Hardware Trojan Detection on Gate Level Netlist*

*2025 CAD Contest Problem A

1st Yan-Cheng Li
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan
likevin1022@gmail.com

***Abstract*—Recent advancements in Machine Learning (ML) have shown significant potential for Hardware Trojan (HT) detection. By extracting features from gate-level netlists, ML approaches can eliminate the dependency on a golden chip, promising high accuracy and reliability. In this paper, we propose a novel graph neural network framework to enhance HT detection. Our method first represents the netlist as a graph and utilizes a Bidirectional Graph Convolutional Network (BiGCN) to capture dependencies from both data-flow and fan-out perspectives. To overcome the class imbalance problem inherent in HT detection, we develop a specialized Threshold-Aware Focal Loss function. Finally, we introduce a post-processing step that filters out detections below a minimum gate count, significantly reducing the false positive rate. The proposed system is evaluated on a private competition dataset and shows marked improvements over ablated versions of the model in key metrics such as F1-score and precision.**

***Index Terms*—Hardware Security, Machine Learning, GNN, EDA, Hardware Trojan**

I. Introduction

The modern semiconductor industry heavily relies on a globalized supply chain, where the design and fabrication of Integrated Circuits (ICs) are often distributed across multiple entities. This paradigm, characterized by the extensive use of third-party Intellectual Property (IP) cores in System-on-Chip (SoC) designs, has enabled unprecedented innovation and reduced time-to-market. However, it has also introduced significant security vulnerabilities. Among the most pernicious threats is the insertion of malicious circuitry, commonly known as Hardware Trojans (HTs), by untrusted parties in the design and fabrication flow. An HT can remain dormant during testing phases, only to be activated under specific conditions post-deployment to leak sensitive information, degrade performance, or cause a complete denial-of-service, thereby jeopardizing the security and reliability of critical systems.

Detecting these stealthy modifications before chip fabrication is of paramount importance. The gate-level netlist, a detailed structural representation of the circuit, serves as a critical stage for pre-silicon security verification. Identifying HTs at this stage can prevent the astronomical costs associated with fabricating compromised hardware.

A. Problem Statement

Despite its importance, detecting HTs at the gate-level remains a formidable challenge. Traditional validation methods, such as logic testing and formal verification, often struggle to achieve sufficient coverage to uncover intentionally hidden Trojans. Post-silicon techniques that rely on side-channel analysis require a trusted ”golden chip” for comparison, which is often unavailable in a zero-trust supply chain model. Moreover, these physical measurements are susceptible to process variations and measurement noise.

In response to these limitations, Machine Learning (ML), particularly Graph Neural Networks (GNNs), has emerged as a promising direction. By treating the netlist as a graph, these methods can learn to identify anomalous patterns without a golden reference. However, existing ML-based approaches face two primary hurdles:

1) **Severe Class Imbalance:** Trojan gates constitute a minuscule fraction of the total gates, causing models to develop a trivial bias towards the majority (benign) class.
2) **Subtle Structural Signatures:** HTs are designed to be stealthy. Capturing the nuanced structural and contextual relationships that distinguish them from legitimate circuits requires highly expressive models.

B. Our Contributions

To overcome these limitations, this paper proposes a novel framework for gate-level hardware Trojan detection built upon a Bidirectional Graph Convolutional Network (BiGCN). Our primary contributions are threefold:

- **A Bidirectional GCN Architecture:** We propose a BiGCN model that processes the netlist graph in both the forward (data-flow) and backward (fan-out) directions, enabling the learning of richer and more comprehensive node embeddings.
- **A Threshold-Aware Focal Loss Function:** We introduce a custom loss function that dynamically adjusts its focus during training, prioritizing hard-to-classify examples near the decision boundary to combat class imbalance.
- **A Post-Processing Filter for Precision Enhancement:** We implement a simple yet effective filter based on a minimum Trojan gate count, which significantly reduces the false positive rate by eliminating spurious, isolated predictions.

C. Paper Organization

The remainder of this paper is organized as follows. Section II reviews related work. Section III details our proposed methodology. Section IV describes the experimental setup. Section V presents and analyzes the results. Finally, Section VI concludes the paper.

II. Background and Related Work

This section provides a foundational understanding of hardware Trojans and reviews the evolution of detection methodologies, culminating in the state-of-the-art that motivates our work.

A. Preliminaries on Hardware Trojans

A Hardware Trojan (HT) is a malicious modification to an IC design, split into a **trigger** and a **payload**. The trigger activates the Trojan, which then executes its payload, ranging from leaking data to causing system failure [1]. As noted by Basak et al. [2], gate-level netlists are a prime target for HT insertion and detection.

B. Conventional and ML-based Detection

Conventional detection methods include logic-based testing, which struggles with coverage, and side-channel analysis, which requires a trusted "golden chip". To overcome these limitations, a paradigm shift towards Machine Learning (ML) has occurred [3]. Early ML methods relied on handcrafted features [4], but their performance was limited.

The advent of Graph Neural Networks (GNNs) represented a significant leap, as they automatically learn features from the graph structure of the netlist [5]. More advanced architectures like Graph Attention Networks (GATs) have also been explored [6]. The most relevant work to our own introduced bidirectional GNNs [7], acknowledging that a gate's context is defined by both its inputs and outputs. However, this prior work did not fully address the critical challenges of class imbalance and false positive reduction, which forms the research gap our work aims to fill.

III. Proposed Methodology

Our framework is designed to overcome the key challenges of HT detection by integrating a bidirectional graph representation, a specialized GNN architecture, a custom loss function, and a targeted post-processing step.

A. Model Selection

In this work, we explored multiple graph-based neural architectures for hardware Trojan detection, including Graph Convolutional Networks (GCN), GraphSAGE, and a hybrid bidirectional model that combines the advantages of both.

GCN: The Graph Convolutional Network serves as a foundational model that aggregates information from neighboring nodes through layer-wise propagation. It effectively captures local structural patterns in the circuit graph, making it suitable for identifying Trojans that are strongly connected to their surrounding gates. However, the receptive field of a standard GCN is limited by the number of layers, which restricts its ability to capture long-range dependencies across complex signal paths.

GraphSAGE: GraphSAGE improves upon GCN by introducing a learnable aggregation mechanism that can sample and combine information from a fixed number of neighbors. This approach enhances scalability and allows the model to generalize better to unseen subgraphs. Moreover, GraphSAGE is capable of capturing diverse neighborhood information through various aggregation functions such as mean, LSTM, or pooling. To further improve its ability to model signal interactions in hardware circuits, a *bidirectional* variant (BiGraphSAGE) can be employed, allowing information to propagate in both forward and backward directions. This extension enables the model to reason about feedback and control-flow dependencies more effectively, as shown in Figure 1.

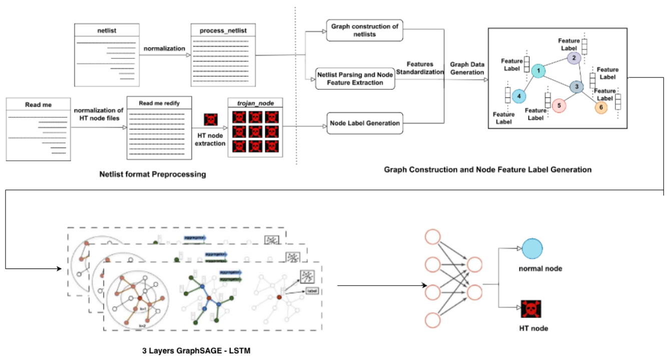
netlist
normalization
process_netlist
Read me
normalization of HT node files
Read me redify
HT node extraction
trojan_node
Netlist format Preprocessing
Graph construction of netlists
Netlist Parsing and Node Feature Extraction
Node Label Generation
Features Standardization
Graph Data Generation
Feature Label
1
2
3
4
5
6
Graph Construction and Node Feature Label Generation
3 Layers GraphSAGE - LSTM
normal node
HT node

Fig. 1. High-level overview of the GraphSAGE HT detection framework.

Hybrid (Bidirectional) Model: To overcome the limitations of both GCN and GraphSAGE, we designed a hybrid bidirectional architecture that propagates information in both forward and backward directions along the signal flow. This bidirectional mechanism enables the model to capture both data and control dependencies between gates, which is particularly beneficial for detecting stealthy Trojans that influence distant or feedback-connected nodes. We also incorporated a custom Threshold-Aware Focal Loss to handle severe class imbalance between Trojan and non-Trojan gates.

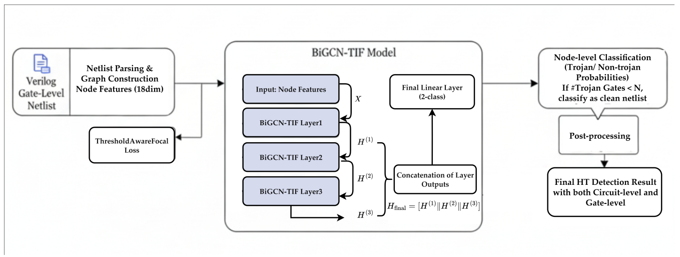
Verilog Gate-Level Netlist
Netlist Parsing & Graph Construction Node Features (18dim)
ThresholdAwareFocal Loss
BiGCN-TIF Model
Input: Node Features
BiGCN-TIF Layer1
BiGCN-TIF Layer2
BiGCN-TIF Layer3
X
$H^{(1)}$
$H^{(2)}$
$H^{(3)}$
Final Linear Layer (2-class)
Concatenation of Layer Outputs
$H_{\text{final}} = [H^{(1)} \| H^{(2)} \| H^{(3)}]$
Node-level Classification (Trojan/ Non-trojan Probabilities) If #Trojan Gates < N, classify as clean netlist
Post-processing
Final HT Detection Result with both Circuit-level and Gate-level

Fig. 2. High-level overview of the proposed HT detection framework.

B. Graph Representation and Architecture

We represent the netlist as a graph $G = (V, E)$ where nodes $v \in V$ are gates. For each node, we extract an 18-dimensional feature vector x_v. We define two edge sets: forward edges E_{fw} representing signal flow, and backward edges E_{bw} representing the reverse connections.

Our model, `BiGCN-TIF`, stacks three bidirectional GCN layers. Each layer processes the forward and backward graphs in parallel and fuses the results. A key feature is the dense concatenation of outputs from all intermediate layers, $H_{final} = [H^{(1)} \| H^{(2)} \| H^{(3)}]$, which allows the final classifier to access features from multiple abstraction levels.

C. Training and Post-Processing

To address class imbalance, we propose a **Threshold-Aware Focal Loss**, which builds upon Focal Loss [8] by adding a dynamic weight that prioritizes hard-to-classify examples near the decision threshold. To improve precision, we apply a post-processing filter: if the count of predicted Trojan gates in a circuit is less than a hyperparameter $N_{min} = 20$, the entire circuit is reclassified as benign.

IV. Experimental Setup

Our experimental methodology was specifically tailored to the format of the hardware security competition. The primary strategy involved creating a custom, high-fidelity training dataset and conducting a data-driven feature engineering process, which we believe are the cornerstones of our success. This section details our dataset generation strategy, the rationale behind our feature selection, evaluation metrics, baseline models, and implementation details.

A. Training Dataset Generation and Strategy

The competition rules provided access to the ground truth for the 30 evaluation circuits. We leveraged this information to construct a specialized training set through a process of Trojan extraction and data augmentation.

1) Positive Sample Generation (Trojan Graphs): We extracted the 20 Trojan sub-circuits and performed data augmentation by remapping them with **Synopsys Design Compiler** using multiple standard cell libraries. This resulted in approx. **360 unique Trojan graph samples**, forcing the model to learn the fundamental logic of the Trojans rather than superficial gate patterns.

2) Negative Sample Generation (Benign Graphs): We extracted a representative set of non-Trojan sub-circuits from the 10 provided clean netlists to serve as negative examples for the classifier.

3) Test Set: The final evaluation was performed on the **original, full 30 competition circuits**.

B. Data-Driven Feature Engineering

In hardware Trojan detection, the quality of the dataset and the relevance of the chosen features are often more critical than the model architecture itself. A model, no matter how complex, cannot learn from uninformative data. Therefore, our primary focus was on a meticulous analysis of the competition dataset and a principled feature selection process. Our final 18-dimensional feature vector is composed of three distinct categories, each justified by our data analysis.

1) Category 1: Compositional Features (9 features): This category describes the fundamental building block of each node: its gate type. Our analysis of the ground truth revealed that Trojans are not built uniformly. The pie chart in Figure 3 shows a clear distributional bias, with `NAND` (39.6%), `XNOR` (20.8%), and `NOR` (16.6%) gates being particularly prevalent across all Trojans. This suggests that the gate type itself is a strong predictive signal.

Furthermore, the heatmap in Figure 4 illustrates that each Trojan possesses a unique "gate signature." For example, the massive Trojan in `design14` is exceptionally rich in sequential elements (`dff`), a characteristic not shared by most other Trojans. To capture these compositional nuances, we selected a 9-dimensional one-hot encoded vector representing the most common gate types:

- `'and', 'or', 'nand', 'nor', 'not', 'buf', 'xor', 'xnor', 'dff'`

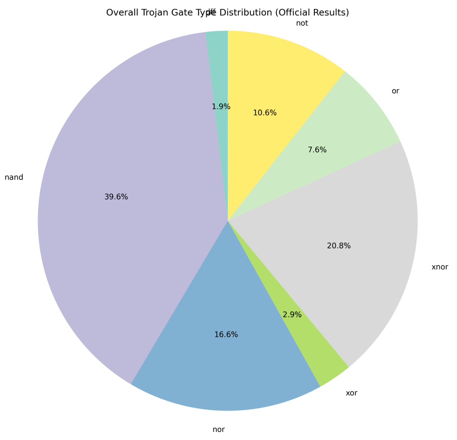

Fig. 3. Non-uniform overall distribution of gate types in Trojans, justifying the use of gate-type features.

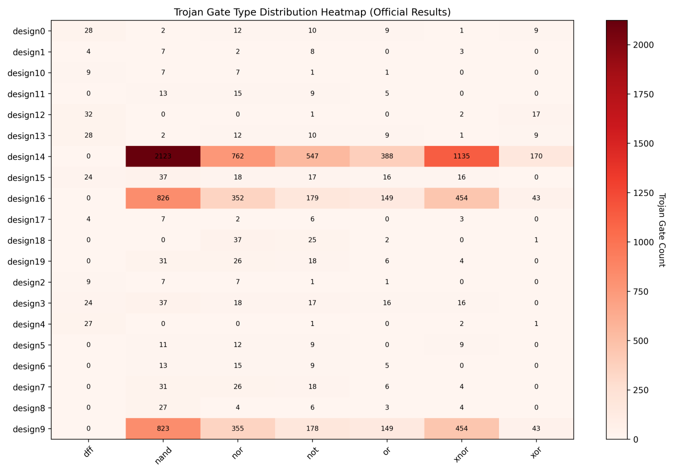

Fig. 4. Heatmap illustrating the unique compositional fingerprint of each Trojan through its gate type distribution.

2) Category 2: Sequential Context Features (5 features): The heatmap analysis (Fig. 4) also highlights that sequential elements like flip-flops (`dff`) are critical components in several large Trojans. Malicious logic often involves hijacking or manipulating the circuit's state, which is controlled by these elements. Simply knowing a gate is a `dff` is insufficient; understanding its role is key. Therefore, we include five binary features to provide a fine-grained context of its pin-level connectivity:

- `'is_ck'`, `'is_d'`, `'is_q'`, `'is_rst'`, `'is_set'`: These flags indicate if a gate is connected to a clock, data, output, reset, or set pin of a sequential element, respectively. They allow the model to better understand the control and state flow of the circuit, which are prime targets for Trojan triggers.

3) Category 3: Topological Anomaly Features (4 features): While compositional features describe *what* a gate is, topological features describe *how* it is connected, which is essential for identifying stealthy, structurally anomalous modifications. The bar chart in Figure 5 shows that Trojan sizes vary dramatically from tens to thousands of gates, implying a wide range of structural complexity. To ensure our model can find Trojans regardless of their size, we incorporate four features that capture well-known structural indicators from hardware security literature:

- **`fan_out`:** This feature counts how many other gates a single gate's output drives. An unusually high fan-out is a classic indicator of a potential trigger distribution network, where a single rare signal activates multiple payload components.
- **`is_reconvergent`:** A binary flag indicating if a gate is part of a reconvergent fan-out structure. Such structures are often exploited by attackers to create complex trigger conditions that are extremely difficult to activate through random testing, making them a suspicious pattern.
- **`in_isolated_subgraph`:** This feature flags gates that belong to small, disconnected clusters of logic. Such isolated subgraphs are highly anomalous in a well-designed circuit and are often indicative of a payload circuit waiting for a trigger signal.
- **`has_tied_inputs`:** A binary flag for gates whose inputs are tied to the same signal or a constant value (VCC/GND). This is an unusual design practice sometimes used by attackers as a low-cost way to implement rare logic conditions for a Trojan trigger.

By systematically combining these three data-driven feature categories, we construct a comprehensive 18-dimensional feature vector that equips our GNN model with the necessary information to effectively and robustly identify malicious circuitry.

C. Evaluation Metrics and Baselines

Evaluation is performed at the circuit-level using Accuracy, Precision, Recall, and F1-Score. To validate our design, we compare our final model against ablated baselines: a **Uni-GCN**, a **BiGCN w/o TIF**, and our model trained with a standard **Cross-Entropy (CE) Loss**.

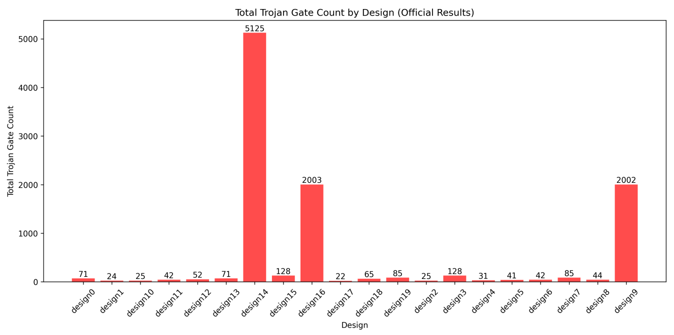

Fig. 5. Vast difference in Trojan size across the 20 designs, motivating the need for scale-invariant topological features.

D. Implementation Details

Our framework was implemented using PyTorch/PyTorch Geometric with an Adam optimizer (LR 10^{-3}), batch size of 16, and our custom loss. Inference uses a probability threshold of $\tau = 0.7$ and a post-processing filter of $N_{min} = 20$.

V. Results and Analysis

This section presents the performance of our framework on the official competition dataset, followed by an ablation study and an analysis of detection consistency.

A. Official Competition Performance

Our model performed exceptionally well on the 30 blind test circuits. As the confusion matrix in Figure 6 shows, we successfully identified **all 20 Trojaned circuits** (100% Recall) while only misclassifying 3 of the 10 clean circuits. This yields a final F1-Score of 93.0% (Table I), validating our approach.

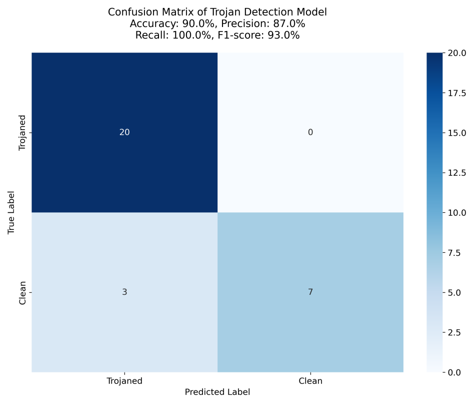

Fig. 6. Confusion matrix of our final model's performance.

B. Ablation Study: Validating Architectural Choices

As visualized in Figure 7, our ablation study confirms the contribution of each design choice. Each enhancement progressively increases the F1-Score while dramatically reducing false positives (FPs), from 10 FPs in the baseline GCN to only 3 in our final model.

TABLE I
FINAL CIRCUIT-LEVEL PERFORMANCE METRICS

Metric	Score
Accuracy	90.0%
Precision	87.0%
Recall (TPR)	100.0%
F1-Score	**93.0%**

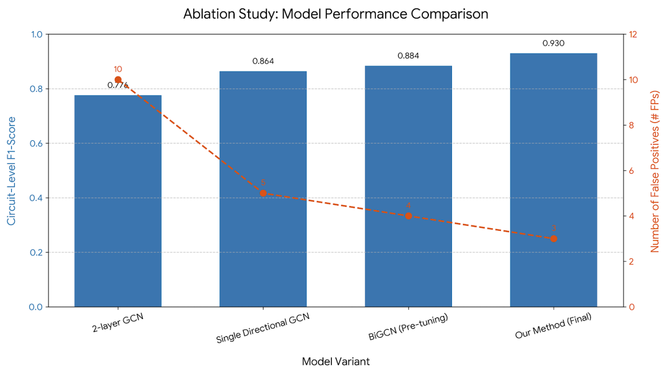

Fig. 7. Ablation study results, showing steady performance improvement.

TABLE II
ABLATION STUDY: CIRCUIT-LEVEL PERFORMANCE OF MODEL VARIANTS

Model Variant	Accuracy	F1-Score	# of FPs
Our Method (Final)	**90.0%**	**93.0%**	**3**
BiGCN (Pre-tuning)	83.3%	88.4%	4
Single Directional GCN	80.0%	86.4%	5
2-layer GCN (Baseline)	63.3%	77.6%	10

C. Analysis of Gate-Level Detection Consistency

Beyond circuit-level accuracy, we analyzed the consistency of gate-level Trojan localization. The box plot in Figure 8 shows a high median gate-level F1-score of 0.688 across the 20 Trojan cases, indicating reliable performance.

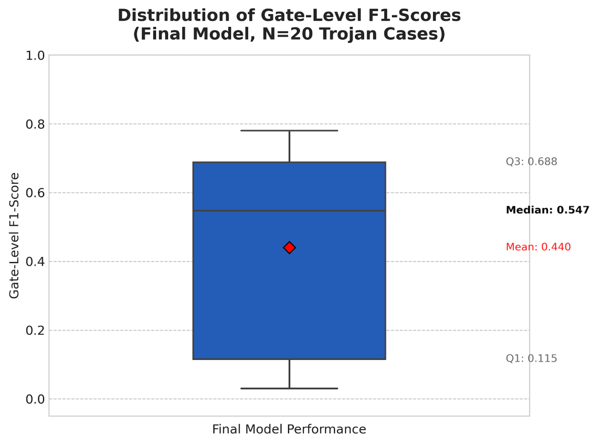

Fig. 8. Distribution of gate-level F1-scores for the 20 Trojaned circuits.

The per-case analysis in Figure 9 details the significant gains our final model achieved over its pre-tuning variant, especially on challenging cases like T2, T4, and T9.

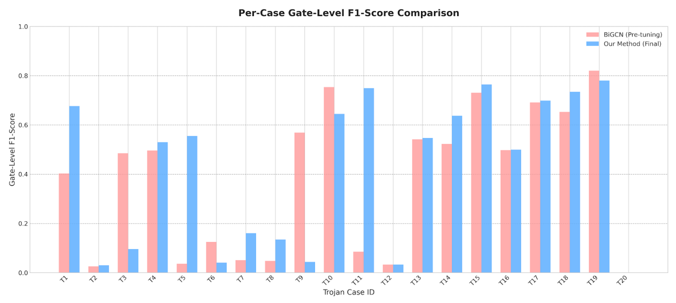

Fig. 9. A per-case comparison of gate-level F1-scores between our final model and its pre-tuning variant.

VI. CONCLUSION

In this paper, we proposed and validated a novel framework for gate-level hardware Trojan detection, tailored for a competitive challenge. By combining a specialized data augmentation strategy using commercial EDA tools with a sophisticated BiGCN-TIF architecture, we developed a highly effective detection model. Our approach, further enhanced by a custom Threshold-Aware Focal Loss and a precision-focused post-processing filter, achieved a perfect recall of 100% and a final F1-Score of 93.0% on the blind competition test set. The ablation studies scientifically confirmed the contribution of each component to the final performance. Future work could involve applying this methodology to a wider range of public benchmarks and exploring architectural variations to further improve gate-level localization precision.

ACKNOWLEDGMENT

The author would like to thank the organizers of the 2025 CAD Contest for providing the challenging problem and dataset.

REFERENCES

[1] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, “Hardware trojan threats: a survey,” in *2010 IEEE international conference on computer design (ICCD)*. IEEE, 2010, pp. 7–1.
[2] A. Basak, M. S. W. D’Silva, and S. Bhunia, “Hardware trojans classification for gate-level netlists using multi-layer neural networks,” in *2017 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2017, pp. 1–6.
[3] J. Koehler, F. Schellenberg, S.-T. S. Hamdi, and M. A. A. Sanad, “Hardware trojan detection using machine learning: A tutorial,” vol. 28, no. 5, 2023, pp. 1–32.
[4] Y. Hao, J. Zhang, and J. Li, “A hardware trojan detection method based on structural features of trojan and host circuits,” in *Journal of Physics: Conference Series*, vol. 1651, no. 1, 2020, p. 012117.
[5] G. Wecel, M. A. A. Sanad, and F. Schellenberg, “Hardware trojan detection using graph neural networks,” *arXiv preprint arXiv:2204.11431*, 2022.
[6] C. Li, Y. Chen, and X. Li, “Gatrojan: An efficient gate-level hardware trojan detection approach using graph attention networks,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2024.
[7] R. Al-Tawy, A.-H. A. E. Mohamed, and H. M. K. Al-Hassani, “A fine-grained detection method for gate-level hardware trojan based on bidirectional graph neural networks,” *Journal of King Saud University-Computer and Information Sciences*, vol. 35, no. 7, pp. 1–13, 2023.

[8] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2980–2988.