability of grouping the gate level netlist.

Random forest approach

As seen in Pseudocode 5.2, the random forest model, which is also a supervised learning model, functions as a non-linear classifier. This model is made up of several decision trees that use random feature extraction and bagging perception to make predictions. This bagging, which is regarded as Bootstrap aggregation, is used by the random forest. Consequently, these random samples are used to generate each model using a replacement process called Bootstrap. These samples are used to train each decision tree, and the ultimate choice is determined by aggregating the outcomes of all the choices. This approach is thus predicated on the idea that an ensemble prediction outperforms an individual prediction. According to Hasegawa et al. (2017), this model is often used in classification and regression-based issues.

Pseudo code 5.2 Random Forest algorithm

1: From the provided feature data set, Y = {f1, f2, f3,...,fr}, choose the random networks.
2. Create a decision tree for every feature data set net.
3. Acquire the forecast outcomes for every decision tree.
4. Determine the maximum voting algorithm for every outcome that is anticipated.
5: Ascertain the model's final prediction by obtaining the greatest number of votes for the predicted outcome.

Unsupervised learning model

A machine learning technique known as an unsupervised learning model uses the unlabeled data from a feature data set to train the model. The netlist is given to the unsupervised model without any explicit label information. This model's primary objective is to analyze the characteristics of each feature and the groups that belong to it based on shared characteristics. Furthermore, the majority of real-time applications work with enormous data sets, and categorizing such a vast collection of attributes takes time. Thus, the idea of the unsupervised learning model resulted from the need of manually establishing the labels for complicated circuits. Furthermore, the suggested model is also capable of handling other Trojan nets, which are not required to be employed in order to train the data set. Considering that the discovery. The prediction accuracy will be lower if the abnormal net does not exist in the training data set. The scheme of abnormal nets in the supervised technique may recognize comparable abnormal nets that are learnt during the training phase. The suggested techniques use an unsupervised learning model to improve the detection rate and group the pre-defined netlist into the appropriate cluster in order to solve this problem. In this study, an unsupervised learning model based on DBSCAN and K-Means is presented for the detection process.

K-Means approach

K-means clustering, a prominent clustering analysis technique in unsupervised machine learning, divides the input sparse feature samples, represented as Y = {f1, f2, f3,..., fr}, into n distinct clusters. whereby each sample in the observations is assigned to a cluster according to the closest mean. Based on the feature set, the centroids and cluster sizes in K-means are initialized. For every feature, the distance metric is computed using the original centroid value.

Determine the centroid value and check to see whether the original and reallocated centroid are the same if all of the features in the set are assigned to the cluster. K-means uses an iterative analytic procedure to cluster the characteristics as normal or diseased. This model's primary goal is to measure each pre-defined cluster's centroid. As a result, setting the centroid is a difficult issue for real-world applications, and the ideal approach is to assign them as far apart as feasible. Additionally, every sample in the feature pool is linked to the nearest centroid. When every sample has been assigned to the appropriate cluster, the group's new centroid is computed. This algorithm repeatedly applies the reallocation approach to enhance the splitting of data points from one cluster to another.

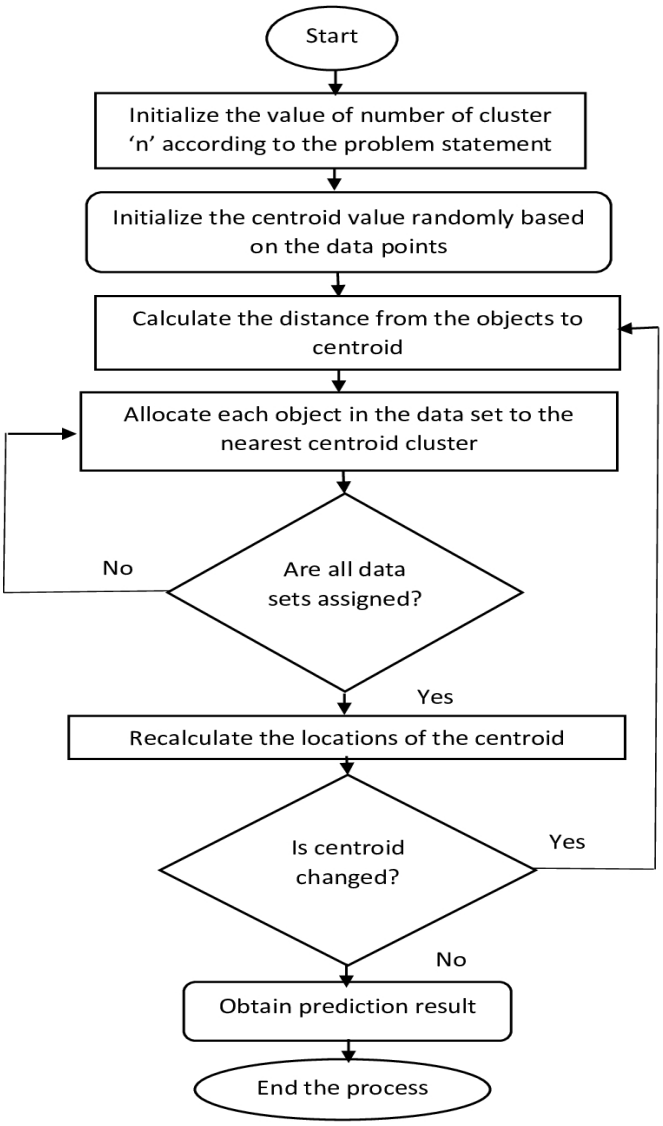
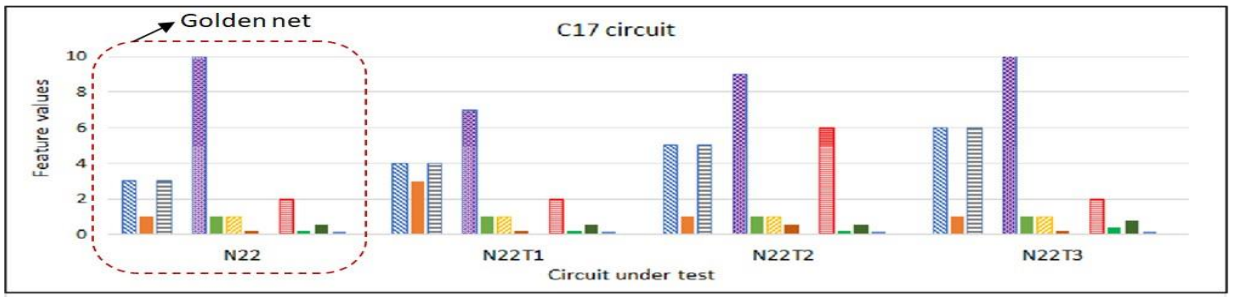


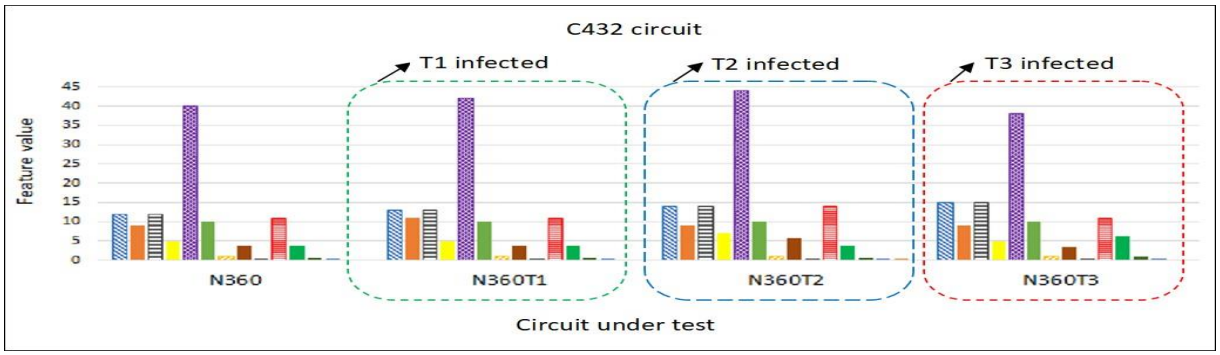
Figure 6: The K-Means algorithm flowchart

Simulation result and analysis

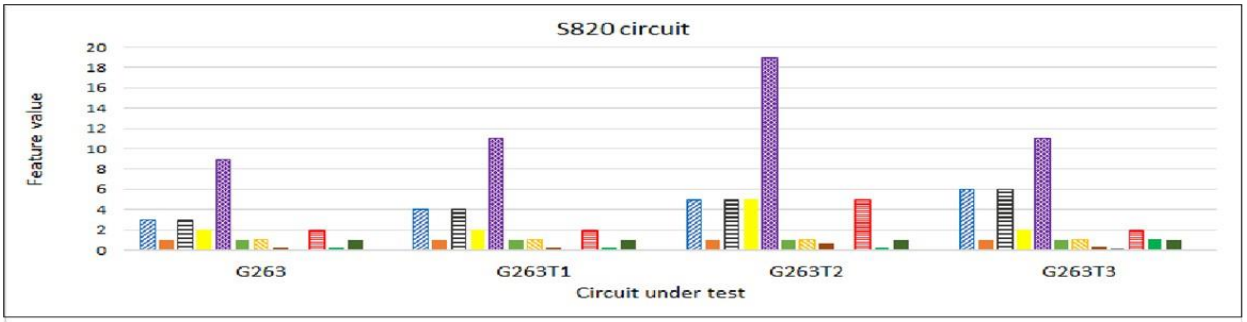
To find the malicious Trojan nets in the ISCAS'85, ISCAS'89, and Trust-HUB benchmark circuits, our suggested approach is put into practice. The original netlist contains several kinds of digital threat modules. When activated, these cyberthreats have the potential to change the original design, and Trojans placed in Trust-HUB circuits may cause functional changes, denial of service attacks, or the loss of confidential data. The Synopsys tool is used to synthesis both the original and the infected designs, and Python scripts are used to carry out the whole process. The precise feature value differs across different kinds of Trojans and authentic nets. Here, a representative C6288 circuit from the ISCAS'85 benchmark is examined using a standard net and several Trojan insertion net types. The factors that are impacted by combinational Trojans in this study include connection, principal output, and toggle rate, whereas sequential Trojans significantly alter the parameters that are affected by load, pins, and switching power. Resistance, static probability, and total load are all affected by the always-on Trojan, but the main characteristics that change when any kind of Trojan is inserted are level, score, and principal input.



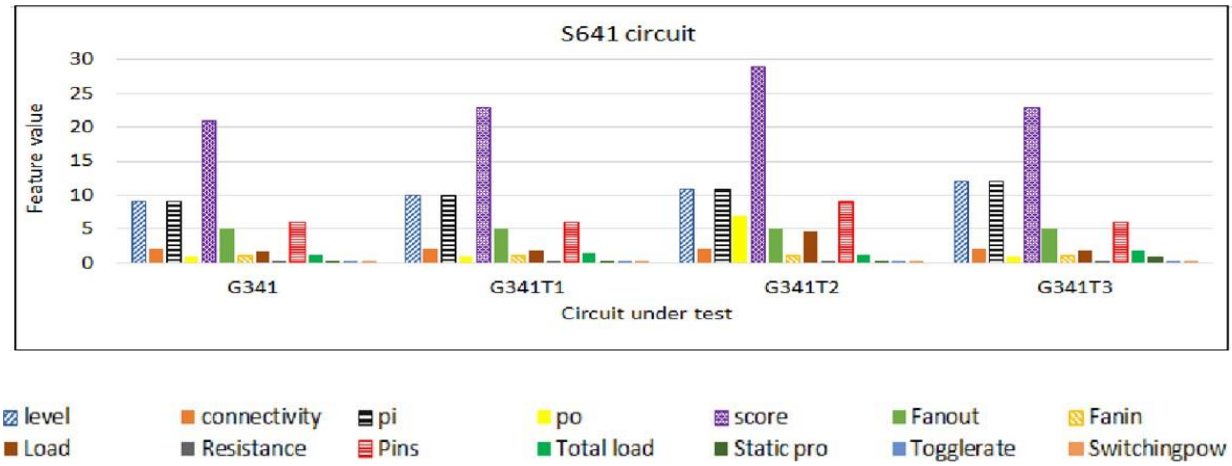
(a)



(b)

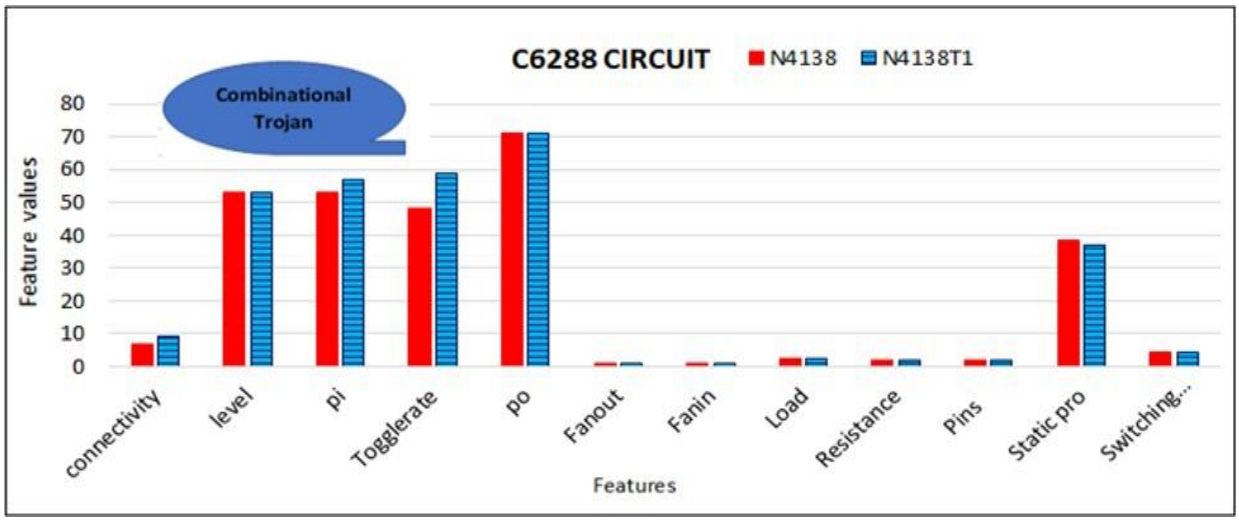


(c)

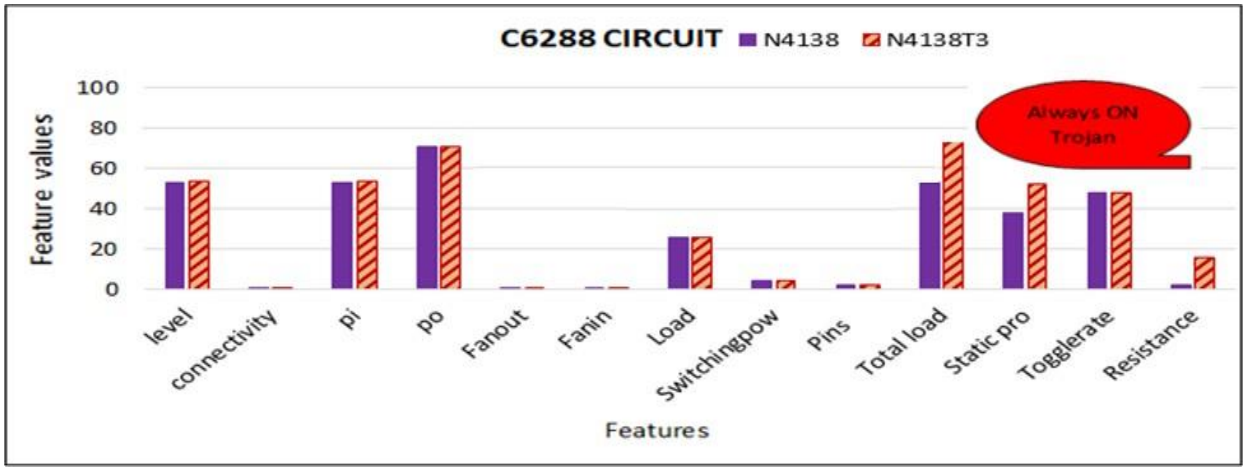


(d)

Figure 7: Report on the analysis of feature values on Trojan nets for the T1-Combinational Trojan, T2 - Sequential Trojan, and T3 - Always-ON Trojan on the benchmark circuits C17, C432, S820, and S641.



(a)



(b)

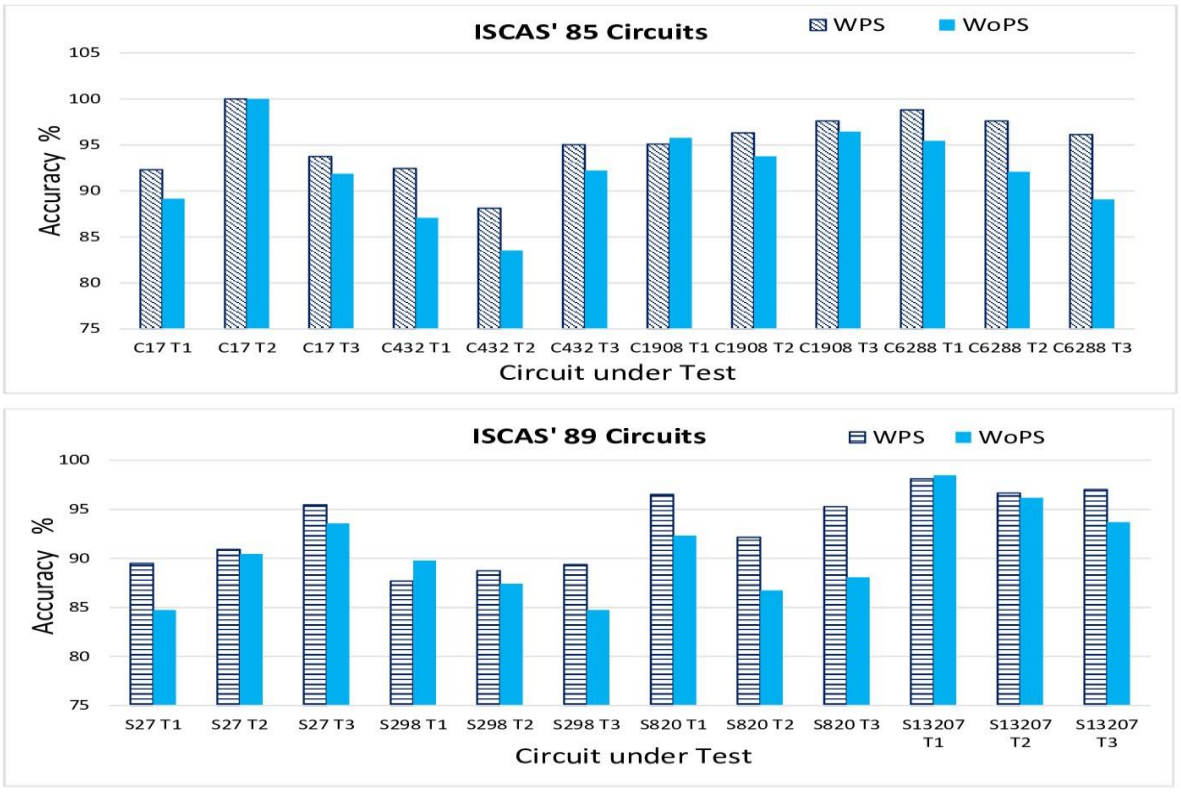
Figure 8: An example of a C6288 circuit feature analysis shows that the circuit is (a) combinational (b) sequential

The statistical analysis of the Trust-HUB circuits' malicious and legitimate networks. The study focuses on Trust-HUB circuits that are compromised by Trojans. These Trojans possess several input trigger events, and they become active when the Trojan module fulfills the internal state's triggering requirement. The nets within the Trojan module are regarded as harmful in this analysis, but the nets outside the Trojan modules are regarded as legitimate nets in the gate level netlist.

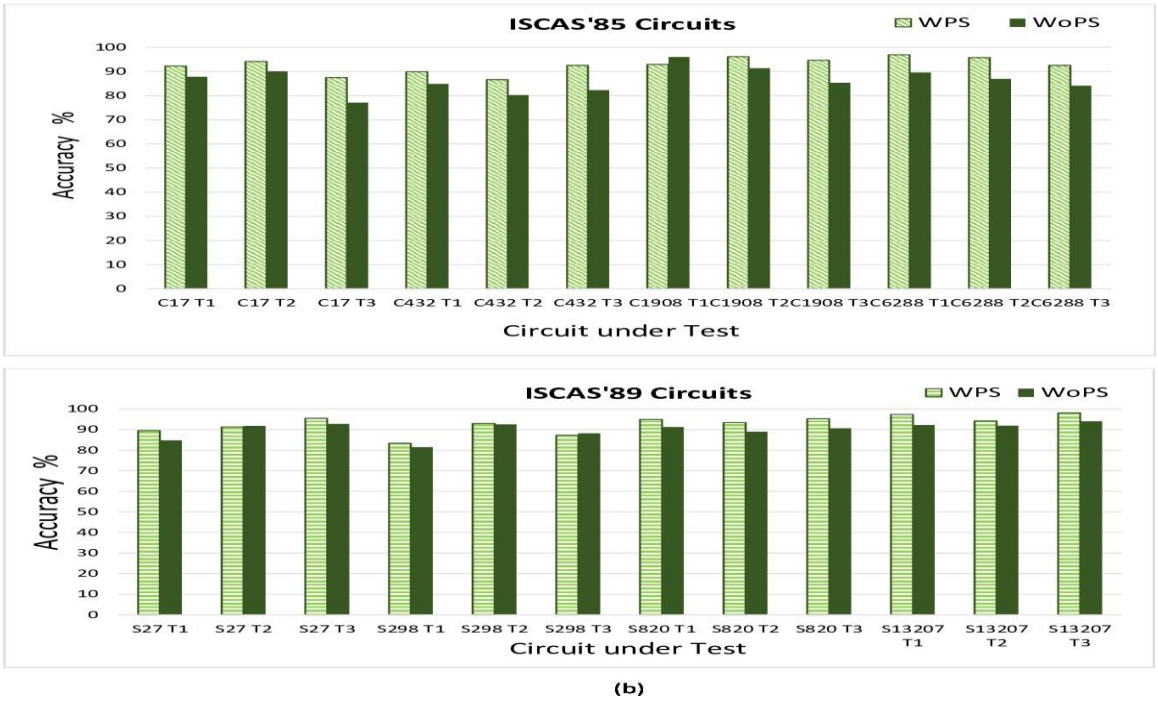
Influence of pre-processing

It makes natural sense to assess the feature distribution using the data pre-processing step. Thus, the framework uses the stacked sparse autoencoder (SSAE) approach to remove the offset caused by related and less relevant features. To train the model, the SSAE chooses the best sparse features that extract and disclose the largest discrepancy features. As a result, the sparse region contains a distribution of the normal and Trojan nets of the data set that this approach recovered. Because the autoencoder only takes into account important characteristics that might lead to a discriminative Trojan prediction for an unknown circuit during training. In

order to choose sparser and more influential features from the original feature set during the training phase, a stacked sparse autoencoder network is created. This network learns all feature categories on its own.



(a)



(b)

Figure 9: Pre-processing Accuracy Metric Analysis: (a) DBSCAN model; (b) Naïve Bayes

The average accuracy of the DBSCAN learning model for the ISCAS'85 circuit is deduced to be 92.19% in the absence of a pre-processing algorithm and increases to 95.26% in the presence of a pre-processing method in the suggested flow. Comparably, ISCAS'89 circuits achieve 90.51% with no pre-processing and 93.01% using an algorithm for pre-processing. Additionally, it is noted that for ISACS'85 circuits, the average accuracy of the Naïve Bayes method is 89.30% and 92.65%, respectively, with and without pre-processing. Furthermore, for

ISCAS'89 circuits, it obtains 90.01% without pre-processing and 92.74% with pre-processing. It is thus seen that the efficient feature set retrieved from the pre-processing unit is used to train both the supervised and unsupervised models, improving the average accuracy in comparison to the traditional feature set. To improve prediction quality and train learning models, feature selection analysis is necessary. The SSAE model therefore demonstrates that the suggested approach increases accuracy by automatically extracting the appropriate features to train the model and without the need for human feature selection involvement.

Performance Analysis of classic supervised algorithms

The mean true positive rate (TPR) for the suggested Naïve Bayes algorithm is 85.49%, whereas the TPR for the proposed random forest algorithm is 81.78%. On the other hand, the mean true negative rate (TNR) for random forest and naïve bayes is 98.35% and 95.39%, respectively. Additionally, it attains an average accuracy of 94.76% for random forest and 86.45% for Naïve Bayes. To be more precise, the supervised learning method yields an average F1-score of 87.24% for random forest and 84.85% for Naïve Bayes. The average accuracy of supervised learning, taking into account ISCAS'85, ISCAS'89, and Trust-HUB circuits, is 92.54% for Naïve Bayes and 94.80% for random forest. Thus, whereas TPR alone performs well in Naïve Bayes, the random forest model of supervised learning classifies the nets better for the unknown gate level net-list. The simulation demonstrates how well the Trojan networks are clustered in the net-list classification phase when supervised information is used, where data labels are supplied during the training phase.

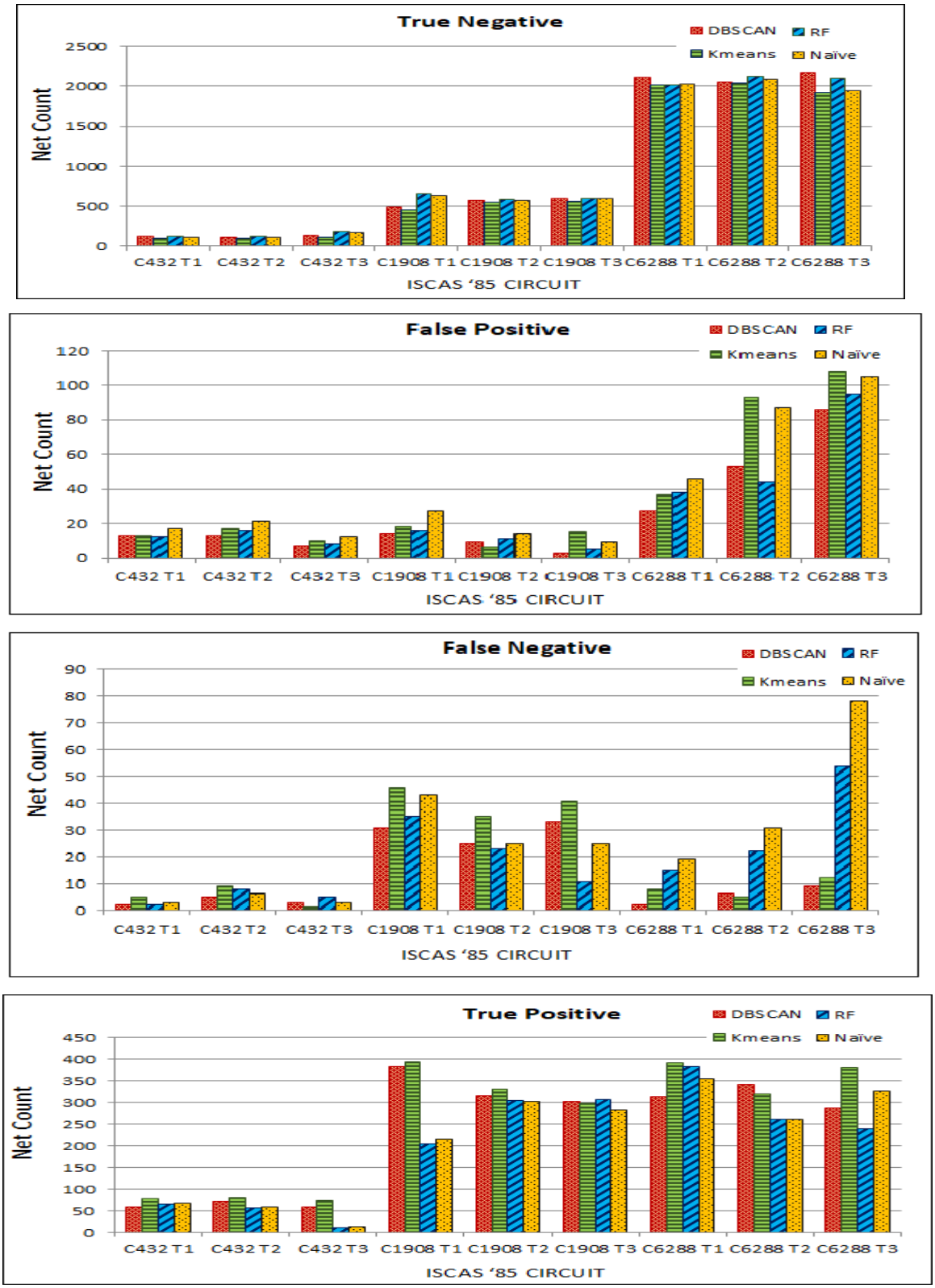


Figure 10: Performance indices for the ISCAS'85 circuit for True Negatives, False Positives, and False Negatives

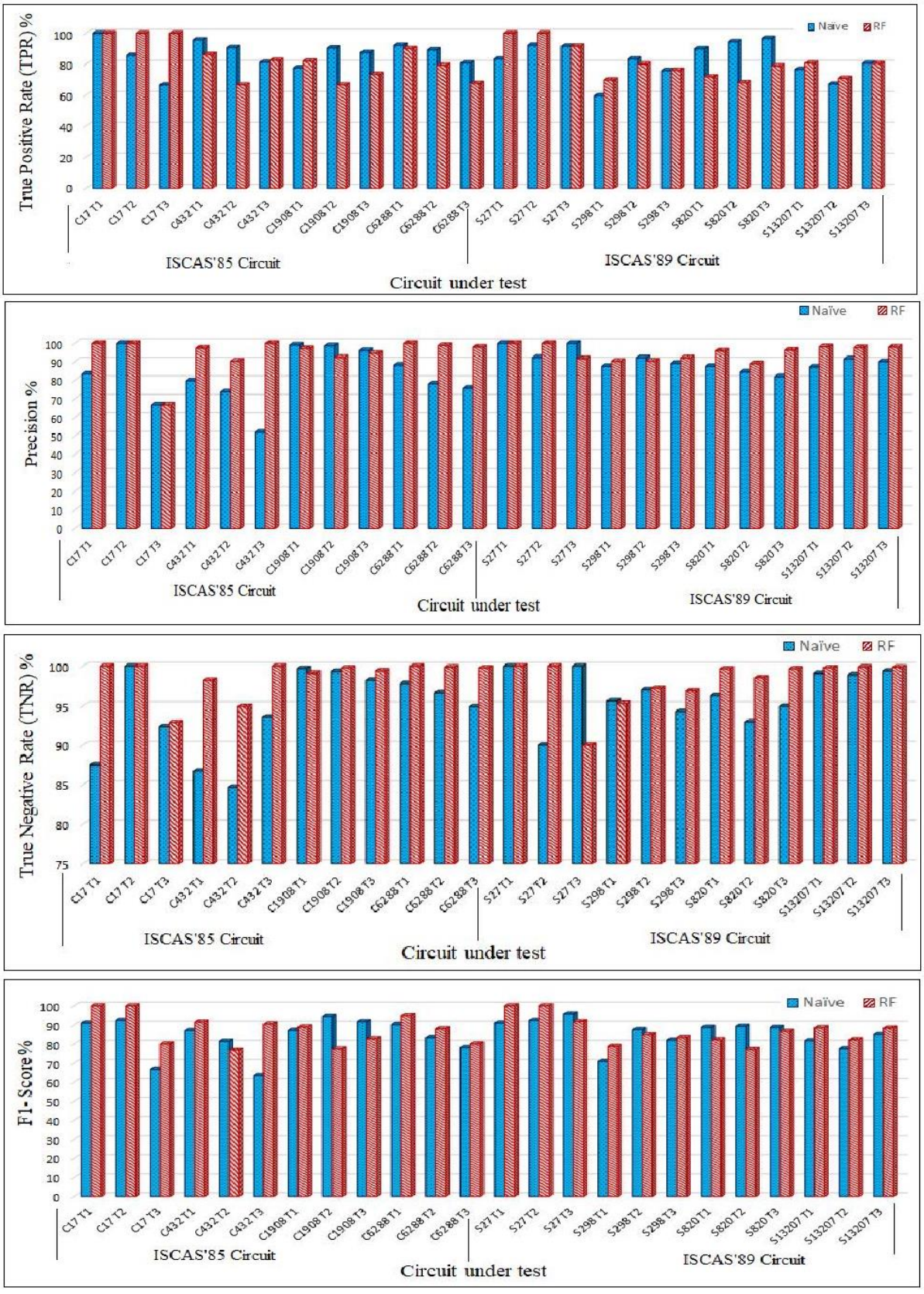
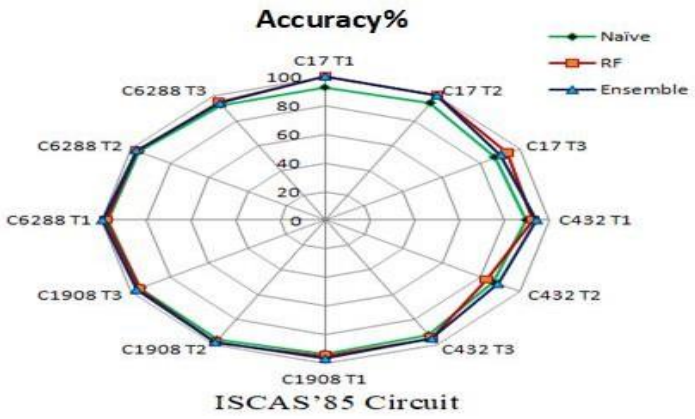


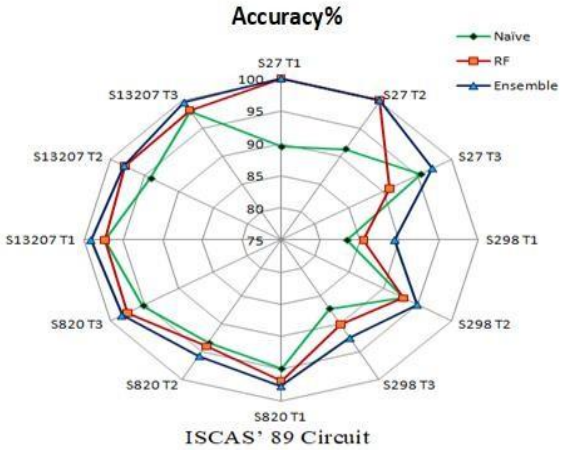
Figure 11: Metrics of the supervised learning model's performance

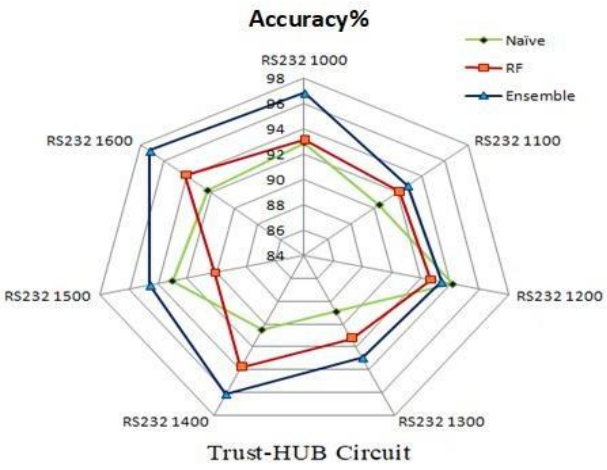
Analysis of Ensemble based supervised algorithm

The accuracy parameter detection results for the supervised ensemble network model. In the study, the Random Forest and Naïve Bayes classifier algorithms are used to compare the accuracy of the ensemble model. Comparing the accuracy of the proposed ensemble model to the current model, it is deduced that the latter achieves a lower detection rate than the former. The ensemble model's average accuracy for the intricately designed benchmark circuits is 96%, which is greater than that of the other supervised models.



(a)





(b) (c)

Figure 12: Comparing the Supervised Ensemble Network's Accuracy

Conclusion

One important consideration while building integrated circuits for system-on-a-chips (SoCs) is security in a variety of hardware applications. A range of supervised and unsupervised algorithm models are put out to forecast malevolent nets inside the netlist. The results of certain ISCAS'86, ISCAS'89 benchmark circuits, and Trust-HUB circuits are accurately

predicted by the ensemble-based machine learning approach, and these predictions are also verified by performance measures. The features for the netlist are extracted, and appropriate feature datasets for the relevant threat modules are then looked through in order to train the learning model. Algorithms are used to extract the manually created characteristics, and some side channel features from the synthesis tool reports are included in the suggested flow. The characteristics for both the real netlist and the many Trojan varieties known as Trojan netlists that are added at nets with low transition probabilities are also retrieved. The feature values of golden nets are compared with combinational, sequential, and always ON Trojan nets throughout the feature generating phase. It has been deduced that functioning Trojans periodically raise the malicious networks' feature value based on the payload. Consequently, the structural and power attributes of the benchmark circuit are likewise altered by the active Trojans. This investigation indicates that although sequential Trojans alter the characteristics load, pins, and switching power, combinational Trojan insertion affects the parameters connection, principal output, and toggling rate. Furthermore, the insertion of all Trojan kinds influences aspects like level, score, and principal input; the always ON Trojan has an impact on resistance, static probability, and total load. As a result, an analysis of feature value based on Trojan kinds is tested at this feature generation step. According to the analysis's findings, in order to improve the dependability of the netlist, differences in the characteristics value are monitored throughout the extraction phase in order to distinguish malicious nets from unidentified test circuits. The stacked sparse autoencoder approach is used by the framework to transform high-dimensional features into low-dimensional datasets. For ISCAS'85 and ISCAS'89 circuits, the average accuracy of the Naïve Bayes method without pre-processing is 86.30% and 90.01%, respectively. Furthermore, the Naïve Bayes pre-processing with stacked sparse autoencoder obtains an average accuracy metric of 92.74% for ISCAS’89 circuits and 92.65% for ISCAS’85 circuits. In a similar vein, the average accuracy of the DBSCAN method without preprocessing is stated to be 90.51% for ISCAS’89 circuits and 92.19% for ISCAS’85 circuits. With pre-processing, the accuracy parameter's average value is seen to increase to 95.26% for ISCAS'85 circuits and 93.10% for ISCAS'89 circuits. It follows that, in comparison to the standard feature set, the optimum features efficiently train the learning model, yielding a greater accuracy rate and better prediction.

References

1. Agrawal, D., Baktir, S., Karakoyunlu, D., Rohatgi, P., & Sunar, B. (2007, May).Trojan detection using IC finger printing. In Security and Privacy, 2007.SP'07.IEEE Symposium on (pp. 296- 310).IEEE.

2. Amelian, A. & Borujeni, S.E., (2018). Aside-channel analysis for hardware Trojan detection based on path delay measurement. Journal of Circuits, Systems and Computers, 27(09), p.1850138.

3. Alkabani, Y., & Koushan far, F. (2008, June).Designer’s hardware Trojan horse. In Hardware- Oriented Security and Trust, 2008. HOST 2008.IEEE International Workshop on (pp. 82- 83). IEEE.

4. Bhasin, S., & Regazzoni, F. (2015, May). A survey on hardware trojan detection techniques.

In Circuits and Systems (ISCAS), 2015 IEEE International Symposium on (pp. 2021-2024). IEEE.

5. Bai, Y., Park, J., Tehranipoor, M., & Forte, D., (2022).Real-time instruction-level verification of remote IoT/CPS devices via side channels. Discover Internet of Things, 2(1), p.1.

6. Banga, M., &Hsiao, M.S.(2008, June). A region based approach for the identification of hardware Trojans. In Hardware-Oriented Security and Trust, 2008.HOST 2008. IEEE International Workshop on (pp. 40-47). IEEE.

7. Bao, C., Forte, D. & Srivastava, A., (2014, March). On application of one-class SVM to reverse engineering-based hardware Trojan detection. In Fifteenth International Symposium on Quality Electronic Design (pp. 47-54). IEEE.

8. Bao, C., Forte, D., & Srivastava, A. (2016).On reverse engineering-based hardware Trojan detection. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 35(1), 49-57.

9. Bhunia, S., Hsiao, M. S., Banga, M., & Narasimhan, S. (2014). Hardware Trojan attacks: threat analysis and countermeasures. Proceedings of the IEEE, 102(8), 1229-1247.

10. Bouchou, M., Wang, H. & Lakhdari, M.E.H., (2017, October). Automatic digital modulation recognition based on stacked sparse auto encoder. In2017 IEEE 17th International Conference on Communication Technology (ICCT) (pp. 28-32).IEEE.

11. Bozdal, M., Samie, M., Aslam, S. and Jennions, I., (2020).Evaluation of can bus security challenges Sensors, 20(8), p.2364.

12. Chen, H., Zhang, X., Huang, K., & Koushanfar, F., (2023). AdaTest: Reinforcement learning and adaptive sampling for on-chip hardware Trojan detection. ACM Transactions on Embedded Computing Systems, 22(2), pp.1-23.

13. Courbon, F., Skorobogatov, S., & Woods, C., (2016, November).Reverse engineering flash EEPROM memories using scanning electron micros copy. In International Conference on Smart Card Research and Advanced Applications (pp. 57-72).

14. Chakraborty, R. S., Narasimhan, S., & Bhunia, S. (2009, November). Hardware Trojan:Threats and emerging solutions. In High Level Design Validation and Test Workshop, 2009.HLDVT 2009. IEEE International (pp. 166-171). IEEE.

15. Chakraborty, R.S.&Bhunia, S., (2009). HARPOON: An obfuscation-based SoC design methodology for hardware protection. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 28(10), pp.1493-1502.

16. Chakraborty, R. S., Wolff, F., Paul, S., Papachristou, C., & Bhunia, S. (2009). MERO: A statistical approach for hardware Trojan detection. In Cryptographic Hardware and Embedded Systems- CHES 2009 (pp. 396- 410). Springer, Berlin, Heidelberg.