IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

A Survey of Smart Home IoT Device Classification Using Machine Learning-based Traffic Analysis

HOUDA JMILA[1], GREGORY BLANC[1], MUSTAFIZUR R. SHAHID[1], AND MARWAN LAZRAG[1].
[1]SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, Palaiseau, France

Corresponding author: Houda Jmila (e-mail: houda.jmila@telecom-sudparis.eu).

The authors are partially funded under the VARIoT project with TENtec n.28263632, co-financed by the Connecting Europe Facility of the European Union.

ABSTRACT

Smart home IoT devices lack proper security, raising safety and privacy concerns. One-size-fits-all network administration is ineffective because of the diverse QoS requirements of IoT devices. Device classification can improve IoT administration and security. It identifies vulnerable and rogue items and automates network administration by device type or function. Considering this, a promising research topic focusing on Machine Learning (ML)-based traffic analysis has emerged in order to demystify hidden patterns in IoT traffic and enable automatic device classification. This study analyzes these approaches to understand their potential and limitations. It starts by describing a generic workflow for IoT device classification. It then looks at the methods and solutions for each stage of the workflow. This mainly consists of i) an analysis of IoT traffic data acquisition methodologies and scenarios, as well as a classification of public datasets, ii) a literature evaluation of IoT traffic feature extraction, categorizing and comparing popular features, as well as describing open-source feature extraction tools, and iii) a comparison of ML approaches for IoT device classification and how they have been evaluated.

The findings of the analysis are presented in taxonomies with statistics showing literature trends. This study also explores and suggests undiscovered or understudied research directions.

INDEX TERMS Classification, Security, device, fingerprinting, identification, internet of things, machine learning, network traffic, survey.

I. INTRODUCTION

IN the last decade, the Internet of Things (IoT) has spread: according to IoT Analytics [1], the IoT market will rise by 18% to 14.4 billion active connections in 2022. Researchers have suggested several definitions of the IoT, but almost all agree that it is a framework of sensors, industrial machines, video cameras, mobile phones, etc., all of which are collectively referred to as IoT devices and can interact directly with one another or over the internet. IoT is used in smart environments (homes, cities, campuses, etc.) to help users understand and control their environment.

Despite its undeniable advantages, IoT expansion raises security and privacy concerns. Most IoT device manufacturers tend to prioritize the three Ps (prototyping, production, and performance) above security [2]. This results in an ineffective security design for IoT devices. As revealed by Wikileaks [3], poorly secured IoT devices are ideal targets for attackers seeking to obtain unauthorized access and infer sensitive information: e.g., smart TVs were converted into listening devices. Attackers can also use compromised devices to inject malicious data and conduct large-scale attacks against third parties or other devices inside the network [4]. *Automatically classifying devices is the first step toward securing IoT networks.* It enables the detection of vulnerable devices and the enforcement of access control.

The growing diversity and heterogeneity of IoT devices, each with its own QoS requirements (cameras require more bandwidth than smart light bulbs, healthcare device traffic must be prioritized, and so on), makes one-size-fits-all network management ineffective. *IoT device classification enables network management automation*. By setting QoS and network management policies based on the type of device,

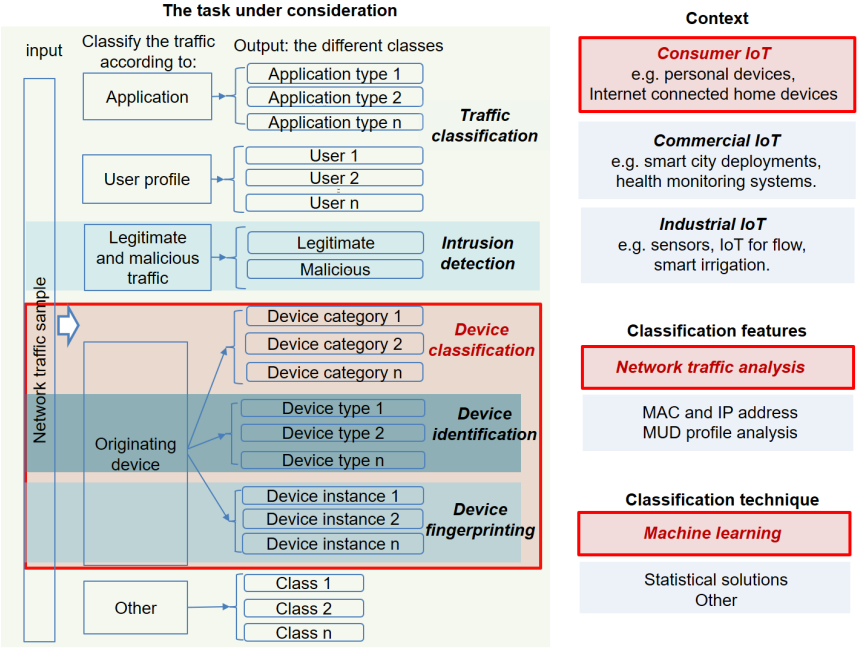

FIGURE 1. The scope of the survey is highlighted in red. We focus on IoT **device classification** in Smart Homes, also called **Consumer IoT** devices. We analyze approaches using **machine learning-based traffic analysis**.

each automatically classified device can be assigned to a class with predetermined policies.

Note that the term **device classification** is often confused with many similar tasks, namely i) traffic classification, ii) intrusion detection, iii) device identification, and iv) device fingerprinting. **Traffic classification** is a broad research field that involves classifying network traffic based on various *parameters* [5] (see Fig. 1). For instance, traffic can be classified as either *legitimate or malicious* based on attack patterns: this is called **intrusion detection**. It can also be classified by the device that generates the traffic (**device classification**). The devices can be categorized into groups of similar devices, such as *devices for energy management* or *devices for health monitoring*, or according to their function, such as *cameras, hubs, home assistants*, etc. **Device identification** classifies devices more finely according to their model or constructor, such as *D-link camera*, *Nest camera, Alexa home assistant, or Google home mini assistant*, etc. **Device fingerprinting** is the finest level of device classification. It gives each device *instance* (e.g., camera A and camera B are two instances of the *Nest Camera*) a distinct fingerprint that is "*impossible to forge and independent of environmental changes and mobility*" [6]. In this study, *we focus on device classification* as a specific case of traffic classification, broader than device identification and device fingerprinting.

A simple way to classify IoT devices is to monitor their MAC addresses and DHCP negotiation [7]. Sivanathan *et al.* [8] outline the shortcomings of this method. First, IP and MAC addresses can be easily spoofed by other devices, making them unreliable identifiers. Furthermore, MAC addresses are not necessarily indicative of device manufacturers, and even if they were, there is no standard for recognizing device brands and types accordingly. To cope with this problem, researchers have examined IoT network traffic and witnessed that IoT devices perform very specific tasks [9]: for example, it is possible to turn on or off a smart bulb or change its brightness and light color, however, a smart bulb can not stream videos or send emails. Therefore, we assume that the IoT network traffic could follow a stable and predictable pattern that may characterize it. Machine learning may reveal hidden network traffic patterns and learn their characteristics, making device classification easier. *This study explores IoT device classification using ML-based network traffic analysis.* To characterize a device, we focus on all the network traffic it creates, which is device-specific and not application-specific because it comprises all the applications (tasks) executed by the device, which can be distinct.

According to [10], IoT devices can be divided into *consumer*, *commercial*, and *industrial* categories. **Consumer IoT devices** include personal devices, such as smartphones, and internet-connected home devices like cameras, home assistants, and smart lamps. Larger organizations employ **commercial IoT devices** for smart city deployments, transportation and electric car monitoring, health monitoring systems, etc. **Industrial IoT devices** improve process control and productivity, such as sensors, robots, and power plant controllers. Some devices, like cameras and sensors, can belong to multiple categories. *This survey focuses on consumer IoT devices, commonly called* ***smart home devices***. This choice is motivated by the rich and abundant literature on smart home devices due to i) the availability of data, compared to its confidentiality in the industrial world, and ii) the large number of

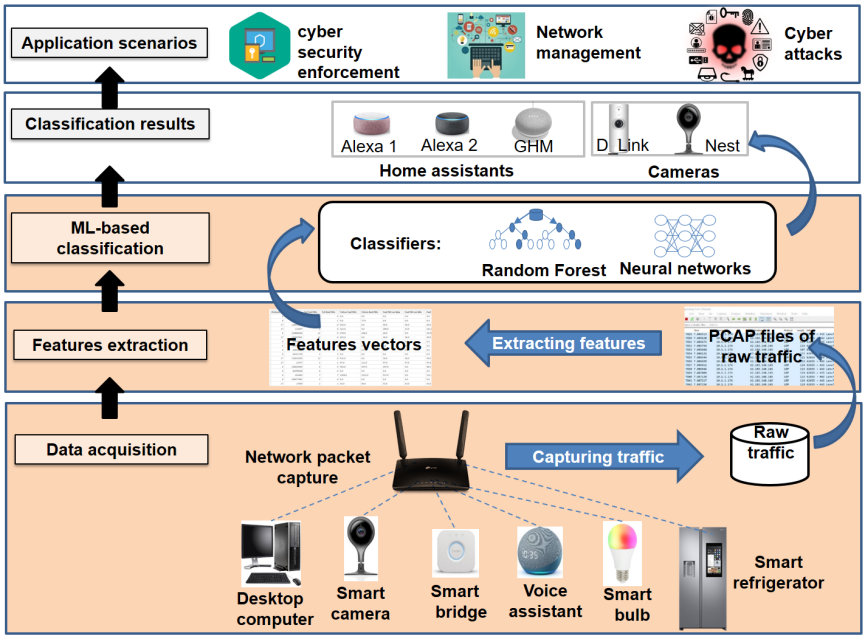

FIGURE 2. Workflow of IoT device classification using ML-based traffic analysis: input includes the devices to be classified. First, raw traffic data is collected as `pcap` files and supplied to the feature extraction procedure, which creates feature vectors (in text-based format) representing the raw traffic. ML algorithms use these files to classify the originating device of each sample. Classification results can be used in various contexts, including cyber security enforcement, network management, and malicious usage.

smart home devices, which represent the largest share of the IoT market (63% according to Gartner [11]). Furthermore, many people, including those unaware of security, use smart home IoT devices, making their protection crucial.

Other surveys have examined IoT device classification-related tasks, but none have focused on the topic of this study. Tahaei *et al.* [7] discussed IoT traffic classification, while [12] examined ML-based internet traffic classification, although neither focused on device classification. Sanchez *et al.* [13] discussed device behavioral fingerprinting but not IoT devices specifically. Yadav *et al.* [6] provided a taxonomy for IoT device identification approaches. However, they did not focus on ML methods and what data collection, feature extraction, and model learning require. To the best of our knowledge, no recent work studies exhaustive IoT classification datasets, no current work explores feature extraction methodologies and compares the most useful and interesting features for IoT device classification, and no previous work examined each step of the IoT classification process as we do.

For a comprehensive literature review, we analyzed papers from different digital libraries like IEEE Xplore, ResearchGate, Google Scholar, etc. First, we performed a keyword search using terms related to i) *IoT devices*, like "IoT devices," "wearable devices," and "IoT gadgets," ii) *classification*, like "classification," "clustering," "identification," and "fingerprinting," iii) *traffic analysis*, like "traffic analysis," "traffic classification," "communication analysis," "network characteristics," "network packets," and "network flows," and iv) machine learning as "machine learning," "deep learning," "artificial intelligence," "supervised learning," "unsupervised clustering," "automated," and "intelligent." Our search was limited to 2018-2022 articles to capture recent advancements. Second, we examined the reference lists and citations of the selected articles to find more papers. Third, we scanned titles and abstracts to reject items that did not fit the scope (task: classification, context: smart home, and classification approach: ML-based traffic analysis). Finally, a deep evaluation of the publications was conducted, and articles with insufficient information on all stages of the classification procedure were removed. At the end of this process, **58 papers** were deemed pertinent to our investigation.

II. ANALYSIS STEPS AND CONTRIBUTIONS

Fig. 2 shows a general flowchart summarizing the multiple steps and actors that can be involved in *IoT device classification using ML-based traffic analysis.* The initial step is data acquisition, which consists of collecting raw traffic from devices in `pcap` files (the `pcap` file format is the de facto standard for packet captures). The second phase is feature extraction, which aims at representing raw traffic with numerical or categorical information in a text-based format (e.g. `csv` (Comma-Separated Values) or `text`) files that ML algorithms can use. The final stage is classification using machine learning algorithms. The classification result can be used for cyber security enforcement, network management, as well as malicious activities like cyber attacks.

To help develop more effective solutions for IoT device classification, this study investigates the literature regarding each stage of the process and attempts to provide answers to

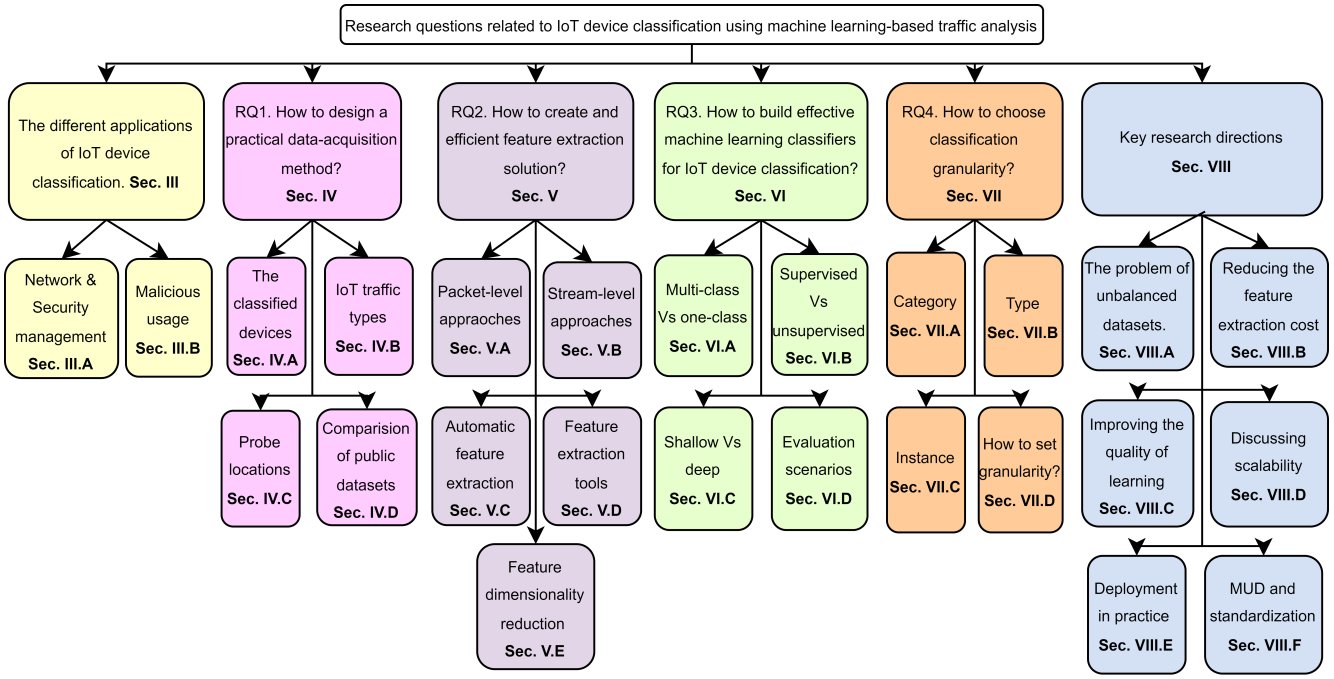

FIGURE 3. Table of content and discussed questions.

the following research topics.

- ***RQ1.*** *How to design a practical data-acquisition method for IoT device classification?* Data acquisition is a crucial step that should enable the practical and realistic capture of the most relevant information about the environment. To design an effective and practical solution for the IoT device classification problem, it is essential to know: i) which devices should be used for data-acquisition to represent a realistic smart home environment, ii) when to collect the traffic to capture the diversity of the devices' operational modes, and iii) where to place the collection probe so as to capture traffic in an effective yet privacy-preserving manner.
- ***RQ2.*** *How to create an efficient feature extraction solution?* Feature extraction is a critical step that must describe the collected traffic as accurately as possible to reflect its patterns. To develop an appropriate feature extraction technique for IoT device classification, it is necessary to know: i) how to represent a single data sample, as a packet or as a flow of packets, in other terms, at what level to extract features (packet-level or flow-level), ii) in the latter scenario, how to define a packet flow (by time interval, number of packets, or connection), and iii) which are the most informative and discriminating features, and how to calculate them.
- ***RQ3.*** *How to build effective machine learning classifiers for IoT device classification?* Classification using machine learning algorithms is the last, but not the least important step. To answer this research question, it is essential to decide: 1) the scope of a classifier (one classifier per device type or one multi-class classifier), 2) the learning strategy (supervised, un-supervised, semi-supervised), and 3) the machine learning techniques to use (deep or shallow algorithms).
- ***Q4.*** *How to choose the classification granularity?* Device classification can be performed at different levels of granularity. It's crucial to understand the pros and cons of each classification level in order to choose the optimal granularity for each context and avoid extra classification costs.

To the best of our knowledge, this is the first paper that covers all of the above mentioned challenges and explores their impact on IoT device classification. As an attempt to address the above-mentioned research questions, this survey also produces the following contributions (Fig. 3, which provides a table of contents, depicts where and how the above questions are handled in this study.):

- An analysis of the various applications for the classification of smart home IoT devices.
- An in-depth examination of IoT traffic data collection strategies. This includes: i) a review of the devices used to represent a smart home setting, ii) a study of IoT traffic types (depending on device operation mode) and their utility for classifying devices, iii) a description of the architecture and different traffic collection points (depending on the traffic probe location) and a debate on how realistic they are, and iv) an evaluation of public datasets for IoT device classification.
- A thorough review of feature extraction approaches. This includes: i) exploring different feature types and

comparing their significance and computation methodologies, ii) exploring deep learning-based automatic feature extraction, iii) describing open-source feature extraction tools, and iv) investigating feature dimensionality reduction for better IoT device categorization.
- A comparison of machine learning approaches for IoT device classification and how they were assessed in the literature.
- An examination and assessment of the various classification granularity levels.
- A summary of contributions in the form of taxonomies and statistics to highlight trends. The statistics were calculated based on a thorough review of each research article with respect to taxonomies (See Tables 2 to 4).

This document follows the classification process from bottom to top, except for the applications of IoT device classification, which will be shown first for sake of clarity.

III. THE DIFFERENT APPLICATIONS OF IOT DEVICE CLASSIFICATION

A. NETWORK AND SECURITY MANAGEMENT

Due to the variety of IoT devices, it is difficult to control them with a single policy. One solution is to describe network and security management rules by device class and assign each device to a class with automated policies. Miettinen *et al.* [14] describe an interesting use-case where newly introduced devices are categorized and the classification result is used to determine whether the device is vulnerable. The decision is based on a vulnerability assessment of the device type carried out by consulting a vulnerability dataset. Consequently, the device is assigned one of the following isolation levels: i) *strict*, where the device can only interact with untrusted devices, ii) *restricted*, where it can communicate with untrusted devices but has limited internet access, and iii) *trusted*, where the device is allowed to communicate with other trusted devices and has unrestricted internet access. This mitigation approach allows vulnerable devices to cohabit with other devices without compromising their security.

Note that detecting vulnerable devices in a smart home is crucial since most IoT devices suffer from poor security design and can be easily compromised by an attacker to gain unauthorized network access or launch massive attacks. For instance, in 2016, the Mirai malware infected millions of IoT devices to launch DDoS (distributed denial-of-service) attacks [4]. The BYOD (Bring Your Own Device) trend, which allows employees to bring their own personal IoT devices at work and connect them to the corporate network, extends the attack surface of companies as compromised personal devices may inject malware into the corporate network and cross-contaminate other devices. Similarly, remote working has exposed professional devices to a less trustworthy environment where they cohabit with possibly more vulnerable smart home devices.

As described above [14], *black listing* approaches detect vulnerable devices that should be disconnected from the network (blocked). IoT device classification can also be used to establish an automatic whitelisting system to ensure only authorized IoT devices can connect to the network, as proposed by Meidan *et al.* [15]. If the determined IoT device type is not in the white list, the organization's SIEM system is alerted to take appropriate action (e.g., disconnect the device from the network).

Discussion: White Vs black listing approaches. White listing is more scalable than blacklisting, which grows with untrusted devices. Moreover, data from authorized (whitelisted) devices is easier to obtain. Nevertheless, using a whitelist would be less robust against adversary attacks, as an attacker may simulate authorized device behavior to avoid the intrusion detection system.

B. MALICIOUS USAGE

IoT device classification can also be exploited by attackers to leak sensitive information about the IoT device and its users.

For instance, Hafeez *et al.* [16] demonstrate that an adversary, with access to upstream traffic from a smart home network, can identify the device types and user interactions with IoT devices, with significant confidence. Dong *et al.* [17] study the case where an adversary attempts to infer the type of IoT devices behind a smart home network even when the traffic of all devices is merged behind the gateway using VPN (Virtual Private Network) and NAT (Network Address Translator) techniques.

Sensitive information revealing device types and user interactions, can be used to infer user activities or home presence [16]: e.g. if the smart lights are in the off state for a long period of time, it means that there is no one at home, opening an opportunity for a break-in. Such passive attacks are hard to identify and mitigate. In this context, Hafeez *et al.* [16] propose a traffic morphing technique helping to hide the traffic of IoT devices, lowering the occurrence of attacks.

IV. APPROACHES TO DATA ACQUISITION

This section describes the data acquisition methodologies found in the literature. In order to organize the findings, we present them along four axes: first, we examine the devices considered for data collection, second we analyze the IoT traffic types that can be captured, third, we discuss data collection scenarios, and finally, we provide a comparative study of public datasets. A taxonomy in Fig. 4 illustrates the main outcomes of this section.

A. THE CLASSIFIED DEVICES

The input to the IoT device classification process is a list of devices to be classified. They can be both IoT and non-IoT devices, also referred to as *single-purpose* and *multi-purpose* devices, since IoT devices are typically intended for a single specific task. An up-to-date list of the most common smart home IoT devices can be found on the website [18]. Examples of non-IoT devices include laptops, cell phones, and Android tablets.

In the literature, some approaches classify *only IoT devices*, and others classify *both IoT and non-IoT devices*.

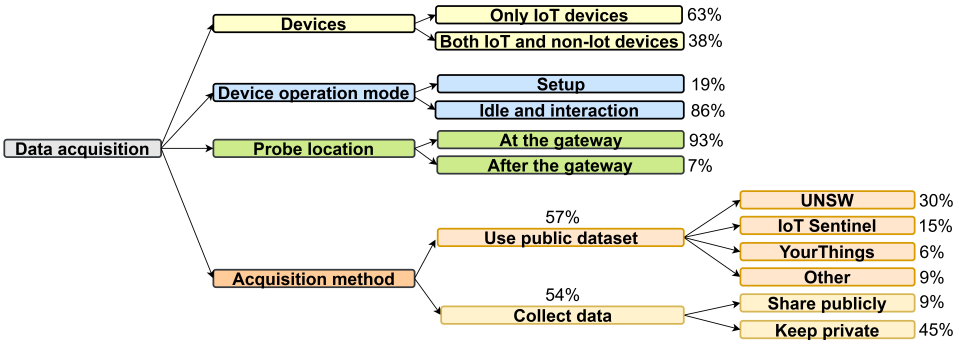

FIGURE 4. Taxonomy of data-acquisition approaches: the approaches are classified according to i) the devices under consideration: only IoT, or both IoT and non-IoT devices, ii) the operation mode of the devices, iii) the probe location, and iv) whether a public dataset is utilized or the traffic is collected by the authors. Percentages show how often each approach is used in the reviewed papers. This highlights the trends discussed in Sec. IV.

Discussion: Fig. 4 shows that the majority of reviewed papers (63%) consider the classification of only IoT devices. However, we think that this is not the most realistic scenario since the traffic must be collected from all the devices connected to the smart home network to ensure its security and automatic management. Since IoT and non-IoT devices cohabit in smart homes, they must be considered during the traffic collection process. However, note that classifying both IoT and non-IoT devices is more challenging since IoT traffic is small and sparse compared to non-IoT data. As shown by Dong *et al.* [17], some IoT devices might be easily confused with non-IoT devices. For example, home assistants have diverse and varied functions (compared to simple single-use devices like light bulbs), making their behavior very similar to non-IoT devices. To address this challenge, we suggest training ML algorithms with mixed (IoT and non-IoT) traffic to boost their generalization capabilities.

B. THE DIFFERENT TYPES OF IOT TRAFFIC

IoT devices generate three types of traffic based on their operation mode, namely: i) *setup traffic* (also called initial traffic) is generated by an IoT device during installation, also called registration or enrollment, ii) *interaction traffic* (also called active traffic) is generated when a device interacts with the user or environment (e.g., a home assistant responding to a voice request from the user), and finally, iii) *idle traffic* represents device activity in the absence of external stimulation. It includes routine communications between the device and the back-end server, as well as keep-alive or heartbeat signals.

1) The setup traffic

When a new device with a new MAC address connects to the network, it follows a device/provider-specific procedure to connect [14]. In most situations, this operation is assisted by a smartphone, laptop, or PC application. The installation procedure typically involves: i) activating the device, ii) connecting with the provider's app, iii) transmitting WiFi credentials, and iv) resetting and connecting to the user's network using the credentials provided.

To collect the installation traffic, existing approaches record the first packets $\{p_1, p_2, p_3, \ldots, p_n\}$ exchanged between the device and the gateway. The decrease in packets exchanged marks the end of the installation phase. To generate enough data, the installation process should be performed multiple times for each device, with a hard reset between each save [14].

2) The interaction and idle traffic

IoT devices generate mostly interaction and idle traffic. Interaction traffic can be triggered either i) by a direct user request, like adjusting light bulb color and intensity, or ii) by a change in the environment observed by the IoT device, such as a sensor that detects motion or a light bulb that detects an inhabitant [19]. Idle traffic mainly includes device-Cloud service exchanges during standby, such as heartbeat messages, regular status updates or notifications [16]. IoT devices generate more traffic when active compared to background mode [20]. This is reasonable since user and environmental interaction stimulates diverse reactions [20].

Discussion: Which traffic type is most suited for IoT device classification? Statistics detailed in Fig. 4 show that 86% of reviewed papers use idle and (or) interaction traffic. Only 19% of reviewed papers rely on setup traffic for device classification. The advantage of setup traffic over idle and interaction traffic is its stability, as the IoT device's behavior during configuration is the same regardless of the environment. Moreover, relying on setup traffic allows for rapid recognition once the device is connected to the network. However, as the initialization state may not appear several times during the IoT device life cycle, setup traffic is scarce, sparse, and difficult to collect in real-world network monitoring. On the other hand, idle and interaction traffic is more abundant and easier to collect, making it better suited for machine learning algorithms, especially deep learning.

C. DIFFERENT LOCATIONS FOR TRAFFIC PROBE

1) A typical network setup for capturing IoT traffic

Fig. 5 shows a typical smart home network architecture. It includes IoT and non-IoT devices connected to an internet

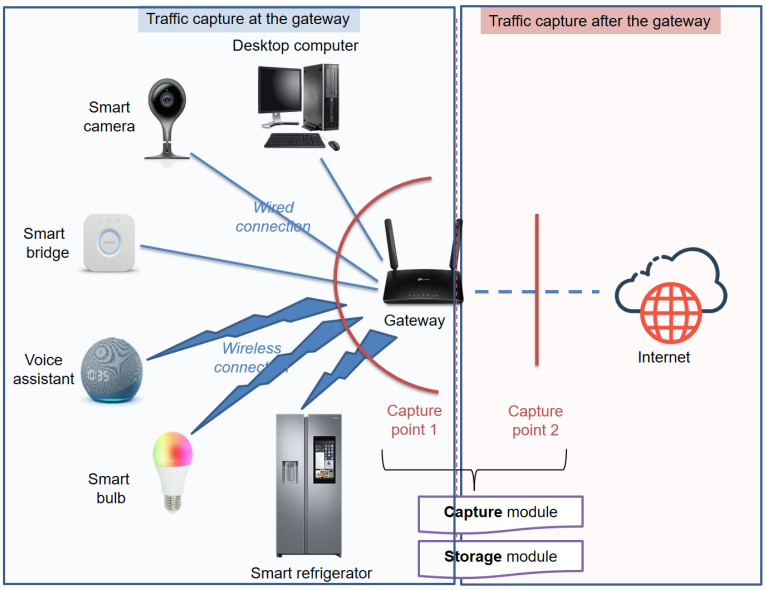

FIGURE 5. A typical network configuration for capturing IoT traffic includes i) IoT and non-IoT devices connected to the gateway via wireless or wired connections and ii) packet capture and storage modules for collecting traffic. There are two capture points discussed in the literature: i) at the gateway and ii) after the gateway.

gateway using wireless or wired connections. At least two tools are required for traffic collection:

- *a Packet capture module* to capture the traffic as `pcap` records comprising entire packets from MAC layer to application layer. Examples include tcpdump [21] or Wireshark [22], and
- *a storage module* to store the traffic data on a distant server, or within the network.

To label the ground truth, the MAC address in the packet header is used to reveal the identity of the device and label the data accordingly.

2) Traffic capture scenarios

The literature considers two scenarios for collecting IoT traffic depending on the location of the probe (capture point): i) at the gateway, i.e. from inside the home device, or ii) after the gateway, i.e. from outside the smart home.

At the gateway, the captured traffic is the one flowing between the devices connected to the home network and the gateway and can be separated by IP or MAC address. Whereas the traffic captured after the gateway contains traffic from all connected devices aggregated using a single public IP address due to the frequent use of NAT at gateways.

Discussion: Which probe location is more practical? Approaches that gather traffic at the gateway assume the ability to intercept and sniff the traffic flowing inside the smart home. However, this clean and controlled experimental setup does not reflect most real-world use cases where traffic is only seen from the outside. A typical application is when Internet Service Providers (ISPs) classify IoT traffic to identify devices inside a smart home and then allocate resources and configure appropriate security rules according to their population and vulnerabilities. But ISPs can not intercept traffic inside the home network. It is then more realistic to collect traffic from outside the smart home after the gateway. However, classifying devices based on such traffic is more challenging because the original packet headers, such as source IP and port, are hidden. Moreover, the widely used VPN-enabled gateways encapsulate the original packets in an encrypted tunnel, hiding the traffic characteristics. This makes device classification even more challenging, and new solutions should be investigated.

Although realistic, this scenario is understudied. This scenario is used in only four papers: [17], [23]–[25]. It is worth noting that Meidan *et al.* [25] and Dong *et al.* [17] made their datasets public so that more research could be done on this topic.

D. PUBLIC DATASETS COMPARISION

57% of reviewed publications use public datasets, either completely or to complement or enrich their data. Most of the datasets we mention in this survey were created for IoT device classification. However, we include other datasets developed for other topics that contain IoT traffic and can be used for IoT device classification.

Table 1 summarizes the datasets listed below. To compare them, we specify for each: i) the devices used to generate the traffic (IoT only, or both IoT and non-IoT), ii) the operation mode of the devices (i.e. setup, interaction, idle), iii) the probe location (i.e., at or after the gateway), iv) the duration of the collection, v) the amount of traffic collected, and we provide vi) a direct access link to the dataset.

1) IoTSentinel dataset [14]

This dataset was collected to identify IoT devices based on their setup traffic. To generate enough traffic, the typical

TABLE 1. Publicly available datasets for IoT device classification

datasets	Devices		Device operation mode			Traffic probe location		Duration & Period	Amount	Access link
	IoT	Non-IoT	Setup	Idle	Interaction	at the gw	after the gw			
IoT Sentinel [14]	31	0	✓	-	-	✓	-	Setup process repeated 20 times per device (Jan. to Apr. 2016)	540 samples	[26]
UNSW [19], [27]	28	3	-	✓	✓	✓	-	2 weeks (Oct. 2016 to Apr. 2017)	≈ 365MB per day	[28]
YourThings [29]	45	0	-	✓	✓	✓	-	10 days (Mar. to Apr. 2018)	≈ 10 GB to 13 GB per day	[30]
IoTFinder [31]	53	0	-	✓	✓	✓	-	1.5 months (Aug. to Sep. 2019)	366 MB	[30]
SHIoT [32]	36	0	-	✓	✓	✓	-	3.458.47 hours (2019)	382.75 GB	[33]
DADABox [34]	41	0	-	✓	✓	✓	-	27 weeks (2020 to 2021)	130.460 records	[35]
HomeMole [17]	10	4	-	✓	✓	✓	✓	49.4 hours (2019)	7.223.282 packets	[36]
IoT-deNAT [25]	8	5	-	✓	-	✓	✓	37 days (2020)	-	[37]
MON(IOT)R [38]	81	0	-	✓	✓	✓	✓	1 month (Sep. 2018 to Feb. 2019)	34.586 samples	[39]
IoT-23 [40]	3	0	-	✓	✓	✓	-	Variable duration per IoT device (2018 to 2019)	387.688 MB	[41]

device configuration process was repeated 20 times for each device. During the setup process, all network traffic between IoT devices and the gateway was recorded. A representative set of 31 IoT smart home devices available on the European market in the first quarter of 2016 was used. There are 27 different device types (4 types are represented by 2 devices each). Most of the devices were connected via WiFi or Ethernet. Some of them utilised ZigBee or Z-Wave.

2) UNSW dataset [19], [27]

This dataset was published by UNSW researchers and covers various IoT research areas. In addition to traffic for IoT device classification, the dataset includes IoT attack traces, IoT MUD profiles, and IoT IPFIX records that can be useful for other IoT-related research topics (the relevance of MUD profiles to device classification is discussed in Sec. VIII). In this paper, we focus on the traffic for IoT device classification. It was first published in [19] and has since evolved. The first version has been extensively used in the literature. The same authors published an updated and more elaborate version in [27]. Recent articles now use the modified version.

This study focuses on the IoT traffic traces reported in [27]. They were collected over 26 weeks, from October 1^{st}, 2016 to April 13^{th}, 2017, but only two weeks' worth of data is available for download.

3) IoTFinder [31] and YourThings datasets [29]

The IoTFinder dataset was created to explore IoT device identification using DNS fingerprints. Thus, the dataset contains `pcap` files of DNS responses for 53 IoT devices from different vendors. The data was collected from August 1^{st}, 2019 to September 30^{th}, 2019.

YourThings dataset was created by the same authors to analyze security properties for home-based IoT devices.

4) SHIoT dataset [32]

This dataset was created for behavior-based IoT device classification. The test bed was implemented at the Faculty of Transport and Traffic Sciences in Zagreb. The dataset contains 144 `pcap` files with 24-hour traffic each.

5) DADABox dataset [34]

This dataset was created to compare some approaches to classifying IoT devices. The testbed was developed at the University of Cambridge, where researchers sporadically interact with IoT devices. The dataset contains 41 different IoT devices, and the data was collected over a period of 27 weeks.

6) HomeMole dataset [17]

This dataset was created to identify IoT devices behind VPN and NAT-enabled gateways in smart homes. Three collection scenarios were developed: i) *a single device environment* in which only one device is considered, ii) *a noisy environment* in which various IoT and non-IoT devices are investigated. Multiple devices may be operating simultaneously at any given time, resulting in traffic aggregation, and iii) *a VPN environment* where VPN is enabled. In this case, traffic is collected before and after the VPN.

7) IoT-deNAT [25]

The dataset was collected to detect vulnerable IoT devices behind a home NAT. The traffic is captured considering only NetFlow's [42] statistical aggregations (i.e., Netflow is a flow-level aggregation of information, usually a 5-tuple header and some counters) instead of the raw data to reduce processing and storage.

8) The MON(IOT)R dataset [38]

This data set examines IoT device information exposure. It contains data from 81 IoT devices deployed in two labs (one at Northeastern University in the United States and the second at Imperial College London in the United Kingdom)

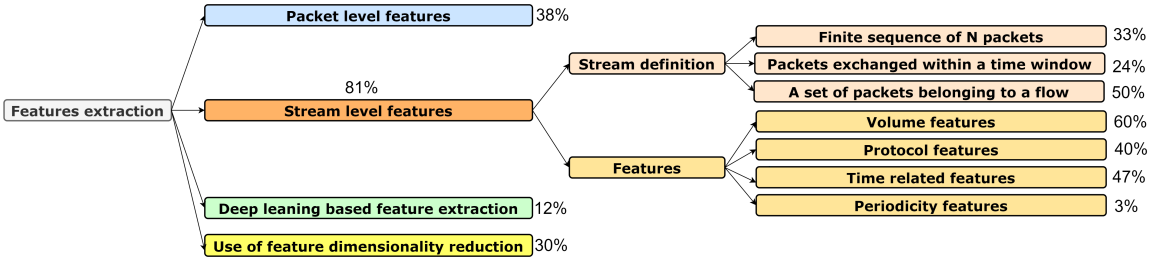

FIGURE 6. Taxonomy of feature extraction approaches. The approaches are classified according to i) the use of header or payload packet level features, ii) the stream definition, iii) the type of used stream level features (volume, protocol, time, or periodicity), iv) the use of automatic feature extraction (DL based), and v) the use of dimensionality reduction. Percentages show how often each approach is used in the reviewed papers. This highlights the trends discussed in Sec. V.

over 30 days between September 2018 and February 2019. Different types of traffic are provided: i) power traffic (487 samples), which is traffic generated by IoT devices when they are turned on, ii) interaction traffic (32,030 samples), iii) idle traffic covering an average of 8 hours per night for one week for each lab, and iv) unlabeled traffic, which is generated when 36 participants use the IoT devices in a studio at their leisure during the data collection period. Data labeling includes the name of the device, where it was used (the US or the UK), when and for how long it was used, and whether or not a VPN was used.

9) IoT-23 dataset [40]

IoT-23 is a dataset containing benign and malicious IoT network activity. The traffic was captured at the Czech Technical University. The dataset contains 20 `pcap` files from infected IoT devices, labeled by the malware that infected them, and 3 `pcap` files containing benign network traffic generated by 3 IoT devices: a smart lamp, a voice assistant, and a smart door lock. The packet captures are labeled with the device that generated the traffic. As done in [43], legitimate traffic can be used for IoT device classification, .

Discussion: How valuable are public datasets? Public datasets enable comparing different solutions. Unfortunately, the available public datasets for IoT device classification are scarce (only 5 of the surveyed papers shared their datasets publicly) and not diversified: most provide idle and interaction traffic, and capture at the gateway, when this is not the most realistic scenario. Since public datasets are not diverse, researchers must collect their own data when examining new scenarios. For instance, Yu *et al.* [44] identify IoT devices based on passively receiving broadcast and multicast packets, and had to collect their own data from different WiFi networks. In conclusion, additional datasets exploring new classification scenarios should be released, and more diversified IoT traffic needs to be collected, in order to boost research on IoT device classification. As shown in Fig. 4, the most used datasets are UNSW (30%), IoTSentinel (15%), and YourThings (6%).

V. FEATURE EXTRACTION METHODOLOGIES

This section describes feature extraction methodologies. First, we discuss packet-level feature extraction: we examine the most commonly used header and payload features and compare them. Second, we analyze stream-level feature extraction. Third, we explore deep learning based automatic feature extraction. Fourth, we provide a list of open-source feature extraction tools, and finally, we highlight the feature dimensionality reduction approaches. Fig 6, gives a taxonomy summarizing the approaches and trends.

Feature extraction is defined in [45] as "the process of defining a set of features (...) which will most efficiently or meaningfully represent the information that is important for analysis and classification." In our case, the feature extraction step consists of describing the network traffic in the most appropriate way to retrieve the maximum amount of information about the device.

In the majority of examined articles, significant work has been dedicated to the extraction of features. Existing approaches are diverse and heterogeneous. The objective of this section is to summarize them in a logical and consistent manner.

Network traffic is the volume of data flowing over a network. It is divided into packets of data and delivered over a network before being reassembled by the receiving computer or device. Packets can be used to describe the network either individually or as a *stream* of packets, also called a *flow* (see Fig. 7).

These two approaches are known as *packet-level* and *flow-level* feature extraction methods, respectively. The following sections present approaches in each category.

A. APPROACHES TO PACKET-LEVEL FEATURE EXTRACTION

These approaches describe each packet individually. A packet consists of a *header* and a *payload*. The header contains protocol information for a given layer, whereas the payload contains the data.

1) The most important packet header features

Extracting features from a packet header is straightforward and has no overhead. One just needs to parse the packet's header fields.

Depending on the layer and protocol, several fields can be present in the packet header. For example, the IPv4 header contains essential routing and delivery information

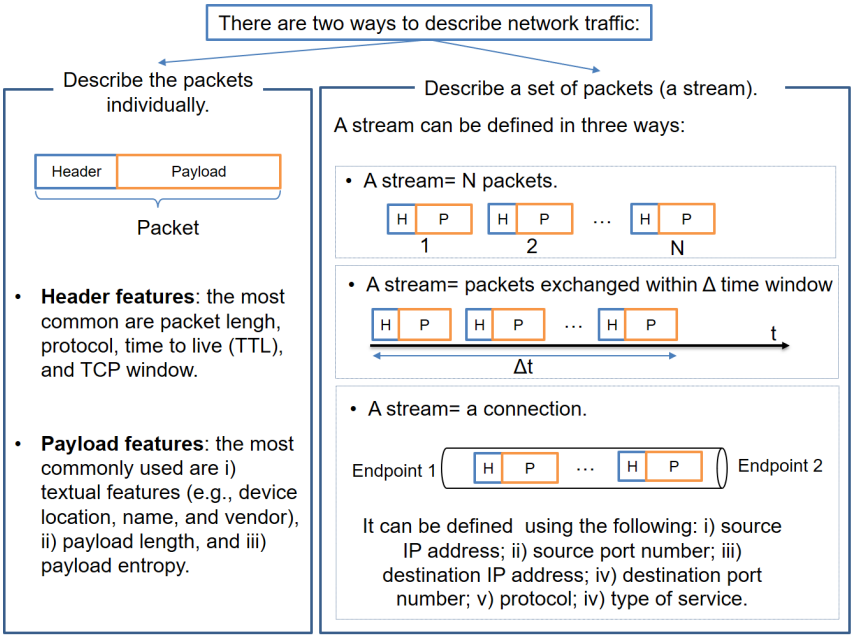

FIGURE 7. The main methods for feature extraction: packet-level and stream-level. For stream-level approaches, three definitions are proposed for the stream.

and consists of 13 fields, including version, header length, service type, total length, time to live, and protocol, etc. Relying on source and destination IP addresses and ports for classification is not recommended due to potential spoofing issues, as mentioned in Sec. I.

The most important header features include i) the **packet length**, which is widely used for IoT device classification [46], and ii) the **TCP window size**, which is very useful for distinguishing between IoT and non-IoT devices as it depends on the memory and processing speed of the device [47]. Small constrained devices, like sensors, have small window sizes, while more powerful devices like video cameras and home assistants have variable and larger window sizes [47].

2) The most important payload features

Typically, payloads consist of the header and payload of the upper layer, which in our case indicates the application payload. It may consist of textual features indicating the device's name, location, manufacturer, type, operating system, services, etc.

The **length of the payload** transported inside a TCP message can indicate the length of the message sent by a given device, and this is device specific [47]. The **entropy of the payload** has been used as a discriminative feature [47], [48]. In [49], the distribution of payload bytes per flow is used for IoT device classification. Encrypted packets may make feature extraction from the payload impossible.

Discussion: Packet-level Vs stream-level. Processing each packet separately for feature extraction is time-consuming and computationally exhausting, requiring large storage and processing resources. The Google Chromecast generates 2,459,538 packets per day, compared to 11,877 traffic flows [32]. Thus, extracting features from packets is more expensive than from flows. Unsurprisingly, most research concentrates on flow-level features (81% of reviewed papers).

B. STREAM-LEVEL FEATURES EXTRACTION METHODS

In this section, we discuss the different stream definitions, we investigate and categorize the most important features, and we examine the approaches to calculating them.

1) Stream definition

Features can be extracted from a set of packets known as a "stream." We have identified three main approaches to defining a stream: i) a stream is a set of N consecutive packets, ordered by arrival time, ii) a stream is a set of packets exchanged within a time window Δ, iii) a stream is a connection between a source and a destination where packets are sent in both directions in a certain order. More information on the approaches using each definition is presented below.

a: A stream as a finite sequence of N packets

In this category, a fixed number N of consecutive packets generated and received from a single IoT device is used to construct a "signature," also called a "fingerprint" of the IoT device. 33% of surveyed papers use this definition, in particular approaches leveraging *setup traffic* (cf. Sec. IV-B1) for device classification, because they use the first packets sent by the devices when connecting to the network. For example, in [14], [50], the authors use the first 12 packets to identify an IoT device, and in [51], 30 packets are used. The authors of [52] extract features from a sequence of 20-21 packets. Shahid *et al.* [53] consider N consecutive packets

where N varies between 2 and 10. 24% of surveyed papers use this definition.

Discussion: Note that determining the optimal value of the flow size, N, is challenging. *Small flows* allow for quick classification but may not be enough to characterize the device, whereas *large flows* can be time and memory-consuming to analyze. Moreover, the appropriate value of N may vary from device to device since IoT devices generate different quantities of data. A small number of packets may be enough to identify certain device types, while a greater number may be required for others. This is problematic because machine learning algorithms require a fixed size for the input. The authors of [14] added padding for devices that emit fewer packets than the required size. Furthermore, capturing the same number of packets for all devices may take a variable amount of time as IoT objects do not generate traffic *at the same rate*. For example, it is possible to capture packets generated by a camera in seconds. However, it takes longer to capture the same number of packets generated by a motion sensor [46]. This makes the data collection process complicated and time-consuming.

b: A stream as a set of packets exchanged within a time window Δ

This consists of subdividing the captured traffic into distinct time-windows of an appropriate duration Δ. For example, Fan *et al.* [54] extract the features every 30 minutes. Pinheiro *et al.* [46] use a window of one second to enable real-time device classification. Hafeez *et al.* [16] use a 10-second time window. Le *et al.* [55] retrieve DNS names requested by a device over a time period ranging from 10 minutes to 24 hours, and found that performance decreases with a decreasing Δ.

Discussion: Note that as for the previous category, the choice of the time window size is important and challenging. *Long time-windows* give richer information about the device but risk increasing classification delay and consuming more memory to store traffic attributes [46]. Moreover, it may result in very similar samples with little feature variation. This could also lead to fewer data samples for learning and testing, and thus be unsuitable for deep learning-based classification approaches. Few and redundant samples may also introduce a bias and overfitting. On the other hand, a *small time-windows* may allow real-time classification but may not contain enough information to reflect the characteristics of the device's behavior. Bai *et al.* [56] showed that a small segmentation window interval degrades the classification results compared to a larger segmentation. In addition, setting the same interval time for all devices can be inappropriate as the devices generate different quantities of traffic. For example, a motion sensor generates close to 140 packets per minute at most, and a camera generates up to 1900 packets per minute on average [56].

c: A stream as a set of packets belonging to a connection

Due to the abovementioned issues, the majority of reviewed papers (50%, see Fig. 6) use this definition of stream. This is based on the RFC 2722 [57] traffic flow definition, stating that a flow is "an artificial logical equivalent to a call or connection." Thus, the flow is the ordered sequence of all packets sent and/or received from a particular source to a particular unicast, anycast, or multicast destination using specific ports and transport protocols.

More concretely, a flow can be defined as a set of packets having in common at least two of the following attributes: i) source IP address, ii) source port number, iii) destination IP address, iv) destination port number, v) protocol, and vi) service type.

Depending on the criteria utilized to define the flow, there are several definition variants. For Marchal *et al.* [20], the flow is a sequence of network packets sent by a given IoT device using a specified communication protocol. A flow is described by Sun *et al.* [49] as a 5-tuple of source and destination IP addresses, source and destination port numbers, and protocol. For Meidan *et al.* [25], the service type is also specified (6-tuplet).

Note that *a collection of flows* can also be used to describe the traffic. The authors of [49] combine features from several flows to provide a high-level characterization of device activities. Meidan *et al.* [58] demonstrated that using a set of consecutive flows gives better classification results since it contains more information about the traffic. The different stream definitions are illustrated in the left part of Fig. 7.

2) Important stream-level features

In this section, we review the various stream-level features that are widely used for IoT device classification. To organize them, we divide them into four categories: i) *volume features* measure the volumetric properties of the stream, ii) *protocol characteristics* describe the protocols on the stream, iii) *temporal characteristics* measure the temporal aspects of the stream, and iv) *periodicity features* reflect the stream's periodicity.

a: Volume features

Examples include packet length statistics, the number of packets or bytes in the entire flow or in a specific direction (incoming or outgoing traffic), the flow rate, etc. For instance, Pinheiro *et al.* [46] identify devices based on statistics of the packet length and number of bytes generated by each device. Sivanathan *et al.* [59] use average packet size and average rate per flow as two principal attributes. Volume features are very important and widely used (in 60% of reviewed papers).

b: Protocol features

Traffic including all protocols and layers, or selected protocols, can be used to extract features. In addition to the widely studied layer-2 to layer-4 protocols, the following application layer protocols have been examined:

- The **Domain Name System (DNS)** is an essential Internet service, and is therefore important to IoT devices communicating with remote Cloud services. The DNS features differentiate IoT from non-IoT devices. IoT devices connect to limited endpoints, mainly their provider servers. This behavior can be captured by the *number of DNS unique queries*, as IoT devices have fewer unique DNS queries than non-IoT devices [60], [61]. Moreover, devices can be identified by the *domain names* they communicate with [27].
  The most frequently used DNS characteristics are: i) the number of unique DNS queries, ii) the number of unique domain names, iii) the most frequently queried domain names, iv) the number of DNS packets, and v) the number of DNS errors. The papers [27], [54], [55], [60]–[63] exploited these features.
- **TLS features:** TLS/SSL is used by many IoT devices to secure internet communication with servers. The TLS protocol consists of two layers: handshake and record protocols. The handshake layer is the most interesting as it comprises of "text-in-the-clear" messages exchanged between devices and servers to create a secure channel and negotiate ciphers and encryption keys. Fan *et al.* [54] use the number of TLS handshakes as a feature. Sun *et al.* [49] analyze the unencrypted data of the TLS handshake and exploit the plaintext data in the `ClientHello`, `ServerHello`, and `Certificate` messages to derive the following features: the list of proposed ciphersuites, the list of announced extensions, and the length of the public key. The authors noted less fluctuation in the distribution of ciphersuites and TLS extensions in IoT devices, compared to non-IoT devices, because they advertise a limited and fixed number of ciphersuites. Thangavelu *et al.* [62] used the following TLS features: the minimum, maximum, and mean of the TLS packet length, the flow duration, and the number of TCP keep-alive probes used in the TLS session. Valdez *et al.* [64] derive features from TLS session initialization messages (`ClientHello` and `ServerHello`). Features include negotiated ciphers, proposed cipher suites, server name, and destination end-point.

c: Time-related features

They measure the temporal aspects of the flow. Examples include the inter-packet arrival time (IAT), i.e. the time interval between two consecutive packets received, the time a flow was active before becoming inactive, the time the last packet was switched [25] and the flow duration, etc. For instance, in [27], [60], the authors calculate the sleep time of a device, the average time interval between two consecutive DNS requests, and the NTP interval. Thangvelu *et al.* [62] consider the flow activity duration. Sun *et al.* [49] calculate idle time as it reflects device activity frequency.

It is worth noting that the **IAT** is one of the most useful time-related features as it varies by device depending on the hardware and software configurations [65]. It is therefore, widely used in the literature ([9], [16], [49], [51], [54], [66]–[69]). In particular, we note that the classification of ZigBee, Z-Wave, and Bluetooth IoT devices is often *exclusively based on IAT* [66], [67].

d: Periodicity features

IoT devices generate background communications that always present relatively constant and periodic patterns. Some researchers [20], [70] extract features from periodic flows. To do this, they first discretize the flow into a binary time series signal representing the existence or not of packets in the traffic each second. Then, they use the Discrete Fourier Transformation to identify the different distinct periods of the signal. Once identified, statistical features are used to describe these periods in detail. Examples include: the number of periods, the maximum and minimum period values, the averages of the occurrence of periods at the minimum period value, and the accuracy and stability of the inferred periods [20], etc. Note that approaches for extracting periodicity features often use the time-window-based stream definition (Fig. 7). Only **2 papers**, namely [20], [70] use periodicity features.

3) How are stream-level features calculated?

We identified two approaches to calculate stream-level features: concatenation and statistics.

a: Concatenation:

Stream-level features can be calculated by concatenating individual packet features. The authors of [51] define a $n \times 7$ feature matrix with 7 packet header features per packet (n packets). Similarly, Wan *et al.* [69] describe a stream of p packets defining a device signature using p vector attributes.

In general, only approaches defining the stream as a set of N packets (see Sec. V-B1a) use this method because concatenating a small number of packets is unlikely to create large signatures.

b: Statistics:

The second way is to perform statistical calculations on packet-level features. Depending on whether the measured feature is numerical (e.g. TTL) or categorical (e.g. protocol type), different statistics can be generated, as described below.

*For **numerical** features, researchers often calculate:*

- The traditional **minimum, maximum, mean, sum, standard deviation, variance**, which are widely used in the literature.
- The **entropy**, which measures the degree of disorder of features. It is a way of describing the nature of the data without focusing on the data itself. For example, the payload entropy indicates the information content of a packet. Packets including text data have less payload entropy than packets carrying audio data [47]. The authors of [47], [48], categorized IoT devices by payload

entropy. Fan *et al.* [54] calculate the entropy of top DNS requests and packet lengths.

- The **skewness** [71] and **kurtosis**, [72] which measure the asymmetry and the "tailedness" of the probability distribution, respectively. In [56], the authors use packet length skewness to explore packets' different lengths in a flow.
- The **augmented Dickey-Fuller (ADF)** test [73], which determines whether or not a given time series is stationary. It was used in [54] to capture how some devices send large packets in a short period of time, causing packet length to shift substantially.
- The **spectral density**, which characterizes a stationary population time series in the frequency domain. The authors of [74] use spectral analysis of packet length to record device communication patterns, differentiate IoT and non-IoT traffic, and determine the device class generating the packet flows.
- Note that when the stream is defined by a time window, finer granularity statistics can be generated by computing the *first quartile, second quartile, and third quartile* of numerical packet features, as [54] does for the "packet length."

*For **categorical** features, researchers often:*

- **List or count feature values**. Huang *et al.* [51] use a binary vector coded according to whether specific protocols exist in the traffic flow. In [49], [56], [70], the authors count the types of protocols involved in the device's communication traffic.
- **Determine the dominant values or their proportion**. For example, Msadek *et al.* [68] identify the set of dominant protocols (the most used). Zhang *et al.* [70] count the proportion of TCP/UDP/ARP in the device communication flow.

C. WHAT ABOUT AUTOMATIC FEATURE EXTRACTION?

While traditional ML algorithms require costly handcrafted features, deep learning approaches may automatically extract and learn the optimum features for the classification, directly from raw data. As DL requires standardized input data of the same type and size for all samples, researchers first convert pcaps into a suitable model input. To do so, Greis *et al.* [75] consider the packet captures (in `pcap` format) collected during the setup phase and transform the first 784 bytes of traffic into a 28×28 **grey-scale image**. Each pixel represents a grey value between 0 (black) and 255 (white). When a setup phase has less than 784 bytes, the remaining pixel values are set to 0 (black). Similarly, Kotak *et al.* [76] use TCP payload to create greyscale images of the device's communication pattern.

Yin *et al.* [77] rely on **traffic vectorization**. They use the first 10 packets to characterize a flow. This number was chosen because the average number of packets in most IoT flows is 10. A flow is described using 2.500 bytes of data (first 10 packets $\times$ 250 bytes). The first 250 bytes of each packet are concatenated. Streams with fewer than 10 packets employ padding.

Discussion: Despite the benefits of these approaches, which simplify and automate feature extraction, transforming data into another format (image, vector, etc.) can lead to semantic information loss. Moreover, this strategy does not take into account expert knowledge, which can help find the most important features. A minority of research papers (12% [75]–[77]) explored this solution.

D. OPEN-SOURCE FEATURE EXTRACTION TOOLS

This section describes the existing feature extraction tools found in the literature. The input of a feature extraction tool is network traffic in `pcap` format collected by a packet capture tool (e.g. tcpdump). The output is text-based format files (often `csv`) containing feature vectors. A feature vector is calculated for each observation.

CICFlowmeter [78] is an open-source feature extractor that produces more than 80 volume- and time-related features per TCP flow. The authors use two methods to measure the attributes. In the first approach, they measure *time-related* features over the full TCP flow, such as the time between packets or the time the flow remains active. In the second approach, they fix the time (e.g., every 1 second) and measure other volume-related attributes (e.g., bytes per second or packets per second).

Bekerman *et al.* [79] present a feature extraction tool, which is implemented on top of Wireshark [22] and extracts 972 behavioral features across different protocols and network layers. The features describe different observations of various granularities, namely i) *a conversation window*, ii) *a group of sessions*, iii) *a session* (e.g., a TCP session), and iv) *a transaction*, i.e., an interaction (request-response) between a client and a server.

Joy [80] extracts features from live network flows with a focus on application layers. The main features are: IP packet arrival lengths and times, the sequence of TLS record arrival lengths and times, other unencrypted TLS data, such as the list of proposed and selected ciphersuites, DNS names, addresses, TTLs and HTTP header elements, etc.

E. FEATURES DIMENSIONALITY REDUCTION FOR BETTER CLASSIFICATION

Feature dimensionality reduction improves classification accuracy and reduces the computational cost. This is a preprocessing phase that identifies relevant features and removes irrelevant or redundant ones. Feature dimensionality reduction is not widely used in IoT device classification. Only 30% of reviewed papers apply this step. This is because most publications rely on expert knowledge to derive an accurate and small set of features, making feature reduction unnecessary. On the contrary, articles using feature extraction tools (see Sec V-D) generate a large number of features and minimize them using feature reduction.

Ghojogh *et al.* [81] review feature dimensionality reduction approaches. They divide them into two groups: 1) *feature*

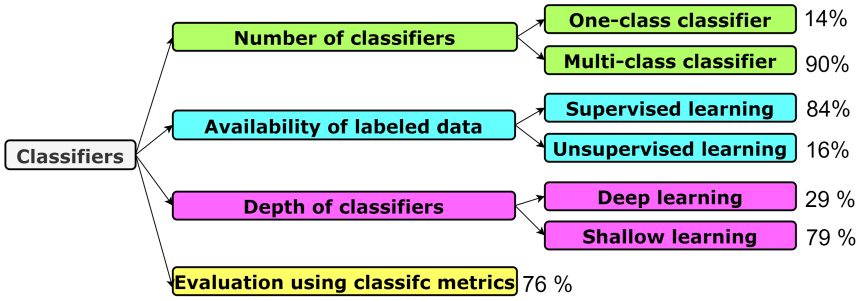

FIGURE 8. Taxonomy of ML based classification approaches. Percentages show how often each approach is used in the reviewed papers (the percentages do not always sum up to 100 because some papers use algorithms from multiple categories).

extraction approaches, where features are *projected* into a lower dimensional subset to extract a new set of features, and 2) *feature selection approaches*, where the best subset of original features is *selected*.

Note that the term "feature extraction" is also improperly used in the literature to represent the process of describing observations by a vector of features (cf. Sec V).

1) Approaches using feature extraction based dimensionality reduction

Thangavelu *et al.* [62] use a common feature extraction method called "*Principal Component Analysis*" (PCA). Fan *et al.* [54] use *Convolution Neural Network-* (CNN) based dimensionality reduction. Similarly, Bao *et al.* [82] use *auto encoders* for dimensionality reduction. Auto encoders learn a mapping from high-dimensional observations to a lower-dimensional representation space such that the original observation can be reconstructed from the lower-dimensional representation [83]. Auto Encoders are widely used for feature learning in general [81]. Similarly, *representation learning* [84] is a feature extraction method used to *learn* automatic discriminative features. It has not been explored yet for IoT device classification.

2) Approaches using feature selection based dimensionality reduction

According to Ghojogh *et al.* [81], there are two feature selection approaches: i) filter methods, and ii) wrapper methods.

a: Approaches employing filter methods

Such methods minimize the feature set by selecting the most discriminative ones. The *Correlation Criteria* is one of the most widely used solutions. It is based on calculating the correlation between each feature and the label vector. The features with the highest correlation value are selected. Sivanathan *et al.* [59] use *Correlation-based Feature Subset* (CFS) and *Information Gain* (IG). Similarly, Cvitic *et al.* [32] use CICFlowmeter for feature extraction (83 features) and then apply IG.

b: Approaches applying wrapper methods

Such approaches select the features based on the classifier's performance. Thus, the selected set can vary from one classifier to another. For instance, in [85], the authors use a *genetic algorithm* based feature selection method. The genetic algorithm determines the smallest set of packet header features in all network layers that contributes significantly to the classification for a given classifier.

VI. CLASSIFICATION

The aim of the classification step is to predict for each traffic input, represented by a vector of features $X = \{x_1, ...x_f\}$, the class c of the device that has generated it. Different classification approaches have been explored in the literature. We will classify them according to i) the number of classes (multi-class classifier or one-class classifier), ii) supervised or unsupervised approaches, and iii) shallow or deep learning algorithms. Fig. 8 illustrates the classification results.

A. MULTI-CLASS VS ONE-CLASS CLASSIFIER

1) Methods using multi-class classifier

Only one classifier is used for the multi-class classification. The trained classification model outputs a vector of class membership probabilities $P^s = \{p_i^s\}_{1 \leqslant i \leq n}$ denoting the likelihood that the inspected traffic sample s comes from device class c_i. The traffic is labelled as originating from the device having the highest probability. To capture unknown devices, a threshold parameter tr can be defined and fine-tuned using the validation dataset. If one probability p_i^s exceeds the threshold parameter tr ($p_i^s > tr$), the traffic is classified as originating from the device class c_i. Otherwise, it is classified as unknown. A device can also be considered as unknown if the feature vector matches more than one class with a low discriminative threshold (0.5 for example). This is the most popular method in the state of the art (90% of reviewed papers).

2) Methods using one-class classifier (a classifier per device)

A minority of reviewed papers (14%) use this classification approach. In the following, we describe how this strategy is employed in the literature. This consists of splitting the

dataset into numerous binary classification problems (focusing on a single class, regardless of all other classes) and then a binary classifier is trained for each device. Each classifier provides either i) a probability p_i that the traffic was generated by a device class c_i or ii) a binary decision on whether the input matches the device type. In the first case, a threshold t (cutoff value) should be set. If $p_i > t$, the traffic is labelled as originating from the device class c_i. t is empirically set to maximize the classifier's accuracy [58]. In the second situation, if a device is accepted by multiple classifiers, the conflict should be resolved, for example, by computing a distance-based metric between the sample to identify and a subset of samples from each class that it has a match for [14], or by applying majority votes [15] to break the tie between multiple matches.

Note that using this strategy, classification accuracy can be increased by evaluating the classification results of more than one sample before choosing the device class. For example, in [58], the authors perform a majority vote on the classification results of *several consecutive TCP sessions* to determine, with an accuracy of 100%, if they were generated by a certain device. The optimal number of consecutive sessions is defined as the minimum number of sessions on which a majority vote provides zero false positives and zero false negatives on the test dataset.

Discussion: Generating a model for multi-class classifiers is challenging in practice: when a new device type is added to the network or the behavior of existing device types legitimately changes (due to firmware upgrades by device manufacturers, for example), the entire model should be retrained for all classes [86]. On the contrary, building a classifier per device avoids costly re-learning if a new device type is added. In addition, building a classifier per device allows for the discovery of new devices: if a sample is rejected by the classifiers, it may be identified as a new device type. Another advantage is its interpretability. When the number of features is important, one classifier per class gives a set of interpretable models instead of one complex model.

However, the one-class classifier approaches are more computationally expensive since the results of more than one classifier should be computed. Moreover, managing conflicts might be time-consuming if a sample fits many device types. As reported in [14], most device type identification time is spent on tiebreaks. Moreover, unbalanced training datasets can affect classifier performance (there are generally fewer samples for one device type compared to the samples of all the remaining samples combined). This issue can be solved by utilizing under-sampling and over-sampling approaches [87].

B. SUPERVISED, UNSUPERVISED

ML-based classification algorithms are often classified into supervised and unsupervised approaches, with the well-known advantages and limitations of each briefly described below.

1) Supervised classification

In supervised classification, labeled datasets are split into training, validation, and test datasets. Datasets can be separated chronologically or randomly. However, temporal partitioning better matches the real world scenario, when a classifier is trained on existing data and then tested on new data. Despite the cost of labeling and the difficulty of detecting new devices not included in training, supervised classification techniques are commonly employed in IoT device classification literature (84% of reviewed papers) due to their high accuracy and ease of implementation.

2) Unsupervised classification

Supervised techniques use labeled device class data. Labeling involves significant human effort, which is tedious and not scalable given the growing number of IoT devices.

Discussion: Unsupervised learning is more scalable since it minimizes human assistance, but it is harder to execute and its accuracy is likely to be lower than supervised approaches. Thus, only 16% of reviewed papers use unsupervised classification approaches. For instance, the authors of [43] propose a classification method using semi-supervised GANs (generative adversarial networks).

C. SHALLOW AND DEEP LEARNING

Deep learning uses multiple layers of nonlinear processing units. All non-deep learning approaches are shallow learning, including most machine learning models before 2006 and neural networks with one hidden layer.

Discussion: Despite the advantages of deep learning, the majority of reviewed papers (79%) still use shallow classification algorithms, probably due to its simplicity and ease of implementation and because some shallow algorithms are intrinsically interpretable, like decision trees. Random Forest is a popular classifier due to its accuracy and speed, but its classification time grows linearly with the number of classes, so it may not scale to a large number of device types.

D. EVALUATION SCENARIOS

Accuracy, precision, recall, F_1 score, and ROC are classic evaluation metrics. **Accuracy** measures the ratio of correctly predicted observations to the total observations. **Precision** indicates what percentage of positive predictions were correct. **Recall** defines what percentage of positive cases a classifier has caught. F_1 **score** is a harmonic average of precision and recall.

Discussion: Most of the reviewed research papers (79%) focus on classic evaluation metrics. However, traditional evaluation does not accurately measure the performance and limitations of classification algorithms. For instance, accuracy gives equal weights to all classes, which is inappropriate if the dataset is unbalanced (e.g. you can have 90% of total accuracy but, in minority classes, most samples are misclassified). The performance of classifiers should be assessed in different scenarios and through diverse metrics and measures.

Below, we describe some other metrics found in the literature to inspire other evaluation methodologies.

a: Measuring classification and learning speed

The *learning time* is significant since classification models that learn rapidly are more adapted to real conditions [20]. The *classification time* (the time required to classify one sample) is critical for instant device identification [14]. In [46], the authors evaluated the training time, *the latency*, i.e., the time spent performing device identification, and *throughput*, i.e. the number of identifications per second.

b: Measuring CPU, memory consumption and computational complexity

In [14], the authors measure the CPU used by the security gateway for the classification and for the enforcement mechanism. In [88], the authors calculate the computational complexity of the different steps of their solution, namely feature extraction, clustering, and model training. The feature extraction cost is estimated to be $m \times O(n)$ where m is the number of features and n is the number of packets in the session. The cost of clustering is calculated based on the steps and loops in the proposed algorithm. The Random Forest training cost depends on the feature vector dimension, the number of decision trees, and the number of training samples.

c: Varying evaluation scenarios

Some papers measure the variation of performance metrics in different scenarios. For example, Huang *et al.* [51] test the scalability of their approach and show that accuracy diminishes with many device types. Meidan *et al.* [15] measure the classification accuracy as a function of the number of consecutive sessions needed for classification. Similarly, Song *et al.* [89] examine the relationship between identification accuracy and the number of packets required for classification. Bai *et al.* [56] measure the classification results under different time window sizes and over different ratios of training and testing datasets. Similarly, Marchal *et al.* [20] assess the evolution of accuracy as the number of training samples changes.

d: Additional evaluation metrics

In addition to classic metrics, other evaluation scenarios can been explored. We give the following examples: i) *robustness* to adversarial attacks [90] to evaluate the classifier quality on ambiguous examples, ii) *explainability* [91], i.e. if the result can be simply interpreted, to provide better acceptance of ML-based solutions in IoT, iii) *transferability* [92], that is, whether a model learned in one context can be applied in another, in order to reduce learning costs and provide "out-of-the-box" tools.

VII. GRANULARITY OF CLASSIFICATION

In the literature, IoT devices are classified at different levels of granularity. Bezawada *et al.* [47] enumerate three classification levels: i) category, ii) type, and iii) instance (cf. Fig. 9). A *device category* is a grouping of similar devices; for instance, devices can be grouped by *function*, e.g., cameras, sensors, or home assistants. A *device type*, however, designates a more specific device model within a general device category. For example, Google Home Mini (GHM) and Amazon Alexa are device types within the category of home assistants. Finally, *a device instance* is a physical device instance of a device type. For example, two different GHMs in the same network are two instances of the GHM device type. In the following, we examine how these different levels of classification have been considered in the literature.

A. CLASSIFYING DEVICES BY CATEGORY

Different definitions of "category" have been proposed in the literature. The most used definition relies on "the main functionality (or purpose) of the device," e.g. refrigerator, TV, watch, or camera, as proposed in [15], [47], [58]. For instance, in [56], the devices are classified into hubs, electronics, cameras, and switches & triggers. In [93], four categories are discussed: IP cameras, smart on/off plugs, motion sensors, and temperature/environmental sensors. A more broader definition is proposed in [94], where the authors classify the devices according to their *application domain* into healthcare, multimedia, hubs, etc. Note that only 22% of papers examined in this survey use this classification level.

As the number of IoT devices grows, so do their applications and features, requiring new device category definitions. To this end, Cvitic *et al.* [32], [95] propose classifying devices according to their "*Cu* predictability index." *Cu* measures the "level of predictability of behavior" of the device. To do this, *Cu* measures the variation in data received and sent by a device over a period of time. Devices that behave in roughly the same way over time are easily predictable, whereas devices whose usage and interaction with the user modifies their behavior (and consequently the data received and sent) are more difficult to predict. The authors derive four device categories based on *Cu*. In doing so, the authors propose a more general definition of the IoT device category.

B. CLASSIFYING DEVICE BY TYPE

This is the most common approach in the literature (81% of surveyed papers). There are several ways to define the device type. For instance, in [14], a device type denotes the "combination of model and software version" of a particular device. In [44], a device type is defined by three parameters: the `manufacturer`, the `manufacturer-type`, and the `manufacturer-type-model`, e.g., "amazon-kindle-v2.0." In [82], a device type is defined by the manufacturer and model (e.g. for security cameras: Simple_Home XCS7_1001).

C. CLASSIFYING DEVICES BY INSTANCE

This is the finest level of granularity, where instances of the same device type must be distinguished. It is also the most difficult and expensive scenario. It should be noted that, in the literature, the use of the term fingerprint does not reflect

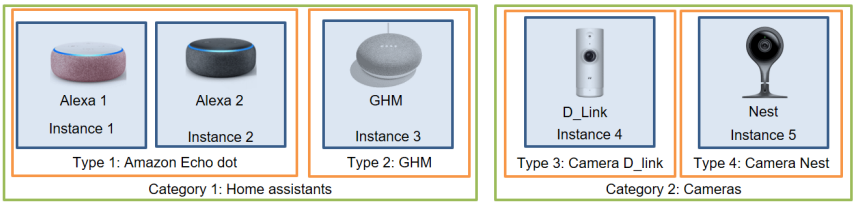

FIGURE 9. IoT devices classification levels. IoT devices can be classified at different levels of granularity: i) category, ii) type, and iii) instance. In this example, *home assistants* and *cameras* are two different categories of IoT devices. *GHM* (Google Home Mini) and *Amazon Echo Dot* are two types of *Home assistants*. Finally, *Alexa 1 and Alexa 2*, are two instances of the Amazon Echo Dot.

FIGURE 10. Taxonomy of the classification granularity: the approaches are classified by granularity of classification. Percentages show how often each approach is used in the reviewed papers.

the definition of *device instance* we propose in this survey but rather refers to device identification, i.e., the classification of devices based on their type. Therefore, proposed solutions for device fingerprinting do not distinguish between device instances.

Discussion: Instance level classification has not been sufficiently explored in the literature. To the best of our knowledge, no solution in the literature exists for this scenario. However, is such a classification really necessary? The answer depends on the use case. For example, when detecting vulnerable devices, instance-based classification is not necessary since instances of the same type share vulnerabilities. However, instance-based classification could be useful in some use-cases. For example, in [96], the authors focus on 5G resource allocation and design a solution for automatically selecting a 5G slice based on the type of IoT device connecting to the network, which is detected through a classification of the radio signal shape. An extension could be envisaged based on instance device classification, where two instances of the same IoT device used by two users *with different rights* are distinguished. This enables better 5G resource management based on the user profile. Instance-based classification could also be useful to track a unique user's device.

D. HOW TO SELECT THE BEST CLASSIFICATION GRANULARITY?

The granularity of classification should be carefully set depending on the application scenario. Category level classification may be sufficient in many situations. For instance, to ensure QoS by giving different priorities to flows (e.g., prioritizing traffic from healthcare devices during periods of high load), it is not necessary to know the manufacturer and software of the device. Even though the category level classification of IoT devices is not very precise, it has the advantage of being scalable.

Device type classification is the most commonly used classification level in the literature due to its better ratio between accuracy and ease of implementation. However, many results [14], [97] have shown that it is difficult to distinguish devices from the same manufacturer or with the same firmware version. This is because these devices usually have similar hardware and software architecture and communicate with the same remote cloud servers using the same protocols. Thus, they often share very similar traffic patterns. Note that this problem is very close to the instance-based classification problem, which is still an open problem.

VIII. KEY RESEARCH DIRECTIONS

In what follows, we consider research directions that have received little or no attention in the literature. Following the paper's rationale, we discuss challenges related to data-acquisition, feature extraction, and machine learning. We address unbalanced data sets and provide solutions in VIII-A, the importance of minimizing feature extraction costs in VIII-B and improving learning quality in VIII-C. In sections VIII-D, VIII-E, and VIII-F, we discuss challenges related to scalability, deployment in practice and lack of standardization, respectively.

A. THE PROBLEM OF UNBALANCED DATASETS

This is a common problem in many ML applications, but it is accentuated in IoT device classification due to the heterogeneous behavior of IoT devices: some devices, like plugs, generate sparse traffic, while others, like cameras, generate large amounts of traffic. This makes the detection of minority class devices difficult. Bai *et al.* [56] report limited data for detecting hubs and Hsu *et al.* [93] remark that it is difficult to distinguish smart plug traffic from IP cam traffic. Thus, having a balanced dataset is more important than the size of the dataset.

Solutions based on data augmentation can be considered during the training phase [48]. However, it is important to avoid introducing biases when over-representing minority classes. There is therefore a trade-off to consider to avoid overfitting the model.

B. REDUCING THE COST OF FEATURE EXTRACTION

It is essential to consider the cost of extracting features. Chakraborty *et al.* [98] distinguish three types of feature ex-

traction costs: i) the computational cost involves computing resources used to calculate the features, ii) the memory cost measures the memory used to store running feature values while computing, and iii) the privacy cost is related to privacy violation, especially for features extracted from the payload that may contain sensitive information. Desai *et al.* [99] propose a framework for ranking features according to their discriminatory power to differentiate between devices. They demonstrate that a small set of highly ranked features is sufficient to achieve an accuracy close to that obtained using all features.

Note that using a limited number of features limits the feature extraction cost but can make the classification approach more vulnerable to adversarial attacks. In fact, it is easier for an attacker to generate traffic that mimics the distribution of values taken by one feature (e.g., packet size, or IAT, etc.) to imitate the behavior of a particular IoT device and bypass the classifier. For instance, Shahid *et al.* [100] generate sequences of packet sizes representing bidirectional flows that look as if they were generated by a real smart device. However, it is more complex to bypass a classifier that takes into account the values of several features because it is difficult to generate traffic that matches the values of all these features at the same time.

C. IMPROVING THE QUALITY OF LEARNING

1) Need for continuous learning

The IoT ecosystem and device behavior evolve rapidly. Thus, classification models must be updated to reflect recent data trends. Kolcun *et al.* [34] note that the accuracy of IoT device classification models falls by 40%, a few weeks after learning, and argue that to preserve the accuracy of the models, they need to be continuously updated. It is then necessary to explore continuous learning ML pipelines that keep the machine-learned models up-to-date [101]. As mentioned in Sec. VI-A2, techniques that train a classifier model per device are more easily re-trained.

2) Scarcity of labeled data

Fan *et al.* [54] note that collecting and labeling data is costly and time-consuming, which cannot be scaled to the overgrowing IoT environment. However, when labeled data is scarce, supervised learning techniques fail. Using semi-supervised or unsupervised approaches are possible solutions. Fan *et al.* [54] proposed an IoT identification model based on semi-supervised learning. To do so, they i) judiciously choose the features describing the traffic, ii) perform a CNN based dimensionality reduction, and then iii) perform the classification using a two-layer neural network, classifying the traffic into IoT and non-IoT, then specifying the class of IoT objects. They managed to get 99% accuracy using only 5% of labeled data.

Generating labeled synthetic data is another solution: e.g, generative adversarial networks (GAN) can generate synthetic data close to the real distribution of training data by capturing the hidden class distribution. In addition, training classifiers with additional synthetic data points gives them better generalization ability [100].

3) Resilience to adversarial attacks

The vulnerability of ML algorithms to adversarial attacks has been demonstrated in several applications, and ML-based IoT device classification is no exception [102]. For example, malicious devices may attempt to mimic the traffic of a legitimate device in order to connect to the network. Fortunately, it is very difficult to do this while preserving the intended malicious functionality [103]. As discussed in [15], the rogue device must be able to generate similar requests to the manufacturer's servers and get similar responses, which is difficult to achieve if device authentication is required.

4) Transferability of the classification model

Kolcun *et al.* [34] reveal that the accuracy of classifiers degrades over time when evaluated on data collected outside the training set. However, it is desirable that classifiers that perform well in one context can be used in another without expensive retraining. Transfer learning [104] is a promising solution that should be explored. For example, it would allow a manufacturer to build a model that learns the behavior of an IoT device and use the model in a smart home to identify the device with little-retraining.

D. DISCUSSING SCALABILITY

Given the exponential growth of the number and types of IoT devices, it is crucial to design scalable solutions. *Scalability must be considered at all stages of the solution design, as explained below.*

1) *Traffic collection:* the collection must be quick, efficient, and non-exhaustive. For instance, **data sampling** [105] (i.e., taking sufficiently representative samples rather than the entire dataset) can be used to improve scalability. However, the choice of the sampling solution must be well thought out as it may be inappropriate for minority and sparse traffic classes, which brings us back to the unbalanced dataset problem, discussed above in Sec. VIII-A.
2) *Feature extraction:* feature extraction should not be complex, long, or costly. It is important to choose a scalable method. For example, packet-level feature extraction is very time- and computation-consuming, and it is therefore not scalable. On the other hand, deep learning (cf. Sec V-C) could be improved to simplify and automate the feature extraction process, and is therefore more likely to be scalable.
3) *ML-based classification:* the number of classifiers (one-class classifier or multi-class classifier) should allow for easier extension to new classes and avoid extensive updating of all the models, as discussed in Sec. VI-A2.
4) *Classification granularity:* Bai *et al.* [56] noticed a decrease in accuracy with the increase in the number of classes. One solution is to carefully choose the classifi-

cation granularity according to the final application, as discussed in Sec. VII-D.

Moreover, with the emergence of edge computing, it is interesting to use the powerful computing and storage capabilities provided by neighboring edge servers to facilitate the IoT device classification and make it more scalable. A first attempt was proposed by Sun *et al.* [88] who designed an edge-based IoT device classification scheme. Transfer learning, discussed above, can also be used for scalability by minimizing learning time.

E. DEPLOYMENT IN PRACTICE

We observe a gap between academic advancements and market implementation since reviewed IoT device categorization solutions are seldom (if ever) deployed.

Indeed, most proposed solutions have not been implemented using a realistic case study. Hence, their contribution to improving the security or management of the IoT system has not been evaluated, making their actual effectiveness uncertain. The lack of such evaluation scenarios is due to the difficulties of implementing and mastering realistic and usually complex ecosystems. In addition, the challenges discussed above need to be addressed to make the solutions more mature and ready for market implementation.

F. MUD AND STANDARDIZATION

Another solution for classifying and identifying IoT devices would be to use the Manufacturer Usage Description (MUD) [106]. The MUD is a standard defined by the IETF [107] that allows IoT device manufacturers to publish device specifications, including intended communication patterns. IoT devices generally perform a specific function [108], and therefore have a recognizable communication pattern, which can be captured formally and concisely as a MUD profile [109]. Unfortunately, current IoT manufacturers do not yet support MUD specifications and mechanism. Hamza *et al.* [109] publicly share their tool called MUDgee [110] to automatically generate MUD profiles of IoT devices.

IX. CONCLUSION

Classifying IoT devices has been proposed as a potential solution to secure and manage the IoT ecosystem. This paper reviewed relevant literature to answer the following research questions:

Q1. How to design a practical data-acquisition method?

Sec. IV showed that it is more practical to collect traffic data from both IoT and non-IoT devices as they co-exist in smart homes and can easily be confused.

There are three device operation modes: setup, idle, and interaction. Traffic generated by the devices during idle and interaction modes is abundant and widely used in the literature. The setup traffic is stable and allows for rapid identification once the device is connected to the network. However, setup traffic is difficult to collect since the initialization phase may not appear multiple times in the device's lifetime.

It is also more genuine to collect traffic from outside the home (i.e., place the probe after the gateway, rather than at the gateway) because this reflects real IoT device classification use-cases. We found that this scenario is understudied in the literature.

Finally, most public datasets we surveyed suffer from the above-mentioned biases, which are also found in papers using them.

Q2. How to build effective machine learning classifiers for IoT device classification?

Sec. V discussed feature extraction approaches and showed that it is more scalable and less expensive to extract features from streams rather than packets.

Traffic can be split by time interval, number of packets, or connection-wise. Dividing traffic by connection (packet flows between two endpoints) is natural and straightforward, but not always appropriate. For instance, to identify a device upon its connection to a network, it is more suitable to consider the first generated packets rather than the whole flow.

Setting the optimal time window or packet count is required for flow splitting, yet this is challenging. Small flows allow for rapid classification but may not be enough to characterize the device. On the contrary, large flows can be time- and memory-intensive to analyze. Moreover, IoT devices generate varying amounts of data at different rates, so the appropriate number of packets may vary.

We identified the most discriminative features and discussed how to calculate them (concatenation, statistics).

Q3. How to build effective machine learning classifiers for IoT device classification?

Analysis in Sec. VI showed that creating one multi-class classifier is not scalable and evolutive because the entire model must be retrained when a new device type is added to the network. On the contrary, building a classifier per device reduces costly re-training, allows discovery of additional device kinds, and makes decisions more interpretable. However, one-class classifier techniques are more computationally demanding since the results of several classifiers must be computed and managed.

Unsupervised learning is more scalable and suited for the rising variety of IoT device types than supervised learning since it minimizes labeling costs. However, unsupervised learning is understudied in the literature. Moreover, more evaluation scenarios and metrics are required for realistic assessment of classification algorithms.

Q4. How to set the classification granularity? In Sec. VII, we discussed category-, type-, and instance-based classification.

Type-instance device classification achieves the best trade-off between accuracy and ease of implementation. Despite being imprecise, the category level classification of IoT devices is scalable and thus more adapted to the IoT context where devices' diversity is growing. To avoid costly classification, granularity level should be set depending on the application context.

We analyzed more issues and suggested new study directions in Sec. VIII. We discussed scalability and implementation in practice and recommended looking at additional challenges like adversarial attack robustness and model transferability.

• • •

REFERENCES

[1] K. L. Lueth, M. Hasan, S. Sinha, S. Annaswamy, P. Wegner, F. Bruegge, and M. Kulezak, "State of iot-spring 2022," IoT Analytics, Tech. Rep., 2022. [Online]. Available: https://iot-analytics.com/number-connected-iot-devices

[2] F. Shaikh, E. Bou-Harb, J. Crichigno, and N. Ghani, "A machine learning model for classifying unsolicited iot devices by observing network telescopes," in *2018 IWCMC*. IEEE, 2018, pp. 938–943.

[3] S. Biddle, "Wikileaks dump shows cia could turn smart tvs into listening devices," https://theintercept.com/2017/03/07/wikileaks-dump-shows-cia-could-turn-smart-tvs-into-listening-devices/, 2017.

[4] Today the web was broken by countless hacked devices – your 60-second summary. [Online]. Available: https://www.theregister.com/2016/10/21/dyn_dns_ddos_explained/

[5] T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *IEEE Communications Surveys Tutorials*, vol. 10, no. 4, pp. 56–76, 2008.

[6] P. Yadav, A. Feraudo, B. Arief, S. F. Shahandashti, and V. G. Vassilakis, "Position paper: A systematic framework for categorising iot device fingerprinting mechanisms," in *Proceedings of the 2nd International Workshop on Challenges in Artificial Intelligence and Machine Learning for Internet of Things*, ser. AIChallengeIoT '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 62–68. [Online]. Available: https://doi.org/10.1145/3417313.3429384

[7] H. Tahaei, F. Afifi, A. Asemi, F. Zaki, and N. B. Anuar, "The rise of traffic classification in iot networks: A survey," *Journal of Network and Computer Applications*, vol. 154, p. 102538, 2020.

[8] A. Sivanathan, H. H. Gharakheili, and V. Sivaraman, "Can we classify an iot device using tcp port scan?" in *2018 IEEE International Conference on Information and Automation for Sustainability (ICIAfS)*. IEEE, 2018, pp. 1–4.

[9] M. R. Shahid, G. Blanc, Z. Zhang, and H. Debar, "Iot devices recognition through network traffic analysis," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 5187–5192.

[10] C. Xenofontos, I. Zografopoulos, C. Konstantinou, A. Jolfaei, M. K. Khan, and K.-K. R. Choo, "Consumer, commercial and industrial iot (in) security: attack taxonomy and case studies," *IEEE Internet of Things Journal*, 2021.

[11] Gartner, "Gartner says 8.4 billion connected "things" will be in use in 2017, up 31 percent from 2016," https://www.gartner.com/en/newsroom/press-releases/2017-02-07-gartner-says-8-billion-connected-things-will-be-in-use-in-2017-up-31-percent-from-2016.

[12] O. Salman, I. H. Elhajj, A. Kayssi, and A. Chehab, "A review on machine learning–based approaches for internet traffic classification," *Annals of Telecommunications*, vol. 75, no. 11, pp. 673–710, 2020.

[13] P. M. S. Sánchez, J. M. J. Valero, A. H. Celdrán, G. Bovet, M. G. Pérez, and G. M. Pérez, "A survey on device behavior fingerprinting: Data sources, techniques, application scenarios, and datasets," *IEEE Communications Surveys & Tutorials*, 2021.

[14] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A. 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)*, 2017, pp. 2177–2184.

[15] Y. Meidan, M. Bohadana, A. Shabtai, M. Ochoa, N. O. Tippenhauer, J. D. Guarnizo, and Y. Elovici, "Detection of unauthorized iot devices using machine learning techniques," *arXiv preprint arXiv:1709.04647*, 2017.

[16] I. Hafeez, M. Antikainen, and S. Tarkoma, "Protecting iot-environments against traffic analysis attacks with traffic morphing," in *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, 2019, pp. 196–201.

[17] 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, pp. 47–59.

[18] "Most popular smart home deviecs," http://iotlineup.com/.

[19] 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, pp. 559–564.

[20] 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*, vol. 37, no. 6, pp. 1402–1412, 2019.

[21] "tcpdump," https://www.tcpdump.org/.

[22] "Wireshark," https://www.wireshark.org/.

[23] X. Ma, J. Qu, J. Li, J. C. Lui, Z. Li, and X. Guan, "Pinpointing hidden iot devices via spatial-temporal traffic fingerprinting," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 894–903.

[24] X. Ma, J. Qu, J. Li, J. C. Lui, Z. Li, W. Liu, and X. Guan, "Inferring hidden iot devices and user interactions via spatial-temporal traffic fingerprinting," *IEEE/ACM Transactions on Networking*, 2021.

[25] 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*, vol. 97, p. 101968, 2020.

[26] Github - IoT_Sentinel Device Fingerprint. [Online]. Available: $https://github.com/andypitcher/IoT_Sentinel$

[27] 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*, vol. 18, no. 8, pp. 1745–1759, 2018.

[28] "UNSW IoT Analytics Database." [Online]. Available: https://iotanalytics.unsw.edu.au/

[29] O. Alrawi, C. Lever, M. Antonakakis, and F. Monrose, "Sok: Security evaluation of home-based iot deployments," in *2019 IEEE symposium on security and privacy (sp)*. IEEE, 2019, pp. 1362–1380.

[30] "YourThings Database," 2018. [Online]. Available: https://yourthings.info/data/

[31] 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, pp. 474–489.

[32] 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*, pp. 1–24, 2021.

[33] "IoT traffic." [Online]. Available: https://www.kaggle.com/datasets/5cae54f093f90a5a0613542573a16ab7c3f5dbf271cac549578266e7c63-2d7f9

[34] 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.

[35] "Dadabox data-set," https://github.com/DADABox/revisiting-iot-device-identification.

[36] "IoT traffic data." [Online]. Available: https://github.com/DongShuaike/iot-traffic-dataset

[37] "IoT-deNAT." [Online]. Available: https://zenodo.org/record/3924770#.YUiPs7gzZPY

[38] J. Ren, D. J. Dubois, D. Choffnes, A. M. Mandalari, R. Kolcun, and H. Haddadi, "Information exposure from consumer iot devices: A multidimensional, network-informed measurement approach," in *Proceedings of the Internet Measurement Conference*, 2019, pp. 267–279.

[39] "The mon(iot)r dataset." [Online]. Available: https://github.com/NEU-SNS/intl-iot

[40] S. Garcia, A. Parmisano, and M. J. Erquiaga, "IoT-23: A labeled dataset with malicious and benign IoT network traffic," Jan. 2020, More details here https://www.stratosphereips.org /datasets-iot23. [Online]. Available: https://doi.org/10.5281/zenodo.4743746

[41] "Iot-23 data-set," https://zenodo.org/record/4743746#.YW_GphpBxPZ.

[42] "Cisco.com, 2017. cisco - netflow."

[43] S. K. Nukavarapu and T. Nadeem, "Securing edge-based iot networks with semi-supervised gans," in *2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*. IEEE, 2021, pp. 579–584.

[44] 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, pp. 55–72.

[45] P. A. Abhang, B. W. Gawali, and S. C. Mehrotra, "Chapter 5 - emotion recognition," in *Introduction to EEG- and Speech-Based Emotion Recognition*, P. A. Abhang, B. W. Gawali, and S. C. Mehrotra, Eds. Academic Press, 2016, pp. 97–112. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128044902000051

[46] A. J. Pinheiro, J. d. M. Bezerra, C. A. Burgardt, and D. R. Campelo, "Identifying iot devices and events based on packet length from encrypted traffic," *Computer Communications*, vol. 144, pp. 8–17, 2019.

[47] B. Bezawada, M. Bachani, J. Peterson, H. Shirazi, I. Ray, and I. Ray, "Iotsense: Behavioral fingerprinting of iot devices," *arXiv preprint arXiv:1804.03852*, 2018.

[48] K. Kostas, M. Just, and M. A. Lones, "Iotdevid: A behaviour-based fingerprinting method for device identification in the iot," *arXiv preprint arXiv:2102.08866*, 2021.

[49] J. Sun, K. Sun, and C. Shenefiel, "Automated iot device fingerprinting through encrypted stream classification," in *International Conference on Security and Privacy in Communication Systems*. Springer, 2019, pp. 147–167.

[50] N. Ammar, L. Noirie, and S. Tixeuil, "Autonomous identification of iot device types based on a supervised classification," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.

[51] Q. Huang, Y. Song, J. Yang, M. Fan, and A. Hu, "A booting fingerprint of device for network access control," in *2019 3rd International Conference on Circuits, System and Simulation (ICCSS)*. IEEE, 2019, pp. 251–254.

[52] S. A. Hamad, W. E. Zhang, Q. Z. Sheng, and S. Nepal, "Iot device identification via network-flow based fingerprinting and learning," in *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. IEEE, 2019, pp. 103–111.

[53] M. R. Shahid, G. Blanc, Z. Zhang, and H. Debar, "Iot devices recognition through network traffic analysis," in *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 2018, pp. 5187–5192.

[54] L. Fan, S. Zhang, Y. Wu, Z. Wang, C. Duan, J. Li, and J. Yang, "An iot device identification method based on semi-supervised learning," in *2020 16th International Conference on Network and Service Management (CNSM)*. IEEE, 2020, pp. 1–7.

[55] F. Le, J. Ortiz, D. Verma, and D. Kandlur, "Policy-based identification of iot devices' vendor and type by dns traffic analysis," in *Policy-Based Autonomic Data Governance*. Springer, 2019, pp. 180–201.

[56] L. Bai, L. Yao, S. S. Kanhere, X. Wang, and Z. Yang, "Automatic device classification from network traffic streams of internet of things," in *2018 IEEE 43rd conference on local computer networks (LCN)*. IEEE, 2018, pp. 1–9.

[57] N. Brownlee, C. Mills, and G. Ruth, "Traffic Flow Measurement: Architecture," RFC 2722 (Informational), RFC Editor, Fremont, CA, USA, Oct. 1999. [Online]. Available: https://www.rfc-editor.org/rfc/rfc2722.txt

[58] Y. Meidan, M. Bohadana, A. Shabtai, J. D. Guarnizo, M. Ochoa, N. O. Tippenhauer, and Y. Elovici, "Profiliot: A machine learning approach for iot device identification based on network traffic analysis," in *Proceedings of the Symposium on Applied Computing*, ser. SAC '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 506–509.

[59] A. Sivanathan, H. H. Gharakheili, and V. Sivaraman, "Managing iot cyber-security using programmable telemetry and machine learning," *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 60–74, 2020.

[60] V. Melnyk, P. Haleta, and N. Golphamid, "Machine learning based network traffic classification approach for internet of things devices," *Theoretical and Applied Cybersecurity*, vol. 2, no. 1, Aug. 2020.

[61] A. Bremler-Barr, H. Levy, and Z. Yakhini, "Iot or not: Identifying iot devices in a short time scale," in *NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2020, pp. 1–9.

[62] V. Thangavelu, D. M. Divakaran, R. Sairam, S. S. Bhunia, and M. Gurusamy, "Deft: A distributed iot fingerprinting technique," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 940–952, 2018.

[63] R. Kolcun, D. A. Popescu, V. Safronov, P. Yadav, A. M. Mandalari, Y. Xie, R. Mortier, and H. Haddadi, "The case for retraining of ml models for iot device identification at the edge," *arXiv preprint arXiv:2011.08605*, 2020.

[64] E. Valdez, D. Pendarakis, and H. Jamjoom, "How to discover iot devices when network traffic is encrypted," in *2019 IEEE International Congress on Internet of Things (ICIOT)*. IEEE, 2019, pp. 17–24.

[65] S. Aneja, N. Aneja, and M. S. Islam, "Iot device fingerprint using deep learning," in *2018 IEEE International Conference on Internet of Things and Intelligence System (IOTAIS)*. IEEE, 2018, pp. 174–179.

[66] L. Babun, H. Aksu, L. Ryan, K. Akkaya, E. S. Bentley, and A. S. Uluagac, "Z-iot: Passive device-class fingerprinting of zigbee and z-wave iot devices," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–7.

[67] H. Aksu, A. S. Uluagac, and E. Bentley, "Identification of wearable devices with bluetooth," *IEEE Transactions on Sustainable Computing*, 2018.

[68] N. Msadek, R. Soua, and T. Engel, "Iot device fingerprinting: Machine learning based encrypted traffic analysis," in *2019 IEEE wireless communications and networking conference (WCNC)*. IEEE, 2019, pp. 1–8.

[69] Y. Wan, K. Xu, F. Wang, and G. Xue, "Iotathena: Unveiling iot device activities from network traffic," *arXiv preprint arXiv:2105.14405*, 2021.

[70] L. Zhang, L. Gong, and H. Qian, "An effiective iot device identification using machine learning algorithm," in *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*. IEEE, 2020, pp. 874–877.

[71] D. P. Doane and L. E. Seward, "Measuring skewness: a forgotten statistic?" *Journal of statistics education*, vol. 19, no. 2, 2011.

[72] K. P. Balanda and H. MacGillivray, "Kurtosis: a critical review," *The American Statistician*, vol. 42, no. 2, pp. 111–119, 1988.

[73] R. I. Harris, "Testing for unit roots using the augmented dickey-fuller test: Some issues relating to the size, power and the lag structure of the test," *Economics letters*, vol. 38, no. 4, pp. 381–386, 1992.

[74] G. Cirillo and R. Passerone, "Packet length spectral analysis for iot flow classification using ensemble learning," *IEEE Access*, vol. 8, pp. 138 616–138 641, 2020.

[75] J. Greis, A. Yushchenko, D. Vogel, M. Meier, and V. Steinhage, "Automated identification of vulnerable devices in networks using traffic data and deep learning," *arXiv preprint arXiv:2102.08199*, 2021.

[76] J. Kotak and Y. Elovici, "Iot device identification using deep learning," in *Conference on Complex, Intelligent, and Software Intensive Systems*. Springer, 2020, pp. 76–86.

[77] F. Yin, L. Yang, Y. Wang, and J. Dai, "Iot etei: End-to-end iot device identification method," in *2021 IEEE Conference on Dependable and Secure Computing (DSC)*. IEEE, 2021, pp. 1–8.

[78] CICFlowMeter Features. [Online]. Available: https://github.com/ahlashkari/CICFlowMeter/blob/master/ReadMe.txt

[79] D. Bekerman, B. Shapira, L. Rokach, and A. Bar, "Unknown malware detection using network traffic classification," in *2015 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2015, pp. 134–142.

[80] Joy feature extracture. [Online]. Available: https://github.com/cisco/joy

[81] B. Ghojogh, M. N. Samad, S. A. Mashhadi, T. Kapoor, W. Ali, F. Karray, and M. Crowley, "Feature selection and feature extraction in pattern analysis: A literature review," *arXiv preprint arXiv:1905.02845*, 2019.

[82] J. Bao, B. Hamdaoui, and W.-K. Wong, "Iot device type identification using hybrid deep learning approach for increased iot security," in *2020 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, 2020, pp. 565–570.

[83] M. Tschannen, O. Bachem, and M. Lucic, "Recent advances in autoencoder-based representation learning," *arXiv preprint arXiv:1812.05069*, 2018.

[84] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[85] A. Aksoy and M. H. Gunes, "Automated iot device identification using network traffic," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–7.

[86] A. Sivanathan, H. H. Gharakheili, and V. Sivaraman, "Inferring iot device types from network behavior using unsupervised clustering," in *2019 IEEE 44th Conference on Local Computer Networks (LCN)*. IEEE, 2019, pp. 230–233.

[87] O. Salman, I. H. Elhajj, A. Chehab, and A. Kayssi, "A machine learning based framework for iot device identification and abnormal traffic detection," *Transactions on Emerging Telecommunications Technologies*, p. e3743, 2019.

[88] Y. Sun, J. Liu, A. K. Bashir, U. Tariq, W. Liu, K. Chen, and M. D. Alshehri, "E-cis: Edge-based classifier identification scheme in green & sustainable iot smart city," *Sustainable Cities and Society*, vol. 75, p. 103312, 2021.

[89] Y. Song, Q. Huang, J. Yang, M. Fan, A. Hu, and Y. Jiang, "Iot device fingerprinting for relieving pressure in the access control," in *Proceedings of the ACM Turing Celebration Conference-China*, 2019, pp. 1–8.

[90] Y. Vorobeychik and M. Kantarcioglu, "Adversarial machine learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 12, no. 3, pp. 1–169, 2018.

[91] N. Burkart and M. F. Huber, "A survey on the explainability of supervised machine learning," *Journal of Artificial Intelligence Research*, vol. 70, pp. 245–317, 2021.

[92] N. Papernot, P. McDaniel, and I. Goodfellow, "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples," *arXiv preprint arXiv:1605.07277*, 2016.

[93] A. Hsu, J. Tront, D. Raymond, G. Wang, and A. Butt, "Automatic iot device classification using traffic behavioral characteristics," in *2019 SoutheastCon*. IEEE, 2019, pp. 1–7.

[94] A. Bassene and B. Gueye, "A group-based iot devices classification through network traffic analysis based on machine learning approach," in *Towards new e-Infrastructure and e-Services for Developing Countries: 12th EAI International Conference, AFRICOMM 2020, Ebène City, Mauritius, December 2-4, 2020, Proceedings 12*. Springer International Publishing, 2021, pp. 185–202.

[95] I. Cvitic, D. Perakovic, M. Perisa, and M. Botica, "Definition of the iot device classes based on network traffic flow features," in *4th EAI International Conference on Management of Manufacturing Systems*. Springer, 2020, pp. 1–17.

[96] A. Thantharate, R. Paropkari, V. Walunj, and C. Beard, "Deepslice: A deep learning approach towards an efficient and reliable network slicing in 5g networks," in *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*. IEEE, 2019, pp. 0762–0767.

[97] V. A. Ferman and M. A. Tawfeeq, "Machine learning challenges for iot device fingerprints identification," in *Journal of Physics: Conference Series*, vol. 1963, no. 1. IOP Publishing, 2021, p. 012046.

[98] B. Chakraborty, D. M. Divakaran, I. Nevat, G. W. Peters, and M. Gurusamy, "Cost-aware feature selection for iot device classification," *IEEE Internet of Things Journal*, 2021.

[99] B. A. Desai, D. M. Divakaran, I. Nevat, G. W. Peter, and M. Gurusamy, "A feature-ranking framework for iot device classification," in *2019 11th International Conference on Communication Systems & Networks (COMSNETS)*. IEEE, 2019, pp. 64–71.

[100] M. R. Shahid, G. Blanc, H. Jmila, Z. Zhang, and H. Debar, "Generative deep learning for internet of things network traffic generation," in *2020 IEEE 25th Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE, 2020, pp. 70–79.

[101] D. Baylor, K. Haas, K. Katsiapis, S. Leong, R. Liu, C. Menwald, H. Miao, N. Polyzotis, M. Trott, and M. Zinkevich, "Continuous training for production in the {TensorFlow} extended ({TFX}) platform," in *2019 USENIX Conference on Operational Machine Learning (OpML 19)*, 2019, pp. 51–53.

[102] O. Ibitoye, O. Shafiq, and A. Matrawy, "Analyzing adversarial attacks against deep learning for intrusion detection in iot networks," in *2019 IEEE global communications conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.

[103] I. Rosenberg, A. Shabtai, Y. Elovici, and L. Rokach, "Adversarial learning in the cyber security domain," *arXiv preprint arXiv:2007.02407*, 2020.

[104] S. J. Pan, "Transfer learning," *Learning*, vol. 21, pp. 1–2, 2020.

[105] S. K. Thompson, *Sampling*. John Wiley & Sons, 2012, vol. 755.

[106] E. Lear, R. Droms, and D. Romascanu, "Manufacturer usage description," *IETF Internet draft*, 2016.

[107] I. Ishaq, D. Carels, G. K. Teklemariam, J. Hoebeke, F. V. d. Abeele, E. D. Poorter, I. Moerman, and P. Demeester, "Ietf standardization in the field of the internet of things (iot): a survey," *Journal of Sensor and Actuator Networks*, vol. 2, no. 2, pp. 235–287, 2013.

[108] P. Sudhakaran and C. Malathy, "Authorisation, attack detection and avoidance framework for iot devices," *IET Networks*, vol. 9, no. 5, pp. 209–214, 2020.

[109] A. Hamza, H. H. Gharakheili, and V. Sivaraman, "Combining mud policies with sdn for iot intrusion detection," in *Proceedings of the 2018 Workshop on IoT Security and Privacy*, 2018, pp. 1–7.

[110] Mudgee. [Online]. Available: https://github.com/ayyoob/mudgee

[111] M. R. Santos, R. M. Andrade, D. G. Gomes, and A. C. Callado, "An efficient approach for device identification and traffic classification in iot ecosystems," in *2018 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2018, pp. 00 304–00 309.

[112] 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, pp. 106–117.

[113] A. Hamza, H. H. Gharakheili, T. A. Benson, and V. Sivaraman, "Detecting volumetric attacks on lot devices via sdn-based monitoring of mud activity," in *Proceedings of the 2019 ACM Symposium on SDN Research*, 2019, pp. 36–48.

[114] L. Deng, Y. Feng, D. Chen, and N. Rishe, "Iotspot: Identifying the iot devices using their anonymous network traffic data," in *MILCOM 2019-2019 IEEE Military Communications Conference (MILCOM)*. IEEE, 2019, pp. 1–6.

[115] X. Wang, Y. Wang, X. Feng, H. Zhu, L. Sun, and Y. Zou, "Iottracker: An enhanced engine for discovering internet-of-thing devices," in *2019 IEEE 20th International Symposium on "A World of Wireless, Mobile and Multimedia Networks"(WoWMoM)*. IEEE, 2019, pp. 1–9.

[116] K. Yang, Q. Li, and L. Sun, "Towards automatic fingerprinting of iot devices in the cyberspace," *Computer Networks*, vol. 148, pp. 318–327, 2019.

[117] Y. Sun, S. Fu, S. Zhang, H. Zhu, and Y. Li, "Accurate iot device identification from merely packet length," in *2020 16th International Conference on Mobility, Sensing and Networking (MSN)*. IEEE, 2020, pp. 774–778.

[118] A. Sivanathan, H. H. Gharakheili, and V. Sivaraman, "Detecting behavioral change of iot devices using clustering-based network traffic modeling," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7295–7309, 2020.

[119] V. A. Ferman and M. A. Tawfeeq, "Gradient boosting algorithm for early detection of unknown internet of things devices," *Journal of Engineering and Sustainable Development (JEASD)*, vol. 25, no. Special_ Issue_2021, 2021.

[120] R. Kumar, M. Swarnkar, G. Singal, and N. Kumar, "Iot network traffic classification using machine learning algorithms: An experimental analysis," *IEEE Internet of Things Journal*, vol. 9, no. 2, pp. 989–1008, 2021.

HOUDA JMILA received her engineering degree in Computer Science from Telecom Sudparis, Institut Polytechnique de Paris, France in 2011 and the Ph.D. in Telecommunications and Computer science in 2015 from the same university. She is currently a post-doctoral researcher at Telecom SudParis. Her research interests includes network security and automated management of resources in virtual networks as well as the machine learning applications to these domains.

GREGORY BLANC is an Associate Professor (Maître de Conférences) at the Networks and Telecommunication Services department of IMT/Télécom SudParis, Institut Polytechnique de Paris, and coordinator of ICT security curriculum. He received his Ph.D in 2012 from NAIST, Japan. During his Ph.D, he chaired the Web 2.0 Application Security WG in the WIDE project. He is co-chair of WG7 on Network Security at Cybersecurity France-Japan, Steering Committee member of the RESSI conference. His research activity covers network security, network virtualization and applications of machine learning to cybersecurity.

MUSTAFIZUR R. SHAHID is currently working as a post-doctoral researcher in Machine Learning and Cybersecurity at Télécom SudParis, Institut Polytechnique de Paris. He received his PhD in Computer Science and Artificial Intelligence in 2021, and his MSc in Computer Science in 2017, from the same institution. His research interests include data science, machine learning, artificial intelligence, computer and network security, and the Internet of Things (IoT).

MARWAN LAZRAG is a research and development engineer at IMT/Télécom SudParis, Institut Polytechnique de Paris. He received his engineering degree in Telecommunications from the Engineering School of Communications (Sup'Com), University of Carthage (Tunisia) in 2019. His activities focus on the Internet of Things and the development of a decision support and security automation platform.

TABLE 2. Papers review with respect to the to data-acquisition and classification granularity taxonomies.

Paper	Data			Devices		Traffic type		Collection point		Classification granularity		
	Collected		Use public	IoT	Non-IoT	Setup	Interaction & idle	at	after			
	Private	Public						the gw		Instance	Type	category
	45%	9%	57%	100%	38%	19%	86%	93%	7%	0%	81%	22%
[15]	✓	-	-	✓	-	-	✓	✓	-	-	-	✓
[14]	-	✓	-	✓	-	✓	-	✓	-	-	✓	-
[58]	✓	-	-	✓	✓	-	✓	✓	-	-	-	✓
[111]	-	-	[19]	✓	✓	-	✓	✓	-	-	-	✓
[56]	-	-	[19]	✓	✓	-	✓	✓	-	-	-	✓
[67]	✓	-	-	✓	✓	-	✓	✓	-	-	✓	-
[65]	✓	-	-	✓	-	-	✓	✓	-	-	-	✓
[47]	✓	-	-	✓	-	-	✓	✓	-	-	✓	✓
[51]	-	-	[14]	✓	-	✓	-	✓	-	-	✓	-
[87]	✓	-	-	✓	-	-	✓	✓	-	-	✓	-
[20]	✓	-	-	✓	-	-	✓	✓	-	-	✓	-
[49]	-	-	[19]	✓	✓	-	✓	✓	-	-	✓	-
[85]	-	-	[14]	✓	-	✓	-	✓	-	-	✓	-
[93]	✓	-	-	✓	-	-	✓	✓	-	-	✓	-
[27]	-	✓	-	✓	✓	-	✓	✓	-	-	✓	-
[62]	✓	-	-	✓	-	-	✓	✓	-	-	✓	-
[112]	✓	-	[19]	✓	-	-	✓	✓	-	-	✓	-
[64]	✓	-	[19]	✓	✓	-	✓	✓	-	-	✓	-
[46]	-	-	[19]	✓	✓	-	✓	✓	-	-	-	✓
[86]	-	-	[113]	✓	-	-	✓	✓	-	-	✓	-
[89]	-	-	[14]	✓	-	✓	-	✓	-	-	✓	-
[68]	-	-	[19]	-	-	-	✓	✓	-	-	✓	-
[52]	-	-	[14]	✓	-	✓	-	✓	-	-	✓	-
[53]	✓	-	-	✓	-	-	✓	✓	-	-	✓	-
[114]	✓	-	-	✓	-	✓	✓	✓	-	-	✓	-
[115]	✓	-	-	✓	-	-	✓	✓	-	-	✓	-
[55]	✓	-	[19]	✓	-	-	✓	✓	-	-	✓	-
[16]	✓	-	-	✓	-	-	✓	✓	-	-	✓	-
[116]	✓	-	-	✓	✓	-	✓	✓	-	-	✓	-
[25]	-	✓	-	✓	-	-	-	✓	✓	-	✓	-
[117]	-	-	[19]	✓	-	-	✓	✓	-	-	✓	-
[54]	-	-	[27]	✓	✓	-	✓	✓	-	-	✓	-
[50]	✓	-	[14]	✓	-	✓	-	✓	-	-	✓	-
[118]	-	-	[27]	✓	✓	-	✓	✓	-	-	✓	-
[76]	-	-	[27]	✓	✓	-	✓	✓	-	-	✓	-
[82]	-	-	[15]	✓	-	-	✓	✓	-	-	✓	-
[61]	-	-	[19], [29]	✓	✓	-	✓	✓	-	-	-	✓
[60]	-	-	[27], [29]	✓	✓	-	✓	✓	-	-	✓	-
[59]	-	-	[27]	✓	✓	-	✓	✓	-	-	✓	-
[74]	-	-	[27]	✓	✓	-	✓	✓	-	-	✓	-
[23]	✓	-	[27]	✓	✓	-	✓	-	✓	-	✓	-
[63]	✓	-	-	✓	-	-	✓	✓	-	-	✓	✓
[44]	✓	-	-	✓	-	-	✓	✓	-	-	✓	-
[17]	-	✓	-	✓	✓	-	✓	-	✓	-	✓	-
[66]	✓	-	-	✓	-	-	✓	✓	-	-	-	✓
[94]	-	-	[27]	✓	✓	-	✓	✓	-	-	-	✓
[70]	✓	-	-	✓	-	-	✓	✓	-	-	✓	-
[75]	-	-	[14]	✓	-	✓	-	✓	-	-	✓	-
[88]	-	-	[27]	✓	-	-	✓	✓	-	-	✓	-
[32]	✓	-	-	✓	-	-	✓	✓	-	-	-	✓
[119]	-	-	[14]	✓	-	✓	-	✓	-	-	-	✓
[24]	✓	-	-	✓	-	-	✓	-	✓	-	✓	-
[77]	-	-	[27], [29]	✓	-	-	✓	✓	-	-	✓	-
[48]	-	-	[14], [27], [29]	✓	✓	✓	✓	✓	-	-	✓	-
[97]	✓	-	[14]	✓	-	✓	-	✓	-	-	✓	-
[34]	-	✓	-	✓	-	-	✓	✓	-	-	✓	-
[43]	-	-	[27], [40]	✓	✓	-	✓	✓	-	-	✓	-
[120]	-	-	[19]	✓	✓	-	✓	✓	-	-	✓	-

TABLE 3. Papers review with respect to the feature extraction taxonomy.

Paper	Stream definition			Extracted features					Dimensionality reduction	Automatic feature extraction
				Packet-level	Flow level					
	N packets	Δt	Connection		Volume	Protocol	Temporal	Periodic		
	33%	24%	50%	38%	60%	40%	47%	3%	31%	12%
[15]	-	-	✓	-	✓	✓	✓	-	✓	-
[14]	✓	-	-	✓	-	-	-	-	-	-
[58]	-	-	✓	-	-	-	✓	-	-	-
[111]	-	-	✓	-	✓	-	✓	-	✓	-
[56]	-	✓	-	-	✓	✓	✓	-	-	-
[67]	-	-	✓	-	✓	-	✓	-	-	-
[65]	✓	-	-	-	-	-	✓	-	-	-
[47]	✓	-	-	✓	-	-	-	-	-	-
[51]	✓	-	-	-	-	✓	✓	-	-	-
[87]	✓	-	✓	-	✓	-	✓	-	-	-
[20]	-	✓	-	-	✓	-	-	✓	✓	-
[49]	-	-	✓	✓	✓	✓	✓	-	-	-
[85]	✓	-	-	✓	-	-	-	-	✓	-
[93]	-	-	✓	-	✓	-	✓	-	-	-
[27]	-	-	✓	✓	✓	✓	✓	-	-	-
[62]	-	✓	-	-	✓	✓	✓	-	✓	-
[112]	-	-	✓	✓	-	-	-	-	✓	✓
[64]	-	-	✓	✓	-	✓	-	-	✓	-
[46]	-	✓	-	-	✓	-	-	-	-	-
[86]	-	-	✓	-	✓	-	-	-	✓	-
[89]	✓	-	-	-	✓	✓	-	-	-	-
[68]	-	✓	-	-	✓	✓	✓	-	-	-
[52]	✓	-	-	-	✓	-	✓	-	✓	-
[53]	✓	-	-	-	✓	-	✓	-	✓	-
[114]	-	✓	-	-	✓	-	-	-	✓	-
[115]	-	-	✓	✓	-	✓	-	-	-	-
[55]	-	✓	-	✓	-	✓	-	-	-	-
[16]	-	✓	-	-	✓	-	✓	-	-	-
[116]	✓	-	-	✓	-	-	-	-	-	✓
[25]	-	-	✓	-	✓	✓	✓	-	-	-
[117]	-	✓	-	-	✓	-	-	-	✓	-
[54]	-	✓	-	-	✓	✓	✓	-	-	✓
[50]	✓	-	-	✓	✓	✓	✓	-	-	-
[118]	-	-	✓	-	✓	-	-	-	-	-
[76]	-	-	✓	✓	-	-	-	-	-	✓
[82]	-	-	✓	-	✓	-	✓	-	✓	-
[61]	-	-	✓	✓	✓	-	-	-	✓	-
[60]	-	-	✓	-	✓	✓	✓	-	-	-
[59]	-	-	✓	-	✓	-	-	-	✓	-
[74]	-	-	✓	-	✓	-	-	-	-	-
[23]	✓	-	-	✓	-	-	-	-	-	-
[63]	✓	-	✓	-	✓	✓	✓	-	-	-
[44]	✓	-	-	✓	-	-	-	-	✓	-
[17]	✓	✓	-	✓	-	✓	-	-	-	-
[66]	-	✓	-	-	-	-	✓	-	-	-
[94]	-	-	✓	-	✓	-	✓	-	✓	-
[70]	-	✓	-	-	✓	✓	-	✓	-	-
[75]	✓	-	-	✓	-	-	-	-	-	✓
[88]	-	-	✓	-	✓	✓	✓	-	-	-
[32]	-	-	✓	-	✓	-	✓	-	✓	-
[119]	✓	-	-	-	✓	✓	-	-	-	-
[24]	✓	-	-	✓	-	-	-	-	-	-
[77]	-	-	✓	✓	-	-	-	-	-	✓
[48]	✓	-	-	✓	-	-	-	-	-	-
[97]	-	-	✓	-	✓	✓	✓	-	-	-
[34]	-	✓	-	-	✓	✓	✓	-	-	-
[43]	-	-	✓	✓	-	-	-	-	-	✓
[120]	-	-	✓	✓	✓	✓	✓	-	-	-

TABLE 4. Papers review with respect to the machine learning taxonomy.

Paper	One-class classifier	Multi-class classifier	Supervised	Unsupervised	Deep	Shallow	Evaluation metrics
%	14 %	90%	84%	16%	29%	79%	76% classic
[14]	✓	-	✓	-	-	✓	classic
[15]	-	✓	✓	-	-	✓	classic
[14]	✓	-	✓	-	-	✓	classic
[58]	-	✓	✓	-	-	✓	classic
[111]	-	✓	✓	-	-	✓	classic
[56]	-	✓	✓	-	✓	✓	classic
[67]	-	✓	✓	-	-	✓	classic
[65]	-	✓	✓	-	✓	-	classic
[47]	-	✓	✓	-	-	✓	classic
[51]	✓	-	✓	-	-	✓	classic
[87]	-	✓	✓	-	✓	✓	classic
[20]	-	✓	-	✓	-	-	classic, speed
[49]	-	✓	✓	-	✓	✓	classic, speed
[85]	-	✓	✓	-	-	✓	classic
[93]	-	✓	✓	-	-	✓	classic
[27]	-	✓	✓	-	-	✓	classic
[62]	-	✓	✓	-	-	✓	classic
[112]	-	✓	-	✓	-	✓	classic
[64]	-	✓	✓	-	-	✓	classic
[46]	-	✓	-	✓	-	✓	classic, speed
[86]	-	✓	-	✓	-	✓	classic
[89]	-	✓	✓	-	-	✓	classic
[68]	-	✓	✓	-	-	✓	classic
[52]	✓	✓	✓	-	✓	-	classic, speed
[53]	✓	-	✓	-	-	✓	classic
[114]	-	✓	✓	-	-	✓	classic
[115]	-	✓	✓	-	-	✓	classic
[55]	-	✓	✓	-	-	✓	classic
[16]	-	✓	✓	-	-	✓	classic
[116]	-	✓	✓	-	✓	-	classic
[25]	-	✓	✓	-	✓	✓	classic, speed
[117]	-	✓	✓	-	-	✓	classic
[54]	-	✓	-	✓	✓	-	classic
[50]	✓	-	✓	-	-	✓	classic, speed
[118]	-	✓	-	✓	-	✓	classic
[76]	-	✓	✓	-	✓	-	classic
[82]	-	✓	-	✓	✓	✓	classic, speed
[61]	-	✓	✓	-	-	✓	classic, speed
[60]	-	✓	✓	-	-	✓	classic
[59]	-	✓	✓	-	-	✓	classic, speed
[74]	-	✓	✓	-	-	✓	classic, speed
[23]	-	✓	✓	-	-	✓	classic, speed
[63]	✓	✓	✓	-	-	✓	classic, speed
[44]	-	✓	✓	-	✓	-	classic
[17]	-	✓	✓	-	✓	-	classic
[66]	-	✓	✓	-	-	✓	classic
[94]	-	✓	✓	-	-	✓	classic
[70]	-	✓	✓	-	-	✓	classic
[75]	-	✓	✓	-	✓	-	classic
[88]	-	✓	-	✓	-	✓	classic, complexity
[32]	-	✓	✓	-	-	✓	classic
[119]	✓	-	✓	-	-	✓	classic
[24]	-	✓	✓	-	✓	-	classic
[77]	-	✓	✓	-	✓	-	classic, speed
[48]	-	✓	✓	-	-	✓	classic
[97]	-	✓	✓	-	-	✓	classic
[34]	-	✓	✓	-	✓	✓	classic
[43]	-	✓	-	✓	✓	-	classic