Journal of Computing and Data Technology

https://journal.krrishpub.com/index.php/jcdt/index

RESEARCH ARTICLE

A Leakage-Controlled Evaluation Protocol for Machine-Learning Intrusion Detection on the TON_IoT Network Dataset

Reginald Ekene Ogu [a,*]

[a] *Department of Computer Engineering, Federal University of Technology, Owerri, Imo State, Nigeria.*

Abstract

Intrusion detectors trained on the TON_IoT network data routinely report near-perfect scores, but the extent to which these scores reflect attack behaviour rather than dataset-specific collection artefacts remains unclear. This study evaluates four classifiers using four progressively stricter evaluation protocols on the 211,043-flow train–test file: naive random splitting with all columns, removal of addressing and free-text identifiers, deduplication before splitting, and holding out entire attacker hosts. A probe trained only on IP addresses and ports reached a binary macro-F1 of 0.9999 and a ten-class macro-F1 of up to 0.932, indicating that the identifiers alone are highly predictive of the labels in this dataset. Exact duplicates made up 9.7% of rows, and 12.1% of test rows under naive cross-validation had a copy in the training fold. Removing identifiers cut multiclass macro-F1 from 0.986 to 0.953 for random forest, with man-in-the-middle F1 falling from 0.903 to 0.757. Under host hold-out, attack recall stayed above 0.95, while false-alarm rates on unseen benign hosts varied substantially across hold-outs. The evaluation protocols, identifier-only probe, and reporting checklist provide practical safeguards for future studies using this dataset.

Keywords: Data leakage, Internet of Things (IoT), Intrusion detection, Shortcut learning.

Article information:

ISSN: 3107-9466 (Online)
Published by: **Krrish Scientific Publications Pvt. Ltd.**
DOI: https://doi.org/10.71426/jcdt.v2.i2.pp207-215
Received: *24 Oct. 2026* | Revised: *08 Nov. 2026* | Accepted: *10 Nov. 2026* | Published: *11 Nov. 2026*
Copyright © 2026 Author(s).
This is an open-access article distributed under the terms of the Creative Commons Attribution–NonCommercial 4.0 International License (CC BY-NC 4.0).

1. Introduction

The Internet of Things has pushed intrusion detection out of the data centre and into homes, factories and city infrastructure, where the devices being protected are cheap, numerous and rarely patched. Machine-learning methods are widely used for this task, and researchers frequently rely on a limited set of public benchmark datasets to compare their performance [1]–[3]. TON_IoT, released by UNSW Canberra, is one of the most widely used of these. It combines IoT and IIoT telemetry, operating-system logs and network flows captured on a testbed with nine attack families [4]–[6].

The TON_IoT network data have been used to build and compare a wide range of detectors, from tree ensembles to graph neural networks [7]–[9], and reported scores on these data are often close to the maximum possible value. Such results warrant careful scrutiny because operational networks differ substantially from controlled testbeds. Previous research has shown that benchmark performance can be inflated by dataset-construction artefacts rather than by the detection of generalisable attack characteristics [10]–[13].

The concern has a precise name: data leakage. Kaufman et al. defined leakage as the introduction of information into training that would not legitimately be available when the model is used [14]. Kapoor and Narayanan later documented leakage across 17 scientific fields and found it behind hundreds of irreproducible machine-learning results [15]. In network intrusion detection, D'hooge et al. showed that metadata such as addresses and timestamps can contaminate detectors trained on CIC-IDS datasets [16]. Similar warnings exist for flow-level feature sets more generally [17], [18].

TON_IoT has a particular structure that makes leakage likely. Its ground truth was produced by tagging traffic from a known set of attacker IP addresses within known time windows [4]. Consequently, the source address may provide a strong proxy for the assigned label. The same small group of attacking machines produces most attack classes, while

*Corresponding author
Email address: reginald.ogu@futo.edu.ng (Reginald Ekene Ogu).

benign traffic comes from many more hosts. The dataset documentation describes this labelling procedure; however, its influence on performance estimates for the commonly used network train–test file warrants direct assessment.

This paper measures that effect directly. Rather than proposing another detector, it treats evaluation design as the object of study. The contributions are as follows:

1. A four-step evaluation ladder (P0–P3) that removes, one at a time, the main routes by which a model can score well on TON_IoT without necessarily learning generalisable attack behaviour: identifier columns, duplicated rows, and host-level overlap between training and testing.
2. An identifier-only probe showing that IP addresses and ports by themselves achieve a binary macro-F1 of 0.9999, providing a reproducible diagnostic of identifier-based shortcuts in this dataset.
3. A per-class and per-host analysis showing which attack types depend most on identifiers, and that false alarms on unseen benign hosts are a substantial source of error under host hold-out evaluation.
4. A short reporting checklist for studies that use the TON_IoT network data.

The rest of the paper is organised as follows. Section 2 reviews related work on benchmark quality and leakage. Section 3 describes the dataset. Section 4 sets out the protocols, models and metrics. Section 5 reports results, Section 6 discusses the implications and limitations of the findings, and Section 7 concludes.

2. Related Work

2.1. IoT intrusion detection benchmarks

Several datasets now compete for the role of standard IoT benchmark. Earlier collections such as UNSW-NB15 and Bot-IoT were followed by TON_IoT [4], [6], X-IIoTID [19], Edge-IIoTset [20] and CICIoT2023 [21]. Booij et al. examined TON_IoT specifically and argued that heterogeneous features and inconsistent attack labels across IoT datasets make cross-study comparison difficult [6]. That call for standardisation was echoed by Sarhan et al., who proposed a common NetFlow feature set and showed that detectors trained on one dataset often transfer poorly to another [17], [18]. Layeghy and Portmann reached a similar conclusion through explainable cross-domain evaluation [22], and Alani and Miri searched for a compact, explainable feature set that would hold across IoT datasets [23].

Reviews of the field tend to emphasise model architecture over evaluation design [1]–[3]. Tree ensembles, convolutional and recurrent networks, graph neural networks and federated learning have all been applied to TON_IoT [8], [24]–[26], and per-sensor ensembles have been built for its telemetry component [27]. Reported scores are frequently close to the maximum, making meaningful comparisons between methods more difficult.

2.2. Dataset flaws and leakage in security ML

A separate line of research has examined whether benchmark scores reliably reflect the intended detection capability. Engelen et al. found labelling and feature-extraction errors in CIC-IDS2017 [12], and Liu et al. measured how widespread such errors were across CIC-IDS2017 and CSE-CIC-IDS2018 [28]. Lanvin et al. showed that correcting those errors changed detection results substantially [13]. Guerra et al. argued more broadly that labelling network traffic is harder than benchmark datasets suggest [29].

Arp et al. catalogued ten pitfalls in machine learning for computer security, including data snooping, spurious correlations and inappropriate baselines, and found these pitfalls to be widespread in the top-tier security papers under examination [10]. Apruzzese et al. questioned whether laboratory evaluations of network intrusion detection reflect operational conditions [11], [30]. Andresini et al. showed that detectors degrade under concept drift that random splits never expose [31].

Leakage is not unique to security. Kapoor and Narayanan's survey [15], Whalen et al.'s guidance for genomics [32] and Rosenblatt et al.'s study of neuroimaging [33] all describe the same pattern: random splitting of data with hidden group structure produces optimistic estimates. The related idea of shortcut learning, in which a model exploits an easy but irrelevant cue, was set out by Geirhos et al. [34].

2.3. Identifier features in TON_IoT studies

Practice on TON_IoT varies. Some studies remove IP addresses and ports before training; others keep these columns, and some do not report the choice at all. D'hooge et al. demonstrated the effect of metadata features on CIC datasets [16], and Verkerken et al. examined how unsupervised detectors generalise across datasets [35]. Hung recently evaluated detectors on CICIoT2023 with an entire attack family withheld from training, a grouped design that is closer to deployment than random splitting [36]. The present study jointly examines identifier dependence, duplicate contamination, and host-level generalisation on the TON_IoT network train–test file.

3. Dataset

3.1. The TON_IoT network train–test file

The TON_IoT datasets were collected at the Cyber Range and IoT Labs of UNSW Canberra on a testbed spanning edge, fog and cloud layers [4], [37]. The network component was captured with Zeek and processed into flow records. Studies typically use the curated train–test file (`train_test_network.csv`), which contains 211,043 flows and 44 columns. The columns comprise 42 features, a binary label and a ten-way type field. The features cover connection metadata (addresses, ports, protocol, service, duration, byte and packet counts, connection state) and application-layer fields for DNS, SSL and HTTP, plus Zeek "weird" events.

Table 1 gives the class distribution. Normal traffic accounts for 50,000 flows, and eight attack classes contribute 20,000 each. Man-in-the-middle (MITM) traffic is the exception, with only 1,043 flows.

Table 1: Class distribution of the TON_IoT network train–test file before and after removal of exact duplicate rows.

Class	Raw flows	After deduplication	Removed (%)
normal	50,000	42,040	15.9
backdoor	20,000	18,711	6.4
ddos	20,000	19,993	0.0
dos	20,000	18,992	5.0
injection	20,000	19,964	0.2
password	20,000	19,861	0.7
ransomware	20,000	14,735	26.3
scanning	20,000	20,000	0.0
xss	20,000	15,137	24.3
mitm	1,043	1,041	0.2
Total	**211,043**	**190,474**	**9.7**

Table 2: Attack-generating source hosts in the deduplicated data (hosts with more than 500 attack flows).

Source host	Attack flows	Attack classes present
192.168.1.30	60,503	ddos, dos, injection, password, scanning
192.168.1.31	30,245	ddos, dos, injection, mitm, password, scanning
192.168.1.32	22,402	scanning, xss
192.168.1.193	19,774	backdoor, ransomware
192.168.1.33	7,284	backdoor, ransomware
192.168.1.37	6,374	backdoor, ransomware
192.168.1.39	692	dos
192.168.1.36	593	xss
192.168.1.34	528	ddos, mitm

3.2. Where the identifiers come from

According to the dataset documentation, ground truth was produced by tagging traffic from the Kali Linux attacking machines, identified by IP address and timestamp [4]. In the train–test file these machines appear mainly in the 192.168.1.30–39 range. Table 2 shows what this means in practice. The nine source addresses listed in Table 2 each contribute more than 500 attack flows and together account for a substantial share of attack traffic. Several of these hosts are tied to only one or two attack classes: 192.168.1.32 produced only scanning and XSS traffic, and 192.168.1.33 and 192.168.1.37 produced only backdoor and ransomware. Benign traffic, by contrast, came from 40 distinct source addresses. Host 192.168.1.193 appears in both benign and attack traffic, highlighting that source-address identity is not a universally unique indicator of class membership.

This structure follows from the testbed collection and labelling procedure. It can bias evaluation when source addresses are included as predictors and flows from the same hosts appear in both training and test sets.

4. Methodology

4.1. Feature groups

The 42 features were divided into three groups. *Identifiers* are `src_ip`, `dst_ip`, `src_port` and `dst_port`. *Free-text fields* are `dns_query`, `ssl_subject`, `ssl_issuer`, `http_uri` and `http_user_agent`; these carry environment-specific strings such as domain names that would not generalise to another network. The remaining 33 columns constitute the *behavioural* feature set, intended to describe connection characteristics rather than host identities. Destination port was treated separately in one ablation because it carries service information as well as addressing. Categorical columns were integer-encoded. The empty marker "-" was kept as its own category to represent an absent protocol-field value.

4.2. The evaluation ladder

Figure 1 summarizes the four-step evaluation ladder for this study.

P0 – naive: All 42 features, raw rows (duplicates retained), stratified five-fold cross-validation. This mirrors common practice.

P1 – identifier-free: As P0, but identifiers and free-text fields are removed, leaving the 33 behavioural features.

P2 – deduplicated: As P1, but exact duplicate rows are removed *before* folds are formed, so no test row has an identical twin in training. Variant P2b adds `dst_port` back to isolate its contribution.

P3 – host hold-out: As P2, but for each of the nine attacking hosts in Table 2, all flows from that host form the test set together with a random 20% of benign source hosts. The model is trained on everything else. This tests whether a detector can recognise attacks from a machine it has never seen, and whether it raises false alarms on benign hosts it has never seen. P3 is run as a binary task because withholding a host removes most of the examples of some attack classes.

A fifth configuration, the *identifier probe*, trains on the four identifier columns alone using deduplicated data. The probe assesses how much label information is available from these identifiers alone.

4.3. Classifiers

Four classifiers were selected to represent commonly used model families for TON_IoT: multinomial logistic regression (LR) on standardised inputs, a decision tree (DT) grown to purity, a random forest (RF) of 100 trees and a LightGBM gradient-boosted ensemble (LGBM). Tree ensembles remain strong baselines on tabular data [38]. The default LightGBM configuration did not produce satisfactory results on the ten-class task in preliminary runs, so the configuration was fixed at 300 trees, learning rate 0.05, 63 leaves and a minimum of 10 samples per leaf. No hyperparameter was tuned on test folds. All experiments used scikit-learn 1.9 and LightGBM with a fixed random seed.

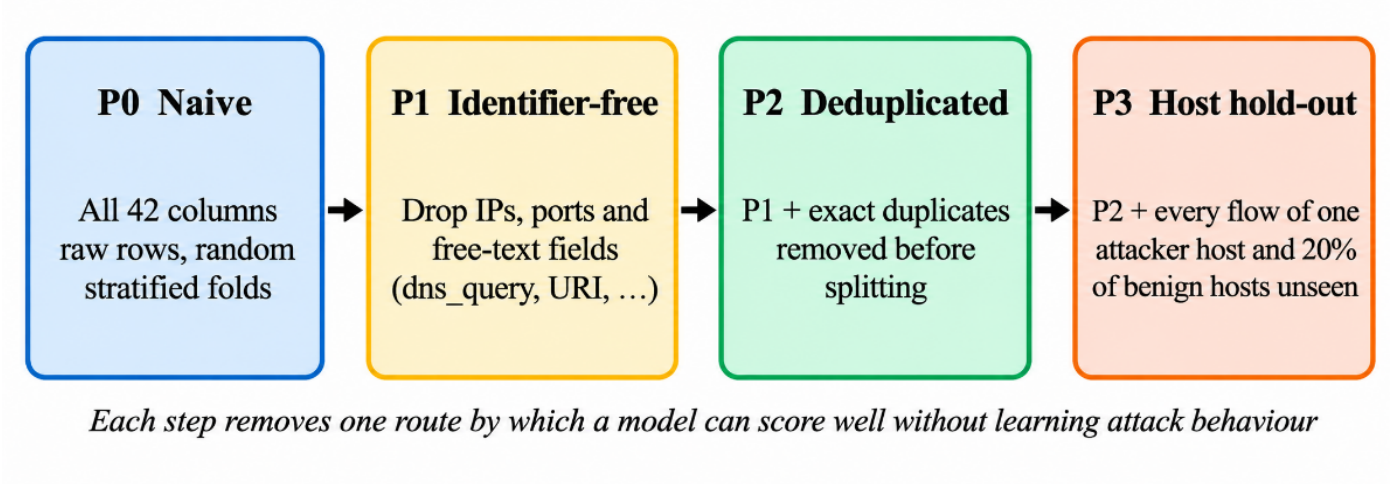


Figure 1: The four-step evaluation ladder. Each protocol removes one route by which a model can achieve a high score without learning attack behaviour.

4.4. Metrics

For class c with true positives TP_c, false positives FP_c and false negatives FN_c, precision and recall are defined in (1).

$$P_c = \frac{TP_c}{TP_c + FP_c}, \qquad R_c = \frac{TP_c}{TP_c + FN_c} \tag{1}$$

The per-class F1 score and its unweighted mean over K classes are defined in (2).

$$F_{1,c} = \frac{2P_cR_c}{P_c + R_c}, \qquad F_1^{\text{macro}} = \frac{1}{K}\sum_{c=1}^{K} F_{1,c} \tag{2}$$

Macro-F1 was preferred to accuracy because accuracy is dominated by the large classes and hides failure on MITM. For each applicable protocol, the difference in macro-F1 relative to the naive baseline is calculated using (3). This difference is termed the *leakage gap* here, although it may also reflect changes in sample composition and evaluation conditions.

$$\Delta_k = F_1^{\text{macro}}(\text{P0}) - F_1^{\text{macro}}(\text{P}_k) \tag{3}$$

Duplicate contamination under P0 was measured as the fraction of test rows with an exact copy in the training fold (4):

$$\rho = \frac{1}{|\mathcal{D}_{\text{te}}|}\sum_{x \in \mathcal{D}_{\text{te}}} \mathbb{1}[x \in \mathcal{D}_{\text{tr}}] \tag{4}$$

For the host hold-out protocol, attack recall on the held-out host h and the false-alarm rate on unseen benign hosts $\mathcal{B}$ are

$$\begin{aligned} \text{AR}_h &= \frac{|\{i \in A_h : \widehat{y}_i = 1\}|}{|A_h|}, \\ \text{FAR}_h &= \frac{|\{i \in N_{\mathcal{B}} : \widehat{y}_i = 1\}|}{|N_{\mathcal{B}}|} \end{aligned} \tag{5}$$

In (5), A_h is the set of attack flows from host h and $N_{\mathcal{B}}$ the set of benign flows from the held-out benign hosts. Results under P0–P2 are reported as the mean and standard deviation over five folds.

5. Results

5.1. The identifiers alone nearly solve the problem

Table 3 and Figure 2 report macro-F1 for every protocol. The identifier probe, which sees nothing but two addresses and two port numbers, reached a binary macro-F1 of 0.9999 with all three tree models. On the ten-class task it scored between 0.912 (DT) and 0.932 (LGBM). Thus, identifiers alone permit near-perfect binary discrimination and high multiclass macro-F1 on this benchmark. Macro-F1 is an average across classes and should not be interpreted as the percentage of individual attacks correctly named.

The importance rankings are shown in Figure 5. When all columns were available, the random forest placed `src_ip` first, with 17.6% of the total impurity reduction. `src_port`, `dst_ip` and `dst_port` also appeared in the top seven, and together the four identifiers accounted for about 40% of the model's importance mass.

5.2. Removing identifiers costs the most where it matters

Binary detection barely moved when identifiers were removed: random forest fell from 1.0000 to 0.9977. The behavioural features are enough to separate normal from attack traffic in this capture. Multiclass typing was another matter. Random forest dropped from 0.986 to 0.953, and LightGBM from 0.988 to 0.955, giving leakage gaps Δ_1 of about 0.033. Logistic regression collapsed from 0.806 to 0.523. This substantial reduction suggests that the linear model relied heavily on the encoded identifier variables. Moreover, treating IP addresses and port values as ordinal quantities can introduce artificial numerical relationships.

The cost was not spread evenly across classes (Figure 3). MITM F1 fell from 0.903 to 0.757, XSS from 0.9996 to 0.920 and ransomware from 0.9999 to 0.948. All three classes are tied to a small number of hosts in Table 2. Classes produced by the two busiest attackers, such as password and scanning, changed by less than 0.01.

Adding `dst_port` back (P2b) recovered about 0.010 of macro-F1 for the tree models. Destination port can be a meaningful feature because it provides information

Table 3: Macro-F1 (mean ± standard deviation over five folds) by protocol and classifier.

Task	Protocol	LR	DT	RF	LGBM
Binary	P0 naive	0.9921 ± 0.0006	0.9999 ± 0.0001	1.0000 ± 0.0000	0.9999 ± 0.0001
	P1 identifier-free	0.9336 ± 0.0005	0.9970 ± 0.0002	0.9977 ± 0.0003	0.9976 ± 0.0002
	P2 deduplicated	0.9785 ± 0.0006	0.9963 ± 0.0002	0.9973 ± 0.0002	0.9974 ± 0.0001
	P2b + dst_port	0.9787 ± 0.0007	0.9971 ± 0.0003	0.9979 ± 0.0003	0.9978 ± 0.0003
	Identifier probe	–	0.9999 ± 0.0001	0.9999 ± 0.0000	0.9999 ± 0.0000
Ten-class	P0 naive	0.8060 ± 0.0039	0.9841 ± 0.0014	0.9860 ± 0.0010	0.9882 ± 0.0011
	P1 identifier-free	0.5233 ± 0.0152	0.9483 ± 0.0020	0.9535 ± 0.0021	0.9554 ± 0.0021
	P2 deduplicated	0.5425 ± 0.0023	0.9464 ± 0.0041	0.9525 ± 0.0022	0.9542 ± 0.0013
	P2b + dst_port	0.6077 ± 0.0027	0.9569 ± 0.0033	0.9629 ± 0.0021	0.9653 ± 0.0015
	Identifier probe	–	0.9116 ± 0.0006	0.9123 ± 0.0007	0.9319 ± 0.0011

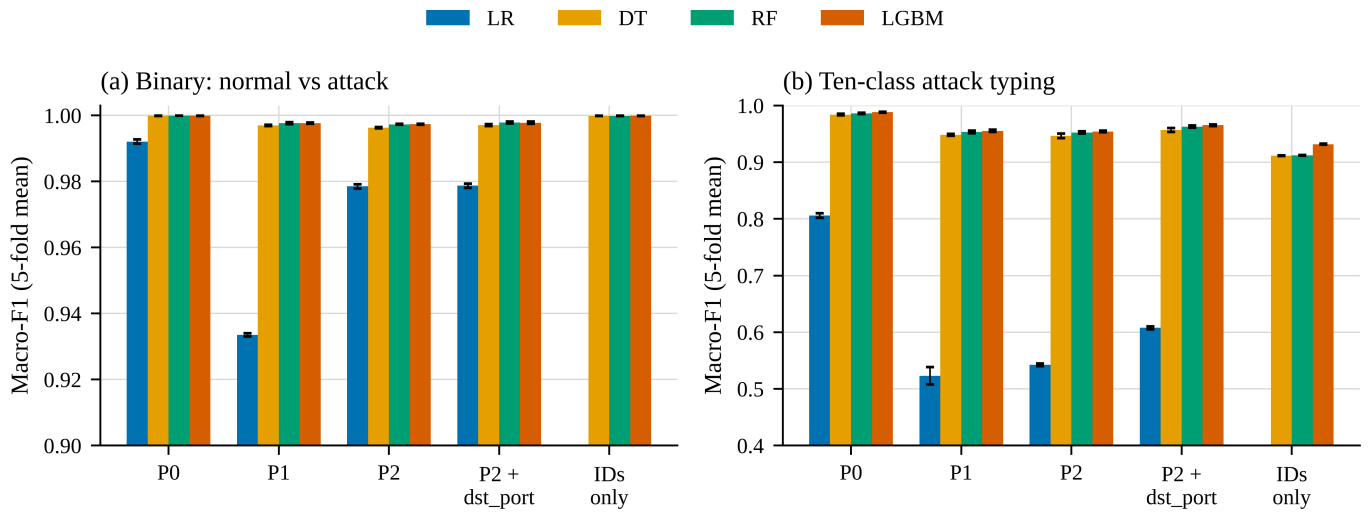


Figure 2: Macro-F1 under each protocol for (a) binary detection and (b) ten-class attack typing. Error bars show the standard deviation over five folds. Note the different vertical scales.

about the targeted service; for example, TCP port 445 is commonly associated with SMB traffic. In this capture, however, benign traffic also used environment-specific high ports (including 10502, 15600 and 51782) that act as fingerprints of particular benign services. The decision to retain `dst_port` should therefore be explicitly justified in the evaluation protocol.

5.3. Duplicates

Exact duplicates made up 20,569 rows, or 9.7% of the file, concentrated in particular classes. More than a quarter of ransomware flows and nearly a quarter of XSS flows were repeats (Table 1). Under P0, 12.1% of test rows on average had an identical copy in the corresponding training fold, giving $\rho = 0.121$. Such overlap can allow a classifier to benefit from repeated examples rather than demonstrating generalisation to independent observations.

In this dataset, deduplication on its own changed headline macro-F1 only slightly (compare P1 with P2 in Table 3). One possible explanation is that duplicated records occur in classes that are already classified accurately; deduplication also changes the class distribution. These possibilities cannot be separated from the reported aggregate results alone. It still matters for validity. An evaluation containing duplicate overlap does not provide the same evidence of generalisation as an evaluation on independent traffic, even when their aggregate scores are similar.

5.4. Unseen hosts: attacks are caught, benign hosts are not trusted

Figure 4 reports the host hold-out experiment. Attack recall on the held-out attacker stayed high: the random forest detected between 95.5% (host 0.34, 528 flows) and 100% of attack flows, with a mean of 99.3%. Under random splitting the same hosts were detected at 97.6–100%, so the drop on the attack side was small.

When a fifth of the benign hosts were withheld, the classifier flagged between 4.5% and 51.1% of the traffic from those hosts as malicious, with a mean false-alarm rate of 25.3% for random forest and 34.1% for LightGBM. Macro-F1 over the held-out test set ranged from 0.42 to 0.97 depending on which hosts were withheld (Table 4). These results are consistent with the models learning host-specific characteristics of benign traffic that do not transfer

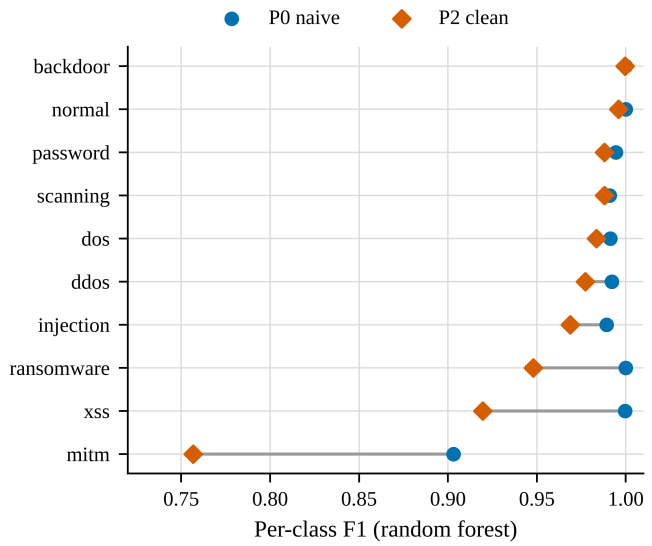


Figure 3: Per-class F1 of the random forest under the naive protocol (P0) and the identifier-free, deduplicated protocol (P2).

Table 4: Host hold-out (P3) results averaged over the nine held-out attacker hosts.

Classifier	Attack recall Mean (range)	False-alarm rate Mean (range)	Macro-F1 Mean (range)
DT	0.980 (0.854–1.000)	0.275 (0.013–0.603)	0.828 (0.326–0.991)
RF	0.993 (0.955–1.000)	0.253 (0.045–0.511)	0.844 (0.419–0.974)
LGBM	0.990 (0.932–1.000)	0.341 (0.093–0.592)	0.800 (0.333–0.969)

reliably to unseen hosts.

This finding has operational implications: a false-alarm rate near 25% on benign traffic from held-out hosts could impose a substantial alert-handling burden. Random splitting can obscure this weakness because traffic from the same benign hosts may appear in both training and test folds.

6. Discussion

6.1. What the headline numbers were measuring

Taken together, the results suggest that the near-perfect scores often reported on the TON_IoT network data measure at least three different things at once. The first is attack behaviour, which is real: identifier-free binary detection under deduplication still reaches 0.997. The second is the identity of the attacking machines, which a probe can exploit almost perfectly. The third is the identity of the benign machines, which only becomes visible when hosts are held out. Of these, behavioural features are more likely to transfer across environments, although transferability was not directly tested in a separate operational network. The second is a classic shortcut in the sense of Geirhos et al. [34]: a feature that is predictive because of dataset-specific collection conditions rather than a robust relationship with attack behaviour.

The finding parallels what D'hooge et al. reported for CIC-IDS2017 [16] and what Arp et al. described as spurious correlation [10]. Explainability tools are increasingly recommended for IoT intrusion detection as a way to build operator trust [39]–[41]. The importance analysis in Figure 5 suggests a second use: an attribution dominated by addresses is a leakage warning, and it is cheap to check. The finding offers a plausible explanation for some of the cross-dataset transfer difficulties reported by Sarhan et al. and Layeghy and Portmann [18], [22]: address-dependent decision rules may not remain useful in networks with different host identifiers.

6.2. Benign diversity is the bottleneck

The asymmetry in Section 5.4 deserves attention. In a controlled testbed, scripted attacks may share reproducible traffic characteristics across hosts. This is one possible explanation for the high recall observed on held-out attacker hosts. Benign traffic is more varied, and in TON_IoT it comes from a modest number of devices and services. Exposure to traffic from a subset of benign hosts does not guarantee accurate classification of traffic from additional hosts. Benchmarks that want to support claims about deployment therefore need more diverse benign traffic, and evaluations need to test on benign hosts the model has not seen. Many intrusion-detection studies emphasise attack-class performance, making the evaluation of benign-host generalisation an important complementary consideration [1], [3].

6.3. A reporting checklist

Based on these results, studies using the TON_IoT network data should, at a minimum:

1. state which columns were used and remove `src_ip`, `dst_ip` and `src_port`; justify any use of `dst_port`;
2. remove free-text fields such as `dns_query`, `http_uri` and certificate subjects unless the study is explicitly about those fields;
3. deduplicate before forming any split and report the number of rows removed;
4. report the identifier-only probe alongside the main results as a leakage reference point;
5. report macro-F1 and per-class F1, with MITM given explicitly;
6. include at least one grouped evaluation, by host or by time, and report false alarms on unseen benign hosts.

Together, these steps can reduce avoidable sources of evaluation bias and make reported detection performance easier to interpret.

6.4. Limitations

Several limitations apply. Only the curated train–test file was used. The full TON_IoT network capture is larger, and the same analysis could be run on it with timestamps available for temporal splitting. The host hold-out protocol is restricted to binary detection, so it does not show how attack typing generalises across hosts. The benign hosts held out in each round were drawn at random, and

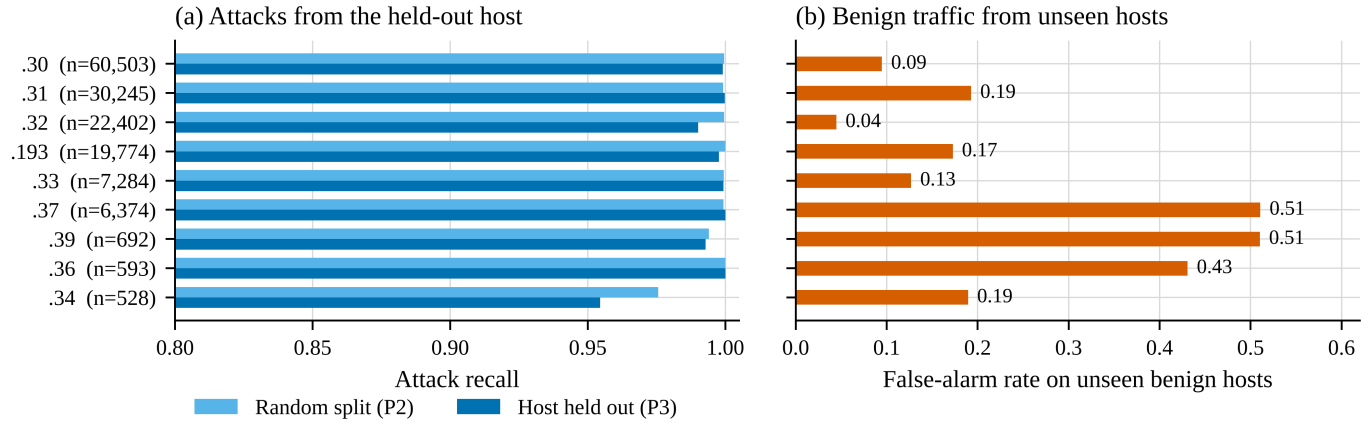


Figure 4: Host hold-out results for the random forest. (a) Attack recall on the held-out attacker under random splitting (P2) and host hold-out (P3). (b) False-alarm rate on benign traffic from held-out benign hosts.

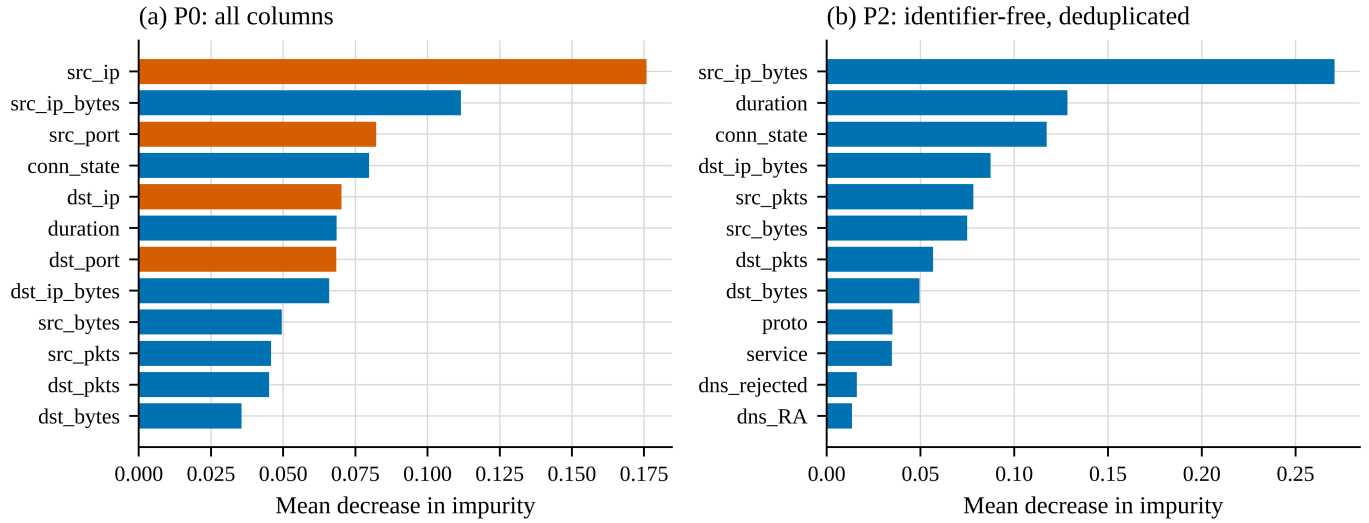


Figure 5: Random-forest feature importance under (a) the naive protocol and (b) the identifier-free, deduplicated protocol. Identifier features are shown in orange.

the false-alarm rate depends on which hosts were chosen; the ranges in Table 4 reflect that variability rather than a confidence interval. Four classical classifiers were tested. Deep-learning models may respond differently to identifier features, and their behaviour requires separate experimental assessment. Finally, the findings concern this dataset's construction and should not be read as a criticism of its authors, who documented the labelling procedure openly [4], [6].

7. Conclusion

On the TON_IoT network train–test file, two IP addresses and two port numbers are enough to separate attacks from normal traffic with a macro-F1 of 0.9999. Removing these four columns leaves binary detection almost unchanged, but multiclass macro-F1 falls by about 0.03, with the largest losses on MITM, XSS and ransomware. Duplicates make up nearly a tenth of the file, and approximately 12.1% of test rows under naive cross-validation have an identical copy in training. Holding out hosts shows that attacks from unseen machines are still caught, while benign traffic from unseen machines is often misclassified. The evaluation ladder, identifier probe, and reporting checklist in Section 6.3 are intended to make these effects visible in future studies and to support more credible assessment of intrusion-detection performance.

Declarations and ethical statements

Conflict of interest: The author declares no conflict of interest.
Funding statement: The author declares that no specific funding was received for this research.
Artificial Intelligence usage statement: During the preparation of this manuscript, the author utilized Claude

AI tools to assist with language refinement, grammar correction, formatting, and improvement of manuscript readability. The author carefully reviewed and edited the content and takes full responsibility for the accuracy, originality, and integrity of the final submitted manuscript.

Availability of data and materials: The TON_IoT datasets analysed in this study are publicly available from IEEE DataPort [37] (https://doi.org/10.21227/fesz-dm97) and from UNSW Canberra at https://research.unsw.edu.au/projects/toniot-datasets. The analysis scripts used in this study are available from the author on reasonable request.

CRediT authorship contribution statement

Reginald Ekene Ogu: Conceptualization, Methodology, Software, Formal analysis, Investigation, Data curation, Visualization, Writing – original draft, review & editing.

Publisher's note

Krrish Scientific Publications Pvt. Ltd. and the ***Journal of Computing and Data Technology*** remain neutral with regard to jurisdictional claims in published maps and institutional affiliations.

References

[1] Heidari A, Jabraeil Jamali MA. Internet of Things intrusion detection systems: a comprehensive review and future directions. *Cluster Computing*. 2023;26(6):3753–3780. Available from: https://doi.org/10.1007/s10586-022-03776-z

[2] Gyamfi E, Jurcut A. Intrusion Detection in Internet of Things Systems: A Review on Design Approaches Leveraging Multi-Access Edge Computing, Machine Learning, and Datasets. *Sensors*. 2022;22(10):3744. Available from: https://doi.org/10.3390/s22103744

[3] Haque S, El-Moussa F, Komninos N, Rajarajan M. A Systematic Review of Data-Driven Attack Detection Trends in IoT. *Sensors*. 2023;23(16):7191. Available from: https://doi.org/10.3390/s23167191

[4] Moustafa N. A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets. *Sustainable Cities and Society*. 2021;72:102994. Available from: https://doi.org/10.1016/j.scs.2021.102994

[5] Alsaedi A, Moustafa N, Tari Z, Mahmood A, Anwar A. TON_IoT Telemetry Dataset: A New Generation Dataset of IoT and IIoT for Data-Driven Intrusion Detection Systems. *IEEE Access*. 2020;8:165130–165150. Available from: https://doi.org/10.1109/ACCESS.2020.3022862

[6] Booij TM, Chiscop I, Meeuwissen E, Moustafa N, den Hartog FTH. ToN_IoT: The Role of Heterogeneity and the Need for Standardization of Features and Attack Types in IoT Network Intrusion Data Sets. *IEEE Internet of Things Journal*. 2022;9(1):485–496. Available from: https://doi.org/10.1109/JIOT.2021.3085194

[7] Gad AR, Nashat AA, Barkat TM. Intrusion Detection System Using Machine Learning for Vehicular Ad Hoc Networks Based on ToN-IoT Dataset. *IEEE Access*. 2021;9:142206–142217. Available from: https://doi.org/10.1109/ACCESS.2021.3120626

[8] Lo WW, Layeghy S, Sarhan M, Gallagher M, Portmann M. E-GraphSAGE: A Graph Neural Network based Intrusion Detection System for IoT. In: *Proceedings of the NOMS 2022–2022 IEEE/IFIP Network Operations and Management Symposium*. IEEE; 2022. p. 1–9. Available from: https://doi.org/10.1109/NOMS54207.2022.9789878

[9] Sarhan M, Layeghy S, Moustafa N, Gallagher M, Portmann M. Feature extraction for machine learning-based intrusion detection in IoT networks. *Digital Communications and Networks*. 2024;10(1):205–216. Available from: https://doi.org/10.1016/j.dcan.2022.08.012

[10] Arp D, Quiring E, Pendlebury F, Warnecke A, Pierazzi F, Wressnegger C, et al. Pitfalls in Machine Learning for Computer Security. *Communications of the ACM*. 2024;67(11):104–112. Available from: https://doi.org/10.1145/3643456

[11] Apruzzese G, Laskov P, Schneider J. SoK: Pragmatic Assessment of Machine Learning for Network Intrusion Detection. In: *Proceedings of the 2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*. IEEE; 2023. p. 592–614. Available from: https://doi.org/10.1109/EuroSP57164.2023.00042

[12] Engelen G, Rimmer V, Joosen W. Troubleshooting an Intrusion Detection Dataset: The CICIDS2017 Case Study. In: *Proceedings of the 2021 IEEE Security and Privacy Workshops (SPW)*. IEEE; 2021. p. 7–12. Available from: https://doi.org/10.1109/SPW53761.2021.00009

[13] Lanvin M, Gimenez P-F, Han Y, Majorczyk F, Mé L, Totel É. Errors in the CICIDS2017 Dataset and the Significant Differences in Detection Performances It Makes. In: *Risks and Security of Internet and Systems*. Lecture Notes in Computer Science, vol. 13857. Cham: Springer; 2023. p. 18–33. Available from: https://doi.org/10.1007/978-3-031-31108-6_2

[14] Kaufman S, Rosset S, Perlich C, Stitelman O. Leakage in data mining: Formulation, detection, and avoidance. *ACM Transactions on Knowledge Discovery from Data*. 2012;6(4):1–21. Available from: https://doi.org/10.1145/2382577.2382579

[15] Kapoor S, Narayanan A. Leakage and the reproducibility crisis in machine-learning-based science. *Patterns*. 2023;4(9):100804. Available from: https://doi.org/10.1016/j.patter.2023.100804

[16] D'hooge L, Verkerken M, Volckaert B, Wauters T, De Turck F. Establishing the Contaminating Effect of Metadata Feature Inclusion in Machine-Learned Network Intrusion Detection Models. In: *Detection of Intrusions and Malware, and Vulnerability Assessment*. Lecture Notes in Computer Science, vol. 13358. Cham: Springer; 2022. p. 23–41. Available from: https://doi.org/10.1007/978-3-031-09484-2_2

[17] Sarhan M, Layeghy S, Portmann M. Towards a Standard Feature Set for Network Intrusion Detection System Datasets. *Mobile Networks and Applications*. 2022;27(1):357–370. Available from: https://doi.org/10.1007/s11036-021-01843-0

[18] Sarhan M, Layeghy S, Portmann M. Evaluating Standard Feature Sets Towards Increased Generalisability and Explainability of ML-Based Network Intrusion Detection. *Big Data Research*. 2022;30:100359. Available from: https://doi.org/10.1016/j.bdr.2022.100359

[19] Al-Hawawreh M, Sitnikova E, Aboutorab N. X-IIoTID: A Connectivity-Agnostic and Device-Agnostic Intrusion Data Set for Industrial Internet of Things. *IEEE Internet of Things Journal*. 2022;9(5):3962–3977. Available from: https://doi.org/10.1109/JIOT.2021.3102056

[20] Ferrag MA, Friha O, Hamouda D, Maglaras L, Janicke H. Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning. *IEEE Access*. 2022;10:40281–40306. Available from: https://doi.org/10.1109/ACCESS.2022.3165809

[21] Neto ECP, Dadkhah S, Ferreira R, Zohourian A, Lu R, Ghorbani AA. CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment. *Sensors*. 2023;23(13):5941. Available from: https://doi.org/10.3390/s23135941

[22] Layeghy S, Portmann M. Explainable cross-domain evaluation of ML-based network intrusion detection systems. *Computers & Electrical Engineering*. 2023;108:108692. Available from: https://doi.org/10.1016/j.compeleceng.2023.108692

[23] Alani MM, Miri A. Towards an Explainable Universal Feature Set for IoT Intrusion Detection. *Sensors*. 2022;22(15):5690. Available from: https://doi.org/10.3390/s22155690

[24] Friha O, Ferrag MA, Shu L, Maglaras L, Choo K-KR, Nafaa M. FELIDS: Federated learning-based intrusion detection system for agricultural Internet of Things. *Journal of Parallel and Distributed Computing*. 2022;165:17–31. Available from: https://doi.org/10.1016/j.jpdc.2022.03.003

[25] Altunay HC, Albayrak Z. A hybrid CNN+LSTM-based intrusion detection system for industrial IoT networks. *Engineering Science and Technology, an International Journal*. 2023;38:101322. Available from: https://doi.org/10.1016/j.jestch.2022.101322

[26] Saba T, Rehman A, Sadad T, Kolivand H, Bahaj SA. Anomaly-based intrusion detection system for IoT networks through deep learning model. *Computers & Electrical Engineering*. 2022;99:107810. Available from: https://doi.org/10.1016/j.compeleceng.2022.107810

[27] Naz N, Khan MA, Alsuhibany SA, Diyan M, Tan Z, Khan MA, et al. Ensemble learning-based IDS for sensors telemetry data in IoT networks. *Mathematical Biosciences and Engineering*. 2022;19(10):10550–10580. Available from: https://doi.org/10.3934/mbe.2022493

[28] Liu L, Engelen G, Lynar T, Essam D, Joosen W. Error Prevalence in NIDS Datasets: A Case Study on CIC-IDS-2017 and CSE-CIC-IDS-2018. In: *Proceedings of the 2022 IEEE Conference on Communications and Network Security (CNS)*. IEEE; 2022. p. 254–262. Available from: https://doi.org/10.1109/CNS56114.2022.9947235

[29] Guerra JL, Catania C, Veas E. Datasets are not enough: Challenges in labeling network traffic. *Computers & Security*. 2022;120:102810. Available from: https://doi.org/10.1016/j.cose.2022.102810

[30] Apruzzese G, Laskov P, Montes de Oca E, Mallouli W, Brdalo Rapa L, Grammatopoulos AV, et al. The Role of Machine Learning in Cybersecurity. *Digital Threats: Research and Practice*. 2023;4(1):1–38. Available from: https://doi.org/10.1145/3545574

[31] Andresini G, Pendlebury F, Pierazzi F, Loglisci C, Appice A, Cavallaro L. INSOMNIA: Towards Concept-Drift Robustness in Network Intrusion Detection. In: *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security*. ACM; 2021. p. 111–122. Available from: https://doi.org/10.1145/3474369.3486864

[32] Whalen S, Schreiber J, Noble WS, Pollard KS. Navigating the pitfalls of applying machine learning in genomics. *Nature Reviews Genetics*. 2022;23(3):169–181. Available from: https://doi.org/10.1038/s41576-021-00434-9

[33] Rosenblatt M, Tejavibulya L, Jiang R, Noble WS, Scheinost D. Data leakage inflates prediction performance in connectome-based machine learning models. *Nature Communications*. 2024;15(1):1829. Available from: https://doi.org/10.1038/s41467-024-46150-w

[34] Geirhos R, Jacobsen J-H, Michaelis C, Zemel R, Brendel W, Bethge M, et al. Shortcut learning in deep neural networks. *Nature Machine Intelligence*. 2020;2(11):665–673. Available from: https://doi.org/10.1038/s42256-020-00257-z

[35] Verkerken M, D'hooge L, Wauters T, Volckaert B, De Turck F. Towards Model Generalization for Intrusion Detection: Unsupervised Machine Learning Techniques. *Journal of Network and Systems Management*. 2022;30(1):12. Available from: https://doi.org/10.1007/s10922-021-09615-7

[36] Hung R-J. An Explainable XGBoost-Based Framework for IoT Attack Detection with Unseen Attack Family Evaluation. *Sensors*. 2026;26(10):3005. Available from: https://doi.org/10.3390/s26103005

[37] Moustafa N. ToN_IoT Datasets [dataset]. *IEEE DataPort*. 2019 Oct 16. Available from: https://doi.org/10.21227/fesz-dm97

[38] Shwartz-Ziv R, Armon A. Tabular data: Deep learning is not all you need. *Information Fusion*. 2022;81:84–90. Available from: https://doi.org/10.1016/j.inffus.2021.11.011

[39] Neupane S, Ables J, Anderson W, Mittal S, Rahimi S, Banicescu I, et al. Explainable Intrusion Detection Systems (X-IDS): A Survey of Current Methods, Challenges, and Opportunities. *IEEE Access*. 2022;10:112392–112415. Available from: https://doi.org/10.1109/ACCESS.2022.3216617

[40] Moustafa N, Koroniotis N, Keshk M, Zomaya AY, Tari Z. Explainable Intrusion Detection for Cyber Defences in the Internet of Things: Opportunities and Solutions. *IEEE Communications Surveys & Tutorials*. 2023;25(3):1775–1807. Available from: https://doi.org/10.1109/COMST.2023.3280465

[41] Le T-T-H, Kim H, Kang H, Kim H. Classification and Explanation for Intrusion Detection System Based on Ensemble Trees and SHAP Method. *Sensors*. 2022;22(3):1154. Available from: https://doi.org/10.3390/s22031154