Balancing Intelligence and Efficiency: Lightweight AI for Intrusion Detection in IIoT

Ahmad Mostafavi*, Hossein Fotouhi†, Maryam Vahabi†, Johan Åkerberg†
*Atlas Copco, Stockholm, Sweden
ahmad.mostafavi@atlascopco.com
†School of Innovation, Design and Engineering, Mälardalen University (MDU), Västerås, Sweden
{hossein.fotouhi, maryam.vahabi, johan.akerberg}@mdu.se

***Abstract*—The proliferation of resource-constrained Internet-of-Things (IoT) and Industrial IoT (IIoT) devices has intensified the need for local, intelligent cybersecurity solutions that do not rely on cloud connectivity. This paper presents a TinyML-based Intrusion Detection System (IDS) framework designed for embedded edge controllers. Two machine-learning IDS models — a Convolutional Neural Network (CNN) and a Random Forest (RF) — were developed and evaluated on the TON_IoT dataset to identify malicious network traffic. Host-side evaluation established baseline performance, while the float-precision RF model was implemented as firmware and validated using hardware-in-the-loop emulation in the Renode RISC-V environment. The resulting binary occupies only 1.6 MB of Flash memory, achieving above 99% detection accuracy with 448 µs average inference latency. These results demonstrate that fully embedded, real-time intrusion detection is feasible within TinyML memory budgets, supporting IEC-62443 compliant anomaly monitoring for industrial devices.**

***Index Terms*—Intrusion Detection Systems (IDS), Artificial Intelligence (AI), Industrial Internet of Things (IIoT), Random Forest (RF), Convolutional Neural Network (CNN), RISC-V**

I. Introduction

The Industrial Internet of Things (IIoT) is transforming industrial operations by embedding sensors, actuators, and smart controllers into critical systems, enabling real-time monitoring, predictive maintenance, and autonomous decision-making [1]. These capabilities promise significant gains in productivity, scalability, and efficiency across sectors such as manufacturing, energy, and transportation. However, the growing reliance on interconnected operational technology and information technology infrastructure also exposes IIoT systems to an expanding cyber threat landscape [2].

Recent research on lightweight Machine Learning (ML) and TinyML-based IDS models has shown promising detection accuracy but often ignores the hardware limitations of embedded platforms. Most approaches evaluate models in simulation only, without verifying execution efficiency on realistic processors. This gap limits their adoption in operational IIoT environments that demand both high detection accuracy and sub-millisecond inference latency.

To address this limitation, this paper develops and evaluates two lightweight IDS implementations based on CNN and RF models. Both models are trained on the TON_IoT dataset [3] and evaluated under binary and multiclass intrusion detection tasks. Their performance is compared in terms of accuracy, model footprint, and inference latency to characterize the trade-off between detection capability and computational efficiency. The float-precision RF model is further implemented as firmware and validated using a Renode-based RISC-V hardware-in-the-loop (HIL) emulation, demonstrating real-time inference capability with a total binary size of 1.6 MB and negligible accuracy degradation. The main contributions of this paper are as follows:

1) Design and evaluation of lightweight CNN- and RF-based intrusion detection models suitable for embedded edge hardware.
2) Hardware-in-the-loop deployment and timing analysis on a simulated 32-bit RISC-V processor using the Renode framework to assess real-time inference capability.
3) Comprehensive empirical evaluation of the trade-off between detection accuracy and computational efficiency for the selected models under resource constraints representative of TinyML devices.

II. IDS Deployment in Critical Infrastructure

The increasing integration of connected devices in industrial operations has intensified the need for secure and efficient IIoT infrastructures. Prior research has explored diverse intrusion detection strategies tailored for resource-constrained environments [4]–[7]. In industrial contexts, IDS solutions must simultaneously ensure high detection accuracy and satisfy stringent latency and memory constraints [6].

Aftab et al. [4] provided a comprehensive overview of ML–based IDS solutions for IIoT, emphasizing the trade-off between computational efficiency and detection performance. Similarly, Andhare and Kumbhar [5] identified lightweight architectures and real-time adaptability as recurring design requirements, noting the scarcity of standardized benchmarking datasets and the lack of validation on embedded platforms. Elrawy et al. [7] further classified IDS mechanisms into signature-based, anomaly-based, and hybrid approaches, while analyzing centralized versus distributed deployment models applicable to IIoT networks. Their taxonomy offers a foundation for assessing IDS design choices in varying industrial topologies.

Recent work by Ericson et al. [1] demonstrated that deep-learning models can be optimized for edge inference through architectural simplification and hardware-aware adaptation,

highlighting the feasibility of efficient embedded inference without relying on aggressive quantization or heavy pruning. Collectively, these studies reveal a clear research trend toward resource-efficient, edge-deployable, and intelligent IDS frameworks. However, few efforts quantitatively evaluate execution latency, memory footprint, and total computational overhead on modern embedded architectures such as RISC-V, which are gaining traction for open and secure IIoT deployments [8].

III. Methodological Setup for Evaluation

This section outlines the model architectures, dataset, optimization techniques, and deployment setup used to evaluate a lightweight IDS on embedded IIoT hardware.

A. ML model selection

Numerous ML algorithms have been investigated for intrusion detection. Simpler models, such as k-Nearest Neighbors (k-NN), provide low computational overhead but typically yield lower detection accuracy. Conversely, more sophisticated models, such as Long Short-Term Memory (LSTM) networks, offer stronger detection capability at the expense of substantial computational complexity, which limits their suitability for embedded deployment [9]. In contrast, RFs and CNNs achieve a favorable balance between detection accuracy and computational efficiency [10], [11], making them well-suited for IIoT applications. These two model families were therefore selected as representative candidates for evaluating lightweight intrusion detection on resource-constrained embedded hardware, where both inference speed and accuracy are critical.

B. Dataset selection

Dataset selection plays a crucial role in evaluating IDS systems, especially when targeting IIoT environments and lightweight AI models. Traditional benchmark datasets such as CICIDS2017 [12] and UNSW-NB15 [13] have been widely used in IDS research; however, they primarily represent conventional enterprise network traffic and lack the heterogeneity and constrained-device behavior characteristic of IoT networks. After a comprehensive review of available benchmarks, the TON_IoT dataset [3], developed by UNSW, was selected as the most suitable dataset for this study. TON_IoT provides a rich collection of telemetry, network flow, and operating system log data obtained from real IoT devices and industrial sensors, enabling realistic evaluation of embedded IDS models under the resource and latency constraints typical of IIoT environments [10], [14]–[17].

C. Feature Engineering and Optimization

The Boruta algorithm [18], [19], applied as a wrapper around RF model, automatically identified the most informative features, while domain knowledge was used to retain attributes known to influence intrusion detection, such as port categories and packet rates.

TABLE I
Classification performance of two ML models (host-side).

Model	Binary/Multiclass	Accuracy	Precision	Recall	F1
RF	Binary	0.9988	0.9988	0.9988	0.9988
RF	Multiclass	0.9882	0.9886	0.9882	0.9883
CNN	Binary	0.9952	0.9952	0.9952	0.9952
CNN	Multiclass	0.8969	0.8986	0.8969	0.8971

TABLE II
Computational efficiency of two ML models (host-side).

Model	Binary/Multiclass	Train Time (s)	Inf. Avg. (ms)	Size (KB)
RF	Binary	9.04	0.0042	6204
RF	Multiclass	9.19	0.0063	5650
CNN	Binary	165.08	0.0150	3211
CNN	Multiclass	212.10	0.146	3223

IV. Model Evaluation

The CNN employed in this study was implemented using TensorFlow [20]. The model processes one-dimensional feature sequences through two convolutional layers with 64 and 128 filters of size 3, each followed by ReLU activation function. A max-pooling layer (pool size 2) reduces spatial dimensionality before flattening. The network concludes with a fully connected dense layer of 128 neurons and a softmax output layer corresponding to the number of attack classes.

The RF model was implemented using the scikit-learn library. A deep configuration employed an ensemble of 100 fully grown decision trees, enabling high representational capacity for complex non-linear patterns in network traffic. Each tree was trained on bootstrapped samples with random feature selection at each split, improving robustness and generalization. Both models were trained and evaluated in floating-point precision using the binary (normal vs. attack) and multiclass (attack type) classification setup. The confusion matrix in Figure 2 represents the result of binary detection for the RF model.

Table I summarizes the classification performance of the RF and CNN models evaluated on the TON_IoT dataset under both binary and multiclass settings. In the binary intrusion detection task (normal vs. attack), the RF achieved an accuracy and F1-score of 0.9988, while the CNN obtained 0.9952, indicating comparable performance across both models. In the multiclass scenario, which distinguishes between multiple attack types, the RF again outperformed the CNN with an accuracy of 0.9882 compared to 0.8969, maintaining near-perfect precision and recall. The superior multiclass performance of the RF indicates its robustness in capturing diverse intrusion patterns, whereas the CNN exhibited higher misclassification among under-represented classes namely Man-in-the-Middle (MitM), Password and Injection as illustrated in Figure 1.

Computational efficiency results are presented in Table II. The RF consistently demonstrated shorter training and inference times despite its larger memory footprint, completing binary training in approximately 9 s and achieving average inference latency of only 0.0042 ms per sample. The CNN, while

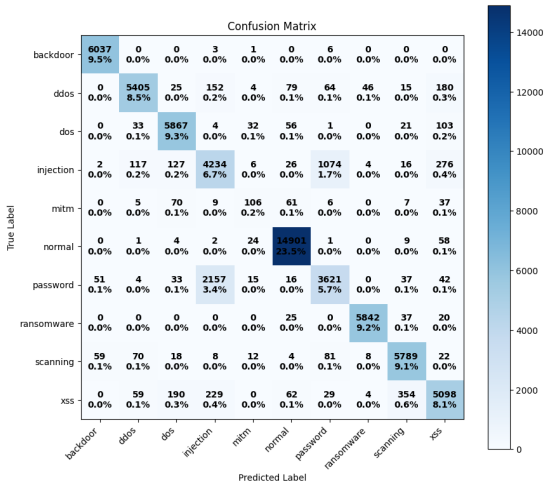


Fig. 1. Confusion matrix for multiclass CNN (host-side evaluation).

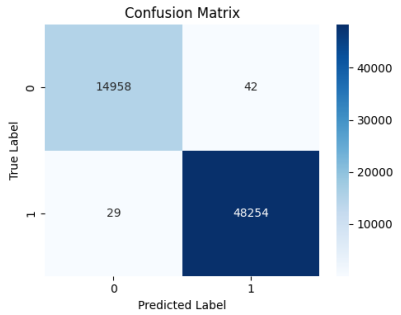


Fig. 2. Confusion matrix for binary detection RF (host-side evaluation).

more compact (∼3.2 MB), required substantially longer training (165–212 s) and higher inference time (0.015–0.146 ms). Although the RF had a larger memory footprint(∼6 MB) than the CNN, it delivered superior accuracy and execution speed, indicating a more favorable balance between performance and computational cost. Based on these results, the RF model was selected for embedded evaluation, where it was translated into bare-metal C code and validated through HIL testing on a RISC-V emulator, as detailed in the following section.

V. Hardware Emulation and Results

Building on the host-side evaluation results, the RF model was implemented for HIL testing on a RISC-V platform. The model's decision-tree structure was translated into portable bare-metal C code with minimal dependencies, resulting in a 1.6 MB ELF firmware suitable for embedded execution as will be described in the following section. Firmware-level validation was performed using the Renode simulator, which supports rapid prototyping and timing analysis of embedded systems without requiring physical hardware. This setup enabled assessment of the model's deployability and runtime behavior under realistic execution constraints.

A. Renode and RISC-V Platform

Renode is an open-source, multi-platform simulation framework developed by Antmicro. It supports a wide range of embedded targets and is particularly well-suited for IoT and edge-AI applications. Our experiments target a 32-bit RISC-V architecture, specifically the `rv32imafdc` ISA, which adds hardware support for floating-point and double-precision operations. A DRAM of size 2 MB was sufficient to run the RF model. The emulator was equipped with UART to be able to receiving network traffic via UART. The RF model was converted to C code using `micromlgen` and integrated into a bare-metal firmware project compiled for RISC-V using `riscv32-unknown-elf-gcc`.

B. Cycle Count Analysis and Inference Timing

The emulated firmware running on a RISC-V soft-core in Renode reported a total cycle count of:

- Total cycles: $2,824,430,009$ cycles
- Average per-sample cycles: $44,832$ cycles

Assuming a conservative 100 MHz clock frequency typical for low-power RISC-V microcontrollers used in edge AI and TinyML the corresponding inference times are calculated as follows:

$$\text{Total Inference Time} = \frac{2{,}824{,}430{,}009}{100 \times 10^6}\ \text{s} \approx 28\,\text{s} \quad (1)$$

$$\text{Average Time per Sample} = \frac{44{,}832}{100 \times 10^6}\ \text{s} \approx 448\,\mu\text{s} \quad (2)$$

These results indicate that the RF model performs inference in under 500 μs per sample on a low-power RISC-V core, making it well-suited for real-time embedded applications such as anomaly detection and edge classification.

C. Firmware Results

The firmware-based evaluation of the RF model on the full test dataset yielded an accuracy of 99.89%, with $62,931$ correct predictions out of $63,000$ samples. The confusion matrix (Figure 3) illustrates the performance of the RF classifier on the test dataset. It correctly identified $48,052$ attacks (true positives) and $14,879$ normal samples (true negatives), while misclassifying only 28 attacks as normal (false negatives) and 41 normal samples as attacks (false positives).

The embedded evaluation on the Renode RISC-V simulator confirmed that the float-precision RF model maintained accuracy comparable to host-side performance, achieving over 99% detection accuracy under binary classification. This consistency demonstrates that the model's decision logic and feature preprocessing were successfully preserved during conversion to bare-metal C code. The compiled firmware executed efficiently within the resource constraints of the simulated RISC-V MCU, validating the feasibility of deploying a full-precision RF-based IDS directly on embedded hardware without relying on external computation, thereby preserving detection reliability while ensuring real-time inference capability.

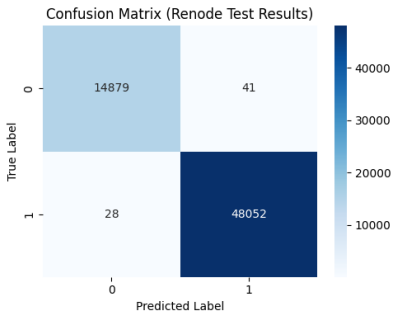


Fig. 3. Confusion matrix for RF run on firmware.

VI. Discussion and Future Work

This work investigated the feasibility of deploying ML–based IDS on low-power embedded platforms using hardware emulation. Two models were trained and evaluated on the TON_IoT dataset: (i) RF classifier and (ii) CNN. Both achieved high binary classification accuracy on the host system (above 99%); however, clear distinctions emerged in their computational efficiency and scalability for embedded implementation.

The RF model consistently outperformed the CNN in both binary and multiclass scenarios, attaining an F1-score of 0.9988 and 0.9883, respectively. Despite its larger model file ($\sim$6 MB), the RF provided faster training and inference and translated directly into compact C code suitable for microcontroller deployment. In contrast, the CNN exhibited higher computational overhead and longer training times, making it less practical for resource-constrained environments. The successful compilation of the RF firmware into a 1.6 MB ELF binary and its execution on a simulated 100 MHz RISC-V processor validate the model's suitability for real-time embedded IDS applications.

One observed limitation, in multiclass detection, is the reduced detection rate for certain attack types, particularly MitM traffic. This weakness stems primarily from class imbalance in the TON_IoT dataset and highlights the importance of more balanced or augmented training data in future studies. Although the RF demonstrated robust overall accuracy, per-class evaluation revealed that rare attack categories remain challenging to identify reliably.

While CNN training required nearly two orders of magnitude more time than RF, the difference is less critical since model training is performed offline. In operational contexts, inference latency becomes the key metric. Here,the RF achieved a per-sample inference time of approximately $448\ \mu s$ on the RISC-V platform, confirming its ability to support real-time detection within the computational limits of typical IoT microcontrollers.

Overall, the results confirm the feasibility of executing full-precision machine learning models directly on embedded processors without the need for quantization or external cloud resources. This capability is essential for latency-sensitive and privacy-critical IIoT applications. Future work will focus on extending the framework to support incremental or online learning, cross-platform portability to ARM and ESP-class devices, and evaluation using diverse datasets that better represent modern IoT attack surfaces. Another approach could be investigating the impact of quantization (the input, the model and the output) on the size and performance of such models. This investigation is crucial for IoT devices with even more restricted resources.

References

[1] A. Ericson, S. Forsström, and K. Thar, "IIoT intrusion detection using lightweight deep learning models on edge devices," in *WFCS*. IEEE, 2024.

[2] J. Åkerberg, J. Furunäs Åkesson, Johanand Gade *et al.*, "Future industrial networks in process automation: Goals, challenges, and future directions," *Applied Sciences*, vol. 11, no. 8, p. 3345, 2021.

[3] University of New South Wales, "ToN-IoT Datasets," https://research.unsw.edu.au/projects/toniot-datasets, 2020, accessed: 2025-08-29.

[4] S. Aftab, Z. S. Shah, S. A. Memon, and Q. Shaikh, "A machine-learning-based intrusion detection for IIoT infrastructure," in *IMTIC*, 2023.

[5] M. S. Andhare and V. S. Kumbhar, "Detecting cybersecurity attacks in industrial IoT: A systematic literature review," in *ICNTE*, 2023.

[6] E. Gyamfi and A. Jurcut, "Intrusion detection in internet of things systems: a review on design approaches leveraging multi-access edge computing, machine learning, and datasets," *Sensors*, vol. 22, no. 10, p. 3744, 2022.

[7] M. F. Elrawy, A. I. Awad, and H. F. A. Hamed, "Intrusion detection systems for IoT-based smart environments: a survey," *Cloud Computing: Advances, Systems and Applications*, vol. 7, no. 1, p. 21, 2018.

[8] Q. Liang, S. Xie, and B. Cai, "Intelligent home iot intrusion detection system based on RISC-V," in *ICPECA*, 2023, pp. 296–300.

[9] F. Dehrouyeh, L. Yang, A. F. Badrkhani, and A. Shami, "On TinyML and cybersecurity: Electric vehicle charging infrastructure use case," *IEEE Access*, vol. 12, pp. 108 703–108 730, 2024.

[10] A. Alsaedi, Moustafa *et al.*, "TON_IoT telemetry dataset: a new generation dataset of iot and iiot for data-driven intrusion detection systems," *IEEE Access*, vol. 8, pp. 165 130–165 150, 2020.

[11] H. Im and S. Lee, "TinyML-based intrusion detection system for in-vehicle network using convolutional neural network," *IEEE Embedded Systems Letters*, 2024.

[12] I. Sharafaldin, A. Lashkari, and A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *ICISSP*, 2018, pp. 108–116.

[13] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *MilCIS*. IEEE, 2015, pp. 1–6.

[14] N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets," *Sustainable Cities and Society*, p. 102994, 2021.

[15] N. Moustafa, M. Ahmed, and S. Ahmed, "Data analytics-enabled intrusion detection: Evaluations of TON_IoT linux datasets," in *TrustCom*, 2020, pp. 727–735.

[16] N. Moustafa, M. Keshk, E. Debie, and H. Janicke, "Federated TON_IoT windows datasets for evaluating ai-based security applications," in *TrustCom*, 2020, pp. 848–855.

[17] N. Moustafa, "New generations of internet of things datasets for cybersecurity applications based machine learning: TON_IoT datasets," in *eResearch Australasia Conference*, Brisbane, Australia, 2019.

[18] M. B. Kursa and W. R. Rudnicki, "Feature selection with the boruta package," *Statistical Software*, vol. 36, no. 11, pp. 1–13, 2010.

[19] M. B. Kursa, W. R. Rudnicki *et al.*, "BorutaPy: Python Implementation of the Boruta Feature Selection Algorithm," GitHub repository, 2025, available: https://github.com/scikit-learn-contrib/boruta_py.

[20] M. Abadi, A. Agarwal, P. Barham, E. Brevdo *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.