IEICE TRANSACTIONS

on Fundamentals of Electronics, Communications and Computer Sciences

DOI:10.1587/transfun.2024EAP1041

Publicized:2024/08/14

A PUBLICATION OF THE ENGINEERING SCIENCES SOCIETY

The Institute of Electronics, Information and Communication Engineers
Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3chome, Minato-ku, TOKYO, 105-0011 JAPAN

PAPER

Hardware Trojan Detection Method Based on Enhanced Local Outlier Factor

Tingyuan NIE[†], *Member*, Jingjing NIE[††], *and* Kun ZHAO[†††], *Nonmembers*

SUMMARY The globalization of the Integrated Circuit (IC) supply chain has introduced the risk of Hardware Trojan (HT) insertion. We propose an unsupervised Hardware Trojan detection method based on the Enhanced Local Outlier Factor (ELOF) algorithm to detect HT efficiently. This method extracts structural and testability features and employs the scoring mechanism of the ELOF algorithm to emphasize the deviation of suspicious HT nets from clusters. Experimental results on Hardware Trojan libraries show that the method achieves an average prediction accuracy (A) of 97.36%, a True Negative Rate (TNR) of 97.81%, a precision (P) of 40.94%, and an F-measure of 49.28%, all of which outperform the Local Outlier Factor (LOF) algorithm and Cluster-Based Local Outlier Factor (CBLOF) algorithm. Notably, the method exhibits superior performance in terms of True Positive Rate (TPR), reaching 70.86%, indicating its efficiency in identifying HT and reducing false negatives. The results demonstrate that the proposed algorithm and feature combination in the approach can significantly enhance the efficiency of Trojan detection.

key words: *hardware Trojan detection, gate-level netlist, feature extraction, unsupervised learning, enhanced local outlier factor*

1. Introduction

Integrated circuits (ICs) serve as the core components of hardware devices and play a vital role in determining the security of Internet of Things (IoT) systems. The globalization of the IC provides opportunities for implanting Hardware Trojans (HTs), leading to issues of information leakage, functional alterations, and decreased hardware reliability [1].

HTs are malicious modules or unauthorized circuit components inserted at various stages of an IC's lifecycle. Existing HT detection methods can be categorized into two major classes: pre-silicon detection and post-silicon detection [2]. Due to factors such as the difficulty of acquiring golden chips, susceptibility to noise, high costs, and the need for specialized equipment, post-silicon detection methods have shown limited efficiency in practical applications, and we do not introduce post-silicon detection methods here.

The pre-silicon detection method can be divided into dynamic detection and static detection [3]. Dynamic detection methods detect Trojans by applying external stimuli and observing the circuit's response. However, Trojan circuits are not easily triggered, and the activation is covert. Static detection methods do not require simulation testing and solely rely on the differences between Trojan and normal circuits. The method is based on the combinational and sequential testability features of Hardware Trojan, forming a 6-dimensional feature vector [4].

Machine learning-based detection methods have substantially improved the efficiency of HT detection. It is generally categorized into supervised learning and unsupervised learning. Supervised learning learns the relationship between input and their corresponding output labels, enabling it to make accurate predictions. Hasegawa et al. identified five structural features of HT by analyzing the differences between Trojan and normal nets. Using the Support Vector Machine (SVM) for gate-level netlist classification, they tuned parameters, significantly improving TPR [3]. Furthermore, research incorporating Neural Network (NN) algorithms made progress based on these five Trojan structural features [5]. Subsequently, a more detailed analysis involving 51 Trojan structural features is proposed. They employed a Random Forest (RF) classifier, adjusting parameters to select 11 crucial features that played a significant role in enhancing F-measure [6]. To further enhance detection performance, 11 features were selected from the 51 structural features. Multiple Layer Neural Network (MNN) algorithms were then applied for Trojan detection in gate-level netlists [7]. Dong et al. introduced a HT detection method based on eXtreme Gradient Boosting (XGBoost). By introducing five new structural features and using an XGBoost classifier, they selected 49 key features out of 56, efficiently achieving machine learning-based HT detection [8]. In [9], the introduction of boundary nets and Trojan nets for machine learning training provided high efficiency in identifying Trojans. Stacked autoencoder and stacked sparse autoencoder models were implemented in [10] for Trojan detection, yielding excellent results. Unsupervised learning emphasizes discovering patterns, structures, or regularities from unlabeled data. Reference [11] utilizes the K-means clustering algorithm for HT detection. However, it is only applicable for detecting combinational trigger-type Trojans and is inefficient for sequential trigger-type Trojans. Unsupervised anomaly detection identifies anomalous data exhibiting different or unexpected behavior within a dataset, also known as outliers or anomalies. Reference [12] reviews local anomaly detection algorithms focusing on the Local Outlier Factor (LOF) algorithm. The LOF is a density-based outlier detection algorithm assessing whether a data point is an outlier by considering the neighbor points. The LOF algorithm calculates the Local Reachability Density (LRD)

[†] The author is with the the School of Information and Control Engineering, Qingdao University of Technology at Qingdao 266520, China (E-mail: tynie@qut.edu.cn).

[††] The author's E-mail is niejingjingqjb@163.com.

[†††] The author's E-mail is sterling1982@163.com.

of the data point and determines the LOF by comparing it with the average LRD of neighboring points. The magnitude of the LOF is used to assess the degree of anomaly for that point. The algorithm PL-HTD (PCA and LOF Hardware Trojan Detection) proposed in Reference [2] combines Principal Component Analysis (PCA) and the LOF algorithm, demonstrating the efficiency of using unsupervised learning to detect HTs. However, it only considers local information of anomalous points which results in lower detection accuracy. Reference [13] introduces the Cluster-Based Local Outlier Factor (CBLOF) algorithm incorporating the scoring mechanism of XGBoost. CBLOF calculates the value for each net to distinguish between normal and Trojan nets.

The current unsupervised HT detection methods exhibit insufficient feature representation when dealing with diverse types of HTs, potentially leading to low detection efficiency. We propose the Enhanced Local Outlier Factor (ELOF) HT detection method and the main contributions are as follows.
Enhancement of HT discrimination: The ELOF considers local and global information on suspicious nets comprehensively, emphasizing the degree of deviation of outliers. Thereby it makes the discriminations between the normal and Trojan nets more pronounced and improves the efficiency.
Highlighted performance to traditional methods: Compared with existing techniques, the method exhibits outstanding performance in unsupervised HT identification. It achieves a better balance between precision and TPR, reduces false negatives, and improves accuracy.
Integration of HT features: The ELOF integrates the structural and testability features of HTs to improve the recognition of different types of HTs and reduce false negatives.

2. Hardware Trojan Detection Method Based on ELOF

The unsupervised HT detection method proposed in this paper follows approximately five main steps:
Step 1 Input Gate-Level Netlist: The input of our model is a gate-level netlist file containing the implanted Trojan circuit. We select benchmark circuits from Hardware Trojan libraries as the subjects of our research.
Step 2 Feature Extraction: By constructing a directed graph model of the gate-level netlist, we extract 56 structural features of the Trojan nets. We perform testability analysis to extract six testability features of the nets. The netlist file is transformed into the data required for learning algorithms.
Step 3 Feature Dimensionality Reduction: Using XGBoost on the selected dataset, we calculate importance scores for the 56 structural features and choose the top 16 features, which, when combined with the six testability features, form a 22-dimensional feature vector.
Step 4 Normalization of Feature Data: To mitigate the negative impact of varying feature value ranges on model training, the data is normalized during data preprocessing to transform the dataset into a standard normal distribution.
Step 5 Detection of HT in Gate-Level Netlist: Calculate the ELOF scores for each net by applying the proposed algorithm. The magnitude of the score is used to distinguish between sets of normal and Trojan nets, with higher scores indicating a higher likelihood of a net being a Trojan.

Previous research has demonstrated the efficiency of feature-based HT detection techniques [14]. The detection techniques rely on signal testability features or netlist structural features [15]. The automatic feature extraction-based deep learning methods have shown the capability for HT detection. Unfortunately, such methods primarily focus on structural features and neglect testability features. In this paper, we collect both structural features and testability features for Trojans. Each type presents distinct characteristics of different Trojans that help improve the model.

Hasegawa et al. [6] proposed 51 structural features for HT. Subsequently, Dong et al. [8] introduced a new feature, ”in_gate_x”, represents the number of x-level logic gates away from net n, resulting in a total of 56 structural features. We extract the structural features by abstracting gate-level netlists into directed graphs. Circuit testability is typically analyzed using the SCOAP (Sandia Controllability / Observability Analysis Program) algorithm [16]. It conducts testability analysis by calculating six metrics for each net, including combinational 0 controllability (CC0), combinational 1 controllability (CC1), combinational observability (CO), sequential 0 controllability (SC0), sequential 1 controllability (SC1), and sequential observability (S0). It considers both combinational trigger-type Trojans and sequential trigger-type Trojans. Compared to normal nets, Trojan nets exhibit lower testability, resulting in higher SCOAP values. In this study, we use the open-source tool Testability Measurement Tool to extract the SCOAP values of nets.

Based on the benchmark circuit selected in this paper, utilizing XGBoost's inherent scoring system, 56 features were evaluated for their importance. Among them, 16 features obtained scores exceeding 10,000 points, i.e., $f_0 - f_4$, $f_{28} - f_{29}$, f_{34}, f_{45}, f_{46}, f_{48}, f_{50}, $f_{52} - f_{55}$. The features represent the number of logic-gate fan-ins x-level away from the net n, the number of up to x-level loops from the input or output side of the net n, the number of constants up to x-level away from the input side of the net n, the minimum level to the primary input or output from the net n, the minimum level to any multiplexer from the input or output side of the net n, and the number of logic-gate x-level away from the net n. We select these 16 structural features and 6 testability features to form a 22-dimensional feature vector.

As shown in Algorithm 1, the proposed ELOF algorithm comprehensively considers local and global information of the data points under examination. The ELOF algorithm calculates a clustering coefficient, indicating the degree to which the point deviates from the cluster. The ELOF value is obtained by multiplying the LOF value by the clustering coefficient to assess the point's anomaly level. The following are the key definitions used in the ELOF algorithm.
Definition 1: k-distance

The k-distance of point p, denoted as $k\_dist(p)$, is the distance between point p and its k-th nearest neighbor. Here, k is a natural number. In regions of higher density, the value of $k\_dist(p)$ is smaller, and vice versa.

Definition 2: k-distance neighborhood

The k-distance neighborhood of point p, denoted as $N_k(p)$, is a set of points within and on the circle centered at point p with a radius of $k\_dist(p)$, as shown in formula (1). $N_k(p)$ represents the k neighbors of point p.

$$N_k(p) = \{o \in D \backslash \{p\} \,|\, d(p,o) \leq k\_dist(p)\} \tag{1}$$

Definition 3: Reachability distance

The reachability distance of point p concerning point o, denoted as $reach\_dist_k(p,o)$, is defined as the maximum between the direct distance between points o and p and the k nearest neighbor distance of point o, as shown in formula (2).

$$reach\_dist_k(p,o) = \max\{k\_dist(o), d(p,o)\} \tag{2}$$

Definition 4: Local reachability density

The local reachability density of point p is the reciprocal of the average reachability distance based on the $MinPts$ nearest neighbors of p. $MinPts$ represents the number of nearest neighbors in the data point's local neighborhood.

$$lrd_{MinPts}(p) = 1/\frac{\sum_{o \in N_{MinPts}(p)} reach\_dist_{MinPts}(p,o)}{|N_{MinPts}(p)|} \tag{3}$$

Definition 5: Local outlier factor

The local outlier factor (LOF) of point p is defined as the ratio of the average local reachability density of the $MinPts$ nearest neighbors of point p to its local reachability density, as shown in formula (4). The LOF value close to 1 indicates that the data point is likely to belong to the same cluster as its neighbors.

$$LOF_{MinPts}(p) = \frac{\sum_{o \in N_{MinPts}(p)} \frac{lrd_{MinPts}(o)}{lrd_{MinPts}(p)}}{|N_{MinPts}(p)|} \tag{4}$$

Definition 6: Large clusters and small clusters

To distinguish between large and small clusters, parameter $\alpha(0 \leq \alpha \leq 1)$ is used. When the proportion of large clusters reaches over 90%, these clusters are called large clusters (LC), while others are called small clusters (SC). If the current number of clusters is β times the number of clusters in the next cluster (usually $\beta = 5$). It is considered that the first few clusters are large clusters, while the other clusters are small clusters. If one of the two conditions is met, large and small clusters can be identified. Let $C = \{C_1, C_2, ...C_k\}$ be the set of clusters generated by the K-means clustering algorithm sorted in descending order based on the number of elements, it meets $|C_1| \geq |C_2| \geq ... \geq |C_k|$. Define two parameters α and β, and let b be the boundary between small clusters and large clusters. The value of b should satisfy either of the following two conditions in formula (5).

$$\begin{cases} (|C_1| + |C_2| + ... + |C_b|) \geq |D| * \alpha \\ \frac{|C_b|}{|C_{b+1}|} \geq \beta \end{cases} \tag{5}$$

Large Cluster $LC = \{C_i | i \leq b\}$, and Small Cluster $SC = \{C_j | j > b\}$.

Algorithm 1 Enhanced Local Outlier Factor Algorithm.

```
Input: Clusters from Clustering Algorithm
    $\alpha, \beta$: coefficients for cluster size ratio
Output: ELOF scores
for each cluster in the set of clusters do
    Classify cluster as large or small based on $\alpha$ and $\beta$
    for each net in the cluster do
        Calculate cluster coefficient ($C$) of the net
        Calculate local outlier factor (LOF) of the net
        Calculate ELOF score for the net using $C$ and LOF
    end for
end for
return ELOF anomaly scores
```

Definition 7: ELOF anomaly value

The ELOF anomaly value of point p is defined as the product of its LOF value and the distance from the point to the center of the nearest large cluster center, as shown in formula (6).

$$ELOF(p) = LOF(p) * C \tag{6}$$

Where $C = min(dist(p, C_i))$ represents the clustering coefficient, indicating the degree to which a point deviates from the cluster. "C_i" represents large cluster.

LOF takes into account the density difference between a point and its neighboring points, considering local information, and plays a fundamental role in detecting outliers. If a point's LOF value is greater than 1, it indicates that the point is sparser compared to its neighbors, making it a potential outlier. ELOF enhances the anomaly detection capability of LOF by introducing the clustering coefficient C) to emphasize the degree to which an outlier deviates. C represents the distance from a data point to the nearest center of a large cluster, reflecting how far the point is from the cluster center and incorporating global information. Specifically, if a point already has a high LOF value (indicating it is considered an outlier) and C is also large (indicating that it is far from the cluster center), the ELOF value will be even larger, further highlighting the point's anomaly. The ELOF considers local and global information on suspicious nets, and emphasizes the degree of outlier deviation, enhancing the accuracy of identifying potential Trojans while reducing false positives.

3. Experiment

We take the Trust_Hub Hardware Trojan Library as the research benchmark suit. Additionally, we have chosen two circuits, s1423-T401 and s1423-T402, from the TRIT-TS benchmark [17]. In practical operations, only the internal nets in the gate-level netlist are considered, while the boundary nets are ignored. Fig. 1 lists the quantities of Trojan nets and normal nets in various benchmark circuits.

The classification of HTs is assessed using four values: true positives, false positives, false negatives, and true negatives. To evaluate the performance of the model, we employ five metrics: TPR (True Positive Rate), TNR (True Negative Rate), Precision, F-measure, and Accuracy. For the details, please refer to [2]. The examples in Fig. 1 show that the

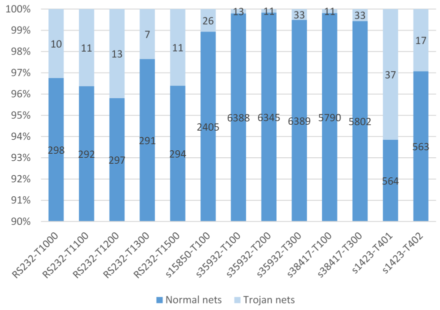

Fig. 1 Statistical results of the normal nets and Trojan nets.

number of normal nets in the netlist is much larger than that of Trojan nets. Even if some of the Trojan nets are correctly detected, the accuracy would still be high.

The hardware platform for this experiment is an Intel Core i5-6500 3.2GHz processor with 16GB of RAM, running on the 64-bit Windows 10 Professional operating system. To achieve optimal performance, we conducted a series of experiments using the ELOF algorithm based on the selected 22-dimensional feature vectors. The parameter K represents the number of clusters needed for K-means clustering, we set the value of K to 8. We adjust the contamination parameter C that represents the anomalies proportion to achieve a better performance for the classifications. Empirically, the value of parameter C should be limited to the range [0, 0.5] [13]. A higher value of C means more normal nets tend to be misidentified as Trojan nets. We first investigated the related studies and determined an initial value for parameter C. We tuned it gradually until the classification performance reached a relative optimal. In the study, it was set to 0.05 for the RS232 series benchmarks, 0.01 for the s15850 and s35932 series, 0.005 for the s38417 series, and 0.1 for the s1423 series. And α and β are coefficients determining small and large clusters, indicating the ratio of points in large clusters to those in small clusters. In previous work, it has been demonstrated that taking $\alpha = 0.9$ and $\beta = 5$ can achieve the best detection results through multiple experiments [13]. In practice, larger or smaller values of α and β will lead to a degradation of Trojan detection performance. Moreover, under the premise of unsupervised learning, the settings can be applied to unseen samples during training.

We conduct a performance analysis of the unsupervised learning model. Based on a 22-dimensional feature vector, we apply the ELOF algorithm for HT detection. The detection results are shown in Table 1, which details the values of parameters K and C, various metrics tested on specific circuits. The average TPR is 70.86%, indicating the algorithm's commendable capability in identifying Trojans. The average TNR is 97.81%, suggesting that the algorithm almost consistently classifies normal nets correctly. The average precision (P) is 40.94%, revealing room for improvement in the overall performance of our algorithm. Furthermore, the average F-measure is 49.28%, considering both precision and TPR, indicating good balance although there is still potential for further optimization. The average prediction accuracy (A) is 97.36%, demonstrating the accuracy and reliability of our algorithm in Trojan detection. To reduce the misclassifications and improve the classification performance, we have adopted strategies like parameter adjustments and showed positive results. In the future, we will develop algorithms to solve the imbalance of learning data and further reduce the misclassifications of Trojans. Another feasible method is to use stacking learning techniques to improve the whole performance of Trojan detection.

To validate the efficiency of our detection method, we conduct a series of comparisons. Firstly, we independently implemented the LOF, CBLOF, and ELOF algorithms using the 22-dimensional feature vector in the same environment. The comparative results are shown in Table 2. We compared the average values of the metrics for the three methods to determine their superiority or inferiority. Table 2 shows the ELOF algorithm proposed in this paper performs well in various metrics. The average TPR is 70.86%, which represents an improvement of 29.38% and 10.80% compared to the LOF and CBLOF algorithms, respectively, indicating that the algorithm can more efficiently identify Trojans. Secondly, the average TNR is 97.81%, with improvements of 0.95% and 0.12% compared to LOF and CBLOF. The average precision (P) is 40.94%, with improvements of 18.53% and 3.86%. The average F-measure is 49.28%, with improvements of 22.15% and 5.33%, indicating that the algorithm achieves a better balance between precision and TPR. The average prediction accuracy (A) is 97.36%, with improvements of 1.83% and 0.22%, suggesting that the algorithm has a certain advantage in accurately identifying Trojans. The comparative results indicate that the proposed method has advantages and performs better in HT detection compared with the LOF and CBLOF.

Furthermore, we compared the performance of our proposed detection algorithm with state-of-the-art techniques including Boundary Net Structure (BNS) [9], Stacked Autoencoder (SA) [10] and Stacked Sparse Autoencoder (SSAE) [10], as well as Support Vector Machine (SVM) [5] and Neural Network (NN) [5]. The specific comparison results are shown in Table 3. The bold in the table indicates that our is stronger than the other five algorithms. In most cases, our algorithm demonstrates the best performance among the benchmark circuits.

We extracted 56 structural features and six testability features to expand Trojan coverage. Sixteen high-importance features were selected from the 56 structural features using XGBoost's intrinsic scoring mechanism, forming a 22-dimensional feature vector. We conducted experiments separately based on 56 structural features, 6 testability features, and the combined 22-dimensional feature vector to verify the efficiency of the feature combination. The results are shown in Table 4. Firstly, when experimenting with only the 56 structural features, the average TPR is 34.67%, the average TNR is 97.09%, the average precision P is 22.73%, the average F-measure is 26.43%, and the average accuracy A is 95.96%. This indicates that the performance of detection

Table 1 Classification results based on ELOF algorithm.

ELOF	K	C	TN	FP	FN	TP	TPR	TNR	P	F	A
RS232-T1000	250	0.05	293	5	0	10	100.00%	98.32%	66.67%	80.00%	98.38%
RS232-T1100	250	0.05	285	7	3	8	72.73%	97.60%	53.33%	61.54%	96.70%
RS232-T1200	250	0.05	293	4	2	11	84.62%	98.65%	73.33%	78.57%	98.06%
RS232-T1300	250	0.05	297	8	0	7	100.00%	97.38%	46.67%	63.64%	97.44%
RS232-T1500	250	0.05	290	4	0	11	100.00%	98.64%	73.33%	84.62%	98.69%
s15850-T100	250	0.01	2396	9	11	15	57.69%	99.63%	62.50%	60.00%	99.18%
s35932-T100	250	0.01	6333	55	4	9	69.23%	99.14%	14.06%	23.38%	99.08%
s35932-T200	200	0.01	6262	53	1	10	90.91%	99.16%	15.87%	27.03%	99.15%
s35932-T300	200	0.01	6353	36	5	28	84.85%	99.44%	43.75%	57.73%	99.36%
s38417-T100	200	0.005	5766	24	7	4	36.36%	99.59%	14.29%	20.51%	99.47%
s38417-T300	200	0.005	5779	23	28	5	15.15%	99.60%	17.86%	16.39%	99.13%
s1423-T401	200	0.1	525	39	16	21	56.76%	93.09%	35.00%	43.30%	90.85%
s1423-T402	200	0.1	514	49	8	9	52.94%	91.30%	15.52%	24.00%	90.17%
Average							70.86%	97.81%	40.94%	49.28%	97.36%

Table 2 Classification comparison for ELOF, LOF, and CBLOF.

Test data	TPR(%)			TNR(%)			P(%)			F(%)			A(%)		
	LOF	CBLOF	**ELOF**	LOF	CBLOF	**ELOF**	LOF	CBLOF	**ELOF**	LOF	CBLOF	**ELOF**	LOF	CBLOF	**ELOF**
RS232-T1000	30.00	100.0	100.0	95.97	98.32	98.32	20.00	66.67	66.67	24.00	80.00	80.00	93.83	98.38	98.38
RS232-T1100	45.45	72.73	72.73	96.58	97.60	97.60	33.33	53.33	53.33	38.46	61.54	61.54	94.72	96.70	96.70
RS232-T1200	30.77	76.92	84.62	96.30	98.32	98.65	26.67	66.67	73.33	28.57	71.43	78.57	93.55	97.42	98.06
RS232-T1300	42.86	85.71	100.0	96.07	97.05	97.38	20.00	40.00	46.67	27.27	54.55	63.64	94.87	96.79	97.44
RS232-T1500	36.36	100.0	100.0	96.26	98.64	98.64	26.67	73.33	73.33	30.77	84.62	84.62	94.10	98.69	98.69
s15850-T100	46.15	57.69	57.69	99.50	99.63	99.63	50.00	62.50	62.50	48.00	60.00	60.00	98.93	99.18	99.18
s35932-T100	69.23	76.92	69.23	99.14	99.15	99.14	14.06	15.63	14.06	23.38	25.97	23.38	99.08	99.11	99.08
s35932-T200	72.73	9.09	90.91	99.13	99.02	99.16	12.70	1.59	15.87	21.62	2.70	27.03	99.08	98.86	99.15
s35932-T300	72.73	54.55	84.85	99.37	99.28	99.44	37.50	28.13	43.75	49.48	37.11	57.73	99.24	99.05	99.36
s38417-T100	18.18	27.27	36.36	99.55	99.57	99.59	7.14	10.71	14.29	10.26	15.38	20.51	99.40	99.43	99.47
s38417-T300	15.15	15.15	15.15	99.60	99.60	99.60	17.86	17.86	17.86	16.39	16.39	16.39	99.13	99.13	99.13
s1423-T401	24.32	45.95	56.76	90.96	92.38	93.09	15.00	28.33	35.00	18.56	35.05	43.30	86.86	89.52	90.85
s1423-T402	35.29	58.82	52.94	90.76	91.47	91.30	10.34	17.24	15.52	16.00	26.67	24.00	89.14	90.52	90.17
Average	41.48	60.06	**70.86**	96.86	97.69	**97.81**	22.41	37.08	**40.94**	27.14	43.95	**49.28**	95.53	97.14	**97.36**

Table 3 Comparison of different metrics with the state-of-the-art methods.

Test data	TPR(%)						TNR(%)						A(%)					
	BNS[9]	SA[10]	SSAE[10]	SVM[5]	NN[5]	ELOF	BNS[9]	SA[10]	SSAE[10]	SVM[5]	NN[5]	ELOF	BNS[9]	SA[10]	SSAE[10]	SVM[5]	NN[5]	ELOF
RS232-T1000	100.00	100.00	100.00	53.33	42.22	**100.00**	98.20	96.14	97.22	30.83	66.92	**98.32**	98.40	96.51	97.46	34.08	63.34	98.38
RS232-T1100	69.00	92.70	94.00	58.33	100.00	72.73	96.80	91.50	94.30	27.00	62.33	**97.60**	93.80	92.80	96.13	28.21	63.78	**96.70**
RS232-T1200	100.00	95.20	95.40	80.00	70.00	84.62	95.80	98.33	99.33	25.57	52.13	98.65	96.30	98.13	99.07	27.30	52.70	98.06
RS232-T1300	100.00	100.00	100.00	88.89	22.22	**100.00**	99.70	98.62	98.98	25.84	73.15	97.38	99.70	98.70	99.05	27.69	71.66	97.44
RS232-T1500	97.40	91.30	100.00	83.33	66.67	**100.00**	97.50	97.00	99.34	23.51	65.89	98.64	97.50	96.50	99.36	25.80	65.92	98.69
s15850-T100	–	–	–	92.59	88.89	57.69	–	–	–	65.75	75.55	**99.63**	–	–	–	66.04	75.69	**99.18**
s35932-T100	–	–	–	93.33	100.00	69.23	–	–	–	59.77	84.56	**99.14**	–	–	–	59.85	84.59	**99.08**
s35932-T200	–	–	–	100.00	87.50	90.91	–	–	–	59.18	86.88	**99.16**	–	–	–	59.29	86.88	**99.15**
s35932-T300	–	–	–	27.03	100.00	84.85	–	–	–	57.99	58.59	**99.44**	–	–	–	57.82	58.82	**99.36**
s38417-T100	–	–	–	100.00	100.00	36.36	–	–	–	75.65	72.22	**99.59**	–	–	–	75.70	72.28	**99.47**
s38417-T300	–	–	–	100.00	75.00	15.15	–	–	–	71.60	76.10	**99.60**	–	–	–	71.82	76.09	**99.13**

using only structural features is relatively low. Secondly, when experimenting with the 6 testability features, the average TPR is 37.40%, TNR is 97.26%, P is 24.31%, F is 28.31%, and A is 96.29%. Relying solely on testability features still does not yield satisfactory detection results. By using the 22-dimensional combined features proposed in this paper, the average TPR significantly improves to 70.86%, while TNR remains high at 97.81%. Precision (P) increases to 40.94%, F-measure improves to 49.28%, and accuracy (A) is 97.36%. The results demonstrate that the method significantly enhances the Trojan detection performance by combining structural and testability features.

4. Conclusion

We proposed an unsupervised Hardware Trojan detection method based on the ELOF. It extracts the structural and testability features of HTs to employ the scoring mechanism to emphasize the deviation of suspicious Trojan nets from clusters, thereby improving the ability of Trojan coverage and distinguishing between normal and Trojan nets. Experimental results on the benchmark circuits demonstrated that the method achieved an average prediction accuracy (A) of 97.36%, TNR of 97.81%, precision (P) of 40.94%, and F-measure of 49.28%. The proposal performed exceptionally well in TPR, reaching 70.86%. The improvements indicated its capability to identify Trojans more accurately and reduce the false negative rate. It demonstrates the efficiency of combining structural and testability features for Trojan detection.

Acknowledgments

This work was partly supported by NSFC (Grants No. 61572269) and Shandong Provincial Natural Science Foundation (Grant No. ZR2021MF101).

References

[1] A. J. Tiempo, and Y. J. Jeong. “Split and Eliminate: A Region-Based Segmentation for Hardware Trojan Detection,” *IEICE TRANSACTIONS on Information and Systems*, vol. 106, no. 3, pp. 349–356,

Table 4 Comparison of classification results for three different feature combinations.

Test data	TPR(%)			TNR(%)			P(%)			F(%)			A(%)		
	56	6	**22**	56	6	**22**	56	6	**22**	56	6	**22**	56	6	**22**
RS232-T1000	40.00	60.00	100.0	96.31	96.98	98.32	26.67	40.00	66.67	32.00	48.00	80.00	94.48	95.78	98.38
RS232-T1100	54.55	45.45	72.73	96.92	96.58	97.60	40.00	33.33	53.33	46.15	38.46	61.54	95.38	94.72	96.70
RS232-T1200	53.85	53.85	84.62	97.31	97.31	98.65	46.67	46.67	73.33	50.00	50.00	78.57	95.48	95.48	98.06
RS232-T1300	57.14	28.57	100.0	96.39	95.74	97.38	26.67	13.33	46.67	36.36	18.18	63.64	95.51	94.23	97.44
RS232-T1500	45.45	54.55	100.0	96.60	96.94	98.64	33.33	40.00	73.33	38.46	46.15	84.62	94.75	95.41	98.69
s15850-T100	53.85	50.00	57.69	99.58	99.54	99.63	58.33	54.17	62.50	56.00	52.00	60.00	99.10	99.01	99.18
s35932-T100	15.38	15.38	69.23	99.03	99.03	99.14	3.13	3.13	14.06	5.19	5.19	23.38	98.86	98.86	99.08
s35932-T200	9.09	9.09	90.91	99.02	99.02	99.16	1.59	1.59	15.87	2.70	2.70	27.03	98.86	98.86	99.15
s35932-T300	18.18	3.03	84.85	99.09	99.01	99.44	9.38	1.56	43.75	12.37	2.06	57.73	98.68	98.52	99.36
s38417-T100	18.18	27.27	36.36	99.55	99.57	99.59	7.14	10.71	14.29	10.26	15.38	20.51	99.40	99.43	99.47
s38417-T300	6.06	12.12	15.15	99.55	99.59	99.60	7.14	14.29	17.86	6.56	13.11	16.39	99.02	99.09	99.13
s1423-T401	37.84	62.16	56.76	91.84	93.44	93.09	23.33	38.33	35.00	28.87	47.42	43.30	88.52	91.51	90.85
s1423-T402	41.18	64.71	52.94	90.94	91.65	91.30	12.07	18.97	15.52	18.67	29.33	24.00	89.48	90.86	90.17
Average	34.67	37.40	**70.86**	97.09	97.26	**97.81**	22.73	24.31	**40.94**	26.43	28.31	**49.28**	95.96	96.29	**97.36**

2023.

[2] C. Dong, Y. Liu, J. Chen, X. Liu, W. Guo, and Y. Chen. “An unsupervised detection approach for hardware trojans,” *IEEE Access*, vol. 8, pp. 158169–158183, 2020.

[3] K. Hasegawa, M. Oya, M. Yanagisawa and N. Togawa. “Hardware Trojans classification for gate-level netlists based on machine learning,” *IEEE 22nd International Symposium on On-Line Testing and Robust System Design (IOLTS)*, pp. 203–206, 2016.

[4] C. H. Kok, C. Y. Ooi, M. Moghbel, N. Ismail, H. S. Choo, and M. Inoue. “Classification of Trojan nets based on SCOAP values using supervised learning,” in *Proceedings of IEEE international symposium on circuits and systems (ISCAS)*, apporo, Japan, pp. 1–5, 2019.

[5] K. Hasegawa, M. Yanagisawa and N. Togawa. “A hardware-Trojan classification method using machine learning at gate-level netlists based on Trojan features,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 100, no. 7, pp. 1427–1438, 2017.

[6] K. Hasegawa, M. Yanagisawa, and N. Togawa. “Trojan-feature extraction at gate-level netlists and its application to hardware-Trojan detection using random forest classifier,” in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS)*, Baltimore, MD, USA, pp. 1–4, 2017.

[7] K. Hasegawa, M. Yanagisawa, and N. Togawa. “Hardware Trojans classification for gate-level netlists using multi-layer neural networks,” in *IEEE International Symposium on On-line Testing & Robust System Design*, 2017.

[8] C. Dong, J. Chen, W. Guo, and J. Zou. “A machine-learning-based hardware-Trojan detection approach for chips in the Internet of Things,” *International Journal of Distributed Sensor Networks*, vol. 15, no. 12, pp. 1550147719888098, 2019.

[9] K. Hasegawa, M. Yanagisawa, and N. Togawa. “A hardware-Trojan classification method utilizing boundary net structures,” *IEEE international conference on consumer electronics (ICCE)*, pp. 1–4, 2018.

[10] M. Priyatharishini, M. N. Devi. “A deep learning based malicious module identification using stacked sparse autoencoder network for VLSI circuit reliability,” *Measurement*, vol. 194, pp. 111055, 2022.

[11] H. Salmani. “COTD: Reference-free hardware trojan detection and recovery based on controllability and observability in gate-level netlist,” *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 2, pp. 338–350, 2016.

[12] O. Alghushairy, R. Alsini, T. Soule and X. Ma. “A review of local outlier factor algorithms for outlier detection in big data streams,” *Big Data and Cognitive Computing*, vol. 5, no. 1, pp. 1–2, 2020.

[13] S. Meenakshi. “Performance enhancement of unsupervised hardware Trojan detection algorithm using clustering-based local outlier factor technique for design security,” *IEEE International Test Conference India (ITC India)*, pp. 1–8, 2022.

[14] S. Yao, X. Chen, J. Zhang, Q. Liu and H. Yang. “FASTrust: Feature analysis for third-party IP trust verification,” *IEEE International Test Conference*, 2015.

[15] C. H. Kok, C. Y. Ooi, M. Inoue, M. Moghbel and F. Hussin. “Net Classification Based on Testability and Netlist Structural Features for Hardware Trojan Detection,” *IEEE 28th Asian Test Symposium (ATS)*, 2019.

[16] L. H. Goldstein, and E. L. Thigpen. “SCOAP: Sandia controllability/observability analysis program,” *Proceesings of the Design Automation Conference*, pp. 190–196, 1980.

[17] K. Huang, and Y. He. “Trigger Identification Using Difference-Amplified Controllability and Dynamic Transition Probability for Hardware Trojan Detection,” *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3387-3400, 2020.

Tingyuan Nie is a Professor at Qingdao University of Technology. He was born in Shandong Province, China, in 1971. He received a B.S. degree from Wuhan University of Technology in 1993, an M.S. degree in 2005, and a Ph.D. degree from Kochi University, Japan in computer engineering. His research interests include VLSI CAD algorithms, hardware security, machine learning, and complex network security. He is a member of IEEE since 2003.

Jingjing Nie is currently holding a master's degree from Qingdao University of Technology. She was born in Liaocheng City, Shandong Province, China in 2000. Received an engineering degree in 2022. Her research interests include VLSI design, hardware security, and machine learning.

Kun Zhao received a B.E. degree from Hefei University of Technology, Hefei, China, in 2004, and an M.S. and Ph.D. degree from Shanghai University, Shanghai, China, in 2008 and 2012, respectively. Since 2016, he has been an Associate Professor at the School of Information and Control Engineering, Qingdao University of Technology, Qingdao, China. His research interests include computer vision, machine learning, and complex networks.