Journal of Hardware and Systems Security
https://doi.org/10.1007/s41635-025-00160-2

Uncertainty-Aware Unimodal and Multimodal Learning for Evolving Hardware Trojan Detection

Rahul Vishwakarma[1] · **Amin Rezaei**[1]

Received: 26 August 2024 / Accepted: 18 February 2025

Abstract
As the semiconductor industry has shifted to a fabless paradigm, the risk of hardware Trojans being inserted at various stages of production has also increased. Recently, there has been a growing trend toward the use of machine learning solutions to effectively detect hardware Trojans with a focus on the accuracy of the model as an evaluation metric. However, in a high-risk and sensitive domain, we cannot accept even a small misclassification. Additionally, it is unrealistic to expect an ideal model, especially when Trojans evolve over time. In this paper, we design an uncertainty-aware machine learning solution which also handles evolving hardware Trojans using our proposed novel conformalized generative adversarial network. We further extend the proposed method for multimodal deep learning along with uncertainty quantification that also addresses the problem of missing modalities. The proposed solutions have been validated on both synthetic and real chip-level benchmarks and proven to pave the way for researchers looking to find informed machine learning solutions to hardware security problems.

Keywords Hardware Trojan · Machine learning · Multimodal deep learning · Uncertainty quantification · Calibrated explainability · Conformal prediction

1 Introduction

The insertion of Hardware Trojans (HT) involves a deliberate modification by an attacker to the design of a hardware component, with potential consequences ranging from device malfunction to the leakage of sensitive information and even physical damage [1]. With the semiconductor industry adopting a fabless model, the risk of HT insertion at various manufacturing stages has increased, posing a substantial security threat to hardware systems. Conventional HT detection methods, including signature-based approaches [2], which analyze Integrated Circuit (IC) functionality, layout, and timing, often prove ineffective against HT insertion attacks, particularly those designed to evolve over time. Consequently, there is a shift towards the adoption of Machine Learning (ML)-based solutions for a more efficient and effective approach to HT detection. However, despite adherence to ML evaluation best practices, there exists the potential for unintended consequences in the context of hardware security [3].

The majority of existing ML-based solutions lack sufficient information about the dataset, especially in cases where class distribution differs significantly, and evaluations need to account for concept drift or the evolution of new incoming datasets. Addressing these concerns, a comprehensive study in [4] seeks to ascertain the extent to which ML serves as a universal solution for the diverse challenges within the hardware security domain [5].

A known issue when employing any ML method is the lack of guarantee that a model claiming a very small misclassification rate will maintain the same performance on unseen data. Concept drift, caused by the attacker's intelligently modified version of HT insertion techniques, introduces uncertainty. A misclassification not only has significant financial and economic implications but can also be life-threatening in high-risk, sensitive domains, such as implantable devices. Consequently, there is a need for additional metrics to complement existing model evaluation techniques, ensuring reliable decisions and coverage guarantees for predictions. Lately, the widespread adoption of a multimodal deep learning approach in diverse domains, including healthcare, for classification tasks has been notable. Nevertheless, concerns

✉ Amin Rezaei
amin.rezaei@csulb.edu

Rahul Vishwakarma
rahuldeo.vishwakarma01@student.csulb.edu

[1] California State University, Long Beach, CA, USA

Published online: 26 February 2025

persist regarding the reliability of model inferences due to challenges like a scarcity of data points and highly imbalanced datasets. Additionally, addressing the issue of missing modalities in multimodal learning presents another hurdle that needs resolution to enhance the utility of the data for the model.

In this paper, we propose a method to detect evolving HTs using uncertainty-aware unimodal and multimodal learning which leverages the algorithm-agnostic statistical inference technique of conformal prediction [6] as shown in Fig. 1. This methodology ensures guaranteed coverage of predictions, particularly when faced with a covariate shift [7]. Our non-invasive approach serves as an overlay for existing ML models, deviating from a point prediction that identifies the presence or absence of a Trojan in a detected circuit. Instead, it provides a set prediction of detected labels, accurately incorporating the correct class 95% of the time on average (i.e., $\alpha = 0.05$). The primary objective of this paper is to facilitate the integration of ML into the problem-solving methodologies of hardware security and promote awareness regarding risk-controlled predictions with assured coverage.

1.1 Motivation

The current scenario in hardware security is characterized by a concerning increase in the complexity and stealthiness of HTs, posing an immediate challenge for detection methods. Upon conducting a thorough examination of the available literature, we identified a notable gap in research specifically addressing the detection of evolving HTs.

Furthermore, recent scrutiny of the Trust-Hub repository's efficacy in handling HTs raised significant doubts about its reliability as a suitable benchmark. Specifically, the effectiveness of the Trust-Hub dataset has been questioned [8], signaling a need for alternative datasets and dispelling the notion that it serves as an infallible silver bullet for designing HT detection algorithms suitable for production environments. In academia, relying on real-world datasets proves challenging, as corporations often refrain from publicizing such data due to confidentiality and Intellectual Property (IP) concerns. To address this limitation, our research endeavors to bridge the gap by generating synthetic datasets that faithfully represent the characteristics of real HTs. This approach is particularly pertinent in light of industry trends, as predicted by Gartner, which forecasts that 60% of data used for Artificial Intelligence (AI) and analytics projects by 2024 will be synthetically generated [9]. The emergent ecosystem of startups securing substantial funding for synthetic data generation [10] further attests to the growing importance of this paradigm.

The motivation behind our research stems from the imperative to address the critical issues of limited input data, the challenge of handling missing modalities, and the necessity to provide robust uncertainty estimation for each prediction in the context of evolving HTs. Through our proposed methodology, we aim to contribute a nuanced understanding

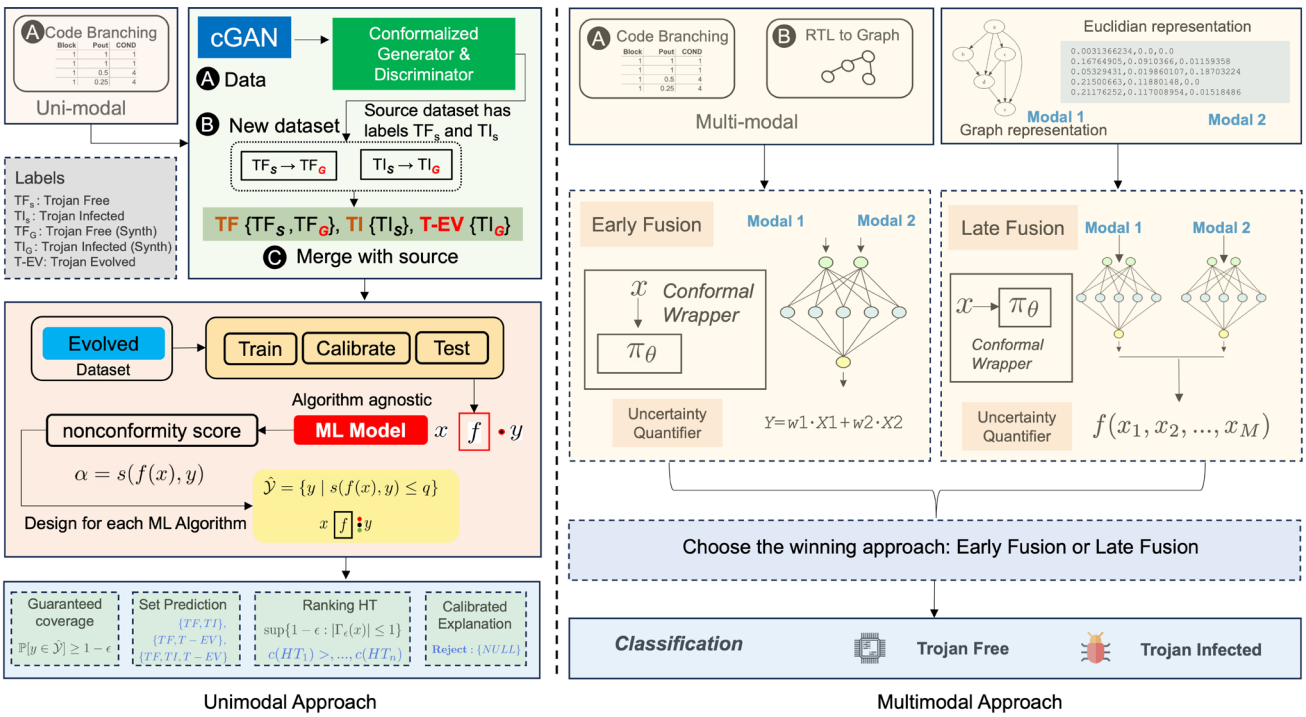

Fig. 1 Proposed solution overview for uncertainty-aware hardware Trojan detection

and effective solutions to propel the field of HT detection forward.

1.2 Contributions

In this paper, our primary focus revolves around unimodal and multimodal deep learning for the identification of HTs, and addressing the inherent challenges associated with this problem, for example, missing modalities and imbalanced dataset. The first challenge pertains to effectively handling missing modalities and implementing uncertainty-aware multimodal fusion approaches. To address this, we utilize graphical representations of circuits [11] and Euclidean data derived from processing the Abstract Syntax Tree (AST) of Register Transfer Level (RTL) files (i.e., Verilog) [12]. While multimodal approaches have been employed to enhance model accuracy in various domains, their application in Trojan identification has been notably absent. For uncertainty-aware multimodal learning, we advocate for implementing logic at the information fusion level of modalities, leveraging p-values aggregation with conformal prediction.

The second challenge we tackle is the quantification of uncertainty associated with HT prediction outcomes and ensuring the validity of predicted labels, especially when dealing with a limited number of highly imbalanced data points. Specifically, our interest lies in the ability of a ML classifier to predict the true label of a new data point with a 95% provable guaranteed coverage, a crucial requirement in risk-sensitive domains. Designing such a system holds potential benefits for decision-makers investigating detected labels as "Trojan-Infected." Additionally, we explore the possibility of ranking detected "Trojan-Infected" circuits to enable more informed treatment decisions.

Comparing our method with state-of-the-art approaches, we illustrate the overarching framework in Fig. 2. The diagram also highlights the current limitations of prevalent ML frameworks for HT detection, including binary predictions, a lack of trust in predictions due to calibration issues, and an absence of coverage guarantees. Our primary contributions are summarized as follows:

- Proposing a multimodal learning approach using graph and Euclidean data of the hardware circuits. This study is the first to investigate and implement a multimodal approach for HT detection, emphasizing uncertainty awareness.
- Suggesting a model fusion approach that systematically assesses each modality's contribution to the overall prediction. This enhances interpretability and facilitates more robust decision-making.
- Addressing the challenges of missing modalities and solving the issue of handling an imbalanced and small dataset by leveraging generative adversarial networks.
- Introducing the notion of HT evolution and providing an innovative method of creating evolving HTs with high precision using a conformalized generative adversarial network.
- Suggesting a tunable significance level through conformal prediction for HT detection by using the concept of guaranteed coverage of the prediction set.
- Defining an algorithm-agnostic and explainability-aware reject prediction made by the ML model. When the model is uncertain about identifying the evolving Trojan, it rejects the prediction, passing it to a human for manual investigation.
- Proposing a ranking mechanism for the evolved Trojans by assigning a confidence score from the prediction.

2 Related Works and Preliminaries

Detection of HTs using traditional ML techniques has primarily focused on modeling methods. Various algorithms aim to enhance overall accuracy by detecting HTs based on features extracted from RTL code, presented as tabular and graphical representations of the circuit. Surveys on ML for HT detection have been conducted in [13–16].

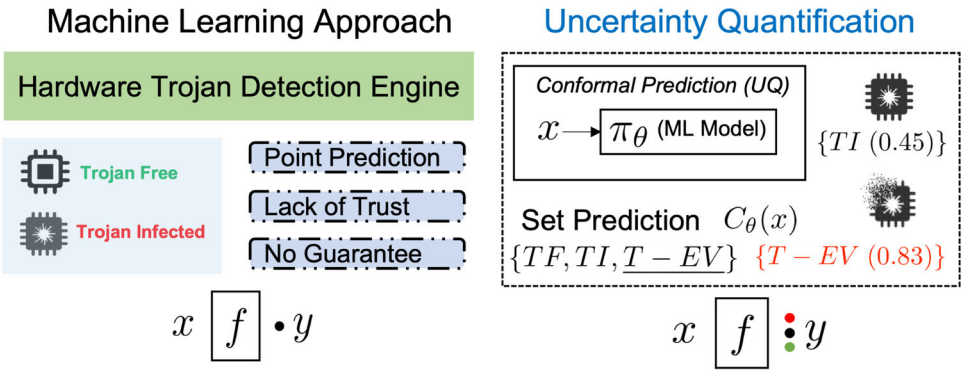

Fig. 2 The input feature is passed to a conventional machine learning hardware Trojan detector

Image classification techniques have been explored in [17] and [18], while multimodal image processing is utilized in [19]. Most papers focus on feature extraction from gate-level netlists, employing ML models such as Support Vector Machine (SVM) [20], Neural Network (NN) [21], eXtreme Gradient Boosting (XGB) [22], and Random Forest (RF) classifier [23].

In addition, Reinforcement Learning (RL) has gained success in various domains, including hardware security. RL-based static detection [24] and RL with adaptive sampling for on-chip detection [25] are notable examples. The common approach involves training the classifier model initially and adjusting hyperparameters to minimize the false negative rate, improving overall accuracy. Graph Neural Network (GNN) [26, 27] and AST [28] are generated for RTL code, yet there is a need to clarify how graphs carry both structural and behavioral attributes of the circuits [29].

Addressing concept drift after deploying a model is crucial. In security applications, [30] maps data samples into a space with fewer dimensions and learns a distance function for evaluating differences. Surprisingly, concept drift in the HT domain, despite potential evolution over time, has not been extensively studied.

On the other hand, explainability in ML for hardware security is explored in [31–33]. They use SHapley Additive exPlanations (SHAP) on benchmark datasets, showing promising results. However, SHAP has drawbacks, including disregarding causality and being influenced by human bias. It assesses feature contributions without explaining their real-world behavior.

Furthermore, multimodal learning has been explored in the AI community, with applications in understanding probability distributions across inputs with multiple modes [34]. Our work targets graph and Euclidean data fusion as modalities of interest, along with uncertainty estimation [35].

In the hardware security domain, dealing with fewer malicious data points necessitates working with small data [36]. This challenge has been addressed in various domains, such as material science [37] and anomaly detection [38].

2.1 Multimodal Learning

Multimodal learning [39] addresses complex problems by integrating information from multiple modalities, such as text, images, and audio, to obtain a comprehensive understanding of a given phenomenon. In our case, we use graphical data and tabular representations of the source circuits. This approach enables models to capture nuanced relationships that may be overlooked when considering each modality in isolation and thus empowers the model to make more robust predictions.

From a mathematical perspective, multimodal learning involves the integration of data representations into a unified framework. Let $X_1, X_2, ..., X_M$ represent M different modalities of data, each with their respective feature spaces $\mathcal{F}_1, \mathcal{F}_2, ..., \mathcal{F}_M$. The task is to learn a mapping f that captures the relationships between these modalities. Mathematically, this can be formulated as:

$$f : \mathcal{F}_1 \times \mathcal{F}_2 \times ... \times \mathcal{F}_M \rightarrow \mathcal{Y} \tag{1}$$

where $\mathcal{Y}$ is the target space, representing the desired prediction.

The challenge lies in effectively combining information from diverse modalities, which can be approached through various techniques such as late fusion or early fusion.

In late fusion [40], features are extracted independently from each modality and then combined at a later stage. This approach treats modalities as separate entities until a decision needs to be made and can be represented as:

$$f(x_1, x_2, ..., x_M) = g(h_1(x_1), h_2(x_2), ..., h_M(x_M)) \tag{2}$$

where h_i represents feature extraction for modality i, and g combines the extracted features.

In early fusion [41], information from different modalities is combined at the input level, resulting in a joint feature representation which can be expressed as:

$$f(x_1, x_2, ..., x_M) = h(x_1, x_2, ..., x_M) \tag{3}$$

where h combines the raw input data from all modalities.

2.2 Calibrated Prediction

Calibration involves ensuring that a model's confidence score accurately reflects the true probability of the prediction's correctness [42]. Let X be the input data, and Y be the output label. Given a training dataset $D = (x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$, the goal is to learn a function f that can predict the correct output label y for a given input x. The output of the model for an input x can be denoted as $f(x)$, and the true probability of the prediction's correctness can be denoted as $P(y = 1|x)$. A calibrated model produces a confidence score $g(x)$ that reflects the true probability of correctness of the prediction. The goal of calibration is to ensure that the confidence score $g(x)$ is well-calibrated, i.e., $P(y = 1|g(x) = p) = p$ for all p in the range $[0, 1]$.

Calibration is a crucial aspect in HT detection since it aids in determining the likelihood of the existence of a Trojan in a circuit, which can have a significant impact on decision-making. In situations where a model's confidence score is high, but the likelihood of a Trojan's presence is low, it is reasonable to assume that the circuit does not contain a Trojan. Conversely, if the confidence score is low but the likelihood

Algorithm 1 Mondrian ICP

Input : Training data D, test instance x, significance level α, number of trees T, and maximum tree depth d.
Output: Prediction set $C(x)$ for x.

Divide D into T disjoint subsets $D_1, \ldots, D_T$;
for $t \leftarrow 1$ **to** T **do**
  Sample D'_t from D_t by recursively partitioning D_t along randomly chosen hyperplanes until each partition contains at most $2d$ points.
  Train a classification model M_t on D'_t.
Compute the conformity scores $s_t(x)$ of x with respect to each model M_t.
Sort the conformity scores $s_t(x)$ in decreasing order.
Compute the p-values p_t of the T conformity scores $s_t(x)$ using the formula $p_t = \frac{T-t+1}{T}$.
Compute the threshold h such that $h = s_t(x)$ if $p_t > \alpha$, otherwise $h = \infty$.
Construct the prediction set $C(x)$ as the set of all labels y such that $s_t(y) \geq h$ for all models M_t.
return $C(x)$

of a Trojan's presence is high, further investigation of the circuit is necessary.

2.3 Conformal Prediction

Conformal prediction [6] as shown in Fig. 3 is an ML framework that quantifies prediction uncertainty by generating prediction sets. It enhances the inference of traditional models, ensuring reliable validity and enabling confidence estimation for individual predictions. In the context of detecting HTs, label-conditional validity is a vital property when dealing with an imbalanced dataset where label proportions differ significantly. This is particularly relevant since the likelihood of encountering a Trojan on a circuit is generally low. In addition, it is worth noting that minority classes are often disproportionately impacted by errors when label-conditional validity is absent [43]. However, this issue can be mitigated by ensuring label-conditional validity, which guarantees that the error rate for even the minority class will eventually converge to the chosen significance level in the long term. Sometimes, conformal prediction may produce uncertain predictions, meaning that prediction sets contain more than one value. This occurs when none of the labels can be rejected at the specified significance level.

When using conformal prediction, the confusion matrix differs slightly from the conventional one due to the unique nature of *prediction sets*, which consist of multiple values rather than a single value. In the case of binary classification, it is essential to consider the number of correctly predicted examples, which have a prediction set containing only the correct label, as well as the number of incorrectly predicted examples, where the prediction set includes only the incorrect label. Additionally, it is important to take into account the number of inconclusive predictions that occur when the prediction set contains both labels, as well as the number of examples with an *empty prediction set*. Furthermore, in some cases, it may be more appropriate to provide a *single value point prediction* instead of a prediction set or interval in a hedged forecast. In such cases, selecting the label with the highest p-value is a simple and reasonable option. The point prediction can be hedged by incorporating additional information that describes the uncertainty.

Our work relies on Mondrian Inductive Conformal Prediction (ICP) [44] in Algorithm 1 and to decrease the rate of false negatives in alert systems, we require class-based authenticity for samples classified as "Evolving Trojan." Additionally, we must ensure that the samples labeled as "Evolving Trojan" are indeed genuine to attain this goal.

When calculating the non-conformity scores, we only consider the scores related to the examples that share the same class as the object x_{n+1}, which we are testing hypothetically as shown below:

$$p_{n+1}^{C_k} = \frac{\left|\left\{i \in 1, \ldots, q : y_i = C_k, \alpha_{n+1}^{C_k} \leq \alpha_i\right\}\right|}{|\{i \in 1, \ldots, q : y_i = C_k\}|}$$

2.3.1 Time Complexity Analysis

The time complexity of the Mondrian ICP algorithm using Mondrian forests is analyzed considering n as the size of the training dataset D, T the number of trees, d the maximum tree depth, and m the number of features. The initial partitioning of D into T disjoint subsets is a linear operation with respect to n, resulting in a time complexity of $O(n)$.

Training each Mondrian tree M_t involves recursive partitioning, which has a complexity of $O\left(d \cdot \frac{n}{T} \cdot m\right)$, and tree training, which incurs $O\left(\frac{n}{T} \cdot m \cdot \log\left(\frac{n}{T}\right)\right)$. Aggregating

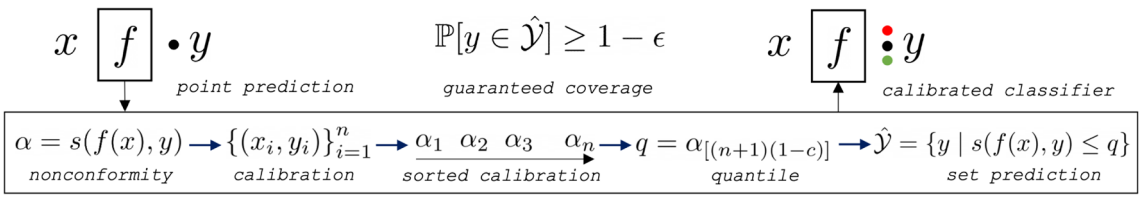

Fig. 3 Illustration of conformal prediction framework, showcasing the key components and stages involved in the process

these across T trees yields a total training complexity of

$$O\left(d \cdot n \cdot m + n \cdot m \cdot \log\left(\frac{n}{T}\right)\right).$$

The subsequent steps-computing conformity scores, sorting, p-value calculation, and constructing the prediction set-contribute $O(T)$, $O(T \log T)$, $O(T)$, and $O(T)$ respectively. Combining all steps, the overall time complexity is

$$O\left(n + d \cdot n \cdot m + n \cdot m \cdot \log\left(\frac{n}{T}\right) + T \log T\right).$$

Assuming T and d are constants, this simplifies to

$$O(n \cdot m \cdot \log n).$$

The best-case scenario arises when Mondrian trees are perfectly balanced with minimal depth, resulting in linear time complexity $\Omega(n \cdot m)$. In average cases, where trees are moderately balanced with typical data distributions, the complexity is $\Theta(n \cdot m \cdot \log n)$, reflecting expected real-world performance. The worst-case occurs with extremely unbalanced trees, large depth, and high feature dimensionality, leading to

$$O(n \cdot m \cdot d + n \cdot m \cdot \log n + T \log T).$$

The Mondrian ICP algorithm with Mondrian forests is computationally efficient for large datasets due to its online tree construction and probabilistic calibration. However, performance is influenced by the number of features m and tree depth d. While it is well-suited for applications requiring fast updates and calibrated outputs, scalability issues may arise with high-dimensional data or deep trees. Future optimizations could target feature selection and adaptive depth control to enhance scalability while preserving predictive robustness. The asymptotic notations summarizing these are presented in Table 1.

2.4 Guaranteed Coverage of Prediction

In the domain of HT detection, it is not only important to have a high level of confidence in the predictions made by a model but also a guarantee of the coverage of each prediction. The property of guaranteed coverage is an inherent property of conformal prediction, which provides statistical guarantees of the correctness of the model's predictions [45]. The theoretical guarantee of coverage is based on the significance level, which is the probability of the model making a mistake. For example, if we set the significance level to 0.05, it means that we allow the model to make mistakes only 5% of the time.

Table 1 Asymptotic notations for Mondrian ICP algorithm

Asymptotic notation	*Complexity*
Big-O (upper bound)	$O(n \cdot m \cdot \log n)$
Big-Ω (lower bound)	$\Omega(n \cdot m)$
Big-Θ (tight bound)	$\Theta(n \cdot m \cdot \log n)$

The theoretical guarantee of coverage is valid for any input x, that the true output label y will be contained in the prediction set $C(x)$ with a probability of at least $1 - \alpha$, where α is the significance level. Mathematically, this can be expressed as:

$$P(y \in C(x)) \geq 1 - \alpha$$

In other words, the probability of making a mistake is bounded by α, and as α decreases, the size of the prediction set decreases, leading to higher confidence in the model's predictions. For example, if we set $\alpha = 0.05$, it means that we are 95% confident that the true output label y is contained in the prediction set $C(x)$ for any input x. The use of conformal prediction provides a strong theoretical guarantee of the correctness of the model's predictions in the context of HT detection, and the corresponding proof is given in Theorem 1.

Theorem 1 *Let $\mathcal{D}$ be a probability distribution over a set $\mathcal{X} \times \{0, 1\}$, where $\mathcal{X}$ is a set of input features and $\{0, 1\}$ is the set of labels. Let $f : \mathcal{X} \to \{0, 1\}$ be a binary classifier, and let $\epsilon \in (0, 1)$ be a confidence level. Then, the conformal prediction algorithm outputs a set of predictions $C(x) \subseteq \{0, 1\}$ for each input $x \in \mathcal{X}$ such that:*

$$\mathbb{P}[(x, y) \sim \mathcal{D}, y \in C(x)] \geq 1 - \epsilon$$

where $(x, y) \sim \mathcal{D}$ denotes sampling a pair (x, y) from the distribution $\mathcal{D}$.

Proof The proof follows from the construction of the conformal prediction algorithm. Given an input x, the algorithm outputs a set of predictions $C(x)$ based on the observed labels of the training examples with similar input features to x. The algorithm guarantees that each prediction in $C(x)$ has a p-value less than or equal to ϵ for any new input with the same feature vector as x. Since the algorithm outputs a set of predictions, the probability that at least one of the predictions is correct is at least $1 - \epsilon$. □

Corollary 1 *Let $\mathcal{D}$, f, and ϵ be as in Theorem 1. For any sample size n, the conformal prediction algorithm outputs a set of predictions $C(x_1), \ldots, C(x_n)$ for each input $x_1, \ldots, x_n \in \mathcal{X}$ such that:*

$$\mathbb{P}[\forall i \in \{1, \ldots, n\}, (x_i, y_i) \sim \mathcal{D}, y_i \in C(x_i)] \geq 1 - \epsilon$$

where $(x_i, y_i) \sim \mathcal{D}$ denotes sampling a pair (x_i, y_i) from the distribution $\mathcal{D}$ for each i.

Proof The proof follows from a union bound over the n samples:

$$\begin{aligned}
&\mathbb{P}[\forall i \in \{1, \ldots, n\}, (x_i, y_i) \sim \mathcal{D}, y_i \in C(x_i)] \\
&\geq 1 - \sum_{i=1}^{n} \mathbb{P}[(x_i, y_i) \sim \mathcal{D}, y_i \notin C(x_i)] \\
&\geq 1 - n\epsilon
\end{aligned}$$

where the second inequality follows from Theorem 1. □

3 Notion of Evolution and Hardware Trojans

Darwin, in his book, *On the Origin of Species*, referred to "descent with modification," instead of *evolution*. Further, a more expansive definition of evolution was given by Futuyma [46]: "[biological evolution] is change in the properties of groups of organisms over the course of generations; it embraces everything from slight changes in the proportions of different forms of a gene within a population to the alterations that led from the earliest organism to dinosaurs, bees, oaks, and humans." Now, we narrow down the notion of evolution for HTs based on the following assumptions:

- The structural (genotype) and behavioral (phenotype) characteristics of HTs change over a period of time, and the changes are induced by the attacker;
- Structural changes can be mathematically formulated for the evolved Trojan as

  $$E_{HT} \rightarrow HT \blacksquare HT_{structural\_changes}$$

  where HT is an existing Trojan and ■ is the operation for structural changes which creates an evolved Trojan E_{HT};
- Behavioral changes are mapped with natural selection, which is the driving force for evolution. The attacker designs the HT such that it adapts to the IC (ecosystem) and its malicious impact is not easily detectable on the circuit. (i.e., it increases its chance of survival.)

We use the above assumption to include the notion of evolution and derive an evolved dataset in an ML-based HT detection engine. In the context of HTs, we can either detect the evolution or predict the evolution way ahead of time. The detection can be performed using anomaly detection [47]; however, here we will be focusing on the prediction of evolution. If we can predict the evolutionary changes in the dataset, a specific treatment can be performed to mitigate the impact of HT insertion. To the best of our knowledge, we have not come across any work in the literature that considers the evolutionary aspect while designing HT detection approaches.

Algorithm 2 Conformalized GAN

Input : Training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$, where $x_i \in \mathbb{R}^p$ and $y_i \in \{0, 1\}$ are the feature vector and label for the i-th example, respectively; significance level α; number of conformal predictors M; GAN generator G; discriminator model D

Output: Conformalized discriminator model D_{CP}

for $m = 1$ **to** M **do**
- $\mathcal{D}_m \leftarrow$ bootstrap sample of $\mathcal{D}$;
- Train GAN generator G_m on $\mathcal{D}_m$;
- Generate synthetic dataset $\mathcal{D}^m_{\text{synth}} = \{G_m(z_i)\}_{i=1}^{n}$, where $z_i \in \mathbb{R}^k$ are random noise vectors;
- Train discriminator model D_m on $\mathcal{D}_m \cup \mathcal{D}^m_{\text{synth}}$;

for $i = 1$ **to** n **do**
- $X_i \leftarrow \{x_i\} \cup \{G_m(z_i)\}_{m=1}^{M}$, where $z_i \in \mathbb{R}^k$ are random noise vectors;
- $\text{CP}_i \leftarrow$ conformal predictor trained on (X_i, y_i) with significance level α;
- $p_i \leftarrow \text{CP}_i(D(x_i))$;

Train conformalized discriminator model D_{CP} on $\{(x_i, y_i, p_i)\}_{i=1}^{n}$;

For each sample x_i in the test set D_{test}, make a prediction based on whether $D(x_i)$ is within the prediction interval I_i:

$$y_i = \begin{cases} 1 & \text{if } D(x_i) \notin I_i \\ 0 & \text{if } D(x_i) \in I_i \end{cases}$$

return D_{CP}

The evolutionary dataset optimization discussed in [48] optimizes any real-valued function over a subset of the space of all possible datasets. It is not feasible to adapt this method for our use case as our real-time data will be Non-Independent and Identically Distributed (Non-IID). An alternate approach can be to use evolutionary algorithms, as discussed in Box2d [49]. There, the problem statement is to evolve the structure of a toy car, provided the geometry of the car shape is translated to chromosomes. The issue with this approach is that we should know how the evolved car looks; however, in our case, we never know the structure of the evolved Trojan.

3.1 Genetic Algorithm

Genetic Algorithms (GAs) [50] have been used to evolve the architecture of NNs for understanding the security of logic locking [51]. The most challenging part of using GA is designing a fitness function. In our case, one possible design of fitness can focus on the ensemble efficiency of detection methods and then compare the similarity of the child Trojan with the list of HTs in a dictionary. However, the limitation of this fitness function is that it will never be able to estimate the fitness of Trojans that are out of distribution.

3.2 Generative Adversarial Network

Based on the game theory and optimization approach, the objective of generative modeling [52] is to analyze a set of training examples and acquire knowledge about the likelihood distribution that created them. Generative Adversarial Network (GAN) has been successfully used for detecting fake images [53] and text-to-image synthesis [54]. In the recent past, there has been a shift in focus towards the utilization of GANs for working with tabular data. An instance of this approach is used for conditional GAN, as demonstrated by [55], which models tabular data and is also effective with imbalanced data. A few of the open source libraries are [56–58]. Here are three prime motivations for using GAN for synthesizing the HT dataset.

3.2.1 Highly Imbalanced Data

In a real-time scenario, the labels for Trojan-Infected circuits are very rare and difficult to detect. This gives rise to the problem of an imbalanced dataset. Based on the existing literature, we believe GANs can be used to generate a more realistic synthetic dataset that complements the training phase.

3.2.2 Non-IID Case for Law of Large Number

The evolved Trojan may or may not be from the same distribution, and for this reason, we have to consider the case of Non-IID random variables. One such example is demonstrated in [59]. We also know that for a large enough dataset with Non-IID samples, the sample mean will converge to the true population mean as the sample size increases. This implies that as the size of the dataset increases, the statistical properties of the data become more reliable and consistent. Thus, the larger the dataset, the more accurate the model is likely to be, provided that there are enough computational resources to effectively process the data. The proof given in [60] for the strong law of large numbers can be generalized to an r-dimensional array of random variables where the sufficient condition becomes $E\left(|X|\left(\log^{+}|X|\right)^{r-1}\right) < \infty$ based on the theorem and corresponding proof for Non-IID given in [61]. The Non-IID case is worthy of our attention, as evolved HT might not represent the same distribution of population in real-time.

Theorem 2 *The strong law of large numbers for Non-IID random variables states that, if $X_1, X_2, \ldots, X_n$ are a sequence of non-identically distributed random variables with finite means $\mu_1, \mu_2, \ldots, \mu_n$, then for any $\epsilon > 0$,*

$$\mathbb{P}\left(\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}(X_i-\mu_i)=0\right)=1-\sum_{i=1}^{n}\frac{\mathrm{Var}(X_i)}{n^2\epsilon^2}.$$

Here, $\mathrm{Var}(X_i)$ denotes the variance of the i-th random variable X_i.

3.2.3 Risk Sensitive Application

Given the potential for significant financial losses, we cannot afford to tolerate even a small probability of misclassification. To mitigate this, we start by designing a near-realistic synthetic dataset using GAN by conformalizing the discriminator and generator.

4 Unimodal Hardware Trojan Detection

First, we consider the data to be unimodal and propose **PALETTE**, an ex**P**lainable fr**A**mework for evo**L**ving hardwar**E** **T**rojan de**TE**ction as shown in the left side of Fig. 1.

The first step is to extract the dataset, and in the case of HTs, we can have images, tables, and graphs as input dataset for HT classification. For example, the features extracted from an IC can be Scanning Electron Microscope (SEM) images, as used in [62, 63]. There has been a growing adoption of GNN for HT detection [26] by first converting the RTL-level code to AST and then either performing a graph classification or transforming the graph to a vector and then performing the classification. In our case, we have used the features extracted based on code branching from the TrustHub chip-level Trojan dataset [12] and the netlist synthetic dataset based on GAINESIS [64].

We then introduce the Conformalized GAN algorithm, which is illustrated in Algorithm 2. Our algorithm is inspired by [65], which leverages principled uncertainty intervals to generate high-quality images from corrupted inputs, and the uncertainty intervals provide a guarantee of containing the true semantic factors for any underlying generative model. The algorithm utilizes conformal prediction to generate evolving HTs and determine its associated level of confidence using prediction intervals; however, we must note that the absence of robustness guarantees in the proposed method leaves it vulnerable to adversarially crafted Trojans designed to evade detection. A comparison of Trust-Hub source dataset with the synthetically generated data, which we call the evolved dataset, is shown in Fig. 4 and the correlation matrix in Fig. 5. In contrast with traditional GANs our proposed method provides a more reliable means for generating evolving HTs. The comparison between real and synthetically generated datasets within the Trust-Hub chip-level Trojan dataset involves examining individual features to discern differences and similarities is shown in Fig. 6. This analysis aims to elucidate how effective and applicable synthetic data is in comparison to real-world instances.

Furthermore, the dataset is further fed as input to the conformal inference engine, which outputs set predictions

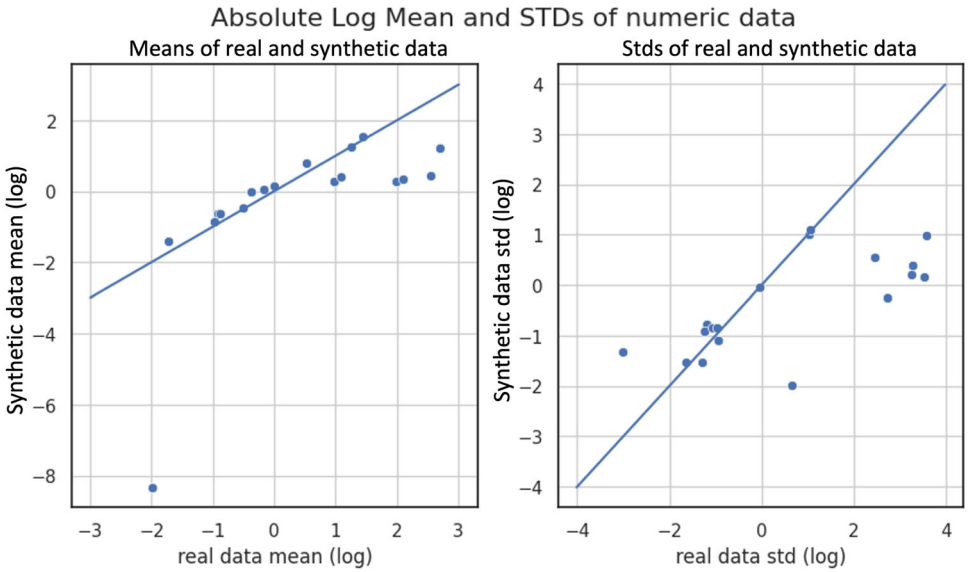

Fig. 4 Comparison of real and synthetically generated dataset on Trust-Hub chip-level Trojan dataset

instead of point predictions based on the significance level. The method is *algorithm-agnostic* as any ML classifier such as statistical or deep learning can be used. The non-conformity score is calculated for each prediction. The p-value represents the probability that the prediction is correct and is used to determine the guaranteed coverage. The important part of the solution is how we interpret the results in a risk-sensitive domain where we cannot tolerate even a single wrong decision.

Finally, we derive four different inferential use cases based on conformal inference. The motive is to quantify the uncertainty associated with each prediction and reduce the False Discovery Rate (FDR) for Trojan-Free (TF), Trojan-Infected (TI), or Evolving Trojan (T-EV).

1. The first is guaranteed coverage, which claims that based on the user-defined significance level, the predicted label will belong to that class. Here, considering the degree of

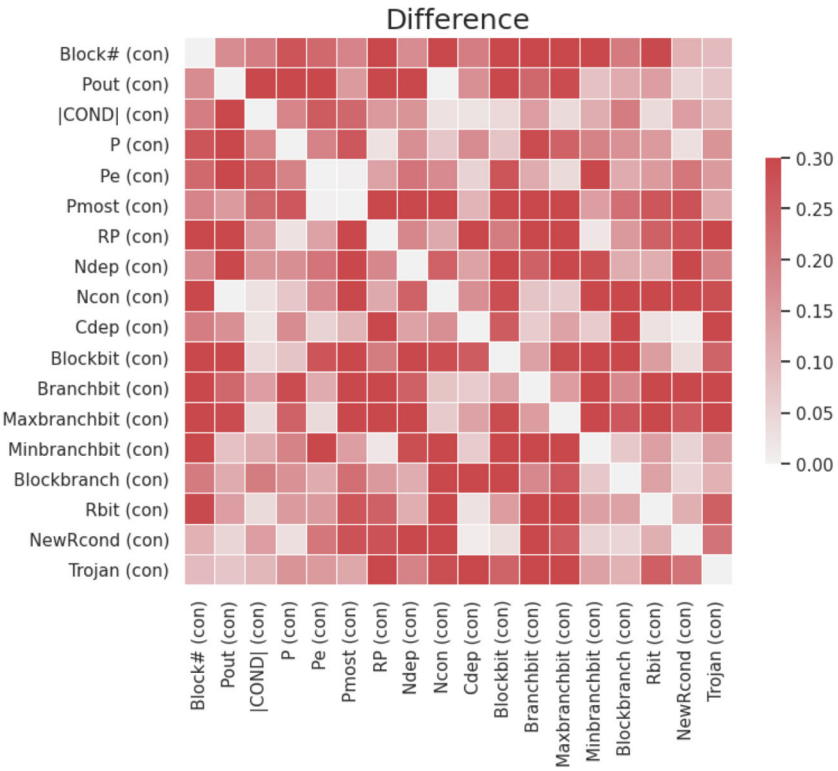

Fig. 5 The correlation matrix illustrates the difference between a real dataset and its synthetically generated counterpart, highlighting differences in feature values

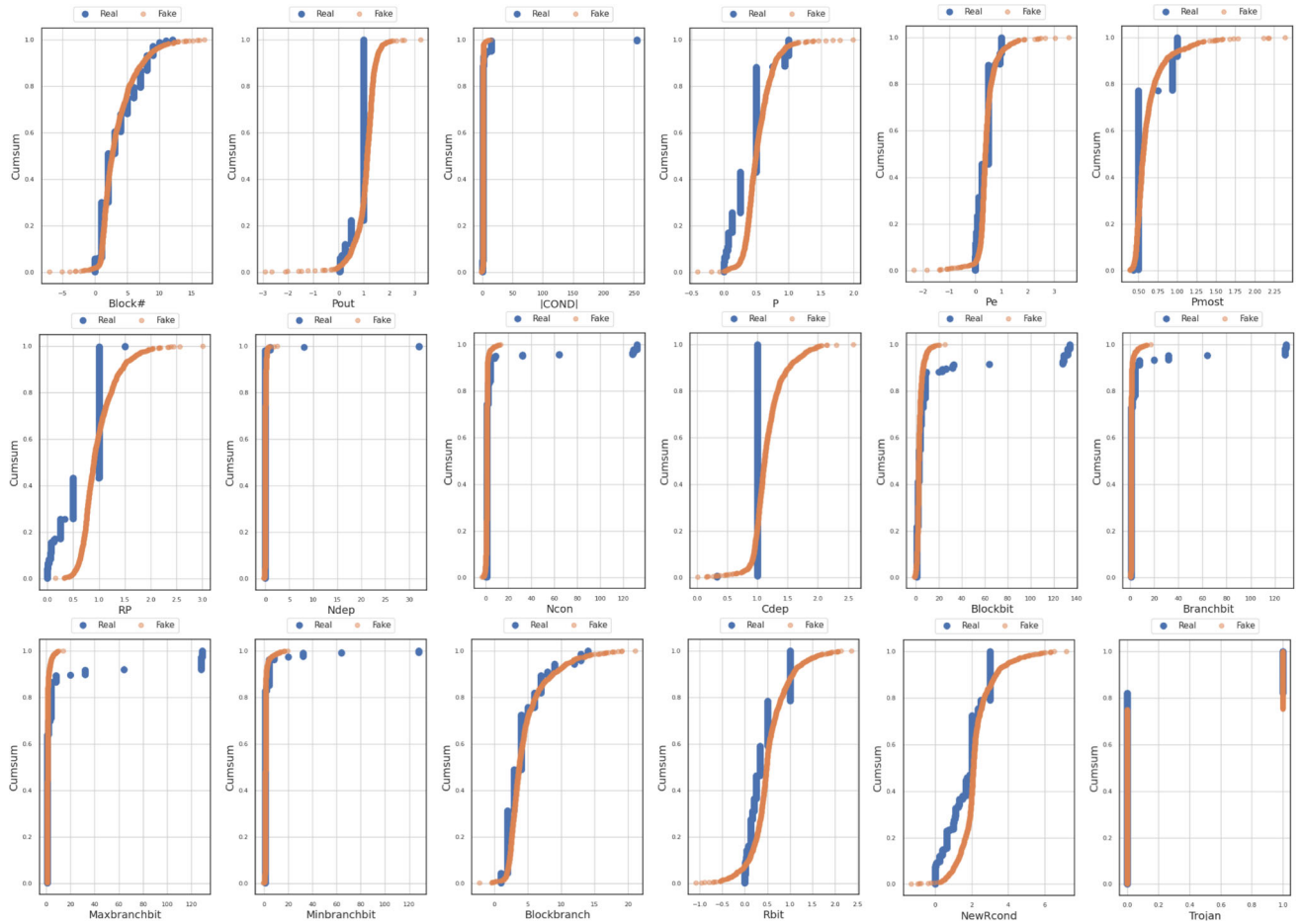

Fig. 6 Comparison between real and synthetically generated datasets within the Trust-Hub chip-level Trojan dataset, focusing on individual features. The analysis aims to highlight distinctions and similarities across features, shedding light on the efficacy and applicability of synthetic data in relation to real-world instances

risk associated with the prediction, a significance level is defined and applied to the p-values of each label for each data point.

2. The second is an inherent property of conformal prediction that results in a set prediction which can have all the labels {TF, TI, T-EV}, a combination of labels {TF, T-EV} or {TI, T-EV}, or a single label {TF}, {TI}, or {T-EV}.
3. The third, ranks the predicted HTs by calculating the confidence of each prediction and using it to rank the severity of being infected with a Trojan (TI, T-EV). The purpose of ranking is to prioritize which one to take action on first for mitigation.
4. Finally, the fourth is the calibrated explanation for the predictions, where the model says: *"I don't know"* and rejects the prediction. The proposed method overcomes the issues of local explanations by SHAP and provides a calibrated approach to reasoning out why a certain prediction has to be rejected. This is achieved by a *NULL* set, indicating that the model is not able to output the prediction for a specific significance level (1 - α).

5 Multimodal Hardware Trojan Detection

Now we emphasize on the need of multimodality for HT detection by proposing **NOODLE**, an u**N**certainty-aware hardware Tr**O**jan detecti**O**n using multimo**D**al deep **LE**arning. While state-of-the-art works on HT detection have focused mainly on choosing the right algorithm and choosing different representations of the dataset for improved accuracy, incorporating different modalities of the same data and feeding it to the ML system has not been investigated. By performing information fusion derived from different modalities, a more refined data representation can be achieved. Furthermore, in a practical scenario, we encounter missing values while collecting data, and this may lead to missing modalities when dealing with a multimodal ML approach.

So, we also need a method that handles missing modalities for any given dataset. Lastly, in the domain of hardware security, it is difficult to get enough data for training, especially the Trojan-Infected circuits, because of the rarity of the event. In such a situation, we need to work with limited data.

Our proposed framework is shown in the right side of Fig. 1 emphasizing the design and implementation, and a pseudocode is also provided in Algorithms 3 and 4. We choose to use two modalities, i.e., graph and tabular data representations. Methods such as multimodal autoencoder [66] have been used for handing missing modalities; however, we use GANs [67] to increase the dataset size to 500 data points as it aims to generate samples that are consistent with the joint distribution of the observed modalities and facilitate more effective multimodal fusion. The data points labeled as Trojan-Free (TF) will be segregated, and only these will be used to generate more data points using GAN so that they represent the same label, and we will do the same for data labeled as Trojan-Infected (TI). Before performing multimodal learning, we first explain the working of uncertainty-aware model fusion.

To perform an uncertainty-aware multimodal fusion, we leverage conformal prediction p-values for the model fusion as described in Algorithm 4. First, we use a Convolutional Neural Network (CNN)-based classifier for graph and tabular data sources with a designed non-conformity score that provides p-values for each label and for each data modal. The below non-conformity score can be used to get calibrated conformal predictions:

$$NS = \sum_{t=1}^{T} B_t(x, y) \tag{4}$$

where $B_t(x, y)$ is the non-conformity score of (x, y) computed from a classifier, h_t. Thus, for every class label $y(j)$, $j \in \{1, ..., M\}$, we have an individual *NULL* hypothesis for each data source, $H0_1, H0_2, ..., H0_N$, where M is the number of class labels, which in our case is either TF or TI, and N is the number of data sources. Thus, for every class label $y(j)$, we obtain N p-values, $p(i)$, $i = 1, ..., N$ (one for each modality). These p-values are then combined into a new test statistic $C(p(1), ..., p(N))$, which is used to test the combined *NULL* hypothesis $H0$ for class label $y(j)$. The conformal prediction region at a specified confidence level, r_E, is then presented as a set containing all the class labels with a p-value greater than $1 - E$. The mentioned steps help in the realization of uncertainty-aware multimodal fusion.

After obtaining a sufficient number of data points for the experiment, we implement multimodal ML using the graph and tabular data. Specifically, we have employed a CNN for binary classification. It is worth mentioning that any ML model can be optimized through hyperparameter tuning to enhance accuracy. However, our primary emphasis is on assessing the effectiveness of uncertainty-aware multimodality by accessing early and late fusions. Finally, the model will be used to make further informed decisions for the detection of HTs.

Algorithm 3 Multimodal deep learning

Input : RTL-level files (Verliog) of circuits
Output: Decision (D) = Trojan-free or Trojan-infected
for *each circuit C* **do**
  Convert C to Graph data **G** and Euclidean data **T**.
  if ∃ *missing modalities* **then**
    perform GAN to impute the missing modality.
Feed the modalities to *CNN*-based classifier.
for *each modalities M* **do**
  Use Algorithm 4 for uncertainty-aware information fusion.
  Perform early fusion.
  Perform late fusion.
Choosing the winning fusion method.
return D.

Algorithm 4 Uncertainty-aware information fusion

Input : Number of data sources N;
Training sets for each data source $T_1 = \{(x_1^{(1)}, y_1), \ldots, (x_n^{(1)}, y_n)\}, \ldots, T_N = \{(x_1^{(N)}, y_1), \ldots, (x_n^{(N)}, y_n)\}$, where $x_i^{(j)}$ is the ith data point belonging to the jth data source and y_i is the class label of the ith data point;
Number of classes M;
Class labels $y^{(i)} \in Y = \{y^{(1)}, y^{(2)}, \ldots, y^{(M)}\}$;
Classifiers $S_1, \ldots, S_N$ for each data source;
Confidence level E.
Output: Conformal prediction regions $r_E = \{y^{(j)} : \hat{p}_j > 1 - E, y^{(j)} \in Y\}$.
Get the new unlabeled example w.r.t each data source $x_{n+1}^{(1)}, \ldots, x_{n+1}^{(N)}$.
Evaluate conformal predictors and classifiers $S_1, \ldots, S_N$ corresponding to each data source, compute p-values $p_j^{(i)}$, where $i = 1, \ldots, N$ corresponds to the ith data source and $j = 1, \ldots, M$ corresponds to the jth class label.
for *each class label* $y^{(j)}$, $j = 1, \ldots, M$ **do**
  Compute p-value, $\hat{p}_j$, of combined hypothesis from N modalities
return r_E.

5.1 Time Complexity Analysis

The time complexity analysis of multimodal deep learning and uncertainty-aware information fusion can be broken down as follows.

Let n represent the number of circuits, m the number of modalities, d the feature dimensionality, and T the number of data sources.

For multimodal deep learning (Algorithm 3), each circuit is first transformed into graph data G and Euclidean data T, with a complexity of $O(n \cdot d)$ due to feature extraction. If any modality is missing, a GAN is used for imputation, which introduces a higher complexity of $O(n \cdot d^2)$ due to iterative updates and backpropagation. Each modality is then processed by a CNN-based classifier, where the complexity is $O(n \cdot m \cdot L)$, with L being the number of CNN layers. Additionally, for each modality, Algorithm 4 (uncertainty-aware information fusion) is invoked, adding to the computational load. Early and late fusion methods are applied over m modalities, contributing $O(n \cdot m)$, and finally, the winning fusion method is selected with negligible overhead $O(1)$.

The total complexity of Algorithm 3 sums up to $O(n \cdot d + n \cdot d^2 + n \cdot m \cdot L + n \cdot m)$, which simplifies to $O(n \cdot d^2)$ in feature-rich scenarios where $d \gg m$.

For uncertainty-aware information fusion (Algorithm 4), the training data is sourced from T data streams with n samples each, resulting in $O(n \cdot T)$ complexity. Conformal predictors evaluate p-values for M classes, leading to $O(n \cdot T \cdot M)$. Sorting these p-values for statistical analysis adds $O(n \cdot M \log M)$. Hypothesis testing and the construction of confidence regions further contribute $O(n \cdot T \cdot M)$. Finally, returning the prediction set incurs $O(n)$.

The total complexity of Algorithm 4 is $O(n \cdot T \cdot M + n \cdot M \log M)$, which simplifies to $O(n \cdot M \log M)$ if $M \log M$ dominates TM.

Combining the complexities of Algorithms 3 and 4, the time complexity of the multimodal deep learning pipeline becomes $O(n \cdot d^2 + n \cdot M \log M)$. In the best-case scenario, with minimal feature dimensionality and a small number of modalities, the complexity is $\Omega(n \cdot d)$, indicating linear complexity. In an average case with moderate feature size and a balanced number of modalities and data sources, the complexity is $\Theta(n \cdot d^2 + n \cdot M \log M)$, representing standard performance in real-world applications. The worst-case scenario involves high feature dimensionality, deep CNNs, and multiple modalities and sources, leading to $O(n \cdot d^2 + n \cdot M \log M)$.

It is worth noting that while multimodal deep learning and uncertainty-aware fusion scale effectively in real-world applications, they are particularly sensitive to the feature dimensionality d, number of modalities m, and data sources T. The CNN training process dominates in feature-rich settings, while uncertainty-aware fusion introduces additional computational overhead due to p-value calculations and sorting. Optimizations such as feature selection, dimensionality reduction, and efficient p-value computations can enhance scalability in high-dimensional datasets. The asymptotic notations summarizing these are presented in Table 2.

Table 2 Asymptotic notations for multimodal deep learning and uncertainty-aware information fusion

Asymptotic notation	*Complexity*
Big-O (upper bound)	$O(n \cdot d^2 + n \cdot M \log M)$
Big-Ω *(lower bound)*	$\Omega(n \cdot d)$
Big-Θ *(tight bound)*	$\Theta(n \cdot d^2 + n \cdot M \log M)$

6 Experimental Results

For experiments, the augmentation of the dataset is implemented using conformalized GAN, leading to the generation of evolved HTs. In the case of multimodal data, we address missing modalities by utilizing conformalized GAN techniques. The experiment is divided into two distinct cases. For the first scenario (i.e., **PALETTE**), characterized by a unimodal dataset, the application of conformal prediction on the source dataset is performed. This phase serves as an illustrative platform, showcasing diverse aspects of inference such as guaranteed coverage, set prediction, ranking of HTs, and calibrated explainability. For the second scenario (i.e., **NOODLE**), the design of a multimodal deep learning architecture adept at handling missing modalities is achieved through the integration of conformalized GAN. Subsequently, an ensemble fusion approach is devised, leading to the creation of a fusion framework. Within this ensemble, the integration of uncertainty-aware information fusion is implemented through conformal prediction. The solutions are implemented using Python (version 3.9) on macOS (version 13.3.1), utilizing a system with 8 GB of RAM and a built-in GPU.

6.1 Unimodal Approach Experiments

For the unimodal approach, we use two sets of data. The first dataset utilized is the synthetic GAINESIS dataset [64], characterized by binary labels. The second dataset involves the use of the Trust-Hub chip-level Trojan dataset [12]. This dataset encompasses RTL source code files for each IP core design, encompassing both malicious and non-malicious functionalities. The malicious functionalities are typically embedded within conditional statements that are rarely executed. Consequently, the ML features are derived from these conditional statements. We first generate 10,000 data points using the proposed conformalized GAN with the given source dataset and pick 20% of the evolved dataset. The generated dataset has labels TF_G and TI_G, where as the source dataset has labels TF_S and TI_S. In our evolved dataset, we create three labels as shown below. First, Trojan-Free (TF) which consists of TF_S and TF_G; second, Trojan-Infected (TI) where we only consider the label TI_S; finally, the third

Table 3 PALETTE's dataset split

	Train	*Calibration*	*Test*
TF	1436	470	471
TI	114	33	44
T-EV	308	117	105
Total count	1858	620	620

label is Evolved Trojan (T-EV) which consists of the label TI_G.

$$Label = \{TF, TI, T-EV\}$$

The dataset is split into training set, calibration set, and test set with a ratio 2:1:1, as shown in Table 3. The experimental results with source code and the dataset are hosted on GitHub.[1]

6.1.1 Baseline Model

For the baseline model, we use logistic regression as a classifier to detect the evolving HTs, and we evaluate the accuracy of the model as a performance metric. If we use logistic regression to detect HTs, the overall accuracy is 0.85, while if we use conformal inference as a wrapper over the logistic regression, the accuracy increases to 0.88 for $\alpha = 0.05$ and 0.90 for $\alpha = 0.1$. This also shows the performance improvement of any classification model when used with underlying conformal inference. The confusion matrix of the base model and conformal inference wrapper is shown in Table 4.

6.1.2 Conformal Inference

The results obtained after implementing conformal inference for detecting evolving HTs are shown in Table 5. Each row represents the circuit and the truth value in columns TF, TI, and T-EV. In addition, the p-values for each label are mentioned in the columns pTF, pTI, and pT-EV. Finally, the detected Trojan is mentioned in column y_pred with $\alpha = 0.05$. The column *Conf* represents the confidence score of each detected label for each circuit, which is obtained by $1 - 2^{nd}\, p_{max}$. An application of conformal inference is the improvement of detection quality for evolving HTs. For example, in Table 5, circuit 2 is detected as TF because the p-values of TI and T-EV are less than the value of $\alpha = 0.05$. The circuit 4 is detected as T-EV, in which we see that p-values for TF, TI, and T-EV are greater than the value of α, so all the labels are set as T (True), and the maximum of the p-value is specified for the detected label. For example, with conformal inference, we can say that with 95% detection guarantee (as $\alpha = 0.05$ decided by the user), circuit 4 is detected as an evolving Trojan with a confidence score of 0.886. This helps the end user have granular-level reasoning for trustworthy and robust decision-making.

[1] https://github.com/cars-lab-repo/PALETTE

Table 4 Confusion matrix of base model and conformal inference with significance level of 95%

	Logistic regression			*Conformal inference*		
	TF	TI	T-EV	TF	TI	T-EV
TF	462	8	8	525	24	44
TI	11	26	2	0	10	7
T-EV	52	0	51	0	0	10

Furthermore, we consider the dataset in two dimensions characterized by three distinct labels: TF, TI, and T-EV. The data distribution follows a bivariate normal distribution, with diagonal covariance matrices assigned to each label. To predict sets, we utilize the MapieClassifier, which operates based on the distribution of softmax scores corresponding to the true labels. As shown in Fig. 7, we compute prediction sets for three different alpha values (0.2, 0.1, and 0.05), each representing varying levels of class coverage. In instances where the class coverage is insufficient, prediction sets may turn out to be empty. This occurs when the model encounters uncertainty at the boundary between two labels. These empty regions, termed the *NULL* set, tend to diminish as we increase the coverage level. In Fig. 8, the training dataset is partitioned into five folds, with each fold used as a calibration set. The plot shows the distribution of conformity scores across each calibration set, with alpha = 0.1. It is noticed that the estimated quantile shows slight variability across the calibration sets.

We also share the results for binary labels (TF, TI) on the GAINESIS dataset in Table 6. The method is validated on 4600 synthetic circuits with and without Trojans, and the corresponding confidence score is shown in the column *Conf*.

In addition, we explore variations of conformal predictors as described in [68]. Table 7 shows that the Mondrian conformal predictor is very strict on detecting the evolved hardware Trojans as compared to the risk-adaptive prediction set *raps*,

Table 5 Conformal inference and associated p-vales for Trust-Hub chip-level Trojan dataset

	TF	TI	T-EV	pTF	pTI	pT-EV	y_pred	Conf
1	T	F	F	0.319	0	0.003	TF	0.997
2	T	F	F	0.243	0.002	0.006	TF	0.994
3	T	T	F	0.161	0.078	0.016	TF	0.992
4	T	T	T	0.114	0.053	0.119	T-EV	0.886
5	T	F	F	0.645	0.001	0.004	TF	0.996
6	F	F	T	0.653	0	0.971	T-EV	0.365
7	T	F	F	0.3	0	0.002	TF	0.998

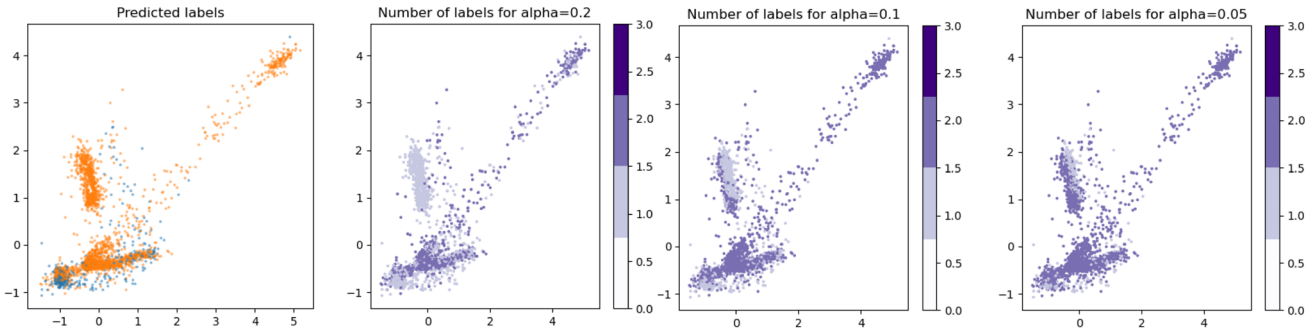

Fig. 7 Comparison of prediction sets generated for a two-dimensional dataset across three different alpha values: 0.2, 0.1, and 0.05

naive, and *top_k* methods with varying significance levels. The *naive* and *top_k* first get the model output of the true class, and *naive* makes the estimated set prediction by getting quantiles from the score distribution, while *top_k* gets the quantiles from the distribution of the ordered positions of the true label. The *raps* method first sorts the model output in decreasing order to get the cumulative output of the true class and then uses it to obtain quantiles from cumulative score distributions. Furthermore, with a very high coverage of 95% ($\alpha = 0.05$), *raps* and *naive* detect almost three times more Trojans as compared to Mondrian, while the detection coverage becomes almost similar when the coverage level is increased.

In Fig. 9, we show the estimated quantiles and the significance level (alpha) in statistical analysis. It shows that as alpha increases, the estimated quantile tends to rise, allowing for a broader acceptance region and a higher likelihood of capturing extreme values. It spans three alpha levels (0.2, 0.1, and 0.05), demonstrating that higher alphas may lead to quantiles in uncertain areas, possibly resulting in the *NULL* set. Conversely, lower alphas create stricter acceptance regions, impacting quantile estimation.

6.1.3 Performance Metrics

Unlike classification task which produces Receiver Operating Characteristic (ROC) and Area Under Curve (AUC), conformal inference produces *effective coverage* and *efficiency*, i.e., average prediction set size, as performance metrics. The limitation of ROC and AUC is that they can be impacted by an imbalanced dataset. In Fig. 10, we show the two different performance metrics for Mondrian conformal predictors. The coverage score measures the proportion of instances in which the True label falls within the predicted region. It is typically measured at different confidence levels. Higher coverage indicates a more conservative prediction method. Now, since validity is guaranteed for all conformal predictors, the key performance metric is efficiency, i.e., the size of the label sets, where smaller sets are more informative and indicate higher efficiency. It is also a direct measure of how good the conformal predictor is at rejecting class labels.

When evaluating conformal prediction methods, there are several metrics that can be used to assess their performance. In Table 8, we show the various performance metrics associated with the detection mechanism for significance levels ranging from 0.05 to 0.9. For example, avg_c indicates the average number of class labels in the prediction sets; this metric serves as a straightforward indicator of the conformal predictor's ability to accurately discard class labels. The significance level is like a threshold that controls how often the ML model makes incorrect predictions. If we set a higher significance level, the model will make fewer errors, but its predictions may be less precise. So, we need to find the right balance to get the best results from our model.

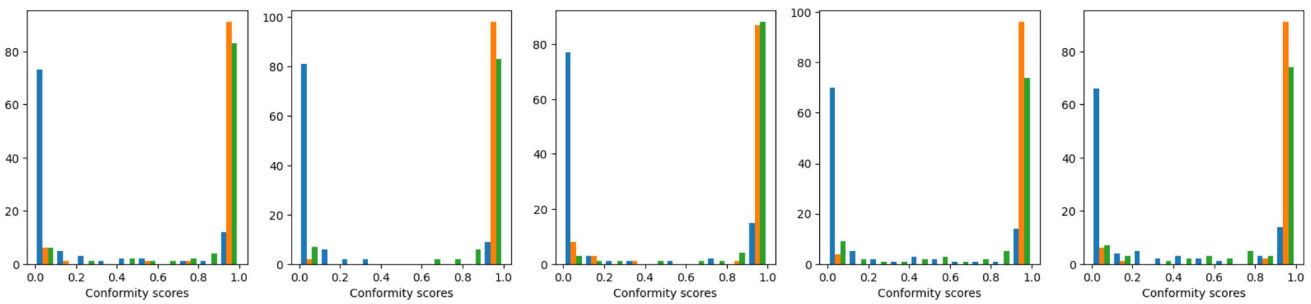

Fig. 8 The distribution of scores observed across calibration folds for the Mondrian Conformal Predictor. The visual representation offers insights into the variation and spread of scores, providing a comprehensive view of the predictor's performance in different calibration scenarios

Table 6 Conformal inference for GAINESIS dataset

circuit	TI	TF	y-pred	Conf
1	FALSE	TRUE	TF	0.891
2	FALSE	TRUE	TF	0.796
3	FALSE	TRUE	TF	0.996
4	FALSE	TRUE	TF	0.997
...	...	...	..	...
4596	FALSE	TRUE	TF	1
4597	FALSE	TRUE	TF	0.991
4598	TRUE	FALSE	TI	0.995
4599	FALSE	TRUE	TF	0.989
4600	FALSE	TRUE	TF	0.992

6.1.4 Risk-Aware Ranking

We leverage *confidence score* from the conformal inference as a ranking mechanism for evolved HTs. For the below-given circuits 12, 13, and 14 from the Trust-Hub dataset, we calculate their confidence score (C) with $\alpha = 0.05$.

$$\alpha_{0.05}(\text{ circuit } 12) = \{T - EV\}_{C=0.88}$$
$$\alpha_{0.05}(\text{ circuit } 13) = \{T - EV\}_{C=0.81}$$
$$\alpha_{0.05}(\text{ circuit } 14) = \{T - EV\}_{C=0.61}$$

Confidence in a model's prediction is determined by its p-value which indicates the probability of obtaining a similar outcome under the *NULL* hypothesis. Higher confidence implies greater accuracy. This metric is defined as:

$$\text{Confidence}(x) = \sup\{1 - \epsilon : |\Gamma_\epsilon(x)| \leq 1\}$$

By ranking the predictions, conformal prediction can offer a more informative way to assess the reliability of individual predictions. The ranked list allows decision-makers to set thresholds or confidence levels for accepting or rejecting predictions based on their position in the ranking. This provides a flexible tool for controlling the trade-off between accuracy and reliability in different applications. In Table 9 we show the *confidence* and *credibility* of the detected labels.

Table 7 Comparison of conformal predictors with corresponding significance level on Trust-Hub chip-level Trojan dataset

Alpha	Mondrian	Raps	Naïve	Top_k
0.05	10	37	35	0
0.5	45	57	57	61
0.9	45	61	61	61

The credibility is obtained by considering the maximum p-value of the given set prediction. Credibility quantifies the quality of the new data points.

6.1.5 Calibrated Explanations for Reject

When the model is not able to detect evolving HT, the model simply says, *"I don't know"* by giving a *NULL* set as the output. In a risk-sensitive domain, a model with no output is better than a decision that is not confident. Our framework also provides the reason for rejecting the decision with a calibrated explanation, as shown in Fig. 11, which is different from traditional explainable methods. If we apply a significance level of 0.5 to the given circuit, none of the p-values for TF (0.45), TI (0.32), and T-EV (0.23) exceed the significance level. As a result, we reject the decision. The explanation for this rejection is based on Local Interpretable Model-agnostic Explanations (LIME) [69].

However, the differentiating factor as compared to SHAP (which disregards causality and is affected by human bias) is that before providing any explanation for the rejection, we ensure that it is calibrated. The approach begins by creating modified versions of the original instance called perturbed instances, where small random changes are introduced. Conformal prediction is then utilized to create prediction regions that estimate the reliability or confidence level of the explanations, and LIME is then used again on these perturbed instances to generate explanations for each of them. The prediction regions obtained through conformal prediction act as a calibration mechanism, guaranteeing that the explanations accurately reflect their level of reliability.

6.2 Multimodal Approach Experiments

For the multimodal approach, we use the features extracted from the Trust-Hub RTL-level (Verilog) Trojan dataset based on code branching features [12] and the graph dataset in [11] which includes RTL source code files (Verilog) for each IP core design containing both malicious and non-malicious functions. The hyperparameter values used for multimodal learning are provided in Table 10 to ensure the reproducibility of the experiment. We also use the synthetic dataset for tabular and graphical representation of the circuits and their relevant performance as loss is shown in Fig. 12. For the Tabular Model, the train loss initially decreases sharply but starts to increase slightly afterward, suggesting potential overfitting. Meanwhile, the test loss decreases initially but then stabilizes, indicating that the model's performance on unseen data has reached a stable point. Similarly, the JSON Model exhibits a steady decrease in both train and test losses, with the test loss plateauing slightly. In both cases, observing the

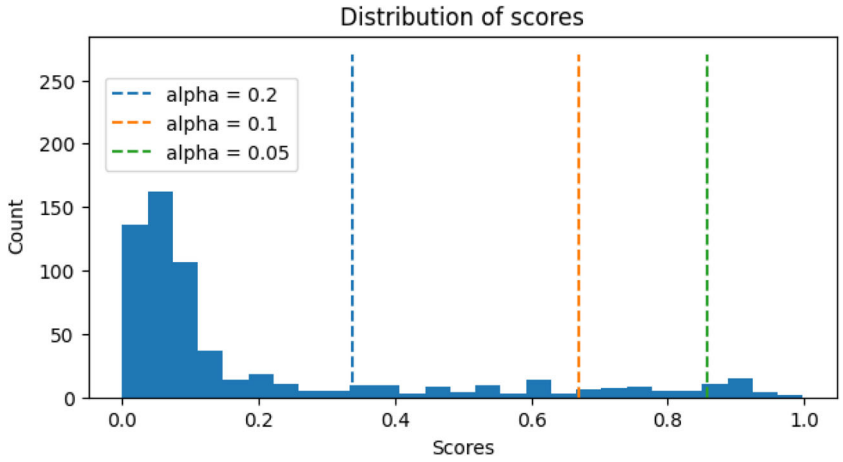

Fig. 9 Distribution of the scores with the calculated quantiles

point where the test loss stabilizes or begins to increase can approximate the convergence point. The experimental results with source code and the dataset are hosted on GitHub.[2]

6.2.1 Brier Score

For any of the classification problem statements, the most common performance metric is model accuracy, followed by various other complementing metrics such as precision, recall and F1-score. However, these metrics can be misleading in situations where the class distribution is imbalanced, as in our case. For this reason, we have used the Brier score as an evaluation metric for assessing the quality of probabilistic predictions in the classification of HTs. The Brier score, which offers insights into accuracy and calibration, is defined as follows:

$$BS = \frac{1}{N}\sum_{i=1}^{N}(p_i - o_i)^2 \tag{5}$$

where N is the number of instances, p_i is predicted probability for instance i, and o_i is the observed outcome for instance i. The Brier score ranges from 0 to 1. A score of 0 indicates perfect accuracy, meaning the predicted probabilities perfectly match the actual outcomes. A score of 1 signifies complete inaccuracy, where the predicted probabilities are entirely different from the actual outcomes.

We begin the evaluation process by independently assessing each modality. This involves conducting binary classification on both the graph dataset and the tabular data. The resulting comparative Brier scores for these classification tasks are presented in Table 11. The experimental outcome demonstrates that, when employing the same CNN-based deep learning model with identical hyperparameters, the graph dataset yields a superior Brier score (0.1798) compared to the tabular data (0.1913). It is worth noting that while we established a baseline model using CNN, any other alternative classification algorithms can also be employed in this context. Then, we test the solution with two different information fusion approaches, i.e., early fusion (feature) and late fusion (decision). As shown in Table 11, the early fusion approach, which combines the graph and tabular data before processing, yields a Brier score of 0.1685. On the other hand, the late fusion strategy, which integrates the graph and table data after individual processing, demonstrated the best performance with a Brier score of 0.1589.

It is worth noting that neither of these data fusion methods can be deterministically labeled as superior [70] as each one of them will demonstrate their potential to produce favorable outcomes when the data distribution changes. For this reason, we implement both of the fusion approaches and choose the approach that provides a better Brier score (i.e., closer to 0), as mentioned in Step 8 of Algorithm 3. The corresponding Brier score distribution with mean interval is also shown in Fig. 13a and b for early and late fusion, respectively. This provides a comprehensive view of predictive accuracy across multiple scenarios and is also useful for comparing models and understanding the variability in performance.

6.2.2 Confidence Calibration Curve

The confidence calibration curve plots observed probabilities of occurrence as a function of the predicted probabilities for the classification model, as shown in Fig. 14. For the model to be perfectly calibrated, it will have all data points along the diagonal; however, in our case, the model is not well calibrated because of the highly imbalanced dataset. These are the cases on which any decision-maker should focus while

[2] https://github.com/cars-lab-repo/NOODLE

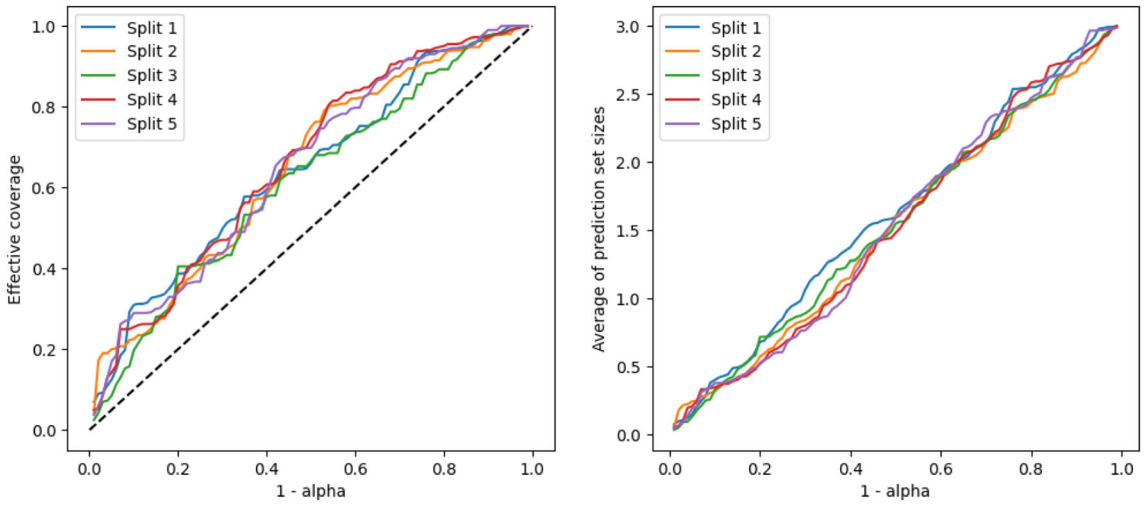

Fig. 10 Effective coverage and average prediction set size for Trust-Hub chip-level Trojan dataset

making a risk-aware decision and not completely relying on accuracy alone. It helps evaluate the alignment between a model's predicted probabilities and the actual likelihood of events.

A histogram at the bottom of Fig. 14 shows the predicted chance for 109 test data. It describes the distribution of the forecasts and helps with visualization of the sharpness, i.e., tendency of the predictions to lie at the extremes of the 0–1 distribution, and is equal to the variance of the predictions.

The calibration curve indicates that the model is miscalibrated at lower confidence levels, displaying overconfidence around a predicted probability of 0.25, where actual outcomes are significantly lower. This miscalibration can lead to unreliable predictions in critical risk assessment scenarios. In contrast, at high predicted probabilities (e.g., 1.0), the model demonstrates strong calibration, with observed outcomes closely matching predictions. The accompanying histogram further illustrates that the model tends to favor extreme probabilities, either very low or very high, while largely avoiding moderate-confidence predictions. This behavior suggests a lack of nuance in the model's probabilistic outputs. The overestimation of rare events at low probabilities could lead to false alarms, particularly in sensitive domains like hardware security, while the reliability of high-confidence predictions suggests that such outputs can be trusted. The scarcity of intermediate predictions may reflect model architecture limitations or a lack of sufficient ambiguous data. Furthermore, large error bars in certain bins highlight the impact of limited sample sizes on calibration reliability, suggesting that increasing the dataset or employing bootstrapping techniques could enhance model performance.

The confidence calibration curve reveals both strengths and weaknesses in the model's probabilistic predictions. While the model exhibits strong reliability at high-confidence levels, it struggles with overconfidence in lower probability

Table 8 Performance metrics of conformal inference on Trust-Hub chip-level Trojan dataset

sig	mean_err	avg_c	n_correct	mean_T-EV
0.05	0.049	1.040	589	0.012
0.1	0.102	0.941	556	0.045
0.2	0.204	0.812	493	0.133
0.3	0.303	0.701	431	0.220
0.4	0.406	0.596	367	0.319
0.5	0.504	0.497	307	0.423
0.6	0.604	0.397	245	0.536
0.7	0.702	0.298	184	0.650
0.8	0.798	0.202	125	0.764
0.9	0.900	0.100	61	0.884

Table 9 Adoption of confidence for risk-aware ranking on Trust-Hub chip-level Trojan dataset

	confidence	credibility	y_pred
1	0.997	0.319	TF
2	0.994	0.242	TF
3	0.922	0.162	TF
4	0.886	0.119	T-EV
5	1	0.645	TF
6	0.999	0.97	T-EV
7	0.998	0.301	TF

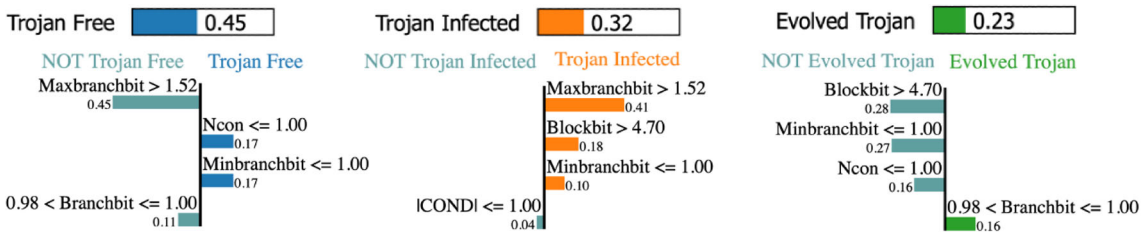

Fig. 11 Calibrated explanation for reject

ranges and avoids making moderate-confidence predictions. To address these limitations, post-hoc calibration techniques such as temperature scaling, Platt scaling, or isotonic regression could be applied to improve the model's calibration, particularly in the low-to-mid probability ranges. Improving calibration is critical for ensuring the model's outputs can be trusted in real-world applications, especially those involving high-stakes decision-making. A well-calibrated model enhances predictive accuracy and provides more reliable confidence estimates, which are essential for risk-sensitive tasks.

6.2.3 ROC-AUC Curve

The Receiver Operating Characteristic (ROC) curve illustrates the balance between sensitivity and specificity in a model. It provides a visual representation of how these two metrics change as the threshold for classifying a condition varies. The Area Under the Curve (AUC), on the other hand, quantifies the likelihood that a randomly chosen pair of circuits, one with the Trojan and one without, will be accurately

Table 10 Hyperparameters for multimodal learning model

Category	Parameter	Value
Data Splitting Parameters	Test Size	20% (`test_size=0.2`)
	Random State	42
Graph Data (Conv Layers)	Filters	64
	Kernel Size	3
	Activation Function	ReLU
	Pooling Type	MaxPooling1D
	Pool Size	2
Tabular Data (Dense Layers)	Neurons	64
	Activation Function	ReLU
Fusion Layer	Dense Layer Neurons	128
	Activation Function	ReLU
	Dropout Rate	0.5
	Output Neurons	1
	Output Activation	Sigmoid
Training Hyperparameters	Optimizer	Adam
	Loss Function	Binary Crossentropy
	Metrics	Accuracy
	Epochs	100
	Batch Size	5
	Verbose	1
Evaluation Metrics	Accuracy Score	Calculated after training
	Confusion Matrix	Calculated after training
	Brier Score Loss	Calculated after training

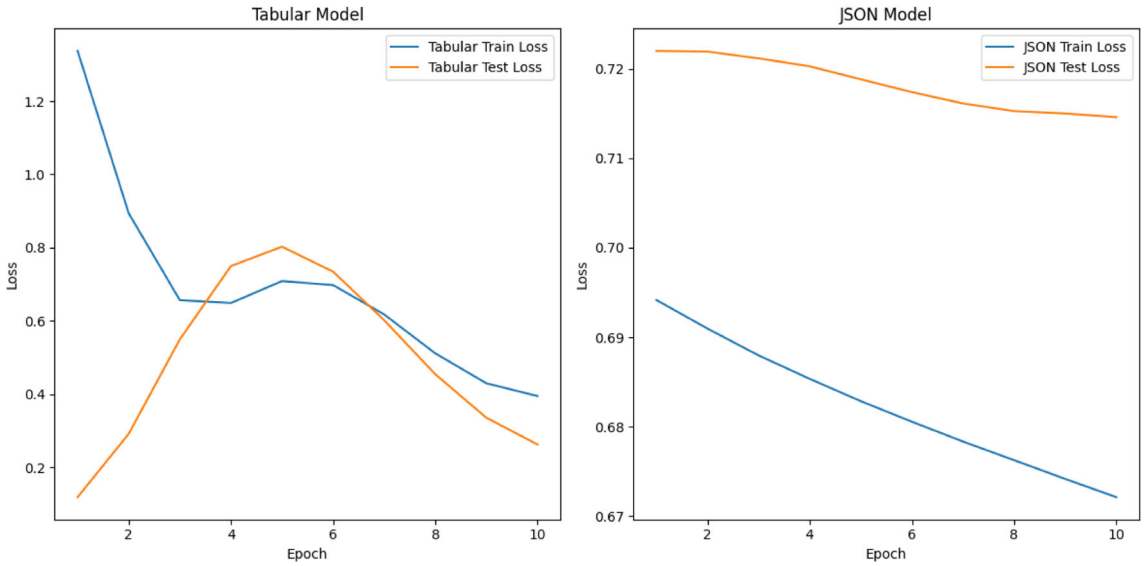

Fig. 12 NOODLE's train and test loss for tabular and graph dataset

classified by the model. The ROC-AUC curve is given in Fig. 15.

The white area represents the optimal zone for model performance, and the lightly shaded red areas represent the zones of acceptable efficacy. The values for ROC-AUC range from 0 to 1, where values near "1" suggest that it can effectively discriminate between TF and TI cases with a high degree of confidence, and if the value is near "0," the model's performance is worse than random guessing. In our case, the value is 0.928, which suggests that the model is performing well.

6.2.4 Radar Plot

The radar plot provides a visual means of presenting complex, multi-dimensional data, as shown in Fig. 16. When appraising the effectiveness of a predictor, there is a tendency to focus narrowly on a limited set of metrics. However, the radar plot provides a method for gaining a comprehensive understanding of performance across diverse dimensions. In a radar chart, each variable is represented along its corresponding axes (some variables have been normalized to conform to the 0–1 range of the radial axis). The model excels in Calibration Loss (0.98), Brier Skill Score (0.96), Calib-in-the-large (0.97), Confidence Score (0.98), Specificity (0.97), and NPV (0.94), indicating reliable probability calibration and effective identification of true negatives. These strengths suggest the model is well-suited for the given applications, where minimizing false positives is crucial. However, the model shows weaker performance in Resolution (0.42), Refinement Loss (0.53), and Sensitivity (0.46), suggesting difficulties in detecting true positives and distinguishing new patterns, which could limit its effectiveness in high-stakes fields like detecting evolving hardware Trojans. To improve, future work should focus on enhancing true positive detection through techniques like dataset rebalancing, better feature selection, or ensemble methods, while preserving the model's robust calibration.

In the given radar plot, we have metrics related to discrimination, which include AUC, resolution, and refinement loss. Following these are combined metrics assessing both calibration and discrimination, namely the Brier score and Brier skill score. As shown in the figure, the model is less sensitive and has high accuracy. This implies that while the model is generally accurate in its predictions, it may not be as effective in identifying all the actual TI cases. This could be due to a higher number of false negatives, which means the model is missing some of the positive cases.

Table 11 NOODLE's Brier score comparison for different modalities

Dataset	Brier Score
Graph-based Data	0.1798
Tabular-based Data	0.1913
Early Fusion (Graph + Tabular)	0.1685
Late Fusion (Graph + Tabular)	0.1589

7 Conclusion

In this paper, we made significant strides in the field of hardware Trojan detection by introducing uncertainty-aware unimodal and multimodal learning. The introduction of guaranteed coverage for prediction sets, facilitated by a tunable significance level through conformal prediction, enhances

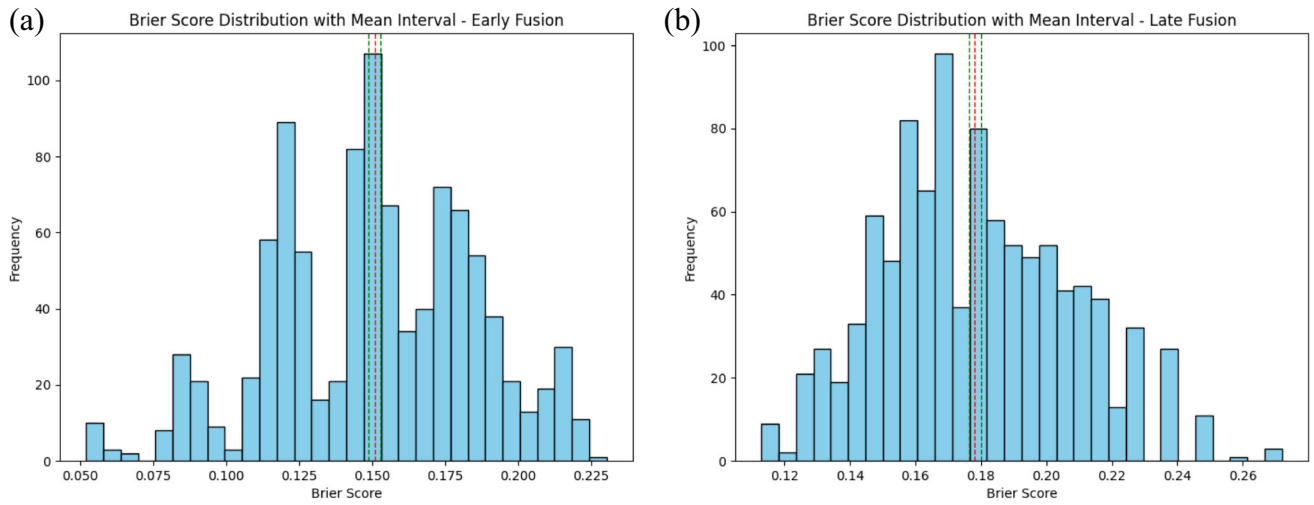

Fig. 13 NOODLE's Brier score. **a** Early fusion and **b** late fusion

the robustness of hardware Trojan detection. Moreover, our algorithm-agnostic reject prediction mechanism, coupled with explainability-aware features, provides a valuable tool for human intervention in cases of model uncertainty regarding evolving Trojan identification. The proposed ranking mechanism, validated on both synthetic and real chip-level benchmarks, contributes to the interpretability and confidence assessment of evolved Trojans. In addition, our assessment addresses both early and late fusion strategies, offering a comprehensive evaluation of the approach's efficacy and also handling missing modalities using GAN. By addressing these pivotal aspects, our work not only pushes the boundaries of hardware Trojan detection but also sets a foundation for more reliable and nuanced approaches to safeguarding hardware systems against evolving threats.

Our results highlighted opportunities for researchers in related hardware security domains such as logic locking [71–74] to rethink the application of ML-based solutions and reconstruct the metrics to evaluate their methods. We do believe

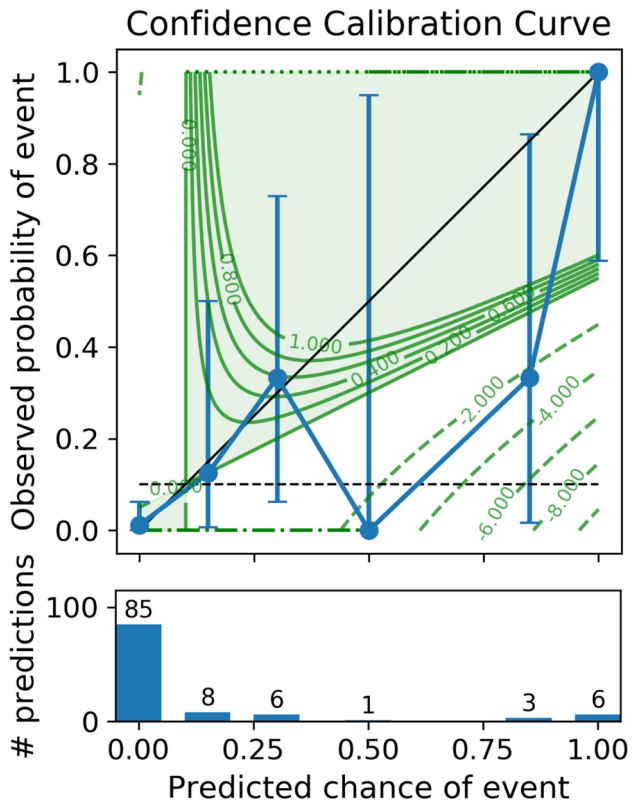

Fig. 14 NOODLE's confidence calibration curve

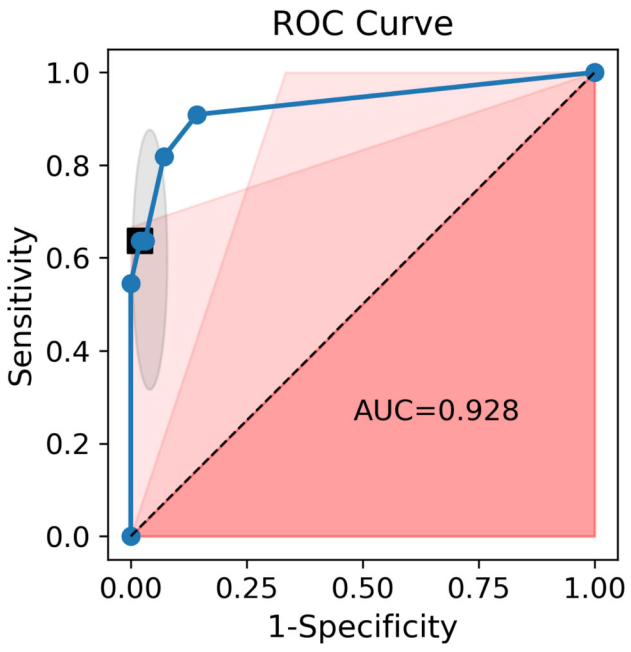

Fig. 15 NOODLE's ROC-AUC curve under late fusion

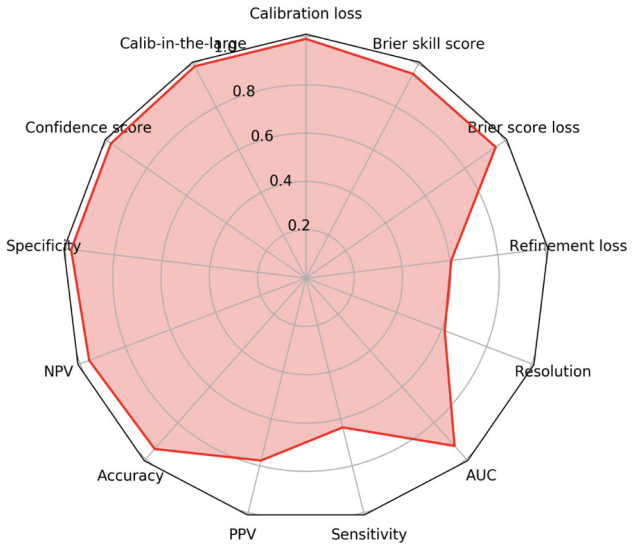

Fig. 16 NOODLE's radar plot for consolidated metrics

that there is no silver bullet for a zero-day attack, but a robust method to minimize the chances of an attack and a proactive approach to defending the attack do help.

Statements and Declarations

Funding This work is supported by the National Science Foundation under Award No. 2245247.

Competing Interests The authors declare no competing interests.

Author Contributions R. V. and A. R. contributed equally to this work.

Data Availability No datasets were generated or analyzed during the current study.

Ethical Approval This declaration is "not applicable."

References

1. Francq J, Frick F (2015) Introduction to hardware trojan detection methods. In: 2015 Design, Automation & Test in Europe Conference & Exhibition (DATE). IEEE, pp 770–775
2. Gbade-Alabi A, Keezer D, Mooney V, Poschmann AY, Stöttinger M, Divekar K (2014) A signature based architecture for trojan detection. In: Proceedings of the 9th workshop on embedded systems security, pp 1–10
3. Ceschin F (2023) Spotting the differences: quirks of machine learning (in) security. USENIX Association, Santa Clara, CA
4. Quiring E, Pendlebury F, Warnecke A, Pierazzi F, Wressnegger C, Cavallaro L, Rieck K (2022) Dos and don'ts of machine learning in computer security. In: 31st USENIX Security Symposium (USENIX Security 22), USENIX Association, Boston, MA
5. Liu W, Chang C-H, Wang X, Liu C, Fung JM, Ebrahimabadi M, Karimi N, Meng X, Basu K (2021) Two sides of the same coin: boons and banes of machine learning in hardware security. IEEE J Emerg Sel Top Circuits Syst 11(2):228–251
6. Shafer G, Vovk V (2008) A tutorial on conformal prediction. J Mach Learn Res 9(3)
7. Tibshirani RJ, Foygel Barber R, Candes E, Ramdas A (2019) Conformal prediction under covariate shift. Adv Neural Inf Process Syst 32
8. Krieg C (2023) Reflections on trusting trusthub. In: 2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD). IEEE, pp 1–9
9. White A (2023) By 2024, 60% of the data used for the development of AI and analytics projects will be synthetically generated. https://blogs.gartner.com/andrew_white/2021/07/24/
10. Nassif J, Tekli J, Kamradt M (2024) Synthetic data: revolutionizing the industrial metaverse. Springer, ???
11. Yu S-Y, Yasaei R, Zhou Q, Nguyen T, Faruque MAA (2021) Hw2vec: a graph learning tool for automating hardware security. arXiv preprint arXiv:2107.12328
12. Salmani H, Tehranipoor M, Sutikno S, Wijitrisnanto F (2023) Trust-Hub Trojan benchmark for hardware Trojan detection model creation using machine learning. https://dx.doi.org/10.21227/px6s-sm21
13. Huang Z, Wang Q, Chen Y, Jiang X (2020) A survey on machine learning against hardware trojan attacks: recent advances and challenges. IEEE Access 8:10796–10826
14. Gubbi KI, Latibari BS, Srikanth A, Sheaves T, Beheshti-Shirazi SA, Pd SM, Rafatirad S, Sasan A, Homayoun H, Salehi S (2023) Hardware Trojan detection using machine learning: a tutorial. ACM Trans Embed Comput Syst
15. Köylü TÇ, Reinbrecht CRW, Gebregiorgis A, Hamdioui S, Taouil M (2023) A survey on machine learning in hardware security. ACM J Emerg Technol Comput Syst
16. Kundu S, Meng X, Basu K (2021) Application of machine learning in hardware Trojan detection. In: 2021 22nd International Symposium on Quality Electronic Design (ISQED). IEEE, pp 414–419
17. Botero UJ, Wilson R, Lu H, Rahman MT, Mallaiyan MA, Ganji F, Asadizanjani N, Tehranipoor MM, Woodard DL, Forte D (2021) Hardware trust and assurance through reverse engineering: a tutorial and outlook from image analysis and machine learning perspectives. ACM J Emerg Technol Comput Syst (JETC) 17(4):1–53
18. Ashok M, Turner MJ, Walsworth RL, Levine EV, Chandrakasan AP (2022) Hardware trojan detection using unsupervised deep learning on quantum diamond microscope magnetic field images. ACM J Emerg Technol Comput Syst (JETC) 18(4):1–25
19. Bowman DC, Emmert JM (2022) Hardware Trojan detection through multimodal image processing and analysis. In: 2022 IEEE International Symposium on Smart Electronic Systems (iSES), pp 712–717
20. Bao C, Forte D, Srivastava A (2014) On application of one-class SVM to reverse engineering-based hardware Trojan detection. In: Fifteenth international symposium on quality electronic design. IEEE, pp 47–54
21. Hasegawa K, Yanagisawa M, Togawa N (2017) A hardware-Trojan classification method using machine learning at gate-level netlists based on trojan features. IEICE Trans Fundam Electron Commun Comput Sci 100(7):1427–1438
22. Dong C, Chen J, Guo W, Zou J (2019) A machine-learning-based hardware-Trojan detection approach for chips in the internet of things. Int J Distrib Sens Netw 15(12):1550147719888098
23. Hasegawa K, Yanagisawa M, Togawa N (2017) Trojan-feature extraction at gate-level netlists and its application to hardware-Trojan detection using random forest classifier. In: 2017 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, pp 1–4
24. Gohil V, Guo H, Patnaik S, Rajendran J (2022) Attrition: attacking static hardware Trojan detection techniques using reinforcement learning. In: Proceedings of the 2022 ACM SIGSAC conference on computer and communications security, pp 1275–1289

25. Chen H, Zhang X, Huang K, Koushanfar F (2023) Adatest: reinforcement learning and adaptive sampling for on-chip hardware Trojan detection. ACM Trans Embed Comput Syst 22(2):1–23
26. Alrahis L, Patnaik S, Shafique M, Sinanoglu O (2022) Embracing graph neural networks for hardware security. In: Proceedings of the 41st IEEE/ACM international conference on computer-aided design, pp 1–9
27. Hepp A, Baehr J, Sigl G (2022) Golden model-free hardware Trojan detection by classification of netlist module graphs. In: 2022 Design, Automation & Test in Europe Conference & Exhibition (DATE). IEEE, pp 1317–1322
28. Han T, Wang Y, Liu P (2019) Hardware Trojans detection at register transfer level based on machine learning. In: 2019 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, pp 1–5
29. Aghamohammadi Y, Rezaei A (2024) Lipstick: corruptibility-aware and explainable graph neural network-based oracle-less attack on logic locking. In: 29th Asia and South Pacific Design Automation Conference (ASP-DAC), pp 606–611
30. Yang L, Guo W, Hao Q, Ciptadi A, Ahmadzadeh A, Xing X, Wang G (2021) Cade: detecting and explaining concept drift samples for security applications. In: USENIX security symposium, pp 2327–2344
31. Pan Z, Mishra P (2023) Hardware Trojan detection using Shapley ensemble boosting. In: Proceedings of the 28th Asia and South Pacific design automation conference, pp 496–503
32. Downing E, Mirsky Y, Park K, Lee W (2021) Deepreflect: discovering malicious functionality through binary reconstruction. In: USENIX security symposium, pp 3469–3486
33. Severi G, Meyer J, Coull SE, Oprea A (2021) Explanation-guided backdoor poisoning attacks against malware classifiers. In: USENIX security symposium, pp 1487–1504
34. Singh P, Srivastava R, Rana K, Kumar V (2021) A multimodal hierarchical approach to speech emotion recognition from audio and text. Knowl-Based Syst 229:107316
35. Ektefaie Y, Dasoulas G, Noori A, Farhat M, Zitnik M (2023) Multimodal learning with graphs. Nat Mach Intell 5(4):340–350
36. Nyiri T, Kiss A (2023) What can we learn from small data. Infocommunications J, Special Issue on Appl Inf, 27–34
37. Xu P, Ji X, Li M, Lu W (2023) Small data machine learning in materials science. npj Comput Mater 9(1):42
38. Ghamisi A, Charter T, Ji L, Rivard M, Lund G, Najjaran H (2023) Anomaly detection in automated fibre placement: learning with data limitations. ArXiv preprint ArXiv:2307.07893
39. Ngiam J, Khosla A, Kim M, Nam J, Lee H, Ng AY (2011) Multimodal deep learning. In: Proceedings International Conference on Machine Learning (ICML), pp 689–696
40. Trong VH, Gwang-hyun Y, Vu DT, Jin-young K (2020) Late fusion of multimodal deep neural networks for weeds classification. Comput Electron Agric 175:105506
41. Nguyen TM, Nguyen T, Le TM, Tran T (2021) Gefa: early fusion approach in drug-target affinity prediction. IEEE/ACM Trans Comput Biol Bioinf 19(2):718–728
42. Ovadia Y, Fertig E, Ren J, Nado Z, Sculley D, Nowozin S, Dillon J, Lakshminarayanan B, Snoek J (2019) Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. Adv Neural Inf Process Syst 32
43. Löfström T, Boström H, Linusson H, Johansson U (2015) Bias reduction through conditional conformal prediction. Intell Data Anal 19(6):1355–1375
44. Boström H, Johansson U, Löfström T (2021) Mondrian conformal predictive distributions. In: Conformal and probabilistic prediction and applications. PMLR, pp 24–38
45. Angelopoulos AN, Bates S (2023) Conformal prediction: a gentle introduction. Found Trends® Mach Learn 16(4):494–591
46. Laland K, Uller T, Feldman M, Sterelny K, Müller GB, Moczek A, Jablonka E, Odling-Smee J, Wray GA, Hoekstra HE et al (2014) Does evolutionary theory need a rethink? Nature 514(7521):161–164
47. Liu B, Vishwakarma R (2022) Anomaly aware log retrieval from disk array enclosures (DAEs). Google Patents. US Patent 11,513,931
48. Wilde H, Knight V, Gillard J (2020) Evolutionary dataset optimisation: learning algorithm quality through evolution. Appl Intell 50:1172–1191
49. Catto E (2010) Box2d. Available from: http://www.box2d.org
50. Holland JH (1992) Genetic algorithms. Sci Am 267(1):66–73
51. Sisejkovic D, Merchant F, Reimann LM, Srivastava H, Hallawa A, Leupers R (2021) Challenging the security of logic locking schemes in the era of deep learning: a neuroevolutionary approach. ACM J Emerg Technol Comput Syst (JETC) 17(3):1–26
52. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2020) Generative adversarial networks. Commun ACM 63(11):139–144
53. Ojha U, Li Y, Lee YJ (2023) Towards universal fake image detectors that generalize across generative models. arXiv preprint arXiv:2302.10174
54. Kang M, Zhu J-Y, Zhang R, Park J, Shechtman E, Paris S, Park T (2023) Scaling up GANs for text-to-image synthesis. arXiv preprint arXiv:2303.05511
55. Xu L, Skoularidou M, Cuesta-Infante A, Veeramachaneni K (2019) Modeling tabular data using conditional GAN. Adv Neural Inf Process Syst 32
56. Ashrapov I (2020) Tabular GANs for uneven distribution. arXiv preprint arXiv:2010.00638
57. Lederrey G, Hillel T, Bierlaire M (2022) DATGAN: integrating expert knowledge into deep learning for synthetic tabular data. arXiv preprint arXiv:2203.03489
58. Zhao Z, Wu H, Van Moorsel A, Chen LY (2023) GTV: generating tabular data via vertical federated learning. arXiv preprint arXiv:2302.01706
59. Yonetani R, Takahashi T, Hashimoto A, Ushiku Y (2019) Decentralized learning of generative adversarial networks from non-iid data. arXiv preprint arXiv:1905.09684
60. Etemadi N (1981) An elementary proof of the strong law of large numbers. Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete 55(1):119–122
61. Smythe RT (1973) Strong laws of large numbers for r-dimensional arrays of random variables. Ann Probab, 164–170
62. Vashistha N, Lu H, Shi Q, Rahman MT, Shen H, Woodard DL, Asadizanjani N, Tehranipoor M (2018) Trojan scanner: detecting hardware trojans with rapid SEM imaging combined with image processing and machine learning. In: ISTFA 2018: Proceedings from the 44th international symposium for testing and failure analysis. ASM International, p 256
63. Shi Q, Vashistha N, Lu H, Shen H, Tehranipoor B, Woodard DL, Asadizanjani N (2019) Golden gates: a new hybrid approach for rapid hardware trojan detection using testing and imaging. In: 2019 IEEE international symposium on Hardware Oriented Security and Trust (HOST). IEEE, pp 61–71
64. Liakos KG, Georgakilas GK, Plessas FC, Kitsos P (2022) Gainesis: generative artificial intelligence netlists synthesis. Electronics 11(2):245
65. Sankaranarayanan S, Angelopoulos AN, Bates S, Romano Y, Isola P (2022) Semantic uncertainty intervals for disentangled latent spaces
66. Jaques N, Taylor S, Sano A, Picard R (2017) Multimodal autoencoder: a deep learning approach to filling in missing sensor data and enabling better mood prediction. In: IEEE International Conference on Affective Computing and Intelligent Interaction (ACII), pp 202–208

67. Creswell A, White T, Dumoulin V, Arulkumaran K, Sengupta B, Bharath AA (2018) Generative adversarial networks: an overview. IEEE Signal Process Mag 35(1):53–65
68. Bates S, Angelopoulos A, Lei L, Malik J, Jordan M (2021) Distribution-free, risk-controlling prediction sets. J ACM (JACM) 68(6):1–34
69. Dieber J, Kirrane S (2020) Why model why? Assessing the strengths and limitations of lime. arXiv preprint arXiv:2012.00093
70. Gallo I, Calefati A, Nawaz S (2017) Multimodal classification fusion in real-world scenarios. In: IAPR International Conference on Document Analysis and Recognition (ICDAR), vol 5, pp 36–41
71. Rezaei A, Afsharmazayejani R, Maynard J (2022) Evaluating the security of eFPGA-based redaction algorithms. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD)
72. Rezaei A, Hedayatipour A, Sayadi H, Aliasgari M, Zhou H (2022) Global attack and remedy on IC-specific logic encryption. In: IEEE international symposium on Hardware Oriented Security and Trust (HOST), pp 145–148
73. Maynard J, Rezaei A (2023) DK lock: dual key logic locking against oracle-guided attacks. In: International Symposium on Quality Electronic Design (ISQED), pp 1–7
74. Aghamohammadi Y, Rezaei A (2023) Cola: convolutional neural network model for secure low overhead logic locking assignment. In: Proceedings of the Great Lakes Symposium on VLSI (GLSVLSI), pp 339–344
75. Vishwakarma R, Rezaei A (2023) Risk-aware and explainable framework for ensuring guaranteed coverage in evolving hardware trojan detection. In: 2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD), pp 1–9
76. Vishwakarma R, Rezaei A (2024) Uncertainty-aware hardware trojan detection using multimodal deep learning. In: 2024 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp 1–6