Evaluating Machine Learning-Based IoT Device Identification Models for Security Applications

Eman Maali
Imperial College London
e.maali19@imperial.ac.uk

Omar Alrawi
Georgia Institute of Technology
alrawi@gatech.edu

Julie McCann
Imperial College London
j.mccann@imperial.ac.uk

***Abstract*—With the proliferation of IoT devices, network device identification is essential for effective network management and security. Many exhibit performance degradation despite the potential of machine learning-based IoT device identification solutions. Degradation arises from the assumption of static IoT environments that do not account for the diversity of real-world IoT networks, as devices operate in various modes and evolve over time. In this paper, we evaluate current IoT device identification solutions using curated datasets and representative features across different settings. We consider key factors that affect real-world device identification, including modes of operation, spatio-temporal variations, and traffic sampling, and organise them into a set of attributes by which we can evaluate current solutions. We then use machine learning explainability techniques to pinpoint the key causes of performance degradation. This evaluation uncovers empirical evidence of *what* continuously identifies devices, provides valuable insights, and practical recommendations for network operators to improve their IoT device identification in operational deployments.**

I. Introduction

The ubiquitous adoption of IoT devices poses security and privacy challenges because they are large in number, wide in variety, interactive and embedded. IoT devices often share hardware and software components and, as a consequence, potential vulnerabilities, as exposed by [20]. Therefore, network operators have relied on the location and identification of devices to attribute vulnerabilities rather than conduct security studies directly. This led the IETF to produce the Manufacturer Usage Description (MUD) specification to facilitate the identification of IoT device types and functionalities, but this also has challenges, *e.g.,* managing device diversity, slow adoption, and limited manufacturer compliance, hindering its widespread effectiveness [4, 21].

Consequently, several proposed solutions to identify IoT devices [1–6, 8, 16, 18, 19, 22–28] have leveraged Machine Learning (ML) techniques. Such techniques analyse passive network data to fingerprint IoT devices, assuming each device exhibits a unique network pattern. Machine Learning-based (ML-based) solutions automatically identify and classify devices based on their network behaviours, ensuring consistent device management, privacy, and updates without manual intervention [29].

Despite advancements, significant challenges limit network operators from deploying proposed ML-based IoT identification solutions. Such solutions assume static environments and overlook the diversity of real-world IoT networks, where devices operate in various modes and evolve behaviours over time [1–6, 8, 16, 18, 19, 22–28]. Furthermore, ML-based solutions do not account for traffic sampling [30, 31]. Therefore, network operators need practical evaluations that are relevant to ML-based solutions.

There have been attempts to evaluate ML solutions for non-IoT devices in the broader Internet domain in terms of generalisation, robustness, fairness, and scalability [32–37]. However, this does not account for IoT specific attributes that set them apart *i.e.,* they are embedded, highly interactive and potentially battery powered. Prior evaluations consider limited real-world factors, for example: (1) Limited understanding of ML-based solutions' performance across operational modes (*e.g.,* idle versus active) [25]. (2) Insufficient evaluation of generalisability across spatial (*e.g.,* identical devices in different networks) and temporal (*e.g.,* changes over time) variations [38, 39]. (3) Lack of investigation into how traffic sampling (*e.g., sFlow*), commonly used to reduce overhead, affects model performance [2]. For example, Ahmed *et al.* [39] shows that Random Forest can avoid performance degradation across specific environments by using certain features.

In this paper, we conduct a rigorous evaluation to enable reproducibility of ML-based IoT device identification. In doing so, we investigate the performance degradation of IoT identification solutions in different settings. Our investigation shows to improve feature selection and robustness, which, to the best of our knowledge, remains unexplored in prior work. We seek to explore the following research questions:

- **RQ1:** How effectively can ML-based IoT device identification solutions operate across different modes of operation and network environments?
- **RQ2:** How can performance degradation in ML-based IoT device identification solutions be pinpointed across different operational deployments, and what are the underlying causes?
- **RQ3:** Can the performance of ML-based IoT device identification solutions be improved by modifying their

Network and Distributed System Security (NDSS) Symposium 2025
24-28 February 2025, San Diego, CA, USA
ISBN 979-8-9894372-8-3
https://dx.doi.org/10.14722/ndss.2025.240118
www.ndss-symposium.org

TABLE I: **Qualitative comparison for selected IoT device identification current literature. Each column corresponds to a discussion of the comparison aspect of related work. We refer to real-world validation as validation. The ✓ implies the availability of code or solutions performed for real-world validation.**

Category	Network Data	Paper	Venue	Algorithms	Features	Dataset	Devices	Public Code	Validation
Rule-based	**Flow Level**	Feng18 [1]	USENIX		Protocol		IoT		
		Saidi20 [2]	IMC		Protocol	[3]	IoT		✓
		Hamza20 [4]	IEEE Trans		Protocol	[5]	Mix	✓	✓
Learning-based	**Flow Level**	*Siva18 [5]	IEEE Trans.	NB, RF	Header		Mix		✓
		Marc19 [6]	IEEE Jour	KNN	Periodicity		IoT		
		*Yang19 [7]	Elsevier	LSTM	Header		IoT		✓
		*Ortiz19 [8]	IoTDI	LSTM	Protocol	[5]	IoT		
		*Meid20 [9]	Elsevier	LightGBM	Header		IoT		
		Cvit21 [10]	Springer	LogitBoost	Header	[10]	IoT		
		*Okui22 [11]	COMPSAC	LightGBM	Protocol	[11]	IoT		
	Packet Level	Than18 [12]	IEEE Trans.	K-means	Attributes		IoT		
		*Pinh19[13]	Elsevier	RF	Header	[5]	Mix		
		*Dong20[14]	AsiaCCS	LSTM	Header	[14]	Mix	✓	
		Yu20 [15]	USENIX	DL	Attributes		IoT		
		*Perd20 [16]	EuroS&P	TF-IDF	Attributes	[16]	IoT		✓
		*Fan22 [17]	IEEE Trans.	CNN	Header/Attributes	[5, 16]	Mix	✓	
		Zhao23[18]	IEEE Tran.	LR	Header	[5, 19]	IoT		

feature selection to address practical challenges?

Answering these questions leads to the following contributions:

- **Reproduction and Evaluation.** We reproduce ten state-of-the-art ML-based IoT identification solutions and evaluate their performance across different operational contexts. This analysis identifies key factors limiting their deployment.
- **Practicality Analysis.** We introduce a framework for evaluating the practicality of ML-based solutions using three attributes: Mode of operation, transferability, and observability.
- **Feature Analysis and Improvement Recommendations.** We pinpoint the causes of performance degradation and provide actionable recommendations for feature selection to guide future research.

Our evaluations reveal that ML-based IoT device identification solutions assuming generalisable communication patterns [3, 39] can fail significantly. Spatial transferability for identical devices in different environments reduces performance by 7.5% to 74%. Temporal transferability shows performance reduction as early as one week by 19.32%, and up to 85.90% after 52 weeks. Additionally, many solutions perform poorly with sampled network traffic like *sFlow*, where model performance reduces by an average of 70.09%. These findings highlight the need for continuous model training and maintenance.

II. Machine Learning-Based IoT Identification

Literature Review. Identification solutions, such as those by Kohno *et al.* (2005) [40] and others [41–46], identify general network devices, relying on the TCP / IP clock skew or similar attributes and did not consider the specific attributes of IoT devices. IoT devices have distinct attributes, including various modes of operation, evolving capabilities (*e.g.,* functionality, update) and specialised communication protocols. Traditional identification solutions must be adapted for IoT identification to provide accurate, scalable, and robust identification [47, 48]. Some IoT devices have constraints on power (battery-powered) that make their network behaviour nondeterministic or event-driven. Traditional network device identification solutions do not consider such scenarios because a common assumption is that communication of devices occurs periodically [49–54]. However, this may not apply to IoT devices. For example, a motion sensor only activates when triggered by motion (event-driven). A key difference between ML-based traditional identification and IoT-specific identification is that IoT devices exhibit diverse behaviour, which requires *tailored* modelling.

The first discussions on the identification of IoT were found in Sivanathan *et al.* (2017) [55], and Lopez *et al.* (2017) [56]. Sivanathan *et al.* and Lopez *et al.* compared their approaches with Internet traffic and machine-to-machine communication classification approaches [49–54]. Thus, we thoroughly surveyed the literature in top security, Internet measurement, and network and system conferences and journals from 2017 onwards. We used IEEE Xplore and Google Scholar, with keywords such as "IoT devices," "smart home devices," "identification," "fingerprinting," "traffic analysis," "traffic classification," "machine learning," "deep learning," and "automated." This process identified over 200 papers (2017 to present). This results in 96 papers with the scope of IoT device identification.

For each paper, we documented the problem, methodology, and results. We grouped the papers by applications, such as anomaly detection and device behaviour. This grouping allowed us to identify common applications that rely on IoT device identification. Additionally, we categorised the methodologies to identify common ML algorithms. Overall, we selected 17 recent papers representative of IoT device identification. The selection of the papers was based on the number of citations, visibility of the conference in which they appear, application, and novelty. Lastly, we categorised these papers by technique, covering rule-based methods [1, 2, 4]

and ML-based methods [5–18]. Many of these papers are ML-based (14 out of 17).

We summarised the 17 papers in Table I and compared the network data used, algorithms, features, datasets, device settings, code availability, and validation. We find that many papers use flow-based network data as input. This usage relates to the prevalence of encrypted traffic and the overhead associated with packet-level inspection [57]. Few papers performed empirical validation with public datasets or published their code. Many papers favour deep-learning techniques for ML algorithms that rely on header information such as packet size, timing, and statistical features.

Researchers often choose deep-learning techniques over traditional ML techniques for IoT device identification, see Table I. Deep-learning techniques can automatically learn complex features from high-dimensional data, reducing the need for extensive manual feature engineering. Architectures such as Convolutional Neural Networks (CNNs) and Long-Short-Term Memory (LSTMs) excel at capturing intricate patterns and temporal dependencies inherent in network traffic. In the surveyed literature, Fan *et al.* [17] employed CNN with features such as time interval, traffic, protocol, and Transport Layer Security (TLS) handshake. Dong *et al.* [14] utilised a bidirectional LSTM model with packet size, sequence, timing, and traffic volume. On the other hand, Ortiz *et al.* [8] represented TCP connections using autoencoders before training with LSTM. The advantages of deep-learning include superior performance with *large* and *heterogeneous* datasets and adaptability to new device types and traffic patterns. However, these benefits come with drawbacks such as significant computational resource requirements, longer training times, and the need for large volumes of *labelled data*. In contrast, traditional ML models are more computationally efficient, require fewer data samples, and offer greater interpretability. However, traditional ML struggles with accurately capturing the complex patterns in the data features necessary to identify IoT devices. Our work empirically documents these differences between traditional and deep-learning models.

Evaluating ML-based IoT Identification. Researchers have studied many evaluation attributes of ML solutions. For example, Beltiukov *et al.* [35] introduced a closed-loop ML pipeline to collect high-quality datasets and created a generalisable approach to network security. Jacobs *et al.* [36] focused on high-fidelity, low-complexity decision trees to help users identify under-specification in their models [37]. Evaluation of ML models expands beyond performance metrics such as accuracy and F1-score. The following is a set of attributes that researchers have studied in prior works:

1) ***Generalisation and Robustness:*** Evaluating a model's ability to generalise and transfer across different domains [32–34].
2) ***Fairness and Accountability:*** Evaluating interpretability and potential bias against specific demographic groups [32, 33].
3) ***Model Scalability:*** Measuring effectiveness in handling increasing data volumes in distributed environments [32–34].
4) ***Data Efficiency:*** Learning from small datasets is crucial in real-world scenarios [33, 34].
5) ***Deployment Compatibility:*** Evaluating ease of integration across various hardware platforms.
6) ***Stability Over Time:*** Examining consistent performance amid shifting data distributions [33, 34].
7) ***Ethics and Societal Impact:*** Considering consequences such as energy consumption and societal effects, such as job displacement [32].
8) ***Cost Metric:*** Evaluating deployment and maintenance costs, including hardware, software, and retraining [32–34].

However, not all these attributes account for identifying IoT specific attributes. Only some prior efforts IoT device identification reflect *certain* IoT attributes. Dadkhah *et al.* [25] captured network traffic from IoT devices in lab environments and analysed device behaviour in different modes of operation. However, they did not investigate how different modes affect device identification. Saidi *et al.* [2] identify IoT devices in real-world conditions by using sampled data from large networks, but their approach performed poorly when the data was incomplete or limited.

Few papers on IoT device identification evaluation reflect *certain* IoT attributes. Kolcun *et al.* [38] investigated performance degradation by evaluating three solutions [5, 16, 18] for IoT device identification, evaluating each model across separate time frames to ensure consistency by using the same datasets, features, and environments. Kolcun *et al.* showed that performance degradation was due to the models, not external factors such as modes of operation. Ahmed *et al.* [39], the closest work to ours, evaluated six publicly available datasets and found that Random Forest (RF) can effectively identify devices across all datasets. While reproducing Ahmed *et al.* [39], we found that the published code [58] did not adequately separate the dataset for training and testing. We contacted the authors, and they provided a technical solution with a sample configuration to reproduce their results. The code contained two methods for specifying the training and testing dataset. The first approach used an explicit variable and pointed to the training and testing dataset. The second approach used five-fold cross-validation to evaluate the dataset. However, after inspecting the code, we found that the function will reuse the training dataset as the testing dataset, regardless of the configured parameters. Technically, the *loadData* function always returns a None value for the test dataset, and the caller function will only use the training dataset [59]. We could not reconcile the paper's claims and findings. Thus, we excluded Ahmed *et al.* [39] from our evaluation.

Although these studies explore related evaluation attributes in IoT device identification and behaviour, they differ in scope (identification vs. behaviour modelling), evaluation conditions (controlled vs. unconstrained), and data completeness (sampled vs. full data). As a result, it is challenging to combine prior works into a single evaluation for operational deployments. We aim to evaluate the practicality of IoT device

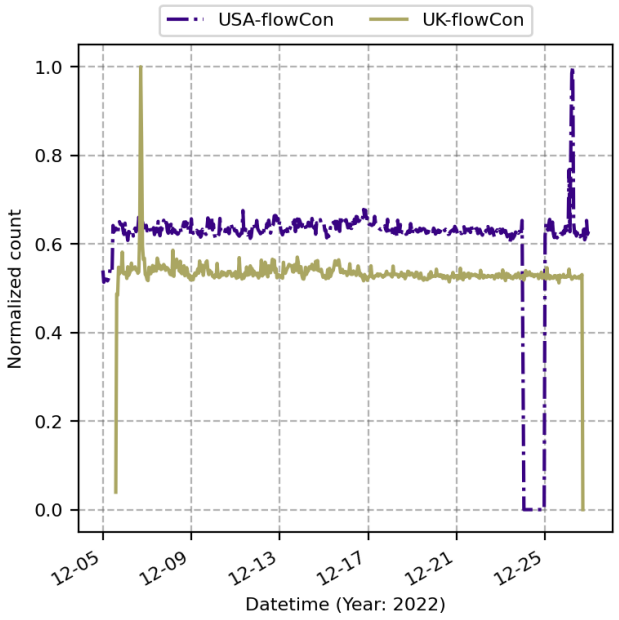


Fig. 1: **The UK and USA testbed devices' number of flows per day between $5^{th} - 27^{th}$ December, and the devices' mode of operation was idle.**

identification by considering the following attributes:

- **Mode of Operation** reflects the variability introduced by the life cycle of IoT devices (*e.g.,* active vs. idle modes). This attribute aligns with stability and robustness over time.
- **Transferability** is the model's ability to generalise across different environments (spatial) and over time (temporal). This attribute aligns with generalisation [32–34].
- **Observability** is the model's ability to maintain accuracy across different network conditions and sampling rates (*i.e.,* congestion, packet loss or incomplete data). This attribute aligns with robustness [32–34].

We exclude other attributes since they are less relevant to deploying IoT identification solutions. For example, it may be unnecessary to evaluate adversarial robustness if the IoT identification techniques suffer in a trusted environment. Similarly, demographic bias focuses on human factors, which is less relevant for deployment considerations.

III. Methodology

In this section, we describe our data collection, reproducibility, features, performance metrics, practicality attributes, explainability analysis, and statistical analysis.

A. *Dataset Collection and Labelling*

Testbed Setup. To rigorously evaluate IoT device identification solutions we need to capture temporal, geographical, and behavioural diversity. We build two testbeds that reflect real-world scenarios across geographically distinct regions. This dual testbed approach exposes differences in market availability, network infrastructure, and user interaction patterns across regions, including overlapping and region-specific devices. The two testbeds in the UK and the USA collect data from 90 IoT devices. The USA has 75 devices, and the UK has 15 devices. See Appendix A Table VIII for a complete list that reflected popular consumer choices [60–63], and represented the major vendors for the US and UK markets. Both testbeds have 13 common IoT devices with some non-IoT devices such as tablets, smartphones, and laptops, spanning the following categories: Cameras (security and video doorbells), smart hubs (home automation bridges for Zigbee, Z-wave, Insteon devices), home automation (smart lights, outlets, thermostats), TVs (including dongles), audio (smart speakers with voice assistants), and appliances (cleaning, cooking, weather stations). Each testbed has a Linux gateway (Ubuntu 20.04.4) with two wired network interfaces: One for Internet access via a public IP and another for the IoT devices on a private network. The testbeds have Wi-Fi adapters that connect to the IoT network and support $2.4GHz$ and $5GHz$ wireless devices. We connected the wired devices directly to a network switch. All devices communicate through standard Network Address Translation (NAT) implemented at the gateway. We used *tcpdump* to capture incoming and outgoing Wi-Fi traffic on the gateway. We configured the DHCP service on the gateway to assign static IP addresses to each device based on their MAC address.

Testbed Instrumentation. We ran idle experiments from 6^{th} to the 27^{th} of December 2022 while students were on break to capture the baseline behaviour of devices with no user interference. Figure 1 shows the daily traffic in the UK and the US testbeds. The average flow on the UK testbed is $6,177$ per day, while the US testbed is $33,843$ per day. We ran active experiments on the UK testbed from the 6^{th} to the 27^{th} of February 2024. Users interacted with the devices by issuing commands to smart speakers and activating lights. On average, we recorded 20 lab accesses daily, each involving at least one device interaction. See Appendix B Table IX for a summary of device activities. To simulate different network conditions such as congestion and interference [64–66], we collected traffic data in April 2024 using four sampling rates to represent data loss, which we used in our evaluation. We also collected traffic traces from both testbeds in January 2023, March 2023, June 2023, and December 2023. We divided all of them into continuous 21-day periods to capture IoT devices' behavioural patterns [47, 48]. We labelled all these traffic traces by year, location, device mode of operation, sample rate, and device model. This approach accounted for both temporal and geographical variability. We considered public datasets as [67–69] but found their collection periods are short.

Testbed Validity. To systematically control different variables and enhance the reliability of the results, our testbeds offer a controlled environment that mimics real-world home deployments. This approach improves the accuracy of the evaluation and ensures the reproducibility and repeatability of experiments, which are essential for empirical research. Our testbeds instrument everyday tasks typical of home users and provide diverse evaluation scenarios, allowing us to set an upper bound on ML-based IoT identification performance

under pristine conditions. Real-world datasets such as [69] may contain uncontrolled and latent factors that impact the evaluation of IoT identification. Lastly, the lab-based nature of the testbeds avoids the privacy concerns inherent in real-world data collection while still allowing the simulation of diverse network conditions and real-world interactions.

TABLE II: **List of features used in each experimental setup under evaluation.**

Algorithm	Experiments Setup	Set of Features
LightGBM	Meid20 [9]	Incoming bytes, incoming packets, TCP flags, flow duration, dst/src ports, Prot. src_TOS IPv4 dst address
	Okui22 [11]	octet Total Count Bytes reverse Octet Total Count Bytes, packet Total Count , reverse Packet Total Count
Random Forest	Siva18 [5]	Flow volume, flow duration SLD entropy, sleep time src/dst port, Cipher-suites DNS/NTP-intervals
	Pinh19 [13]	Total, mean, and standard deviation bytes transmitted
LSTM	Yang19 [7]	TOS field, TTL, IP DF, src/dst port, TCP Win, TCP RTO, TCP MSS, TCP OPT
	Ortiz19 [8]	TCP connection fields
	Dong20 [14]	Dst port, protocol, frame length time interval
CNN	Fan22 [17]	Total packets, total bytes flow rate, flow duration min/MAX/AVG time interval NTP/DNS duration entropy (10 domains) number of DNS queries TLS handshaking count
TF-IDF	Perd20 [16]	DNS-domain frequency

B. *Features and Features Extraction*

We must account for the different features used by ML-based IoT identification solutions to enable a rigorous evaluation. Thus, we must be consistent with features employed in existing literature in IoT device identification. Features are commonly extracted at three distinct levels of abstraction: (1) Packet-level features, which describe individual packet attributes. (2) Flow-level features, which capture aggregated patterns over communication sessions. (3) Deep-learning features designed to leverage advanced representation-learning techniques [32, 70].

In practice, we used *tshark* [71] to parse raw packets and extract flows. We used the feature extraction tools YAF [72] and CICFlowmeter [68]. We used *Softflowd* [73] to sample traffic for observability evaluation. Note that *Softflowd* extracts only simple features *e.g.,* packet size or time between packets. This extraction balances the trade-off between detail and volume of data storage and processing [30, 31, 74]. To ensure compatibility with ML models and preserve categorical variable information, we used standard encoding (*e.g.,* one-hot or label encoding) [32, 75].

C. *Reproducibility of Prior Work*

To ensure that the models and configurations were as consistent as possible, we matched hyper-parameters and data preprocessing steps and directly used code where publicly available, *e.g.,* [39]. Code was only modified when we encountered an outdated library, such as [14], to ensure library compatibility. For [17], we used a docker container with the same library version found in the original code. Where code was unavailable [7–9, 11, 13, 16, 55], we re-implemented the code described in the original papers assuring configurations were as consistent as possible.

Similarly, some datasets used in the papers were private. Therefore, we reused the code on our datasets and carefully adjusted parameters to maintain the integrity of the experiments, such as input size or preprocessing steps, to align with the structure of our data. For example, we modified window size parameters to match the sampling frequency of our dataset and ensured that this change did not affect the overall feature extraction process. Reproducing prior work is challenging - we made conservative parameter assumptions and chose the best-performing results. We aimed to align with published results, and of the top 14 papers, we reproduced 10. The remaining four needed more feature extraction information.

D. *Performance Metrics*

It is well established that accuracy metrics are unsuitable for evaluating imbalanced datasets because the majority class can dominate predictions [34, 76]. The True Positive Rate (TPR) and False Positive Rate (TPR) are the established approach [34, 76–78]. For our experiments, we use the area under the Precision-Recall Curve (AUCPR) metric to provide insights into a model's ability to correctly identify instances, particularly those belonging to the minority class in imbalanced datasets [77, 78].

E. *Practicality Attributes*

As mentioned in Section II, our work focuses on practical attributes that are relevant to IoT deployment, thus:

Mode of Operation*:* IoT devices, once powered on, exhibit four possible modes: Setup, idle, active, and update [47]. However, in typical operating environments, IoT devices are predominantly idle or active [47, 55]. To simplify our evaluation, we focus on the two fundamental modes, idle and active, to reduce complexity while demonstrating our evaluation. We initially train the original model on a mixed-mode dataset that contains idle and active samples and test the model for (1) Idle scenario, (2) active scenario, and (3) mixed scenario, both idle and active samples. We evaluated traffic from the two testbeds and averaged the results across each scenario.

Transferability: This attribute evaluates the temporal or spatial generalisability of the model. ***Temporal Transferability*** evaluates a model's capacity to maintain accurate predictions over different periods. ML solutions try to learn the statistical properties of the target variable, which may change over time. This change is known as concept drift [34, 79]. We evaluated the performance of IoT identification papers with the following time delta periods between training and testing: 1 week, 2 weeks, 3 weeks and 1 month, 3 months, 6 months and 12 months. We tested the same devices across different time deltas while keeping the mode of operations mixed for both labs. ***Spatial Transferability*** evaluates a model in a different environment setting. We performed the spatial transferability experiment in two distinct geographical locations focusing on common IoT devices overlapped in the collection time: (1) Trained on the UK testbed and tested on the US testbed, and (2) trained on the USA testbed and tested on the UK testbed. In both scenarios, the devices' mode of operation was idle.

Observability: This attribute evaluates the robustness of IoT identification solutions when limited data is available. Most scenarios in the current ML-based IoT identification solutions use full sample rates (1:1). The default sampling rate for large networks such as *sFlow* is 1:100 [31]. However, to reflect real-world sampling, we performed experiments using three further rates to represent data loss: 1:100, 1:1000, and 1:5000 [30, 31]. These sampling rates provide an upper limit for network congestion, interference, and packet loss.

F. Explainability Analysis

To indicate features that are more influential for the model's predictions, we relied on ML explainability [75, 80, 81] and tailored them to study IoT identification solutions. We designed our ML explainability around fundamental principles for interpreting ML-based systems: t-Distributed Stochastic Neighbour Embedding [81], Shapley Additive exPlanations [80] and feature importance [75]. We do not claim any of the techniques as a novel contribution. We limit our ML explainability to traditional ML features and not deep-learning. Deep-learning features have complex transformations and interactions within neural network architectures. Interpreting feature importance is challenging [82, 83]. For example, a deep-learning model may accurately classify an IoT device, but it is difficult to determine which specific features—like packet size or flow volume—were most critical. This difficulty arises because the model processes these features across multiple and embedded layers, making it hard to trace their contributions to the final decision [82, 83]. We shall explore deep-learning explainability in future work.

t-Distributed Stochastic Neighbour Embedding: To evaluate how effectively feature sets, drawn from various studies, differentiate between device categories (*e.g.,* type or vendor). We used t-Distributed Stochastic Neighbour Embedding (t-SNE), a dimensionality reduction technique commonly used to visualise high-dimensional data in two or three-dimensional space [81]. We evaluate the degree of separation between these categories in the training datasets. Well-separated clusters in the visualisation indicate that the feature set captures relevant patterns effectively while overlapping points suggest less discriminatory power between clusters.

Shapley Additive exPlanations: To understand how each group of features collectively influences the model's predictions, we treated the ML-based IoT device identification as a black-box model, *i.e.,* we do not examine internal workings [80]. We used Shapley Additive exPlanations (SHAP), a local explainability technique. A positive SHAP value for a feature indicates that its presence increases the prediction (*i.e.,* more important), while a negative value suggests that its absence increases the prediction (not so influential). SHAP values reveal how a group of features interact with each other to influence predictions. We optimised SHAP calculations by leveraging efficient implementations from the SHAP library, designed explicitly for tree-based models. We ensured the features represent the correct sequence of events, keeping the order in which data points occur, thus preserving temporal relationships and applying SHAP values to these features. This optimisation ensures we effectively indicate the model's behaviour while handling time-dependent data.

Feature Importance: To evaluate whether the patterns learned from the training data generalise to new instances, we conducted permutation importance tests. These tests determine feature importance by shuffling the values of each feature and measuring the resulting impact on model performance [75, 77]. Features with more significant performance drops when shuffled are considered more important. We perform permutation importance: First, for each feature independently while keeping the values of other features constant to evaluate its importance. Secondly, we shuffle the full feature vector for each sample in the dataset to evaluate the importance of the whole feature set collectively rather than individual features.

G. Statistical Analysis

To identify changes in feature distributions that may indicate model degradation or feature instability. We relied on comparing distributions. We designed our distribution comparison based on the fundamental principle of similarity: Population Stability Index [84], and Kolmogorov-Smirnov Test [85]. We do not claim any of these techniques as novel contributions.

Population Stability Index: We seek to detect the feature distribution shifts that influence the performance of ML models. We used the Population Stability Index (PSI), a statistical measure used to quantify changes in the distribution of a variable over time or between different datasets [84]. PSI provides insight into the model's adaptability and potential performance when exposed to new data by assessing whether the characteristics of a training and testing dataset remain consistent. The PSI results are interpreted across three ranges. A PSI value below 0.1 indicates a negligible shift. Values between 0.1 and 0.25 indicate a slight shift, while a PSI value of 0.30 or greater is a major shift [84].

Kolmogorov-Smirnov Test: We seek to measure distributional shifts and evaluate whether the training data adequately represents the testing environment, a factor in determining the

TABLE III: **Reproducibility and mode of operation evaluation for learning-based IoT identification solutions. We serve reproducibility results as a baseline for comparison throughout practicality metric scenarios.**

Algorithm	Experiments Setup	Reported Results	Reproduced AUCPR	Train: Mix		
				Test: Idle	Test: Active	Test: Mix
LightGBM	Meid20 [9]	AUCPR 0.96	0.88	0.54	0.57	0.78
	Okui22 [11]	AUCPR 0.98	0.90	0.59	0.57	0.78
Random Forest	Siva18 [5]	Accura. 0.99	0.87	0.49	0.27	0.86
	Pinh19 [13]	F-score 0.95	0.89	0.74	0.58	0.87
LSTM	Yang19 [7]	Precis. 0.95	0.78	0.11	0.12	0.11
	Ortiz19 [8]	F-score 0.82	0.76	0.11	0.11	0.10
	Dong20 [14]	Accura. 0.99	0.67	0.11	0.10	0.11
CNN	Fan22 [17]	Accura. 0.99	0.68	0.12	0.12	0.12
TF-IDF	Perd20 [16]	pAUC 0.99	0.80	0.56	0.59	0.59

robustness of models across varied conditions. We used the Kolmogorov-Smirnov (K-S) Test, a non-parametric statistical test used to evaluate the similarity between two distributions [85]. The (K-S) test measures the maximum distance between the cumulative distribution functions (CDFs) of two datasets, providing a way to determine whether they differ significantly. The (K-S) test results are interpreted using the p-value. A p-value < 0.05 indicates that the distributions are significantly different, while a p-value > 0.05 suggests negligible difference [85].

IV. Evaluation Results

A. Experimental Setup

Environment. For all experiments, we deployed ML models on a local server running Ubuntu 20.04.4, with Nvidia GPU RTX $A5000$, Intel CPU 11th Gen Intel(R) Core(TM) i9-11900K @ $3.50GHz$, and $128GB$ RAM.

Learning Models. We implemented the ML model for LightGBM, Random Forest (RF), Long Short-Term Memory (LSTM), Convolutional Neural Network (CNN), and Term Frequency-Inverse Document Frequency (TF-IDF) in Python 3.9 using scikit-learn for RF, TensorFlow for LSTMs and CNNs, and LightGBM. The parameters were inherited from the baseline model to provide a fair comparison, and no hyper-parameters tuning was performed.

B. Reproducibility of Prior Work

We reproduced all results using a baseline dataset with full sampling (1:1), mixed (idle and active), and testbed locations. Table III shows the reported results in the 3rd column and our reproduced results in the 4th column. We use this as the baseline for all subsequent comparisons. For example, Meid20 [9] reported an AUCPR of 0.96 while we produced an AUCPR of 0.88. Siva18 [5] reported an accuracy of 0.99. Our reproduction achieved an AUCPR of 0.87. Pinh19 [13] reported an F-score of 0.95, and our reproduction achieved an AUCPR of 0.89. Pred20 [16], the pAUC 0.99, and our reproduction produced an AUCPR of 0.80.

C. Mode of Operation

We train all models baseline using a dataset with full sampling (1:1), mixed (idle and active), and from both testbed locations. Idle scenario data ranges from the 6^{th} to 27^{th} of December 2022, and active scenario data from the 6^{th} to 27^{th} of February 2024.

Table III shows that the AUCPR of LightGBM reduced by $\approx 35.23\%$ in both idle and active scenarios compared with an AUCPR of 0.78 in the mixed scenario for Meid20 [9] and Okui22 [11]. In the mixed scenario, the testing samples resemble those of the training dataset (both idle and active mode). Interestingly, we observed that devices such as Amazon Echo Show, Roku TV, and Facebook Portal exhibit multiple levels of idleness. For example, the Amazon Echo Show can perform background tasks without receiving user commands, *e.g.,* shuffling photos on display. Alternatively, it can be completely idle and not engage in any activity. Indeed, a device-wise statistical analysis of the multiple idleness levels confirms different distributions, *i.e.,* p-values of the (K-S) test were < 0.01. In active mode, IoT devices exhibit more frequent and varied communication patterns based on user behaviour, *i.e.,* users activate a smart speaker to play music or an e-book. These wide ranges introduce non-static behaviour observed when devices transit between different active patterns. In these cases, static feature-based models such as LightGBM struggle to learn changing patterns, causing a reduction in performance. For example, the Amazon Echo Show has an average daily inter-arrival time for packets of 48.75 seconds and a standard deviation of 154.08. Yet, Google Home shows an inter-arrival time of 7.51 seconds and a standard deviation of 33.20.

For RF, Siva18 [5] AUCPR was reduced by 43.68% and 68.97% in idle and active scenarios, respectively, and achieved an AUCPR of 0.86 in the mixed scenario. We investigated this behaviour by comparing the test-train distribution in the mixed scenario using features from Siva18 [5] listed in Table III. We find PSI < 0.001 in the mix scenario, indicating no change in the test-train feature distribution. Yet, in idle and active scenarios, some devices did exhibit a change in their distribution, such as the Google Home Mini with a PSI value of 0.34 and the TP-Link plug with a PSI value of 3.09. While RF with Pinh19 [13], AUCPR reduced by 16.85% for idle and 34.83% for active mode, and achieved an AUCPR of 0.87. We measured PSI values using features from Pinh19 [13] (Table III) for both idle and mixed datasets, again observing no change in the test-train distribution, *i.e.,* PSI < 0.001. However, in active mode, we observe a change in one feature (the mean of the transmitted bytes) with a PSI of 0.15, indicating a slight shift in distribution. Overall, these results show that RF performs well in specific operation modes but does not adapt to changes in active mode.

For LSTM and CNN in Yang19 [7], Ortiz19 [8], Dong20 [14] and Fan22 [17], the AUCPR reduced by 82.35% in idle, active, and mix scenario. The AUCPR reduction indicates that LSTM and CNN models overfit or memorise patterns in the training modes. We can see the lack of generalisation between idle to active modes concretely in the Amazon Echo Show, where packet sizes change from $9.89KB$ to $14.36KB$, and packet frequency changes from $2pps$ to $5pps$. Lastly, for TF-IDF in Pred20 [16], the AUCPR reduced

TABLE IV: **Temporal transferability detailed Area Under the Precision-Recall Curve (AUCPR) assessment for learning-based IoT identification solutions. We refer to Week as 'W', meanwhile 'M' for Month.**

Algorithm	Experiments Setup	Baseline AUCPR	Train: Now						
			Test: 1W	Test: 2W	Test: 3W	Test: 1M	Test: 3M	Test: 6M	Test: 12M
LightGBM	Meid20 [9]	0.88	0.71	0.73	0.70	0.76	0.72	0.65	0.65
	Okui22 [11]	0.90	0.81	0.82	0.80	0.60	0.59	0.53	0.52
Random Forest	Siva18 [5]	0.87	0.87	0.87	0.87	0.83	0.80	0.29	0.37
	Pinh19 [13]	0.89	0.89	0.87	0.87	0.84	0.84	0.64	0.67
LSTM	Yang19 [7]	0.78	0.14	0.14	0.11	0.11	0.11	0.12	0.11
	Ortiz19 [8]	0.76	0.11	0.11	0.11	0.11	0.11	0.11	0.11
	Dong20 [14]	0.67	0.11	0.11	0.10	0.11	0.10	0.11	0.16
CNN	Fan22 [17]	0.68	0.14	0.14	0.14	0.11	0.10	0.11	0.11
TF-IDF	Perd20 [16]	0.80	0.60	0.60	0.60	0.65	0.61	0.62	0.54

by 30% in idle, active, and mixed scenarios. The cosine similarity of the TF-IDF feature vectors for all devices in all scenarios is between $0.1-0.3$. These similarity values indicate a change in DNS frequency patterns across different modes, which results in performance reduction.

D. Temporal Transferability

We train all models baseline using a dataset with full sampling (1:1), mixed (idle and active), and from both testbed locations. For training, we used December 2022, while for testing, we used datasets on the following dates: January 2023, March 2023, June 2023, and December 2023, using mixed-mode operations in both testbeds.

Table IV shows that for periods < 3 months the LightGBM AUCPR in Meid20 [9] reduced by 20.45%, and by 11.11% in Okui22 [11]. Equally, for periods > 3 months we see a reduced by 26.14% and 42.22% for Meid20 [9] and Okui22 [11], respectively. We measured PSI values using features from Meid20 [9] and Okui22 [11] for all train-test periods, observing slight change in the test-train distribution, *i.e.,* PSI $0.1 - 0.25$. Notably, AUCPR of LightGBM in Meid20 [9] varies over the first month, from 0.73 at 2 weeks, 0.70 by 3 weeks, and increases to 0.76 in 1 month. LightGBM attempts to adjust to new patterns in Meid20 [9], causing this behaviour. Between 2-3 weeks, there is a change in the test-train data, causing a PSI value of > 0.35. LightGBM adjusts to these patterns to improve performance between 3 weeks and 1 month to increase the AUCPR. However, beyond 1 month, there was a change in the test-train distribution PSI value, which was > 0.25. While LightGBM attempts to adjust for this change in distribution, it is too great, causing a reduction in AUCPR value.

RF in Siva18 [5] results in a reduce of AUCPR in both periods < 6 months (by 8.05%) and > 6 months (by 57.47%). For features in Siva18 [5], we measure PSI values for periods < 6 months, observing no change in the train-test distribution (PSI < 0.001). Conversely, there is a change in the train-test distribution when > 6 months, *i.e.,* PSI > 0.35. The AUCPR for RF in Pinh19 [13] reduced by in both periods < 6 months (by 5.62%) and > 6 months (by 28.09%). PSI values reveal stability in train-test distributions as the PSI value is < 0.25. The variable performance of RF in both Siva18 [5] and Pinh19 [13] indicate the model's sensitivity to temporal change, especially when PSI values indicate greater than moderate drift (PSI $0.1 - 0.25$).

The AUCPR of LSTM and CNN in Yang19 [7], Ortiz19 [8], Dong20 [14], and Fan22 [17] reduce by 85.90% across all periods. When evaluating the feature distribution, we observe a significant change in train-test distribution in all experiment setups, *i.e.,* PSI > 0.35. Indeed, the reduction in AUCPR (0.16 to 0.10) again indicates that LSTMs and CNNs overfit training patterns, failing to generalise to new patterns over time. Lastly, the AUCPR of TF-IDF in Perd20 [16] initially reduced from 0.80 during the 3 weeks to 0.60 before rising to 0.65 (*i.e.,* TF-IDF score $< 0.05 - 0.1$) in the 1 month, then reduced again to 0.54 (*i.e.,* TF-IDF score $> 0.05 - 0.1$). By empirical observation, DNS records change in the first 3 weeks and repeat at 1 month. This phenomenon is known as Domain Cycling [86, 87], where domains undergo regular changes or rotations, often reverting to previous states after certain intervals. Furthermore, this indicates that DNS domain frequency can effectively be used to distinguish between IoT devices over long temporal periods.

TABLE V: **Spatial transferability detailed Area Under the Precision-Recall Curve (AUCPR) evaluation for learning-based IoT identification solutions.**

Algorithm	Experiments Setup	Baseline AUCPR	Train: US	Train: UK
			Test: UK	Test: US
LightGBM	Meid20 [9]	0.88	0.27	0.31
	Okui22 [11]	0.90	0.26	0.50
Random Forest	Siva18 [5]	0.87	0.50	0.64
	Pinh19 [13]	0.89	0.50	0.34
LSTM	Yang19 [7]	0.78	0.20	0.20
	Ortiz19 [8]	0.76	0.20	0.20
	Dong20 [14]	0.67	0.20	0.20
CNN	Fan22 [17]	0.68	0.20	0.20
TF-IDF	Perd20 [16]	0.80	0.74	0.57

E. *Spatial Transferability*

We train all models baseline using a dataset with full sampling (1:1), mixed (idle and active), and from both testbed locations. For both training and testing, we used devices common in both testbeds, examining idle mode data collected on the $6^{th} - 27^{th}$ of December 2022 with full sampling (1:1).

Table V shows that LightGBM AUCPR in Meid20 [9] reduced by 69.32% in "train: USA, test: UK" and reduced by 64.77% in "train: UK, test: USA". Similarly, the AUCPR in Okui22 [11] reduced by 71.11% in "train: USA, test: UK" and reduced by 44.44% in "train: UK, test: USA". For each of these train-test combinations, we observe a difference in distribution as PSI values are > 0.35. Such distribution changes indicate LightGBM's difficulty in capturing changes in different spatial features.

The AUCPR of RF in Siva18 [5] reduced by 42.53% in the "train: USA, test: UK" scenario and by 26.44% in "train: UK, test: USA". The AUCPR of RF in Pinh19 [13] also reduced by 43.82% in "train: USA, test: UK" and in the "train: UK, test: USA" scenarios reduced by 61.80%. Again, we observe that these train-test combinations have differences in distribution (PSI values > 0.35). Interestingly, we can see this shift using the port feature used in Siva18 [5]. For example, the Google Home Mini's average daily frequency for destination port 123 (Network Time Protocol (NTP) [88]) was 37 in the UK and 97 in the USA. Similarly, port usage varies spatially, concretely: Ring Doorbell uses port 5001 in the UK, and Amazon Echo Show uses port 5000 in the USA. Such changes indicate challenges in generalising to different locations.

For LSTM and CNN in Yang19 [7], Ortiz19 [8], Dong20 [14], and Fan22 [17], we observe the same pattern as in previous cases the inability to generalise (AUCPR reduced by 73.68% to a consistent AUCPR of 0.20), due to the change in distribution between locations (PSI values > 0.35). Lastly, the AUCPR of TF-IDF in Perd20 [16] reduced by 7.5% in the "train: USA, test: UK" scenario, and reduced by 29% in the "train: UK, test: USA" scenario. Our empirical observations show that devices query more domains in the US than in the UK. We observed 56 unique domains in the USA and 35 in the UK during the same 21-day observation period, with all devices across both testbeds operating in idle mode. Concretely, Amazon Echo Show queries 25 domains in the US and 22 in the UK. Additionally, there are uneven queries per domain across regions, *e.g.,* DNS queries to 'amazonalexa.com' are more frequent in the USA (442 per day) than in the UK (180 per day). This is further evident by the similarity scores between the USA and UK domains being < 0.05, but > 0.05 between the UK and USA domains. Such variance is due to several factors. USA domains act as testing grounds for updates and beta testing, so devices access more servers, causing more DNS queries [89]. Complex Content Delivery Network (CDN) routing in the USA allows connections to a broader range of endpoints, increasing diversity in how DNS queries are resolved [90]. Lastly, USA data privacy laws permit frequent data collection, causing numerous DNS queries for analytics, while UK laws are less permissive [89–91]. These variations indicate that TF-IDF lacks the flexibility to adapt to the distinct characteristics of IoT data across different geographical regions.

TABLE VI: **Observability detailed Area Under the Precision-Recall Curve (AUCPR) evaluation for learning-based IoT identification solutions. We use / when the features are packet-based the *Softflowd* could not capture it.**

Algorithm	Experiments Setup	Baseline AUCPR	Train: 1:1		
			Test: 1:100	1:1000	1:5000
LightGBM	Meid20 [9]	0.88	0.30	0.23	0.25
	Okui22 [11]	0.90	/	/	/
Random Forest	Siva18 [5]	0.87	0.29	0.25	0.25
	Pinh19 [13]	0.89	/	/	/
LSTM	Yang19 [7]	0.78	/	/	/
	Ortiz19 [8]	**0.76**	0.071	0.072	0.071
	Dong20 [14]	0.67	0.071	0.072	0.071
CNN	Fan22 [17]	0.68	/	/	/
TF-IDF	Perd20 [16]	0.80	/	/	/

F. *Observability*

We train all models baseline using a dataset with full sampling (1:1), mixed (idle and active), and from both testbed locations. We test 1:1 against sample rates of 1:100, 1:1000, and 1:5000 collected in April 2024. Table VI shows the AUCPR of LightGBM in Meid20 [9] reduced with the sample rate (65.91% at 1:100, by 73.86% at 1:1000, by 71.59% at 1:5000). Our empirical observation that the increased number of packets reduced the granularity of IoT device behaviours. For example, the number of packets sampled per-flow for Amazon Echo Show at at full sample(1:1) the mean is $\mu = 1.0438 \times 10^5$ with a standard deviation of $\sigma = 3.9539 \times 10^4$ of mean packet frequency increase by 11983.8% at 1:100, by 93949.1% at 1:1000, by 396057.3% at 1:5000). In such cases, static feature-based models such as LightGBM struggle to capture changing patterns, causing a reduce in performance when varying data sampling. The AUCPR of RF in Siva18 [5] also reduced with the sample rate (66.67% in 1:100, by 71.26% in both 1:1000 and 1:5000). Similarly to LightGBM, we observe high variance in features in Siva18 [5], particularly between the 100- and 1000-second intervals. For example, flow volume for the TP-Link Lightbulb increased from $0.0004MB$ ($\sigma = 0.001MB$) at $1s$ to $0.44MB$ ($\sigma = 7.59MB$) at 1:100, to $1.93MB$ ($\sigma = 3.33MB$) at 1:1000, to $0.39MB$ ($\sigma = 0.66MB$) at 1:5000. This makes it harder to identify the patterns, preventing RF from fully capturing the necessary dynamics. Lastly, for LSTM in Ortiz19 [8] and Dong20 [14], we observe a reduction in AUCPR (by 90.66% to a consistent AUCPR of 0.0071). For example, in Ortiz19 [8], the increased number of TCP packets reduces the granularity of IoT device behaviours. The number of TCP packets sampled per-flow for WEMO Plug at at full sample(1:1) the mean is $\mu = 1.2289 \times 10^4$ with a standard deviation of $\sigma = 3.6884 \times 10^3$ increase by 7382.5% at 1:100, by 179742.3% at 1:1000, by

382868.6% at 1:5000. In such cases, the increase in sampled data results in losing important temporal patterns on which the LSTM was trained, leading to a significant reduce and lack of generalisation.

G. *Takeaways*

The results in this section indicate that network operators would benefit from RF models with feature sets that reflect device outbound traffic patterns such as those seen in Pinh19 [13] as they generalise well, particularly across modes of operation. Training the model in idle mode and then conducting predictions for different periods was also beneficial as IoT devices remain idle for extended periods, and the range of possible behaviours beyond that are too large to model.

Models generally degrade in performance over time due to the changes in device capability, *e.g.,* updates or functionalities. Combining LightGBM for longer-term predictions and RF for shorter-term identification can achieve more consistent temporal performance, as seen from their AUCPR in Table IV. Additionally, retaining models more frequently (*e.g.,* every two weeks) helps prevent the drop-off in temporal degradation.

We observe models with DNS query (as in Siva18 [55]) features reduce temporal degradation. As such, we suggest using RF with DNS frequency. We reason this is due to the cyclical nature of DNS domains combined with the reduced overfitting from the ensemble learning of RF. We also see DNS frequency as a useful feature for spatial generalisation using TF-IDF. This is because TF-IDF represents DNS domains more nuancedly, capturing the significance of specific domains for spatial identification.

Our results (Table VI) show that full packet captures sample and transferring to higher sample rates reduces performance. While using full packet sampling benefits test time identification, it can be impractical in larger networks (*e.g.,* overhead, packet loss, packet delay). Thus, alternative techniques can be employed, such as network segmentation, where networks can be divided into smaller segments.

V. Explainability Results

In this section, we explain the reduction in AUCPR performance observed in Section IV using ML explainability.

We selected experiments using packet- or flow-level features, as explained in Section III-F. We analyse the following papers, Meid20 [9], Okui22 [11], Siva18 [5], and Pinh19 [13]. See Table II for the full features of each paper.

Figure 2 (a-d) shows Meid20 [9], Okui22 [11], Siva18 [5], and Pinh19 [13] projected into a 2D space using t-SNE features from the original papers. We use Amazon Echo Show, Google Home Mini, Ring Doorbell, TP-Link Lightbulb, TP-Link Plug, Philips Hub, Wansview Camera, WEMO Plug, and Yi Camera. We focused on device-type instead of vendors because our primary goal is to evaluate features. Device-type identification presents more distinct and separable characteristics than vendor-specific identification. For example, a smart thermostat and a smart speaker exhibit different network traffic patterns based on their functions, making them easier to differentiate by device type. Two smart speakers from different brands (*e.g.,* Amazon and Google) perform similar functions with subtle network differences, making vendor-specific identification far less distinct. However, we observe no clear separation by device-type. The clusters are neither tightly grouped nor distinctly separated.

Figure 2 (e-h) shows the SHAP beeswarm plot for Meid20 [9], Okui22 [11], Siva18 [5], and Pinh19 [13] for the baseline training dataset. Figure 2(e-h) shows that 0.33% of the total features have zero SHAP values in different experiments. The zero SHAP value features do not contribute to the model predictions. In Meid20 [9], features such as *TCP Urgent (URG), Congestion Window Reduced (CWR)* and *Explicit Congestion Notification (ECN) flags* have zero SHAP values. In Okui22 [11], the *total forward packet count* feature has zero SHAP value. In Siva18 [5], the *duration of NTP/DNS* features have zero SHAP values. We observed features with higher absolute SHAP values, which indicates their significance. For example, in Meid20 [9] *TCP Finish (FIN) flag* showed a high SHAP value, in Siva18 [5] *second level domain (SLD) entropy* a high SHAP value. Other features, such as *destination port* in Meid20 [9] and Siva18 [5], have minimal impact on the model's predictions.

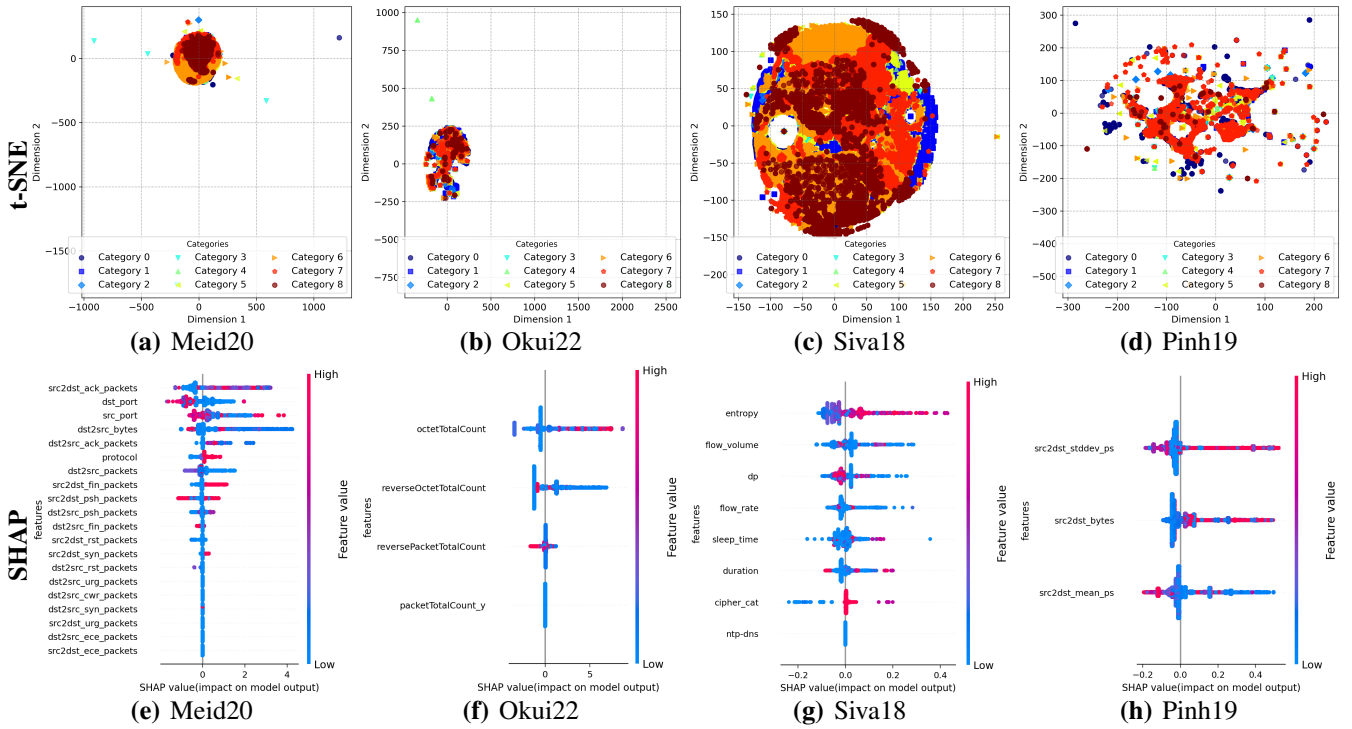
The lack of distinct clustering in the t-SNE analysis and the presence of zero SHAP values show that the features used in the training datasets lack sufficient discriminatory power to differentiate between device categories.

Next, for Meid20 [9], Okui22 [11], Siva18 [5], and Pinh19 [13], we present the most important features for the testing datasets. For space limitation, we provide empirical interpretations for each feature only once, unless a feature's behaviour changes across different practicality attributes.

A. *Mode of Operation*

For the full permutation of the three modes of operation using Meidan20 [9] we find that *TCP CWR flag* is the most important feature with a median of 2.60. The *TCP ECN-Echo (ECE) flag* is the least important with a median of 0.0. In the single permutation of the three modes of operation, the *TCP Push (PSH) flag* is the most important feature with a median of 0.006, while the *TCP ECE flag* is the least important feature with a median 0.0. The above results occurred because devices with high data rates, such as Yi Camera, Wansview Camera, and Ring Doorbell, set *TCP CWR flag* more frequently than devices with lower data rates, such as TP-Link Lightbulb or a thermostat. The Yi Camera set the *TCP CWR flag* 10 times per hour in idle mode and 5 times per hour in active mode while streaming video. The TP-LINK Lightbulb set *TCP CWR flag*, 1-3 times per day in both idle and active modes. Similarly, devices such as Yi Camera and Wansview Camera use *TCP PSH flag* more frequently than devices such as TP-Link Lightbulb or Plug. We observed that all devices set *TCP ECE flag* in response to congestion in the same way, regardless of their type.

For the full permutation of the three modes of operation using Okui22 [11] we find that *total forward packet* is the



(a) Meid20 **(b)** Okui22 **(c)** Siva18 **(d)** Pinh19

(e) Meid20 **(f)** Okui22 **(g)** Siva18 **(h)** Pinh19

Fig. 2: **t-SNE analysis, and SHAP of IoT devices from the UK testbed, from 6^{th} to the 27^{th} of December 2022. Devices' modes of operation were mixed. We use 0: Amazon Echo Show, 1: Google Home Mini, 2: Ring Doorbell, 3: TP-Link Lightbulb, 4: TP-Link Plug, 5: Philips Hue Hub, 6: Wansview Camera, 7: WEMO Plug, 8: Yi Camera.**

most important feature with a median of 0.50. The *total reverse packet* is the least important with a median of 0.12. In the single permutation of the three modes of operation, *total forward packet* is the most important with a median of 0.51. The *total reverse packet* remains the least important with a median of 0.12. *total forward packet* is the total sending packet. The above results occurred because devices such as Philips Hue Hub, TP-Link Lightbulb, and WEMO Plug report usage data every hour to their manufacturer's cloud. Devices such as Amazon Echo Show, Google Home Mini, and Roku TV generate traffic based on user interactions, causing event-triggered traffic patterns. The *total reverse packet* are the packets the device receives. In idle mode, devices from the same vendor, such as the TP-Link Lightbulb and TP-Link plug, receive approximately 50 packets from the TP-Link Server.

For the full permutation of the three modes of operation using Siva18 [5], the *SLD entropy* is the most important feature with a median of 0.11. The *flow duration* is the least important with a median of -0.02. However, for the single permutation, *flow rate* becomes the most important with a median of 0.06, and *cipher suites* is the least important with a median of 0.03. Siva18 [5] used entropy to measure the most common internet domain contacted by the device. The top domain used by Yi Camera in idle mode is 'www.yitechnology.com' for status updates. While in active mode, while steaming video, the Yi Camera connects to 'www.yitechnology.com' and 'www.yicloud.com'. The top domains used by the Amazon Echo Show in idle mode are 'alexa.amazon.com', 'api.amazonalexa.com', and 'devices.amazon.com', and in active mode are 'alexa.amazon.com', 'video.amazon.com', and 'api.music.amazon.com'. The top domain for the TP-Link Lightbulb in idle and active modes is 'cloud.tplinkwifi.com'.

For the full permutation of the three modes of operation using Pinh19 [13], the *total transmitted bytes in one minute* is the most important feature with a median of 0.15. The *mean of transmitted bytes in one minute* is the least important with a median of 0.09. For the single permutation of the three modes of operation, the *mean of transmitted bytes in one minute* is the most important feature with a median of 0.07, and *total transmitted bytes in one minute* is the least important with a median of 0.04, showing a reversal in feature importance. *total transmitted bytes in one minute* and *total forward packets* are different names for the same feature and exhibit the same behaviour. However, *total transmitted bytes in one minute* and *mean of transmitted bytes in one minute* are perfectly correlated. This correlation explains the reversal in feature importance.

B. *Temporal Transferability*

For the full permutation of the seven periods using Meid20 [9], the *TCP CWR flag* is the most important feature with a median of 3.3. The *TCP ECE flag* is the least important with a median of 0.0. For the single permutation of the seven periods, the *total bidirectional bytes* is the most important feature with a median of 0.002. The *TCP ECE flag* is the least

important with a median of 0.0. The above results occurred because devices such as Yi Camera, Wansview Camera, and Ring Doorbell set *TCP CWR flag* more frequently than devices such as TP-Link Lightbulb over the seven periods. We observe a trend of consistent bytes traffic from scheduled devices such as Philips Hue Hub and TP-Link Lightbulb. Interactive devices such as the Amazon Echo Show have a sporadic traffic pattern.

For the full and single permutations of the seven periods using Siva18 [5], the *destination port* is the most important feature with a median of 0.07. For the full permutation of the seven periods, the *sleep time* is the least important with a median of 0.05. For the single permutation, the *flow rate* is the least important with a median of -0.003. The above results occurred because TP-Link devices use port 9999 for communication with their cloud services over the seven periods. Amazon Echo Show uses port 3478 for STUN (Session Traversal Utilities for NAT), which helps with NAT traversal for VoIP and video calls, and Wansview Cameras uses port 8899 for their cloud communication and for accessing the camera's stream. We observed that devices such as the Philips Hue Hub might have had fixed sleep times, resulting in predictable intervals of activity (*e.g.,* sending packets every hour). Later, after a set of inactivity duration, the device enters sleep mode (longer than the initial interval). We observed a significant overlap between devices' flow rate (ratio between flow volume and flow duration), for example, Amazon Echo Show and Google Home Mini, Amazon Echo Show and Yi Camera.

For the full and single permutations of the seven periods using Okui22 [11], the *total forward bytes* is the important feature with a median of 0.23. The *packet total count* and *packet total count* are the least important, with a median of 0.0. We observe consistent byte traffic (*total forward bytes*) from scheduled devices such as Philips Hue Hub and TP-Link Lightbulb. Interactive devices such as the Amazon Echo Show have a sporadic traffic pattern. We observed the same total packet count for the Amazon Echo Show, Google Home Mini, Wansview Camera, Yi Camera, TP-Link Plug, and WEMO Plug.

For the full and single permutations of the seven periods using Pinh19 [13], the *total transmitted bytes in one minute* is the most important feature with a median of 0.31. The *standard deviation of transmitted bytes in one minute* is the least important with a median of 0.20. For the single permutation of the seven periods, the *standard deviation of transmitted bytes in one minute* is the most important with a median of 0.16. The *transmitted bytes in one minute* is the least important with a median of 0.09. The above results occurred because we observed a trend of consistent transmitted bytes traffic from scheduled devices such as Philips Hue Hub and TP-Link Lightbulb. Interactive devices such as the Amazon Echo Show have a sporadic traffic pattern. However, *total transmitted bytes in one minute* and *standard deviation of transmitted bytes in one minute* are perfectly correlated. This correlation explained the reversal in feature importance.

C. *Spatial Transferability*

For the single and full permutations of both the UK and the USA datasets Meid20 [9], the *destination port* is the most important feature with a median of 0.10. The *TCP ECE flag* and *TCP CWR flag* are the least important, with a median of 0.0. The above results occurred because we observe that in both the UK and USA, devices such as Yi Camera, Wansview Camera, and Ring Doorbell, which have a high data transmission rate, set *TCP CWR flag* more frequently than devices such as TP-Link Lightbulb or thermostat, which have low data transmission rate. Devices in the USA are more likely to set *TCP CWR flag* than devices in the UK. The USA does not have a robust regulatory framework for IoT device services and advertisement.

For the single and full permutations of both the UK and the USA datasets using Okui22 [11], the *total forward bytes* is the most important feature, with a median of 0.50. The *packet total count* is the least important with a median of 0.0. The above results occurred because we observe that *total forward bytes* in the USA is higher than in the UK, even though both regions show similar trends. This difference is mainly due to the greater freedom that devices in the USA have regarding GDPR and the range of connections they can establish. We observed the same devices' packet count in both locations, for example, TP-Link Lightbulb (US) and Google Home Mini (UK).

For the single and full permutations of both the UK and the USA datasets using Siva18 [5], the *flow volume* is the most important feature with a median of 0.13. The *cipher suites* is the least important with a median of -0.003. For the single permutation of the UK and the USA datasets, the *flow rate* is the most important feature with a median of 0.07. The *flow duration* is the least important with a median of -0.05. The above results occurred because of an observed *flow volume* in the USA that is more than double that in the UK, although both regions show similar trends. This difference is mainly due to the greater freedom that devices in the USA have regarding GDPR and the range of connections they can establish. We observed *cipher suites* for the same device in the UK and USA are different, again due to the existence of GDPR in the UK. We observed the same value for devices' *flow rate* and *flow duration* in both locations, for example, YeeLight (US) and TP-Link Lightbulb (UK).

For the single and full permutations of both the UK and the USA datasets using Pinh19 [13], the *total transmitted bytes in one minute* is the most important feature with a median of 0.11. The *standard deviation of transmitted bytes in one minute* is the least important feature with a median of 0.023.

D. *Observability*

Our observability results are limited to Meid20 [9] and Siva18 [5] because the sampling rate used in the experimental set-up decreased the number of features [30, 31, 74] (see Section IV-F).

For the single and full permutations of four sampling rates using Meidan20 [9], the *flow duration* is the most important

feature with a median of 0.31. The *incoming bytes* is the least important feature with a median of 0.002. For the single permutation, the *incoming bytes* is the most important feature with a median of 0.14. The *flow duration* is the least important feature with a median of 0.04. At a 1:100 sampling rate with a single permutation, we lost the exact flow duration due to sampling gaps compared to a sampling rate of 1:1. We observed only flow duration from devices that transmitted continuously (*e.g.,* Yi Camera). We could still observe information for *incoming bytes*. The disrupted data pattern makes *incoming bytes* more stable and indicative of device type. At 1:1000 and 1:5000 sampling rates with full permutations, *incoming bytes* vary widely compared to a 1:1 sampling rate. We observed that the Yi Camera has a longer *flow duration* because it streams video, while the TP-Link Plug has a shorter flow duration, only sending intermittent data for status updates. We were able to observe how long each device transmits data continuously.

For the full permutations of four sampling rates using Siva18 [5], the *flow duration* is the most important feature with a median of 0.11. The *flow rate* is the least important feature with a median of -0.01. For the single permutation of four sampling rates, the *flow rate* is the most important with a median importance of 0.07. The *flow rate* and *flow duration* are perfectly correlated. This correlation explains the reversal in feature importance. The *destination port* is the least important with a median of 0.02. With sampling, we could not observe the ports 9999 used by TP-Link devices, 3478 for the Amazon Echo Show, and port 8899 for the Wansview Camera.

E. *Takeaways*

The results in this section indicate the significance of feature representation. Simple first-order statistical features (*e.g.,* mean, variance) may not capture distributional information effectively, and instead, features such as entropy are more suitable. Features with high variance between high- and low-transmission devices should also be avoided. Such variance can lead to overlapping patterns, reducing the ability to distinguish device types accurately. We suggest using features with binary values, such as being set or not set. An example is the TCP congestion windowing flag, which can provide a clear, non-overlapping indicator of device transmission behaviour.

VI. Evaluation Improvements

In this section, we explore feature selection modifications aimed at enhancing IoT device identification performance.

A. *Experimental Setup*

Collected Dataset. We used the same traffic traces from the two testbeds at different periods (*i.e.,* 2022, 2023, 2024). We used 21 days to capture the behaviours of IoT devices.

Learning Model. Section IV-G showed that RF consistently shows the least decline in performance for IoT device identification performance across all practicality evaluation scenarios. Through the practicality of implementing the improvement process, we used an RF model as a baseline using tailored features and hyper-parameter tuning the model. The resulting model is an RF with 121 trees, each with a maximum depth of 13.

Tailored Features. We compiled a list of over 117 features from the literature review, removed duplicates, and eliminated correlated features to ensure independence Figure 3 in Appendix C. This process reduced the feature set to 81. We evaluated the importance of these features, discarding any with zero importance. Only features with an importance score above 0.5, which significantly impact the model's predictions, were retained. Using the Pareto Principle (80/20 rule) [75, 92], we identified that 20% of the features often explain 80% of the performance. From the 81 features, we identified 17 that aligned with this principle. Further feature importance analysis showed that only 10 features consistently dominated. Consequently, we removed seven features with zero importance to finalise the selection. We focused on the top 10 because fewer features allow us to understand decision-making better. For example, the feature 'sleep time' may suggest a device is a TP-Link Lightbulb. When the feature 'sleep time' is combined with 'packet size', it becomes clear that the device is an Amazon Echo Show. The final tailored feature list is: ***source port, destination port, second level domain entropy, sleep time, flow volume, TCP flags, DNS queries, cipher-suites, maximum inter-arrival time, TLS handshaking***.

Baseline. The baseline samples were from both geographical locations, with devices operating in mixed modes at a 1:1 sample rate. Starting with the tailored feature list, we systematically reduced the feature set by removing those with the lowest importance. After each reduction, we retrained the model, establishing a new baseline. We then evaluated the retrained model on the scenarios. This reduction continued until we identified the minimal feature set that maintained acceptable performance across all scenarios.

B. *Improvement Results*

Table VII shows that Iteration 1 is the most effective across all evaluated scenarios. Iteration 1 achieves AUCPR values exceeding 80% in 70% of scenarios. Iteration 1 shows the best performance in the following scenarios: Train: Mix, Test: Idle 0.64, Train: Mix, Test: Active with 0.69, Train: Mix, Test: Mix with 0.42, Train: US, Test: UK with 0.80, Test: 1 month with 0.93, and Test: 3 months with 0.82. The PSI values for tailored features across all these scenarios maintain a value < 0.01, with (K-S) test p-values > 0.05, indicating that the distributions are similar. However, Iteration 1 AUCPR in mixed scenario reduced from 0.94 to 0.42. The PSI values for tailored features in mixed mode are > 0.30, indicating that the distributions have changed. This decrease in AUCPR shows that mixed operational modes are challenging for the model.

Table VII shows that Iteration 6 is the worst across all evaluated scenarios, achieving AUCPR values < 0.60 in all scenarios. For example, the performance in the mixed mode AUCPR reduced from 0.75 to 0.14. Similarly, in Iteration 6, AUCPR reduced from 0.75 to 0.16 in the temporal transfer-

TABLE VII: **Improvement evaluation results Area Under the Precision-Recall Curve (AUCPR) for IoT identifications tailored features, under device mode of operation and transferability metrics. We use 'M' to refer to months.**

			Mode of Operation			Spatial Transferability		Temporal Transferability			
Iteration	**Features Subset**	**Baseline AUCPR**	**Train: Mix Test: Idle**	**Train: Mix Test: Active**	**Train: Mix Test: Mix**	**Train: UK Test: US**	**Train: US Test: UK**	**Train: Now Test: 1M**	**Train: Now Test: 3M**	**Train: Now Test: 6M**	**Train: Now Test: 12M**
0	dst port, src port SLD entropy, sleep time TCP flags, flow volume Cipher-suites, DNS queries TLS handshaking MAX inter-arrival time	0.94	0.62	0.29	0.29	0.91	0.86	0.85	0.84	0.27	0.33
1	dst port, src port SLD entropy, sleep time TCP flags, DNS queries MAX inter-arrivaltime	0.98	0.64	0.69	0.42	0.74	0.80	0.93	0.82	0.27	0.30
2	dst port, SLD entropy, Sleep time MAX inter-arrivaltime	0.97	0.63	0.68	0.42	0.75	0.79	0.93	0.80	0.28	0.30
3	dst port, SLD entropy, flow volume	0.96	0.36	0.36	0.36	0.73	0.78	0.91	0.80	0.28	0.31
4	dst port, SLD entropy, sleep time	0.98	0.35	0.91	0.91	0.80	0.80	0.83	0.74	0.26	0.31
5	Sleep time, SLD entropy	0.90	0.52	0.72	0.23	0.66	0.71	0.84	0.69	0.23	0.20
6	Sleep time	0.75	0.34	0.43	0.14	0.55	0.56	0.53	0.49	0.16	0.15

ability tests. The PSI between the training and testing dataset features was > 0.35. These PSI results show that the feature distribution has changed. Iteration 2 and 4 succeeded in 7 out of 10 scenarios. For example, in Iteration, 3 recorded success in 6 out of 10 scenarios, with an AUCPR of 0.96. However, the AUCPR for the idle, active, and mixed modes was only 0.36. Iteration 5 succeeded in 6 out of 10 scenarios while the baseline AUCPR of 0.90 reduced to 0.52 in idle mode, 0.32 mix mode, and the temporal transferability was 0.20 at 12 months. Across all these iterations, the PSI between the training and testing dataset features was > 0.35. These values show that the feature distributions remain unchanged.

C. *Takeaways*

The results in this section indicate that a broad range of features better capture the device behaviours, maintaining high AUCPR scores, while models with single features suffer when distribution shifts. Thus, we suggest that models be designed to learn the most relevant environmental features. For example, adaptive algorithms can select important features based on the real-time context, such as mode of operation, transferability, or observability.

VII. Limitations

The data used in our evaluation is limited because it does not cover all IoT smart home categories; for example, we did not include smart kitchen appliances. This was done to ensure that the same model of IoT device was used in both US and UK labs to ensure data comparability. Our data included only Wi-Fi IoT communication. We excluded Zigbee and Z-Wave to establish a solid methodological foundation before extending to other communication technologies. Including multiple protocols would add significant complexity due to differences in their communication stacks, hardware requirements, and operational behaviours. Future evaluation would include other IoT communication protocols.

Our data analysis methods were limited by excluding feature selection techniques such as Mutual Information Analysis (MIA) or Principal Component Analysis (PCA). We did this to focus on the features examined in the original works that we evaluated. However, adding MIA or PCA to future work could offer further insights into how feature combinations contribute to device identification, revealing the most relevant features that provide complexity reduction.

We excluded deep learning features from our ML explainability as those features have complex transformations and interactions within neural network architectures. This complexity arises because the model processes these features across multiple and embedded layers, making it hard to trace their contributions to the final decision [82, 83]. Future evaluation could explore deep learning explainability.

VIII. Conclusion

This work rigorously evaluated ML-based IoT device identification solutions using curated datasets and model explainability techniques. Our evaluation revealed key factors that impact the performance of IoT identification solutions in real-world settings, including variability in device operation modes, spatial and temporal factors, and observability, which are crucial for situational awareness of network operators' needs. We recommend that network operators retrain and evaluate model performance weekly to maintain fidelity above the required threshold. However, this retraining raises the crucial question of the minimum amount of data network operators must have. In future work, we aim to investigate how updating IoT devices impacts their communication patterns, potentially leading to false positives or incorrect identification. Additionally, we will explore systematic methods for evaluating deep-learning features in IoT device identification.

References

[1] X. Feng, Q. Li, H. Wang, and L. Sun, "Acquisitional rule-based engine for discovering internet-of-things devices,"

in *27th USENIX Security Symposium (USENIX Security 18)*, 2018.

[2] S. J. Saidi, A. M. Mandalari, R. Kolcun, H. Haddadi, D. J. Dubois, D. Choffnes, G. Smaragdakis, and A. Feldmann, "A haystack full of needles: Scalable detection of IoT devices in the wild," in *Proceedings of the ACM Internet Measurement Conference*, 2020.

[3] T. Hu, D. J. Dubois, and D. Choffnes, "Behaviot: Measuring smart home iot behavior using network-inferred behavior models," in *Proceedings of the 2023 ACM on Internet Measurement Conference*, 2023.

[4] A. Hamza, D. Ranathunga, H. H. Gharakheili, T. A. Benson, M. Roughan, and V. Sivaraman, "Verifying and monitoring IoT network behavior using MUD profiles," *IEEE Transactions on Dependable and Secure Computing*, 2020.

[5] A. Sivanathan, H. H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath, and V. Sivaraman, "Classifying iot devices in smart environments using network traffic characteristics," *IEEE Transactions on Mobile Computing*, 2018.

[6] S. Marchal, M. Miettinen, T. D. Nguyen, A.-R. Sadeghi, and N. Asokan, "Audi: Toward autonomous IoT device-type identification using periodic communication," *IEEE Journal on Selected Areas in Communications*, 2019.

[7] K. Yang, Q. Li, and L. Sun, "Towards automatic fingerprinting of IoT devices in the cyberspace," *Computer Networks*, 2019.

[8] J. Ortiz, C. Crawford, and F. Le, "DeviceMien: Network device behavior modeling for identifying unknown IoT devices," in *Proceedings of the International Conference on Internet of Things Design and Implementation*, 2019.

[9] Y. Meidan, V. Sachidananda, H. Peng, R. Sagron, Y. Elovici, and A. Shabtai, "A novel approach for detecting vulnerable IoT devices connected behind a home NAT," *Computers & Security*, 2020.

[10] I. Cvitić, D. Peraković, M. Periša, and B. Gupta, "Ensemble machine learning approach for classification of IoT devices in smart home," *International Journal of Machine Learning and Cybernetics*, 2021.

[11] N. Okui, M. Nakahara, Y. Miyake, and A. Kubota, "Identification of an IoT device model in the home domain using IPFIX records," in *2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, 2022.

[12] V. Thangavelu, D. M. Divakaran, R. Sairam, S. S. Bhunia, and M. Gurusamy, "Deft: A distributed iot fingerprinting technique," *IEEE Internet of Things Journal*, 2018.

[13] A. J. Pinheiro, J. de M. Bezerra, C. A. Burgardt, and D. R. Campelo, "Identifying IoT devices and events based on packet length from encrypted traffic," *Computer Communications*, 2019.

[14] S. Dong, Z. Li, D. Tang, J. Chen, M. Sun, and K. Zhang, "Your smart home can't keep a secret: Towards automated fingerprinting of IoT traffic," in *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*, 2020.

[15] L. Yu, B. Luo, J. Ma, Z. Zhou, and Q. Liu, "You are what you broadcast: Identification of mobile and IoT devices from (public) WiFi," in *29th USENIX Security Symposium (USENIX Security 20)*, 2020.

[16] R. Perdisci, T. Papastergiou, O. Alrawi, and M. Antonakakis, "IoTfinder: Efficient large-scale identification of IoT devices via passive DNS traffic analysis," in *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2020.

[17] L. Fan, L. He, Y. Wu, S. Zhang, Z. Wang, J. Li, J. Yang, C. Xiang, and X. Ma, "AutoIoT: Automatically Updated IoT Device Identification With Semi-Supervised Learning," *IEEE Transactions on Mobile Computing*, 2022.

[18] J. Zhao, Q. Li, J. Sun, M. Dong, K. Ota, and M. Shen, "Efficient iot device identification via network behavior analysis based on time series dictionary," *IEEE Internet of Things Journal*, 2023.

[19] R. Trimananda, J. Varmarken, A. Markopoulou, and B. Demsky, "Packet-level signatures for smart home devices," in *Network and Distributed Systems Security (NDSS) Symposium*, 2020.

[20] N. N. V. Database, "CVE-2021-28372," https://nvd.nist.gov/vuln/detail/CVE-2021-28372, 2021.

[21] A. Hamza, D. Ranathunga, H. H. Gharakheili, M. Roughan, and V. Sivaraman, "Clear as mud: Generating, validating and applying iot behavioral profiles," in *Proceedings of the 2018 Workshop on IoT Security and Privacy*, 2018.

[22] A. Mudgerikar, P. Sharma, and E. Bertino, "Edge-based intrusion detection for iot devices," *ACM Transactions on Management Information Systems (TMIS)*, 2020.

[23] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," *Machine Learning*.

[24] Y. Wan, K. Xu, F. Wang, and G. Xue, "Characterizing and mining traffic patterns of iot devices in edge networks," *IEEE Transactions on Network Science and Engineering*, 2020.

[25] S. Dadkhah, H. Mahdikhani, P. K. Danso, A. Zohourian, K. A. Truong, and A. A. Ghorbani, "Towards the development of a realistic multidimensional IoT profiling dataset," in *2022 19th Annual International Conference on Privacy, Security & Trust (PST)*. IEEE, 2022.

[26] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A.-R. Sadeghi, and S. Tarkoma, "Iot sentinel: Automated device-type identification for security enforcement in iot," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017.

[27] R. A. Sharma, E. Soltanaghaei, A. Rowe, and V. Sekar, "Lumos: Identifying and Localizing Diverse Hidden IoT Devices in an Unfamiliar Environment," in *31st USENIX Security Symposium (USENIX Security 22)*, 2022.

[28] J. Holland, P. Schmitt, N. Feamster, and P. Mittal, "New directions in automated traffic analysis," in *Proceedings*

of the 2021 ACM SIGSAC Conference on Computer and Communications Security, 2021.

[29] CISA, "The Internet of Things: Impact on Public Safety Communications," https://www.cisa.gov/sites/default/files/publications/CISA%20IoT%20White%20Paper_3.6.19%20-%20FINAL.pdf#:~:text=URL%3A%20https%3A%2F%2Fwww.cisa.gov%2Fsites%2Fdefault%2Ffiles%2Fpublications%2FCISA%2520IoT%2520White%2520Paper_3.6.19%2520.

[30] J. Mahdavi and J. Mahdavi, "InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks," 2001. [Online]. Available: https://rfc-editor.org/rfc/rfc3176.txt

[31] sFlow.org, "sFlow Version 5 Specification," https://sflow.org/sflow_version_5.txt, 2012.

[32] T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *IEEE Communications Surveys & Tutorials*, 2008.

[33] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *2010 IEEE Symposium on Security and Privacy*. IEEE, 2010.

[34] D. Arp, E. Quiring, F. Pendlebury, A. Warnecke, F. Pierazzi, C. Wressnegger, L. Cavallaro, and K. Rieck, "Dos and don'ts of machine learning in computer security," in *31st USENIX Security Symposium (USENIX Security 22)*, 2022.

[35] R. Beltiukov, W. Guo, A. Gupta, and W. Willinger, "In search of netunicorn: A data-collection platform to develop generalizable ml models for network security problems," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '23, 2023. [Online]. Available: https://doi.org/10.1145/3576915.3623075

[36] A. S. Jacobs, R. Beltiukov, W. Willinger, R. A. Ferreira, A. Gupta, and L. Z. Granville, "Ai/ml for network security: The emperor has no clothes," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022.

[37] A. D'Amour, K. Heller, D. Moldovan, B. Adlam, B. Alipanahi, A. Beutel, C. Chen, J. Deaton, J. Eisenstein, M. D. Hoffman *et al.*, "Underspecification presents challenges for credibility in modern machine learning," *Journal of Machine Learning Research*, 2022.

[38] R. Kolcun, D. A. Popescu, V. Safronov, P. Yadav, A. M. Mandalari, R. Mortier, and H. Haddadi, "Revisiting IoT device identification," *arXiv preprint arXiv:2107.07818*, 2021.

[39] D. Ahmed, A. Das, and F. Zaffar, "Analyzing the feasibility and generalizability of fingerprinting internet of things devices," *Proceedings on Privacy Enhancing Technologies*, 2022.

[40] "Remote physical device fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, 2005.

[41] R. M. Gerdes, T. E. Daniels, M. Mina, and S. F. Russell, "Device identification via analog signal fingerprinting: A matched filter approach."

[42] S. J. Murdoch, "Hot or not: Revealing hidden services by their clock skew," in *Proceedings of the 13th ACM conference on Computer and communications security*, 2006.

[43] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, "Wireless device identification with radiometric signatures," in *Proceedings of the 14th ACM international conference on Mobile computing and networking*, 2008.

[44] J. Franklin, D. McCoy, P. Tabriz, V. Neagoe, J. V. Randwyk, and D. Sicker, "Passive data link layer 802.11 wireless device driver fingerprinting." in *USENIX Security Symposium*, 2006.

[45] L. C. C. Desmond, C. C. Yuan, T. C. Pheng, and R. S. Lee, "Identifying unique devices through wireless fingerprinting," in *Proceedings of the first ACM conference on Wireless network security*, 2008.

[46] S. Bratus, C. Cornelius, D. Kotz, and D. Peebles, "Active behavioral fingerprinting of wireless devices," in *Proceedings of the first ACM conference on Wireless network security*, 2008.

[47] M. H. Mazhar and Z. Shafiq, "Characterizing smart home iot traffic in the wild," in *2020 IEEE/ACM Fifth International Conference on Internet-of-Things Design and Implementation (IoTDI)*. IEEE, 2020.

[48] D. Kumar, K. Shen, B. Case, D. Garg, G. Alperovich, D. Kuznetsov, R. Gupta, and Z. Durumeric, "All things considered: An analysis of {IoT} devices on home networks," in *28th USENIX security symposium (USENIX Security 19)*, 2019.

[49] A. Orrevad, "M2m traffic characteristics: When machines participate in communication," 2009.

[50] M. Laner, P. Svoboda, N. Nikaein, and M. Rupp, "Traffic models for machine type communications," in *ISWCS 2013; The Tenth International Symposium on Wireless Communication Systems*. VDE, 2013.

[51] H. Kim, K. C. Claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee, "Internet traffic classification demystified: myths, caveats, and the best practices," in *Proceedings of the 2008 ACM CoNEXT conference*, 2008.

[52] W. Zhou, L. Dong, L. Bic, M. Zhou, and L. Chen, "Internet traffic classification using feed-forward neural network," in *2011 International conference on computational problem-solving (ICCP)*. IEEE, 2011.

[53] T. Auld, A. W. Moore, and S. F. Gull, "Bayesian neural networks for internet traffic classification," *IEEE Transactions on neural networks*, 2007.

[54] X. Xie, B. Yang, Y. Chen, L. Wang, and Z. Chen, "Network traffic classification based on error-correcting output codes and nn ensemble," in *2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery*. IEEE, 2009.

[55] A. Sivanathan, D. Sherratt, H. H. Gharakheili, A. Radford, C. Wijenayake, A. Vishwanath, and V. Sivaraman, "Characterizing and classifying iot traffic in smart

cities and campuses," in *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2017.

[56] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Network traffic classifier with convolutional and recurrent neural networks for internet of things," *IEEE Access*, 2017.

[57] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys (CSUR)*, 2009.

[58] D. Ahmed, "Analyzing the Feasibility and Generalizability of Fingerprinting Internet of Things Devices," https://github.com/dilawer11/iot-device-fingerprinting, 2022.

[59] D. Khan, "Iot device fingerprinting," 2024. [Online]. Available: https://github.com/dilawer11/iot-device-fingerprinting/blob/main/src/iotpackage/ModelTraining.py#L933

[60] E. Crockett, "85 Top IoT Devices," https://www.datamation.com/mobile/85-top-iot-devices/, 2023.

[61] "Consumer iot market trends and key players," https://www.mordorintelligence.com, 2023.

[62] Deloitte, "Connectivity and mobile trends survey 2023: Consumers make their homes smarter," https://www2.deloitte.com/us/en/insights/industry/telecommunications/connectivity-mobile-trends-survey.html#consumers-make-their-homes-smarter, 2023.

[63] 2Smart, "Iot in the consumer electronics market: Trends for 2024," https://2smart.com, 2024.

[64] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services," Tech. Rep., 1998.

[65] M. A. Khan and K. Salah, "Iot security: Review, blockchain solutions, and open challenges," *Future generation computer systems*, 2018.

[66] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling tcp throughput: A simple model and its empirical validation," in *Proceedings of the ACM SIGCOMM'98 conference on Applications, technologies, architectures, and protocols for computer communication*, 1998.

[67] A. Parmisano, S. Garcia, and M. J. Erquiaga, "Stratosphere Laboratory. A labeled dataset with malicious and benign IoT network traffic," https://www.stratosphereips.org/datasets-iot23.

[68] A. Ahmadi, A. Shams, and R. Jalili, "CICFlowMeter," https://github.com/ahlashkari/CICFlowMeter, 2022.

[69] D. Y. Huang, N. Apthorpe, F. Li, G. Acar, and N. Feamster, "Iot inspector: Crowdsourcing labeled network traffic from smart home devices at scale," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2020.

[70] A. Callado, C. Kamienski, G. Szabó, B. P. Gero, J. Kelner, S. Fernandes, and D. Sadok, "A survey on internet traffic identification," *IEEE Communications Surveys & Tutorials*, 2009.

[71] G. Combs, "tshark: Network Protocol Analyzer," https://www.wireshark.org/docs/man-pages/tshark.html, Year Accessed.

[72] C. M. Inacio and B. Trammell, "YAF: Yet another flowmeter," in *24th Large Installation System Administration Conference (LISA 10)*, 2010.

[73] M. Irino, "softflowd: Netflow probe for ipv4/v6, software implementation," 2024. [Online]. Available: https://github.com/irino/softflowd

[74] S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," in *2014 science and information conference*. IEEE, 2014.

[75] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, 2003.

[76] A. C. Müller and S. Guido, *Introduction to Machine Learning with Python: A Guide for Data Scientists*, 2016.

[77] E. Alpaydin, *Introduction to machine learning*, 2020.

[78] J. Davis and M. Goadrich, "The relationship between Precision-Recall and ROC curves," in *Proceedings of the 23rd International Conference on Machine Learning*, 2006.

[79] A. Paleyes, R.-G. Urma, and N. D. Lawrence, "Challenges in deploying machine learning: a survey of case studies," *ACM Computing Surveys*, 2022.

[80] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in Neural Information Processing Systems*, 2017.

[81] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, 2008.

[82] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," *arXiv preprint arXiv:1506.06579*, 2015.

[83] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*. Springer, 2014.

[84] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[85] F. J. Massey Jr, "The kolmogorov-smirnov test for goodness of fit," *Journal of the American Statistical Association*, 1951.

[86] P. Mockapetris, "Domain names: Implementation and specification," 1987. [Online]. Available: https://tools.ietf.org/html/rfc1035

[87] Cloudflare, "Dns load balancing: What is it and how does it work?" 2020. [Online]. Available: https://www.cloudflare.com/learning/dns/dns-load-balancing/

[88] D. L. Mills, "Network time protocol (version 3) specification, implementation and analysis," Request for Comments: 1305, 1992.

[89] A. Analysis, "Subscribers to ad-supported tiers have surpassed 100m in the us," 2024.

[90] European Union, "Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016

on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (general data protection regulation)," 2016.

[91] State of California, "California consumer privacy act of 2018 (ccpa)," 2018.

[92] P. Domingos, "A few useful things to know about machine learning," *Communications of the ACM*, 2012.

APPENDIX A
IOT DEVICES UNDER EXPERIMENTS

Table VIII presents IoT devices the presence of the device in both the UK and USA testbeds. We used these activities to collect traffic in Section III-A.

APPENDIX B
IOT DEVICES ACTIVITIES

Table IX presents sample activities of IoT devices performed in the UK testbed; these activities a sample not limited to. We used these activities to collect active experiments in Section III-A.

APPENDIX C
UNCORRELATED FEATURES CORRELATED MATRIX

Figure 3 presents the uncorrelated features correlated matrix, where we identified the top features used in the reevaluation process to measure practicality improvement in Section VI.

TABLE VIII: **IoT Devices Under Test. The number of devices is summarised as follows: $N_{uk} = 15$, $N_{us} = 75$, $N_{uk} \cap N_{us} = 13$, and $N_{uk} \cup N_{us} = 90$. We use • to indicate the presence of the device in both the UK and USA testbeds.**

Category	Devices
Home Appliances	iRobot Roomba, Belkin WeMo Crockpot, Belkin WeMo Motion Sensor, Eufy Vacuum
Audio/Speakers	Google Home Mini •, Google Home, Amazon Echo Show •, Amazon Dot, Amazon Echo, Apple HomePod, Sonos Speaker
Smart Cameras	Yi Camera, Wansview Camera, Nest Camera, Ring Video Doorbell •, August Doorbell, AVTech Push, Axis Camera, D-Link Camera, Belkin NetCamera HD+, Canary Camera, Logitech Logi Circle, Nest Bell, Nest Cam IQ, Netgear Arlo Camera, Skybell Video Doorbell, Wyze Cam Pan, Wyze Cam
Home Automation	Amazon Smart Plug, Belkin WeMo Link, Belkin WeMo Switch, Chamberlain Garage Opener, Eufy Light, Eufy Plug, Nest Guard, Nest Protect, Piper NV, Rachio 3, Sonos Beam, TPLink Kasa Smart Plug •, TPLink Kasa Smart Bulb •, Wyze Bulb, YeeLight
Hub/Controller	Philips Hub •, Eufy HomeBase, Google OnHub, Google HomeHub, HomeTroller, Insteon Hub Pro, Logitech Harmony Hub, MiCasaVerde VeraLite, Samsung SmartThings Hub, Wink 2 Hub, Withings Home
Smart TV	Amazon Fire TV, Apple TV, LG webOS TV, Roku TV, Samsung SmartTV
Game Consoles	Sony PlayStation 4, Xbox One X
Non-IoT	Bose SoundTouch, Kodi (RPi), Nvidia Shield, iPhone10, iPad Mini (Gen2), Galaxy S10, Openhab (RPi), QNAP TS-120, Securifi Almond, Switch, Synology NAS, Webthing (RPi), Webthing Gateway (RPi), Western Digital EX2 Ultra

TABLE IX: **A sample activities of IoT devices performed during active experiments in the UK testbed.**

Device	Description
Amazon Echo Show	"Alexa, what's the time?" "Alexa, play relaxing music." "Alexa, How is the time?" "Change the volume of the Echo Show." "Alexa, set the alarm for 50 minutes."
Google Home Mini	"Hey Google, what's the time?" "Google, play jazz music." "Hey Google, Do I need to shop today?" "Hey Google, Change the volume of the Echo Show." "Hey Google, turn on the WEMO plug."
Ring Viedo Doorbell	Watch live video feed. Change the view of camera. Alerts of door ringing. communicate with visitors. Recording the video feed.
Wansview Camera	Watch live video feed. Change the view of camera. Motion detection. Recording the video feed.
Belkin WEMO Plug	Turn(on/off) the plug. Control the switch (on/off) using Google.
TPLINK Smart Bulb	Control the light (on/off) using Alexa. Turn (on/off) the bulb via switch. Adjust the Colour Temperature Change the Colour of lighting.
Philips Hue Bridge	Control the Philips Hue lights (on/off). Schedule the Philips Hue lights. Change the Philips Hue lights brightness levels.

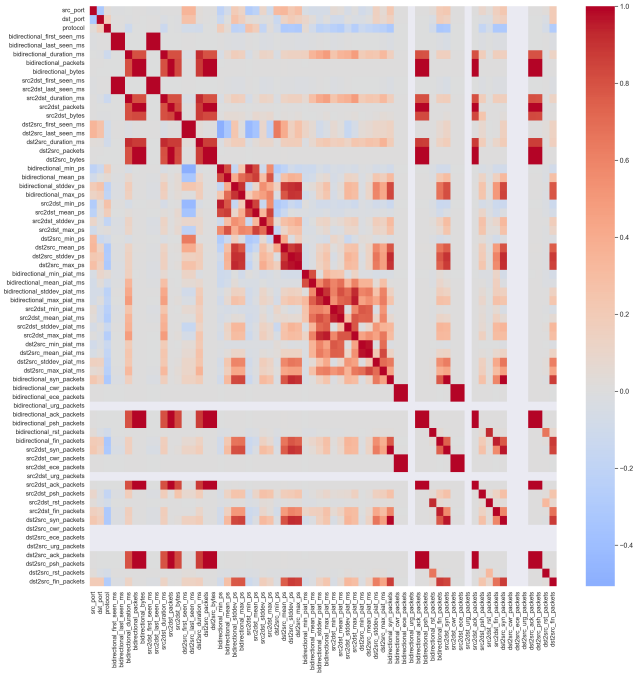


Fig. 3: **Correlation matrix for uncorrelated features used in IoT device identification systems.**