Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Cross-Domain Generalization Failure in Lightweight Intrusion Detection Models for IIoT Networks

Md Azizul Hakim

azizulhakim8291@gmail.com

Bangladesh Sweden Polytechnic Institute https://orcid.org/0009-0008-7738-6200

Md Shihab Uddin

Bangladesh Sweden Polytechnic Institute https://orcid.org/0009-0004-8088-1510

Talha Ibne Anich

Bangladesh Sweden Polytechnic Institute https://orcid.org/0009-0004-9090-2274

Research Article

Keywords: Intrusion Detection System, Industrial Internet of Things, Cross-Domain Generalization, Lightweight Machine Learning, Explainability, Adversarial Robustness, Class Imbalance, Edge Deployment

Posted Date: July 2nd, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10207045/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Cross-Domain Generalization Failure in Lightweight Intrusion Detection Models for IIoT Networks

MD Azizul Hakim[a,*], Md Shihab Uddin[b] and Talha Ibne Anich[c]

[a]Department of Computer Science and Technology, Bangladesh Sweden Polytechnic Institute, Kaptai, Rangamati, Chittagong, 4530, Bangladesh
[b]Department of Electrical Engineering, Bangladesh Sweden Polytechnic Institute, Kaptai, Rangamati, Chittagong, 4530, Bangladesh
[c]Department of Computer Science and Technology, Bangladesh Sweden Polytechnic Institute, Kaptai, Rangamati, Chittagong, 4530, Bangladesh

ARTICLE INFO

Keywords:
Intrusion Detection System
Industrial Internet of Things
Cross-Domain Generalization
Lightweight Machine Learning
Explainability
Adversarial Robustness
Class Imbalance
Edge Deployment

ABSTRACT

Lightweight machine learning models are increasingly proposed for intrusion detection in Industrial Internet of Things (IIoT) networks due to their suitability for resource-constrained edge deployment. However, most reported results evaluate these models only within their training network, leaving behavior on unseen industrial networks largely unverified. This study trains four representative lightweight architectures on one widely used IIoT dataset and evaluates them, without retraining, on two independent and structurally distinct IIoT datasets, using a feature representation restricted to attributes available across all three sources. Explainability analysis, corroborated across two architecturally distinct top-performing models, shows that both rely overwhelmingly on coarse port-category features, and a direct comparison of port-category prevalence across datasets reveals that the most influential category occurs in source-domain attack traffic at 96 to 435 times the rate observed in the two target domains, indicating that coarsening port resolution relocates rather than removes a documented shortcut. Evaluation under naturally imbalanced class distributions, rather than the balanced distributions common in prior work, reveals a further effect: the evaluation protocol used can reverse which target network appears to pose the greater generalization challenge. Adversarial robustness and the capacity to recover performance through limited exposure to target-domain data are also assessed; robustness to adversarial perturbation is found to be unrelated to a model's ability to generalize across networks, and recovery through limited adaptation varies considerably by architecture. These findings suggest that deployment readiness for lightweight IIoT intrusion detection should be assessed using cross-network evaluation under realistic class distributions, rather than within-domain accuracy alone.

1. Introduction

Industrial Internet of Things (IIoT) networks now connect sensors, controllers, and actuators across manufacturing, energy, and critical infrastructure systems, creating an attack surface that conventional, server-grade intrusion detection cannot always reach. This has driven sustained interest in *lightweight* machine learning models for intrusion detection: classifiers small enough to run on constrained edge hardware, where computation, memory, and energy budgets are limited. A growing body of work reports such models achieving near-perfect detection accuracy on standard IIoT traffic benchmarks, suggesting that the problem of resource-efficient intrusion detection is largely solved.

This impression rests on a narrow form of evaluation. In almost all published studies, a model is trained and tested on data drawn from the same capture, often a single split of one dataset collected on one network. Real IIoT deployments do not work this way. A model trained on one industrial network is routinely expected to operate on another, with different devices, different traffic patterns, and different attacker behavior. Whether the accuracy reported under same-dataset evaluation has any bearing on this realistic deployment scenario is rarely tested, and almost never tested for the specific class of lightweight models that edge deployment requires.

This gap motivates a direct question: *do lightweight intrusion detection models that perform well within their training network retain that performance when applied, without retraining, to a different IIoT network?* A closely related question follows immediately: if performance does not transfer, is the cause identifiable, and can it be addressed cheaply, through a small amount of local adaptation, rather than full retraining?

This paper answers both questions through a systematic cross-dataset study. Four representative lightweight architectures, a decision tree Loh (2011) and three compact neural network designs, are trained on one IIoT dataset and evaluated, without retraining, on two independent IIoT datasets that differ in source network, device composition, and traffic capture format. To isolate the cause of any performance loss, feature reliance is examined directly through explainability analysis, and a controlled ablation tests whether reducing the resolution of the most influential feature removes the problem. Evaluation is repeated under both artificially balanced and naturally imbalanced class distributions, since the former is the de facto standard in prior work but does not reflect deployment conditions. Adversarial robustness is assessed independently of cross-dataset performance, and a final experiment measures whether fine-tuning on a small,

*Corresponding author
azizulhakim8291@gmail.com (M.A. Hakim); shihab9586@gmail.com (M.S. Uddin); mail.talhaibneanich@gmail.com (T.I. Anich)
ORCID(s): 0009-0008-7738-6200 (M.A. Hakim); 0009-0004-8088-1510 (M.S. Uddin); 0009-0004-9090-2274 (T.I. Anich)

labeled sample from the target network can recover what zero-shot transfer loses.

The contribution of this work is fourfold:

- It provides direct evidence, across two independent target datasets, that strong within-domain performance for lightweight IIoT intrusion detection models does not imply cross-network performance Latha and Bommi (2023).
- It shows, through explainability analysis, a targeted ablation, and a direct cross-dataset comparison of port-bucket prevalence, that coarsening port resolution into broad categorical buckets rules out raw port memorization as the cause of failure, but that reliance shifts to the coarser port-bucket categories themselves; the most influential of these categories occurs in source-domain attack traffic at approximately 96 times the rate observed in Gotham's attack traffic and 435 times the rate in WUSTL-IIoT-2021's, confirming that reducing feature granularity relocates this shortcut rather than removing it.
- It shows that evaluation under artificially balanced class distributions, common in prior work, substantially misrepresents performance relative to the natural, imbalanced conditions a deployed system would encounter.
- It demonstrates that the capacity to recover performance through small-scale fine-tuning on target-network data is architecture-dependent, rather than a property of lightweight models in general.

Together, these results indicate that the readiness of a lightweight intrusion detection model for IIoT deployment cannot be inferred from within-domain accuracy, and that cross-network evaluation should be treated as a standard requirement rather than an optional extension.

2. Related Work

2.1. Lightweight Intrusion Detection for IIoT

A substantial body of work targets intrusion detection models small enough for edge deployment in IIoT settings. Ferrag et al. introduced Edge-IIoTset Ferrag et al. (2022), a testbed-generated dataset spanning fourteen attack categories across IoT and IIoT devices, and reported high detection accuracy using a range of classifiers, evaluated entirely within the dataset's own train/test split. Zolanvari et al. introduced the WUSTL-IIoT line of datasets through a vulnerability-analysis study of industrial IoT networks Zolanvari et al. (2019), which has since been used by numerous lightweight-classifier studies, again under single-dataset evaluation. Belarbi et al. released the Gotham dataset Belarbi et al. (2025), a large-scale, reproducible capture of real heterogeneous IoT device traffic, intended in part to address the lack of diversity in existing IoT security benchmarks. These three datasets, used respectively as the training domain and two evaluation domains in the present study, were each introduced with in-domain results only; none of their originating publications report performance when a model trained on one is tested on another.

2.2. Cross-Dataset Generalization in Intrusion Detection

A smaller, more recent line of work has begun to test whether intrusion detection performance survives a change of network. Cantone et al. Cantone et al. (2024) trained four classifiers on one enterprise network dataset and evaluated them on three independent variants, reporting near-perfect within-dataset accuracy collapsing to chance-level performance across datasets, and attributed the collapse to dataset-specific anomalies identified through visualization rather than to a single confounding feature. D'Hooge et al. D'hooge et al. (2019) reported a comparable failure restricted to two attack categories across two enterprise datasets, concluding that the classifiers tested could not operate reliably on traffic from a network unseen at training time. Verkerken et al. Verkerken et al. (2022) extended this observation to unsupervised models, reporting a substantial average drop in detection performance under cross-dataset evaluation relative to within-dataset evaluation. Apruzzese, Pajola, and Conti Apruzzese et al. (2022) proposed XeNIDS, a structured framework for cross-evaluating ML-based network intrusion detection systems across multiple datasets, showing that cross-evaluation can both extend a model's effective detection surface at no additional labeling cost and expose risks that single-dataset evaluation conceals. Layeghy and Portmann Layeghy and Portmann (2023) combined explainability analysis with cross-domain NIDS evaluation, the closest existing work to the present study in spirit; their analysis identifies feature-importance shifts across datasets but does not test a targeted ablation isolating a single confounding feature, compare balanced against natural class distributions, or assess few-shot recovery. These studies establish that cross-dataset failure is a recurring property of network intrusion detection generally; none of them isolate lightweight architectures specifically, and none test IIoT datasets, which differ from the enterprise network traffic these studies examine in device composition, protocol mix, and traffic volume.

A second, closely related line of work has identified the specific mechanism most often blamed for inflated intrusion detection performance: reliance on fields that identify a flow's origin rather than its behavior. Engelen et al. Engelen et al. (2021) showed that classifiers trained on a widely used benchmark dataset could achieve high accuracy using destination port alone, demonstrating that apparent detection performance can reflect a learned association between an attack's source and a fixed port number rather than any property of malicious traffic itself. This finding has informed subsequent dataset-design work; a recent study on IoT intrusion detection explicitly removed IP- and port-based identifiers from its feature set to obtain a more behavior-driven, and presumably more transferable, representation Dharini

et al. (2026). The present study tests this mechanism directly, rather than designing around it: feature reliance is measured before and after reducing port resolution, and cross-dataset performance is compared under both conditions, allowing the contribution of this specific, previously documented shortcut to be isolated from whatever remains.

A small number of more recent studies have moved cross-dataset evaluation specifically into the IIoT/IoT setting. Bilal et al. Bilal et al. (2026) evaluated federated intrusion detection models across three IIoT datasets, including Edge-IIoTset, reporting macro-F1 losses of up to thirty percentage points when a model trained on one dataset is tested on another under a single-source, non-federated condition. This study excludes IP addresses and port identifiers from its feature set entirely by design, so it does not test whether port reliance contributes to the observed transfer loss, and its architectures are evaluated within a federated aggregation process rather than in isolation, which prevents attributing generalization behavior to any single lightweight model. This confirms that cross-dataset failure extends to IIoT data specifically, strengthening rather than undermining the motivation for the present work; it does not, however, isolate architecture-level behavior, test the port-shortcut hypothesis directly, compare balanced against naturally imbalanced evaluation, or assess few-shot adaptation.

2.3. Adversarial Robustness of Intrusion Detection Models

Separately from generalization, the robustness of intrusion detection classifiers to adversarially perturbed input has been studied as a deployment risk in its own right. Vitorino et al. Vitorino et al. (2023) examined the vulnerability of IoT intrusion detection models, including tree-based classifiers, to realistic adversarial perturbation, reporting substantial degradation under attack. Debicha et al. Debicha et al. (2023) surveyed evasion attacks and defenses for network intrusion detection broadly, noting that robustness is rarely evaluated alongside generalization in the same study. The present work follows this observation by assessing adversarial robustness independently from, but alongside, cross-dataset performance for each architecture.

2.4. Domain Adaptation and Few-Shot Recovery

Where generalization failure has been observed, a parallel body of work has examined whether it can be mitigated without full retraining. Singla et al. Singla et al. (2020) used adversarial domain adaptation to prepare intrusion detection models with minimal labeled target-domain data, reporting recovered performance using a small fraction of the data otherwise required. Kheddar et al. Kheddar et al. (2023) reviewed deep transfer learning approaches for intrusion detection in industrial control networks, covering a range of adaptation strategies but noting limited comparison of how adaptability varies across lightweight model architectures specifically. The present study addresses this directly by applying the same fine-tuning procedure, under matched conditions, to all four evaluated architectures.

2.5. Summary of the Gap

Across these bodies of work, no study combines (i) lightweight IIoT-specific architectures evaluated in isolation, (ii) more than one independent target dataset, (iii) a direct test of the port-shortcut mechanism rather than an assumption that it explains the failure or a design choice to remove it by default, (iv) a comparison of evaluation under balanced versus naturally imbalanced class distributions, and (v) a matched few-shot adaptation test across all evaluated architectures, as summarized in Table 1. This study is constructed to close that combined gap rather than any single piece of it in isolation.

Table 1: Coverage of related cross-dataset IDS studies against the five criteria this paper combines: (i) lightweight architectures evaluated in isolation, (ii) more than one independent target dataset, (iii) a direct test of the port-shortcut mechanism, (iv) balanced vs. natural class-distribution comparison, (v) matched few-shot adaptation across architectures.

Study	(i)	(ii)	(iii)	(iv)	(v)
Cantone et al. Cantone et al. (2024)	–	✓	–	–	–
D'Hooge et al. D'hooge et al. (2019)	–	–	–	–	–
Verkerken et al. Verkerken et al. (2022)	–	–	–	–	–
Apruzzese et al. Apruzzese et al. (2022)	–	✓	–	–	–
Layeghy & Portmann Layeghy and Portmann (2023)	–	✓	–	–	–
Bilal et al. Bilal et al. (2026)	–	✓	–	–	–
This work	✓	✓	✓	✓	✓

3. Datasets and Feature Engineering

Three IIoT/IoT network traffic datasets are used: **Edge-IIoTset** Ferrag et al. (2022), the training domain, comprising fourteen attack categories across IoT and IIoT testbed devices; **Gotham 2025** Belarbi et al. (2025), a reproducible capture of real per-device IoT traffic across 104 devices, used as the first cross-domain target; and **WUSTL-IIoT-2021** Ahuja et al. (2021), an Argus flow-export capture from an industrial SCADA testbed, used as the second cross-domain target. The three datasets were collected with different tools, at different layers (packet versus flow), and expose different native fields, so no raw feature is available in all three.

A minimal *common schema* was constructed instead: source/destination port, transport protocol, and four TCP flag bits (FIN, SYN, RST, ACK). Raw port numbers were not used directly. Prior work has shown that destination port alone can yield high detection accuracy on benchmark intrusion datasets Engelen et al. (2021), because port values often identify a flow's dataset of origin rather than its behavior. To prevent this shortcut and obtain a feature representation with a chance of transferring across networks, each port is instead mapped to one of four categorical buckets, *none*, *well-known* ($<$ 1024), *registered* (1024–49151), or *dynamic* ($\geq$ 49152), and one-hot encoded Seger (2018) separately for source and destination. Protocol is one-hot encoded into TCP, UDP, ICMP, or OTHER, the last capturing any protocol value not matching the first three (e.g., GRE, ESP, or unparsed/missing protocol fields). This

Table 2
Dataset summary. Source/target role, size after capping, and class balance.

	Edge-IIoTset	Gotham 2025	WUSTL-IIoT-2021
Role	Source (train)	Target 1	Target 2
Capture type	Packet-level	Packet-level	Flow-level (Argus)
Devices/files	1	104	1
n (balanced eval)	299,461	118,444	236,959
n (natural eval)	—	189,396	299,832
Attack share (natural)	—	10.6%	7.3%
TCP flag bits available	Yes	Yes	No

yields a 16-dimensional schema (4 flag bits + 8 port-bucket indicators + 4 protocol indicators), identical across all three datasets despite their different native formats. WUSTL-IIoT-2021's Argus export exposes no TCP flag information; the four flag features are zero-filled for this dataset, a disclosed limitation rather than a missing-data error. Attack labels are harmonized to a binary attack/benign distinction across all three datasets despite differing native taxonomies: Edge-IIoTset's fourteen named attack categories, Gotham's per-device anomaly labels, and WUSTL-IIoT-2021's binary `Target` field are each mapped to a single binary label, discarding multiclass distinctions for the cross-domain experiments (Section 5.4's multiclass analysis uses Edge-IIoTset's native categories directly, in-domain only) All models are trained once on Edge-IIoTset and evaluated, without retraining, on Gotham and WUSTL-IIoT-2021. Each target dataset is evaluated twice: once under class-balanced sampling, the convention in most prior cross-dataset studies, and once under its natural, heavily benign-skewed class distribution, to test whether balanced evaluation overstates real-world performance. Table 2 summarizes role, size, and class balance for all three.

4. Methodology

4.1. Model Architectures

Four lightweight classifiers are trained on the 16-dimensional common schema (Section 3): **DecisionTree** Loh (2011), a depth-limited tree (max_depth = 10); **SmallMLP** Taud and Mas (2017), a three-layer feed-forward network ($16 \to 16 \to 8 \to 2$, ReLU); **Small1DCNN** Ige and Sibiya (2024), two 1D convolutional layers (8 and 16 channels, kernel size 3) followed by global average pooling and a linear classifier, with no normalization layers; and **SmallLSTM** Graves (2012), a single-layer LSTM (hidden size 16) reading each feature vector as a length-16 sequence of scalars, followed by a linear classifier on the final hidden state. This treats the 16-dimensional feature vector as a sequence purely to obtain a lightweight recurrent baseline; no temporal ordering between flows is assumed or exploited, and the choice trades a small increase in parameter count (Table 3, Section 5.7) for a recurrent inductive bias as a point of comparison against the feed-forward and convolutional architectures. All three neural architectures are trained with the Adam optimizer minimizing class-weighted cross-entropy, using a learning rate of 1×10^{-3}, batch size 256, for 15 epochs.

$$\mathcal{L} = -\sum_{i=1}^{N} w_{y_i} \log \hat{p}_{i,y_i}, \tag{1}$$

where $\hat{p}_{i,y_i}$ is the predicted probability of the true class y_i for sample i, and w_{y_i} is the inverse class frequency, set to 1 for all classes during initial training and re-computed per fine-tuning batch during few-shot adaptation (Section 4.7).

4.2. In-Domain and Cross-Domain Evaluation

Edge-IIoTset is split 70/10/20 into train, validation, and test sets, stratified by label. Each model is trained once on the training split and evaluated on the held-out test split (in-domain). The same trained models, without any further parameter updates, are then evaluated on Gotham and on WUSTL-IIoT-2021 (cross-domain), each under both a class-balanced sample and the dataset's natural class distribution (Table 2). All four conditions use the same `StandardScaler` (zero mean, unit variance), fit once on Edge-IIoTset's training split. Detection performance is reported as the F1 score van Rijsbergen (1979) on the attack class,

$$F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{2}$$

which, unlike accuracy, does not reward a classifier for defaulting to the majority (benign) class under the natural, imbalanced evaluation condition.

4.3. Efficiency Profiling

For each trained model, four lightweight-deployment metrics are recorded: serialized model size on disk, single-sample inference latency (mean, median, 95th and 99th percentile over 1,000 repeated calls), peak memory during inference, and parameter count (node count for DecisionTree, learnable parameter count for the neural models). Training wall-clock time is recorded separately for each architecture under identical hardware conditions.

4.4. Explainability

To diagnose what each model relies on, SHAP Lundberg and Lee (2017) values are computed for the best in-domain model (by F1) over a fixed sample of 200 test instances: `TreeExplainer` for DecisionTree, `KernelExplainer` for the three neural models. Mean absolute SHAP value per feature, averaged over the sample, ranks feature contribution to the attack-class prediction. Decision-tree feature importances (mean decrease in impurity) are reported as a second, model-native measure of feature reliance, to cross-check the SHAP ranking.

4.5. Adversarial Robustness

Robustness to evasion is tested with the HopSkipJump black-box attack Chen et al. (2020); Nicolae et al. (2018), applied identically to all four models (including DecisionTree, which gradient-based attacks such as FGSM cannot target). A fixed random sample of 100 in-domain test instances is perturbed

(max_iter = 20, max_eval = 500, init_eval = 100), and robustness is reported as the accuracy drop,

$$\Delta\text{Acc} = \text{Acc}_{\text{clean}} - \text{Acc}_{\text{adv}}, \tag{3}$$

between the model's accuracy on the clean sample and on its adversarially perturbed counterpart.

4.6. Statistical Validation

Two checks establish that the reported differences are not artifacts of a single train/test split. First, each architecture is retrained from scratch across five random seeds, and in-domain and cross-domain F1 are reported as mean ± standard deviation. Second, McNemar's test Pembury Smith and Ruxton (2020) is applied pairwise between all six model pairs, on identical test instances, separately for the in-domain condition and the cross-domain condition on Gotham under balanced evaluation (the same condition used for multi-seed validation, Section 5.3):

$$\chi^2 = \frac{(|b - c| - 1)^2}{b + c}, \tag{4}$$

where b and c are the off-diagonal counts of the 2×2 contingency table of correct/incorrect predictions for the two models being compared (continuity-corrected). A pair is reported as significantly different at $p < 0.05$.

4.7. Few-Shot Domain Recovery

To test whether the cross-domain gap can be closed cheaply, Gotham is split in half into a fixed adaptation pool and a fixed evaluation set, held constant across all conditions. For fractions $\{0, 0.01, 0.05, 0.10, 0.25\}$ of the adaptation pool, each model is updated using only that fraction of labeled target-domain data: DecisionTree Loh (2011) is refit on the union of its original training data and the sampled fraction; the three neural models are fine-tuned for 5 additional epochs at one-tenth the original learning rate, with class weights recomputed on the fine-tuning batch (Eq. 1) to counter the class imbalance typical of small samples. This procedural difference is intentional, reflecting how each model type is conventionally updated in practice, but it means DecisionTree's recovery reflects retraining on its full original training set plus the new sample, while the neural models' recovery reflects fine-tuning on the new sample alone; the two are not directly comparable as a controlled test of architecture alone. F1 on the fixed evaluation set (Eq. 2) is tracked across all five fractions, including the fraction = 0 zero-shot baseline, for a direct, fraction-by-fraction recovery curve.

5. Results

Results are organized to separate what each model achieves from why. Section 5.1 establishes that all four architectures fit the training domain equally well, removing architecture choice as a source of the differences that follow. Section 5.2 reports the central finding: detection performance collapses when each model, without retraining, is applied to traffic from a different IIoT network, and this collapse holds under two independent target datasets and under both balanced and naturally imbalanced evaluation. Sections 5.3 through 5.8 then test, in turn, whether this collapse is statistically reliable, what each model relies on to make its decisions, how robust each model is to adversarial perturbation, whether the gap can be closed with a small amount of target-domain data, what each architecture costs to deploy, and how that cost compares to published lightweight intrusion detection systems. Table 3 reports exact in-domain and cross-domain figures throughout, and Table 4 reports the size comparison in Section 5.8; Figures 1, 2, and 3 each visualize a finding not otherwise shown.

5.1. In-Domain Performance

All four architectures achieve closely matched, strong performance on the Edge-IIoTsetFerrag et al. (2022) test split (Table 3, top rows). F1 ranges from 0.971 to 0.972 across all four models, with precision consistently above 0.99 and recall consistently above 0.95. No model separates meaningfully from the others under this condition: the gap between the strongest and weakest in-domain F1 is 0.001, well within the range expected from a single train/test split alone. At the level of individual predictions, DecisionTree Loh (2011) and SmallMLP Taud and Mas (2017) disagree on only 5 of 59,893 in-domain test instances.

This near-identical performance is itself a methodological outcome, not an incidental one. Under the raw, unbucketed feature representation used in early experimentation, DecisionTree and SmallLSTM Graves (2012) substantially outperformed SmallMLP and Small1DCNN Ige and Sibiya (2024) in-domain, a gap traceable to the high-resolution port values both models could exploit. Once ports are reduced to the four coarse buckets described in Section 3, that gap disappears: all four architectures converge to comparable performance on the same, coarser feature set. The architectures examined here therefore do not differ meaningfully in their capacity to fit Edge-IIoTset; whatever differences emerge in subsequent sections arise from how each architecture behaves once that fit is tested elsewhere.

5.2. Cross-Domain Performance

Without any retraining, the same four models applied to Gotham dataset Belarbi et al. (2025) and to WUSTL-IIoT-2021 Ahuja et al. (2021) show a substantial decline in detection performance relative to the in-domain results of Section 5.1 (Table 3, middle and lower rows). Under the natural class distribution, F1 falls from approximately 0.97 in-domain to between 0.18 and 0.28 on Gotham, and to between 0.09 and 0.13 on WUSTL-IIoT-2021 (Fig. 1). The decline is consistent across all four architectures and both target datasets; under natural-distribution evaluation, no model retains more than 29% of its in-domain F1 once applied outside its training network.

Under the natural class distribution, the decline is larger on WUSTL-IIoT-2021 than on Gotham for every model. WUSTL-IIoT-2021 differs from Edge-IIoTset in capture method (flow-level Argus export versus packet-level capture) and lacks TCP flag information entirely (Section 3),

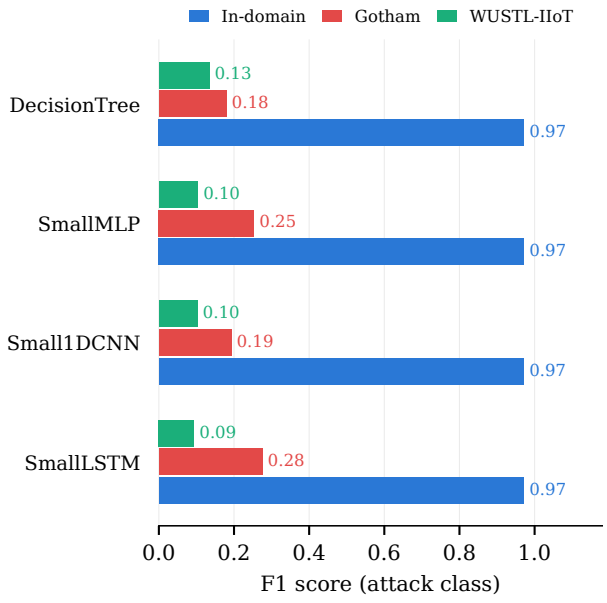


Figure 1: In-domain F1 versus cross-domain F1 (natural class distribution) for all four models on both target datasets. Cross-domain performance collapses uniformly relative to in-domain performance, and the collapse is larger on WUSTL-IIoT-2021 than on Gotham for every model.

whereas Gotham retains the same packet-level fields as the source domain. The larger drop on the more structurally dissimilar target is consistent with the failure reflecting a loss of transferable signal as the network and capture conditions diverge from the training domain, rather than a fixed quantity independent of how different the target network is.

Comparing balanced and natural evaluation reveals a second, independent finding. On WUSTL-IIoT-2021, balanced-distribution F1 ranges from 0.39 to 0.53, while natural-distribution F1 for the same models falls to 0.09–0.13, a fourfold reduction. The effect is smaller but still present on Gotham (e.g. SmallLSTM: 0.284 balanced versus 0.275 natural). Precision is the metric most affected: under the natural distribution, where attack traffic is rare (7.3% on WUSTL-IIoT-2021, 10.6% on Gotham), false positives accumulate against a much larger pool of benign traffic, driving precision to 0.050–0.072 for every model on WUSTL-IIoT-2021 . Recall is comparatively stable between the two evaluation conditions for every model, indicating that the models retain a similar ability to flag attacks once outside their training domain, but the practical cost of that detection, in false alarms relative to true ones, is substantially understated by balanced evaluation alone. This evaluation choice does more than understate the cost of detection: it can reverse which target dataset appears harder. Under balanced evaluation, every model scores higher on WUSTL-IIoT-2021 than on Gotham (e.g. DecisionTree: 0.530 versus 0.171); under natural evaluation, this ordering flips, and every model scores lower on WUSTL-IIoT-2021 than on Gotham (e.g. DecisionTree: 0.134 versus 0.180). Which target network appears to pose the greater generalization challenge therefore depends on the evaluation protocol used, not only on the network itself.

Among the four architectures, SmallLSTM attains the highest cross-domain F1 on Gotham (0.275 natural) but the lowest on WUSTL-IIoT-2021 (0.093 natural); DecisionTree shows the reverse pattern, weakest on Gotham (0.180) but strongest on WUSTL-IIoT-2021 (0.134). No single architecture dominates across both target datasets, indicating that cross-domain ranking is not fixed but depends on which network the model is deployed to.

5.3. Statistical Validation

Two checks test whether the results in Sections 5.1 and 5.2 reflect a stable property of each model rather than an artifact of one particular train/test split.

Repeating training across five random seeds leaves the central finding unchanged. In-domain F1 is highly stable for every model (standard deviation $\leq$ 0.002 for three of the four architectures, 0.002 for Small1DCNN), confirming that the near-identical in-domain performance reported in Section 5.1 is not a coincidence of one split. Cross-domain F1 (Gotham, balanced distribution) is also stable for three of the four models (DecisionTree: 0.179 ± 0.011; SmallMLP: 0.252 ± 0.020; SmallLSTM: 0.254 ± 0.021), confirming that the collapse reported in Section 5.2 holds across seeds rather than reflecting one unfavorable split. Small1DCNN's cross-domain F1 (same condition) is noticeably less stable across seeds (0.277 ± 0.106), indicating that its cross-domain behavior, while consistently poor, is also the least predictable of the four architectures from one training run to the next.

McNemar's test Pembury Smith and Ruxton (2020), applied pairwise on identical test instances, gives a more cautious picture for in-domain comparisons than the raw F1 values alone might suggest. Despite an in-domain F1 spread of only 0.004 across all four models (Table 3), three of the six pairwise comparisons are statistically significant at $p < 0.05$: DecisionTree differs significantly from both Small1DCNN and SmallLSTM, and SmallMLP differs significantly from SmallLSTM. The remaining three pairs, including DecisionTree versus SmallMLP, are not significant. This is consistent with what the in-domain test set size implies: with roughly 60,000 test instances, even small, consistent differences in per-sample correctness can reach statistical significance without corresponding to a difference of practical consequence. The significant pairs should accordingly be read as evidence that small in-domain differences are not due to chance, not as evidence that any one architecture is meaningfully better suited to the in-domain task than another.

For cross-domain predictions, all six pairwise comparisons are significant at $p < 0.05$, including pairs whose absolute F1 values are close, such as SmallMLP versus SmallLSTM (0.253 versus 0.275, natural distribution). Combined with the multi-seed results, this indicates that the differences in cross-domain F1 between architectures, while all of them poor in absolute terms, are systematic rather than incidental: a given architecture's specific degree of cross-domain failure

Table 3
In-domain and cross-domain detection performance. Cross-domain results are reported under both balanced and natural class distributions.

Condition	Model	Acc.	Prec.	Rec.	F1
In-domain	DecisionTree	0.972	0.993	0.951	0.972
	SmallMLP	0.972	0.993	0.951	0.972
	Small1DCNN	0.972	0.994	0.950	0.971
	SmallLSTM	0.972	0.993	0.950	0.971
Cross-domain: Gotham (balanced)	DecisionTree	0.441	0.101	0.563	0.171
	SmallMLP	0.667	0.170	0.581	0.263
	Small1DCNN	0.617	0.124	0.451	0.194
	SmallLSTM	0.686	0.185	0.609	0.284
Cross-domain: Gotham (natural)	DecisionTree	0.449	0.107	0.567	0.180
	SmallMLP	0.634	0.162	0.582	0.253
	Small1DCNN	0.593	0.122	0.455	0.192
	SmallLSTM	0.658	0.178	0.612	0.275
Cross-domain: WUSTL-IIoT (balanced)	DecisionTree	0.363	0.363	0.976	0.530
	SmallMLP	0.277	0.302	0.741	0.429
	Small1DCNN	0.278	0.300	0.723	0.424
	SmallLSTM	0.247	0.278	0.659	0.391
Cross-domain: WUSTL-IIoT (natural)	DecisionTree	0.078	0.072	0.976	0.134
	SmallMLP	0.062	0.056	0.743	0.104
	Small1DCNN	0.071	0.055	0.726	0.102
	SmallLSTM	0.056	0.050	0.665	0.093

is a reproducible property of that architecture on a given target dataset, not noise around a single common failure rate.

5.4. Feature Reliance

SHAP values are computed for both DecisionTree and SmallMLP, the two best in-domain models at full precision (F1 = 0.97155 and 0.97146 respectively, a difference within the multi-seed noise floor reported in Section 5.3), to confirm the port-bucket reliance finding is not an artifact of an arbitrary tie-break (Fig. 2). Port-bucket indicators dominate for both models, though the specific feature relied on most differs: DecisionTree's top feature is `dst_port_none`, while SmallMLP's is `dst_port_dynamic`. Across both models, the five highest-ranked features are entirely port-bucket indicators, accounting for 83% of total importance for DecisionTree and 73% for SmallMLP. TCP flags and protocol contribute comparatively little, and five of the sixteen features, `src_port_registered`, `dst_port_registered`, `protocol_UDP`, `protocol_ICMP`, and `protocol_OTHER`, receive zero importance. The model's native Gini-based feature importances agree: individually, `src_port_wellknown` (0.366), `dst_port_wellknown` (0.298), and `dst_port_none` (0.270) account for 93% of total split importance between them.

This concentration explains the protocol features' negligible contribution. Edge-IIoTset's training traffic is 88.4% TCP and 11.5% ICMP, with effectively no UDP; protocol carries almost no discriminative signal despite ICMP comprising 11.5% of training traffic, indicating that protocol type does not correlate strongly with the attack label in this dataset, rather than reflecting an absence of protocol variation. Whether this reliance transfers can be tested directly by comparing port-bucket prevalence on attack traffic across all three datasets. `dst_port_none`, DecisionTree's top feature by SHAP value, is present in 40.5% of Edge-IIoTset's attack traffic, but only 0.42% of Gotham's and 0.09% of WUSTL-IIoT-2021's, a 96-fold and 435-fold difference respectively. The features each model has learned to treat as its strongest indicator of an attack are, in both target networks, almost entirely absent from attack traffic, regardless of which specific port-bucket category each model happens to rely on most. Port-bucket reliance was the explicit target of the ablation in Section 3: raw port values were replaced with four coarse buckets specifically because high-resolution ports are a documented source of dataset-specific shortcut learning Engelen et al. (2021). Figure 2 shows that this reliance was reduced, not removed. The model still depends overwhelmingly on which port bucket a flow falls into, just at a coarser resolution than the original numeric value. The cross-domain collapse reported in Section 5.2 therefore cannot be attributed to raw port memorization alone, since that specific mechanism was already constrained; a port-bucket-level dependency of this magnitude is sufficient on its own to explain why performance does not transfer to networks with a different port-bucket distribution.

5.5. Adversarial Robustness

Robustness to evasion is assessed independently of cross-domain transfer, using the HopSkipJump black-box attack Chen et al. (2020) on 100 in-domain test instances per model (Eq. 3); clean accuracy here is computed on this same 100-instance subsample, which is why it differs slightly from the full-test-split in-domain accuracy of 0.972 reported

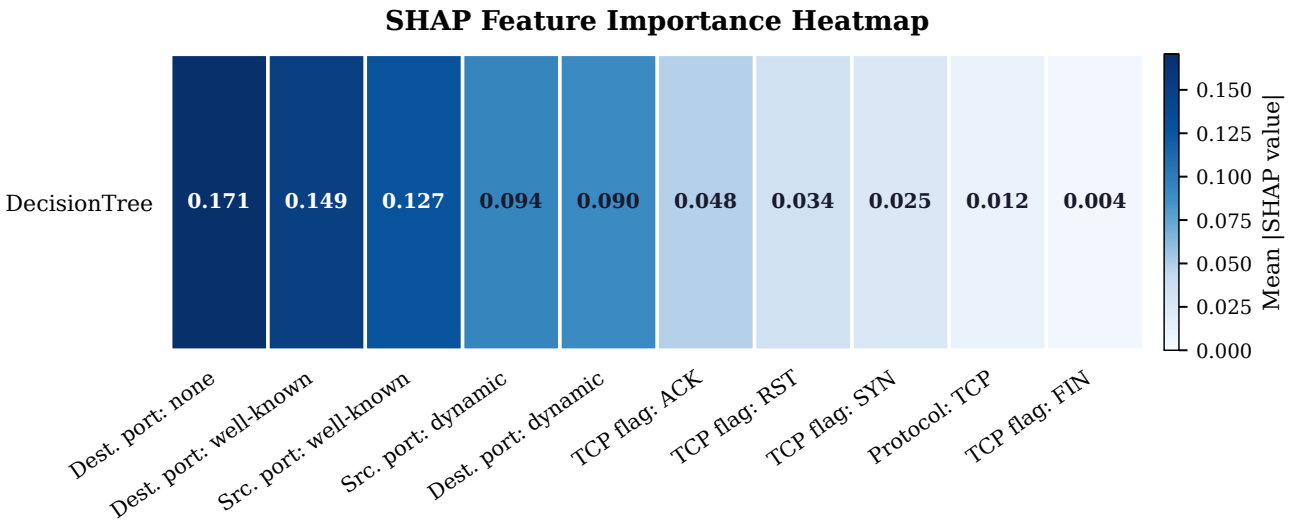


Figure 2: Mean absolute SHAP value per feature for DecisionTree, ranked by importance. The ten highest-importance features are shown, of which the top five account for 83% of total importance across all sixteen features; the remaining six features, including all five zero-importance features, are omitted. Port-bucket features dominate; protocol and TCP flag features contribute comparatively little. The same category-level pattern holds for SmallMLP (Section 5.4).

in Table 3. All four models start from comparable clean accuracy on this subsample (0.94–0.95) but diverge sharply once perturbed. DecisionTree and Small1DCNN degrade by 0.44–0.45, retaining roughly half their clean accuracy under attack. SmallMLP and SmallLSTM degrade by 0.88, falling to 0.06 accuracy, near the level of guessing a single class on a balanced sample.

This pattern does not track either in-domain performance or cross-domain performance: SmallMLP and SmallLSTM are among the strongest cross-domain performers (Section 5.2) yet the least robust to adversarial perturbation, while DecisionTree, weakest on Gotham, is comparatively robust. Generalization to a different network and robustness to an adversarial input are therefore distinct properties of these architectures, and improving one offers no indication of the other. Unlike the cross-domain and in-domain results above, this evaluation reflects a single 100-sample run without repetition across seeds; the relative ranking between models is consistent with the magnitude of the gap observed, but confidence intervals comparable to those reported elsewhere in this paper are not available for these estimates.

5.6. Few-Shot Domain Recovery

The zero-shot point on each recovery curve (fraction = 0) is evaluated on the fixed held-out half of the balanced Gotham sample reserved for few-shot evaluation, not the full Gotham sample used in Table 3; this is why, for example, DecisionTree's zero-shot F1 of 0.170 here differs slightly from its Gotham-balanced F1 of 0.171 in Table 3. Fine-tuning on a small sample of labeled Gotham data, following the procedure in Section 4.7, produces markedly different outcomes across the four architectures (Fig. 3). DecisionTree shows no improvement until 25% of the adaptation pool (14,805 samples), at which point F1 nearly quadruples relative to its zero-shot baseline, from 0.170 to 0.638. SmallLSTM improves immediately at 5% (2,961 samples), reaching 0.585, but then declines at 10% and 25%, ending at 0.429, below its own 5% peak. SmallMLP changes little before 10%, then rises gradually to 0.289 at 25%, a modest gain over its 0.264 baseline. Small1DCNN does not improve at any fraction tested; its few-shot F1 remains at or below its zero-shot value throughout.

No single relationship between data quantity and recovery holds across all four models. DecisionTree needs a comparatively large sample before any benefit appears; SmallLSTM benefits from a small sample but loses some of that benefit as more data is added; SmallMLP requires data but recovers only partially; Small1DCNN does not recover regardless of sample size. The capacity to close the cross-domain gap through limited local adaptation is therefore a property of the architecture being adapted, not solely a function of how much target-domain data is made available.

5.7. Efficiency Profiling

All four models remain within the resource bounds expected of edge deployment: model size ranges from 4.7 KB (SmallMLP) to 7.9 KB (SmallLSTM), and mean inference latency per sample ranges from 0.11 ms (SmallMLP) to 0.40 ms (SmallLSTM). Training cost differs by over two orders of magnitude (425×), from 0.20 s for DecisionTree to 85.0 s for SmallLSTM, reflecting the cost of gradient-based training Bengio (2012) relative to a single tree fit. None of these differences track the cross-domain or few-shot results of Sections 5.2 and 5.6: DecisionTree is both the cheapest to train and one of the two architectures that recovers well under few-shot adaptation, while SmallLSTM is the most expensive to train yet performs comparably or

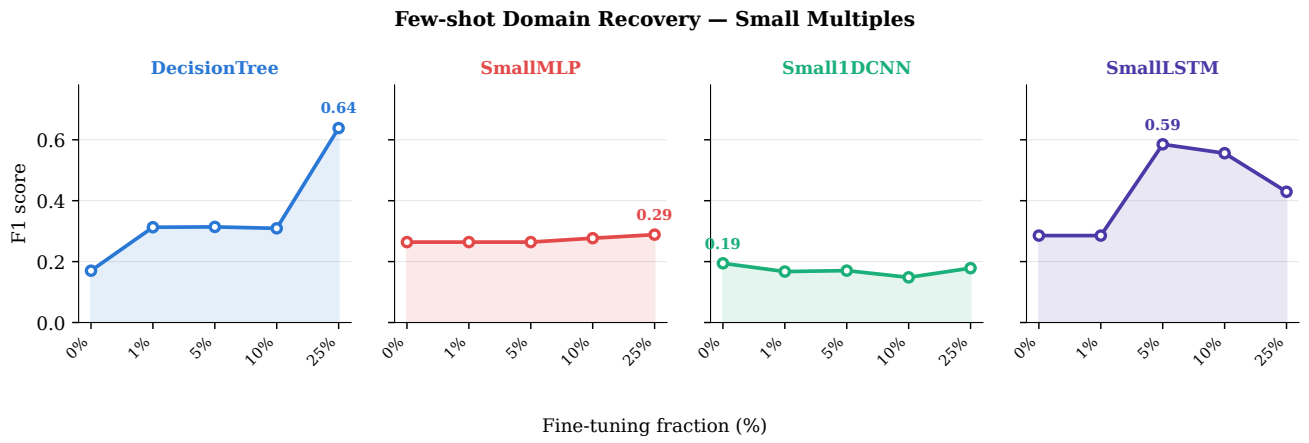


Figure 3: F1 on the Gotham evaluation set after fine-tuning each model on increasing fractions of labeled Gotham data. Recovery is architecture-dependent: DecisionTree and SmallLSTM show substantial gains, SmallMLP improves slowly, and Small1DCNN does not improve.

better under cross-domain transfer on Gotham; its few-shot recovery, by contrast, peaks early and declines (Section 5.6), ending well below DecisionTree's recovered performance. Efficiency and adaptability are independent considerations for model selection in this setting, not substitutes for one another.

5.8. Comparison to Published Lightweight IDS Footprints

Table 4 confirms that all four architectures evaluated here sit comfortably within typical edge-deployment budgets, alongside recent lightweight intrusion detection studies for IoT, IIoT, and resource-constrained edge settings more generally. The comparison is not like-for-like: the studies in Table 4 achieve compactness through compression, quantization, or distillation applied to a richer native feature set, while the smaller size reported here follows directly from the minimal 16-dimensional common schema used throughout this study (Section 3), not from compressing an otherwise larger model. One comparison point, DNN-KDQ Umar et al. (2025), targets edge deployment generally and is benchmarked on an enterprise network dataset rather than an IoT or IIoT capture; it is included as a edge-deployment efficiency reference rather than a domain-matched comparison. All models in Table 4, including those evaluated here, fall well below the 250–500 KB deployment budget commonly cited in the broader TinyML literature tin (2025); the differences within the table (4.7–39.9 KB) are accordingly a confirmation that all of these approaches are genuinely lightweight by that standard, rather than a finding with independent deployment consequences.

6. Discussion

The results in Section 5 support a specific, narrower claim than "lightweight models do not generalize." Generalization fails consistently and substantially across two independent IIoT datasets, but the mechanism behind that failure is now partially understood, and the failure is not uniform across architectures once adaptation is introduced.

6.1. Coarsening Features Relocates Shortcut Reliance, It Does Not Remove It

The port-bucketing ablation (Section 3) was designed to test whether raw port memorization, a documented shortcut in network intrusion detection Engelen et al. (2021), explains the cross-domain collapse. It does not. DecisionTree's feature importances and SHAP values (Section 5.4) show that the model still depends overwhelmingly on which coarse port-bucket category a flow falls into, with three categories accounting for 93% of split importance; SmallMLP, an architecturally unrelated model tied with DecisionTree in-domain, independently shows the same category-level dependence, indicating this is not an artifact of one model's particular fit. Coarsening removed the original, fine-grained shortcut while leaving a coarser version of the same mechanism intact. Section 5.4 shows this is sufficient to explain the collapse: the port-bucket feature DecisionTree relies on most heavily by SHAP value occurs at 96 times the rate in source-domain attack traffic compared to Gotham, and 435 times the rate compared to WUSTL-IIoT-2021. This indicates that reducing feature resolution is not, by itself, a reliable defense against shortcut learning in intrusion detection; the shortcut persists at whatever resolution the feature representation still preserves, and transfers only as well as that resolution's distribution does.

6.2. Balanced Evaluation Overstates Deployment-Relevant Performance

The natural-distribution results (Section 5.2) show that balanced-class evaluation, the convention in most prior cross-dataset studies, substantially overstates performance under conditions a deployed system would actually face. The effect is largest on WUSTL-IIoT-2021, where natural-distribution F1 is roughly one quarter of its balanced-distribution value, driven primarily by a collapse in precision

Table 4
Model size comparison against recently published lightweight intrusion detection systems.

Model / Study	Size	Latency
DecisionTree (this work)	5.7 KB	0.16 ms
SmallMLP (this work)	4.7 KB	0.11 ms
Small1DCNN (this work)	4.8 KB	0.21 ms
SmallLSTM (this work)	7.9 KB	0.40 ms
TCN tcn (2026)	16.3 KB	–
TinyML FNN lun (2025)	31 KB	–
Dynamic-Quant. IDS dis (2025)	<32 KB	–
BiGRU-MHA-LSTM tcn (2026)	39.9 KB	–
DNN-KDQ Umar et al. (2025)	20.2 KB	0.07 ms

as false positives accumulate against a much larger pool of benign traffic. This suggests that F1 or accuracy figures reported under artificial class balance should not be read as estimates of operational performance. The effect is not limited to overstating performance on a single target: balanced evaluation reverses which of the two target datasets appears more difficult relative to natural evaluation (Section 5.2), indicating that comparative claims about which deployment environment poses the greater generalization challenge are themselves protocol-dependent, not solely a property of the environments being compared. For context, Ferrag et al. report 99.54% accuracy for binary classification on Edge-IIoTset using the dataset's full native feature set Ferrag et al. (2022); the in-domain accuracy reported here (97.2%, Table 3) reflects the cost of restricting the feature set to sixteen dimensions deliberately chosen for cross-dataset compatibility (Section 3); the 2.3-point difference is the price paid for a schema designed to generalize, not evidence of a weaker model. No equivalent cross-dataset baseline exists for direct comparison on these three datasets, which is the gap this paper addresses.

6.3. Adaptability Is Architecture-Dependent, Not Data-Dependent

The few-shot results (Section 5.6) show that no single rule relates the amount of target-domain data available to the degree of recovery obtained. DecisionTree requires a comparatively large adaptation sample before any benefit appears, then recovers substantially. SmallLSTM recovers with a much smaller sample, but its performance declines as more adaptation data is added rather than continuing to improve; a plausible contributor is the per-batch class-weight recomputation used during fine-tuning (Eq. 1), which changes as the adaptation sample's class balance shifts with size, though confirming this would require repeating the experiment under fixed class weights. SmallMLP improves only modestly even with the largest sample tested, and Small1DCNN does not improve at any sample size. A substantial part of DecisionTree's recovery may reflect this procedural difference rather than architecture alone, since it is refit on its full original training set plus the new sample while the neural models are fine-tuned on the new sample only; this study does not include a DecisionTree-refit-on-small-sample-alone condition that would isolate the two factors, and the magnitude of DecisionTree's recovery should accordingly be read as a property of the update procedure used, not of decision trees as an architecture class. The architecture-dependence claim rests more securely on the comparison between SmallLSTM and Small1DCNN, which are fine-tuned under identical procedures yet diverge sharply in outcome (Section 5.6); this pair, not the DecisionTree comparison, is the cleanest evidence for Contribution 4. This comparison should still be read with one caveat: SmallLSTM processes the 16-dimensional feature vector as an artificial sequence of scalars rather than a natural temporal signal (Section 4), so its divergence from Small1DCNN may partly reflect this specific parameterization choice rather than a general property of recurrent architectures.

6.4. Generalization, Robustness, and Efficiency Are Independent Axes

Three properties examined in this paper, cross-domain generalization, adversarial robustness, and computational efficiency, do not track one another. DecisionTree is the cheapest to train, among the more adversarially robust models, and one of the two architectures that recovers well under few-shot adaptation; SmallLSTM is the most expensive to train yet performs comparably or better on cross-domain transfer while being among the least adversarially robust. A model that is favorable on one of these axes offers no guarantee on the others, and a deployment decision that optimizes for one without examining the rest risks an unexamined weakness on another. This independence is not absolute: on WUSTL-IIoT-2021 specifically, DecisionTree is both the strongest cross-domain performer and among the more adversarially robust models, an alignment rather than an independence; the broader pattern holds because SmallMLP and SmallLSTM break it in the opposite direction on Gotham.

6.5. Practical Implications for Model Selection

No single architecture is preferable on every axis examined here. If deployment can budget for a modest labeled sample from the target network, DecisionTree offers the strongest combination of low training cost and substantial few-shot recovery, though the confound noted in Section 6.3 regarding

its update procedure should be kept in mind. If zero-shot performance on an unfamiliar network is the priority and a known adversarial robustness cost is acceptable, SmallLSTM performs best of the four on Gotham, though not on WUSTL-IIoT-2021. Small1DCNN is not recommended under any condition tested here: it does not generalize well zero-shot, does not recover with adaptation, and offers no efficiency advantage over the other three architectures. These recommendations are specific to the two target networks tested and should be revisited for deployment targets that differ structurally from both.

6.6. Limitations

Five limitations bound the scope of these findings. First, the common feature schema is restricted to fields available across all three datasets; WUSTL-IIoT-2021 in particular contributes no TCP flag information, so its cross-domain results reflect only the port-bucket and protocol portion of the schema. Second, Edge-IIoTset's training traffic is 88.4% TCP and 11.5% ICMP, with effectively no UDP traffic, limiting how much protocol diversity the source domain offers for a model to learn from. Third, adversarial robustness is reported from a single 100-sample evaluation per model without repetition across seeds, in contrast to the multi-seed and significance testing applied to the cross-domain results; the relative ordering between models should be read with correspondingly lower confidence than the rest of this paper's findings. Fourth, few-shot recovery is evaluated on Gotham only; Section 5.2 shows that architecture ranking under zero-shot transfer differs between Gotham and WUSTL-IIoT-2021, so whether the same architecture-dependent recovery pattern observed here also holds on WUSTL-IIoT-2021 is untested. Fifth, the explainability analysis in Section 5.4 is conducted on DecisionTree and SmallMLP; both rely predominantly on port-bucket categories, though on different specific features, strengthening the generality of the port-shortcut finding. Whether Small1DCNN and SmallLSTM rely on the same category of feature is not directly verified, and is inferred only indirectly from their shared cross-domain collapse pattern. Sixth, WUSTL-IIoT-2021's larger cross-domain collapse relative to Gotham is attributed to general structural dissimilarity, but its zero-filled TCP flag features are a specific, isolable contributor not separated from other sources of dissimilarity in this study; a controlled ablation removing flag features from all three datasets uniformly would be needed to attribute the larger collapse specifically to this factor rather than to capture-method or protocol differences more broadly. Seventh, the class-weight recomputation hypothesis offered for SmallLSTM's non-monotonic few-shot recovery (Section 6.3) was not tested directly; confirming it would require repeating the few-shot experiment with fixed class weights across all fractions.

7. Conclusion

This paper tested whether lightweight intrusion detection models that perform well on the network they were trained on retain that performance on a different IIoT network. Across two independent target datasets, they do not: detection performance collapses substantially and consistently, a result stable across random seeds and statistically significant. This failure persists even after a feature-resolution ablation specifically designed to remove the most obvious explanation, raw port memorization, indicating that the same shortcut mechanism operates at coarser feature resolutions and is not eliminated by reducing granularity alone. Evaluating under each target network's natural, imbalanced class distribution rather than the artificially balanced convention common in prior work reveals a further, independent problem: balanced evaluation not only overstates performance but can reverse which deployment target appears more difficult. Recovery through small-scale fine-tuning on target-domain data is possible for some architectures but not others, indicating that adaptability is a property of model design rather than of lightweight models as a class. Taken together, these findings indicate that within-domain accuracy is not a reliable basis for judging deployment readiness, and that cross-network evaluation, under realistic class distributions, should be treated as a standard requirement for lightweight IIoT intrusion detection research rather than an optional addition.

CRediT authorship contribution statement

MD Azizul Hakim: Conceptualization, Methodology, Software, Formal analysis, Visualization, Writing – original draft. **Md Shihab Uddin**: Investigation, Formal analysis, Visualization, Writing – Review and Editing. **Talha Ibne Anich**: Investigation, Formal analysis, Visualization, Writing – Review and Editing.

Declaration of generative AI and AI-assisted technologies in the manuscript preparation process

During the preparation of this work the authors used Sonnet 4.6 in order to paraphrase some portion of text and to check the grammar of the manuscript. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors. All costs associated with this study, including computational and data resources, were borne by the authors.

Data availability

Edge-IIoTset Ferrag et al. (2022), Gotham 2025 Belarbi et al. (2025), and WUSTL-IIoT-2021 Ahuja et al. (2021) are publicly available from their original sources. Code to reproduce the experiments in this study is available here.

References

, 2025. From tiny machine learning to tiny deep learning: A survey. ArXiv:2506.18927.

, 2025. Lightweight intrusion detection system for iot with improved feature engineering and advanced dynamic quantization. Discover Internet of Things doi:10.1007/s43926-025-00203-8.

, 2025. Lightweight machine learning models for intrusion detection on iot devices Benchmarking Lightweight ML Models for Edge-IoT Botnet Detection under Resource Constraints.

, 2026. A lightweight multi-classification intrusion detection model for edge iot networks. Electronics 15, 938. doi:10.3390/electronics15050938.

Ahuja, N., Singha Roy, S., Kumar, N., Jain, R., 2021. Machine learning based intrusion detection system for software defined industrial internet of things networks. arXiv preprint arXiv:2103.01410 .

Apruzzese, G., Pajola, L., Conti, M., 2022. The cross-evaluation of machine learning-based network intrusion detection systems. IEEE Transactions on Network and Service Management 19, 5152–5169. URL: http://dx.doi.org/10.1109/TNSM.2022.3157344, doi:10.1109/tnsm.2022.3157344.

Belarbi, O., Spyridopoulos, T., Anthi, E., Rana, O., Carnelli, P., Khan, A., 2025. Gotham dataset 2025: A reproducible large-scale iot network dataset for intrusion detection and security research. arXiv preprint arXiv:2502.03134 .

Bengio, Y., 2012. Practical recommendations for gradient-based training of deep architectures, in: Neural networks: Tricks of the trade: Second edition. Springer, pp. 437–478.

Bilal, M.A., Islam, I.U., Idrees, S., Qasim, M., Khan, M.J., Khan, J., 2026. Dataset-centric evaluation of federated intrusion detection models in iot networks. Scientific Reports 16, 2683. doi:10.1038/s41598-025-32567-w.

Cantone, M., Marrocco, C., Bria, A., 2024. Machine learning in network intrusion detection: A cross-dataset generalization study. IEEE Access 12, 144489–144508.

Chen, J., Jordan, M.I., Wainwright, M.J., 2020. Hopskipjumpattack: A query-efficient decision-based attack, in: 2020 ieee symposium on security and privacy (sp), IEEE. pp. 1277–1294.

Debicha, I., Cochez, B., Kenaza, T., Debatty, T., Dricot, J.M., Mees, W., 2023. Review on the feasibility of adversarial evasion attacks and defenses for network intrusion detection systems. arXiv preprint arXiv:2303.07003 doi:10.48550/arXiv.2303.07003.

Dharini, N., Janani, V., Katiravan, J., 2026. Efficient detection of intrusions in ton-iot dataset using hybrid feature selection approach. Scientific Reports .

D'hooge, L., Wauters, T., Volckaert, B., De Turck, F., 2019. In-depth comparative evaluation of supervised machine learning approaches for detection of cybersecurity threats, in: 4th International Conference on Internet of Things, Big Data and Security (IoTBDS), pp. 125–136.

Engelen, G., Rimmer, V., Joosen, W., 2021. Troubleshooting an intrusion detection dataset: the cicids2017 case study, in: 2021 IEEE Security and Privacy Workshops (SPW), IEEE. pp. 7–12.

Ferrag, M.A., Friha, O., Hamouda, D., Maglaras, L., Janicke, H., 2022. Edge-iiotset: A new comprehensive realistic cyber security dataset of iot and iiot applications for centralized and federated learning. IEEe Access 10, 40281–40306.

Graves, A., 2012. Long short-term memory. Supervised sequence labelling with recurrent neural networks , 37–45.

Ige, A.O., Sibiya, M., 2024. State-of-the-art in 1d convolutional neural networks: A survey. IEEE Access 12, 144082–144105.

Kheddar, H., Himeur, Y., Awad, A.I., 2023. Deep transfer learning for intrusion detection in industrial control networks: A comprehensive review. Journal of Network and Computer Applications 220, 103760.

Latha, R., Bommi, R., 2023. An analysis of intrusion detection systems in iiot, in: 2023 Eighth International Conference on Science Technology Engineering and Mathematics (ICONSTEM), IEEE. pp. 1–10.

Layeghy, S., Portmann, M., 2023. Explainable cross-domain evaluation of ml-based network intrusion detection systems. Computers and Electrical Engineering 108, 108692.

Loh, W.Y., 2011. Classification and regression trees. Wiley interdisciplinary reviews: data mining and knowledge discovery 1, 14–23.

Lundberg, S.M., Lee, S.I., 2017. A unified approach to interpreting model predictions. Advances in neural information processing systems 30.

Nicolae, M.I., Sinn, M., Tran, M.N., Buesser, B., Rawat, A., Wistuba, M., Zantedeschi, V., Baracaldo, N., Chen, B., Ludwig, H., et al., 2018. Adversarial robustness toolbox v1. 0.0. arXiv preprint arXiv:1807.01069 .

Pembury Smith, M.Q., Ruxton, G.D., 2020. Effective use of the mcnemar test. Behavioral Ecology and Sociobiology 74, 133.

van Rijsbergen, C.J., 1979. Information Retrieval. 2 ed., Butterworth-Heinemann.

Seger, C., 2018. An investigation of categorical variable encoding techniques in machine learning: binary versus one-hot and feature hashing.

Singla, A., Bertino, E., Verma, D., 2020. Preparing network intrusion detection deep learning models with minimal data using adversarial domain adaptation, in: Proceedings of the 15th ACM Asia conference on computer and communications security, pp. 127–140.

Taud, H., Mas, J.F., 2017. Multilayer perceptron (mlp), in: Geomatic approaches for modeling land change scenarios. Springer, pp. 451–455.

Umar, H.G.A., et al., 2025. Energy-efficient deep learning-based intrusion detection system for edge computing: a novel DNN-KDQ model. Journal of Cloud Computing 14, 32. doi:10.1186/s13677-025-00762-9.

Verkerken, M., D'hooge, L., Wauters, T., Volckaert, B., De Turck, F., 2022. Towards model generalization for intrusion detection: Unsupervised machine learning techniques. Journal of Network and Systems Management 30, 12.

Vitorino, J., Praça, I., Maia, E., 2023. Towards adversarial realism and robust learning for iot intrusion detection and classification. Annals of Telecommunications 78, 401–412.

Zolanvari, M., Teixeira, M.A., Gupta, L., Khan, K.M., Jain, R., 2019. Machine learning-based network vulnerability analysis of industrial internet of things. IEEE internet of things journal 6, 6822–6834.