




Article

IoT Device Identification Using Unsupervised Machine Learning

Carson Koball , Bhaskar P. Rimal , Yong Wang , Tyler Salmen and Connor Ford

The Beacom College of Computer and Cyber Sciences, Dakota State University, Madison, SD 57042, USA

* Correspondence: yong.wang@dsu.edu

Abstract: Device identification is a fundamental issue in the Internet of Things (IoT). Many critical services, including access control and intrusion prevention, are built on correctly identifying each unique device in a network. However, device identification faces many challenges in the IoT. For example, a common technique to identify a device in a network is using the device's MAC address. However, MAC addresses can be easily spoofed. On the other hand, IoT devices also include dynamic characteristics such as traffic patterns which could be used for device identification. Machine-learning-assisted approaches are promising for device identification since they can capture dynamic device behaviors and have automation capabilities. Supervised machine-learning-assisted techniques demonstrate high accuracies for device identification. However, they require a large number of labeled datasets, which can be a challenge. On the other hand, unsupervised machine learning can also reach good accuracies without requiring labeled datasets. This paper presents an unsupervised machine-learning approach for IoT device identification.

Keywords: internet of things; device identification; machine learning; unsupervised machine learning



Citation: Koball, C.; Rimal, B.P.; Wang, Y.; Salmen, T.; Ford, C. IoT Device Identification Using Unsupervised Machine Learning. *Information* **2023**, *14*, 320. <https://doi.org/10.3390/info14060320>

Academic Editors: Spyros Panagiotakis and Evangelos K. Markakis

Received: 4 March 2023

Revised: 23 May 2023

Accepted: 25 May 2023

Published: 31 May 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The Internet of Things (IoT) is a term used to describe the interconnection of computing devices embedded in everyday objects to the Internet through the home, business, or institutional networks [1]. IoT devices and applications present significant security challenges, including limited device capabilities, lack of standardization, insufficient trust and integrity, and software vulnerabilities [2]. As a result, device identification is challenging in the IoT. Many critical services, such as access control and intrusion detection, are built on correctly identifying each unique device [3].

As users are identified in a digital network by their unique identities, IoT devices also require their unique identities when connecting to a network. Identities of Things (IDoT), a general term, has been adopted to describe IoT entities (e.g., users and devices). Four primary authentication factors could be used to identify users: something you know (e.g., username and password), something you possess (e.g., a physical token or a smart card), something you are (e.g., fingerprint or face recognition), and something you do (e.g., voice or sign). IoT devices can only be identified by something they have. A common technique to identify a device in a network is using the device's MAC address. However, MAC addresses can be easily spoofed.

Identity in IoT devices consists of attributes and dynamic values along with the member in varying contexts [4]. It can be a collection of things, should have a purpose, and should be treated uniformly across platforms. There are many representations of identities, and they can rely on globally unique identifiers [5,6], a combination of user characteristics [7], a set of attributes of the users [8], or a set of claims [9]. These approaches all possess a commonality based on the fact that they link an identity unique to a particular entity [4].

Machine-learning (ML)-assisted approaches attract many research interests because they can capture dynamic characteristics from the devices for device identification [10–17]. ML-assisted approaches fall into two general categories, i.e., supervised ML-assisted approaches [10,11] and unsupervised ML-assisted approaches [12–15,17]. Supervised ML-assisted approaches demonstrate great accuracy when used for device identification. However, they often require a large number of labeled datasets, which could be a challenge. On the other hand, unsupervised ML can reach reasonable accuracies without labeled datasets [12–15,17]. Additionally, the unsupervised method allows for greater flexibility regarding dynamic IoT networks where devices may join or leave at any time. Therefore, this paper focuses on unsupervised ML approaches for device identification.

The unsupervised approach presented in the paper utilizes an ensemble-based approach for device identification. A K-Nearest Neighbors classifier is used to identify each IoT device. The dataset proceeds through four steps: preprocessing, outlier removal, feature selection, and clustering before a device prediction is made. The key contributions of this paper include, but are not limited to, (1) demonstrating how to use unsupervised ML for device identification; (2) evaluating the performance of supervised ML and unsupervised ML using the same dataset; (3) discussing possible benefits that unsupervised ML may bring to the field of device identification.

The remainder of this paper is structured as follows. Section 2 introduces related work. Section 3 presents the proposed unsupervised ML-assisted approach for IoT device identification. The evaluation results are presented in Section 4. Finally, the conclusion and discussion for future research are discussed in Section 5.

2. Related Work

The surveys in [18,19] show that supervised, unsupervised, semi-supervised, and deep-learning approaches could all be used for device identification. Supervised ML is used in [10,11,20–22]. The authors in [20] proposed a supervised ML-assisted approach for identifying known IoT devices. A proprietary tool developed in [23] was used in [20] to extract features from captured network traffic. Using the same feature extraction tool, a two-stage meta classifier for IoT device identification was studied in [10]. The first stage classifier differentiates IoT and non-IoT devices. The second stage classifier identifies a specific IoT device class. The classifier considered in [20] is Random Forest-based. Other classifiers considered in the supervised ML include Decision Trees, Logistic Regression models, Support Vector Machines (SVM), GBM, and XGBoost models [10,11]. These papers show the utmost accuracy in identifying devices in the IoT. However, supervised ML requires labeling to train the models, which may be expensive or impossible to acquire.

In [12], authors used unsupervised clustering to identify IoT device types in network flow traffic. Network traffic was broken up into time granularities of 1–8-min packet flows for each device on the network. Depending on the device, the final clustering used K-Means with 128 or 256 clusters. This research takes a heuristic approach to identify flows in a packet capture by taking the data in 1–8-min intervals of packets. In [13], authors used data captured directly from the devices to identify cycles in the flow data relating to how often and how predictable the transmission of data is. They then use K-Nearest Neighbors and arbitrary labeling to cluster devices. This approach is much slower than other algorithms.

Unsupervised deep learning is used in [14,15], where ML autoencoders are combined with clustering algorithms to identify arbitrary device types. Work in [14] focused on identifying compromised devices using packet statistics, whereas [15] considered how variational autoencoders arbitrarily identify devices on a network using a combination of periodic features such as those in [13] and flow statistics [15].

As discussed in [12–15,17], unsupervised learning can reach accuracies as good as or better than those in supervised approaches, while having much higher accuracy in both unseen and compromised devices.

3. Unsupervised Machine-Learning-Assisted Approach for IoT Device Identification

The unsupervised process detailed in this paper utilizes an ensemble-based approach to device identification, where each base model that comprises the ensemble network is a one-class classifier. Figure 1 details the various steps that comprise the one-class classifiers implemented in this paper. These steps will be discussed more thoroughly later in this section.

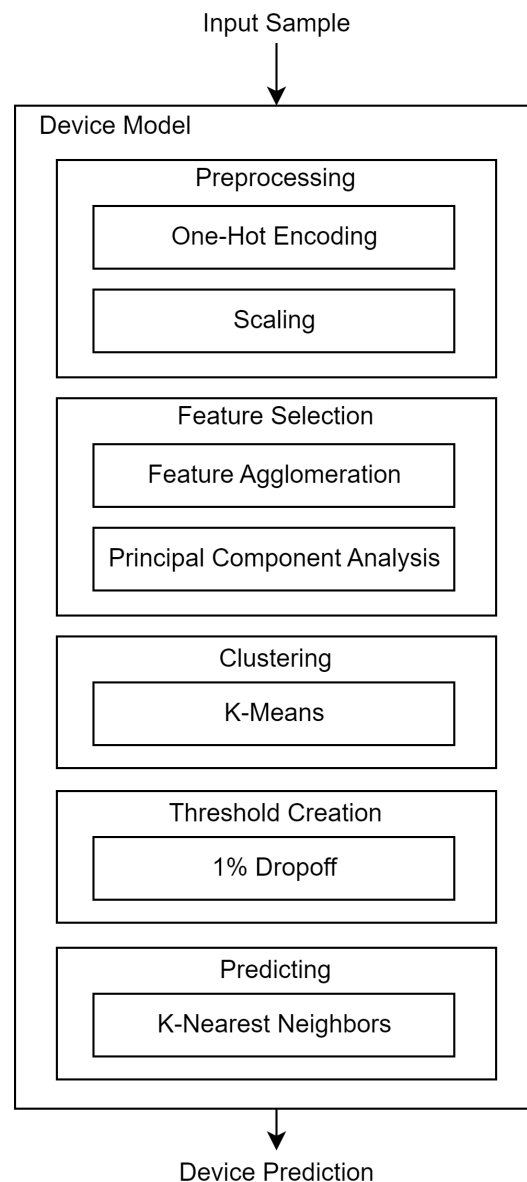


Figure 1. An overview of a one-class classifier.

The combined results from each base model will ultimately decide what a given sample will be classified as, as seen in Figure 2. The goal of a one-class classifier is to distinguish which samples do and do not belong to the class it represents. In the discussion of IoT device identification, the class would be an IoT device. This approach differs from a one-vs-rest scheme for two reasons. The first reason is that introducing a new device or removing an existing device from the network will require the entire ensemble network to be retrained in the one-vs-rest scheme. In the one-class scheme, however, introducing a new device will only require a new model for that device to be trained, whereas removing an existing device will only require that the existing model for the said device is discarded. The second reason this approach differs from a one-vs-rest scheme is that this approach

allows a sample to be considered to be an unknown sample, such as the case where the sample belongs to a device not in the network. In many one-vs-rest schemes, this unknown sample would have been predicted as one of the devices in the trained model network. As a result, the approach used in this paper allows for better scalability as the number of devices increases in a given network.

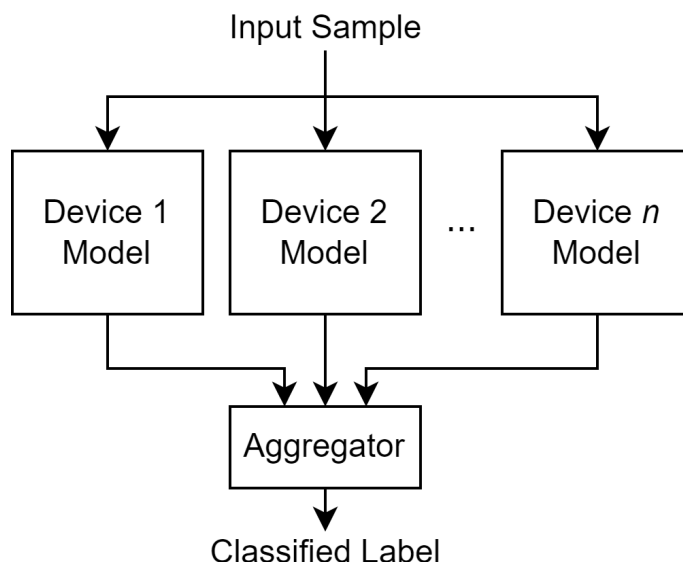


Figure 2. An overview of a one-class ensemble network.

3.1. Pre-Processing

The first step of the training stage is to process the data to augment the model's ability to capture information from the dataset. One such method of processing is through one-hot encoding. To include nominal features, such as the network protocol used in a flow, said features must be encoded to remove any inferred order between values. Additionally, since the proposed model utilizes distance-based algorithms in both the training and predicting stages of its lifespan, it is imperative to properly scale the raw data the model takes as input. Without scaling, features with inherently larger ranges will dominate features with inherently smaller ranges in Euclidean-based distance comparisons. Furthermore, the information found in these features with smaller ranges can be overlooked without scaling. In the implementation of the model in this paper, standardization is performed on each feature, where each feature is independently scaled to fit a normal distribution.

3.2. Feature Selection

The second step in the training stage is the selection of relevant features that will be used for analysis. The first portion of this step is using Feature Agglomeration (FA) [24]. A total of 100 clusters from the FA algorithm were used in the model. The second portion of this step is the use of Principal Component Analysis (PCA) [25]. All 100 features extracted from the PCA were also used in the model.

3.3. Clustering

The third step in the training stage is clustering the processed data. Clustering allows for more effective capture of each device's traffic patterns and reduces computation time in predicting. The model in this paper utilizes K-Means clustering [26]. A low K value would result in the K-Means algorithm overgeneralizing the data, creating centroids that are insignificant to the traffic patterns of each device. Alternatively, a K value too high would result in the K-Means algorithm overfitting on the data, creating centroids that are fitted extremely well to sampled noise in the training data, leaving little room for generalization in unseen data. As with [12], the K value for each model was determined by testing an incrementally larger K to a maximum of 1000. The sum of averaged squared distances of

each sample to its closest centroid, or inertia, was recorded for each value of K tested. An elbow point was then identified where an increase in K would not substantially reduce the inertia of the data points, and this K value was chosen for the model. Additionally, each centroid will be assigned a corresponding training distribution value based on the total training samples assigned to the centroid. This will be used as a tie-breaking metric in the predicting stage of the model's lifespan, detailed in Section 3.5.

3.4. Threshold Creation

The final step in the training stage is to create the threshold that decides classification behavior. This threshold defines whether an input sample is or is not predicted as the device of a given model. As with [12], each centroid created with the K-Means algorithm has an assigned set of samples that belong to it. The distance between each point and its centroid is then calculated, and the threshold for the centroid is defined as the distance that includes no more than 99% of the samples that belong to it. As a result, each centroid will have a corresponding threshold distance value that will exclude 1% of the samples that belong to it from the training dataset. Another method of identifying a distance threshold was explored utilizing DBSCAN [27].

3.5. Predicting

From the training stage of the model, a series of centroids, as well as their respective training distribution values and distance thresholds, are stored. Additionally, the preprocessing models, FA, and PCA models are stored after being fitted from the training dataset. The model is given the sample as input to predict a new test sample, where the preprocessing models transform its feature values. Next, the sample's features are selected based on the FA and PCA models, which ultimately reduce the number of features that will be observed. To make the prediction, the model implements a K-Nearest Neighbors classifier fitted with the centroids defined by the K-Means algorithm in the training stage. The centroid closest to the transformed test sample is then considered the centroid to which the testing sample belongs. However, if the distance between the centroid and the testing sample is greater than the distance threshold set for the centroid, the testing sample will be considered a negative sample. Conversely, if the distance between the centroid and the testing sample is equal to or smaller than the distance threshold set for the centroid, the testing sample would be considered a positive sample, as seen in Figure 3.

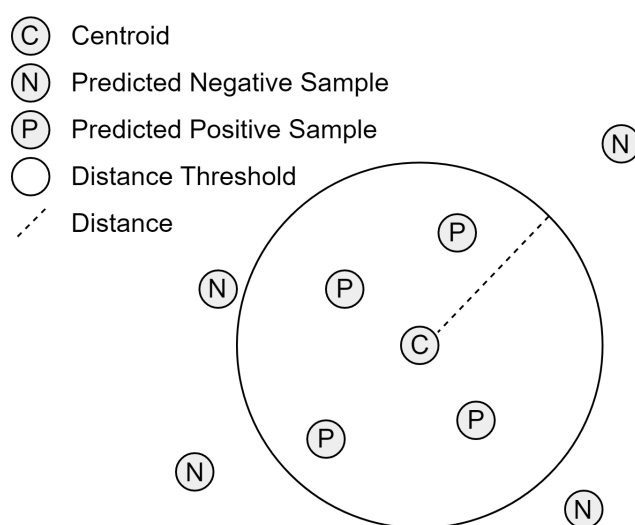


Figure 3. Example of a distance threshold and its effect on predicted samples in a 2-dimensional space.

This model could effectively act as a classifier for the device it represents in a single-device environment. However, in environments with more than one device, any given

sample can be predicted positively for multiple device models. Consequently, the network must distinguish which positive sample will be assigned to which device. This final tie-breaking comparison utilizes the training distribution value assigned to each cluster in the clustering step. The training distribution value for each centroid of these device models is compared to all the devices considered sample positive. The device model whose centroid has the highest training distribution value will ultimately claim the sample as its own.

3.6. Testing Dataset

As shown in Figure 4, a testing network was established to generate a testing dataset. A total of 16 IoT devices are connected to the testing network. The network traffic is collected through nTAP and RaspAP [28]. nTAP is a passive, full-duplex monitoring device that provides visibility into the network regardless of traffic. nTAP collects network traffic before it reaches the firewall. RaspAP provides Internet access for IoT devices and is used to collect network traffic at the Wi-Fi access point. *tcpdump* is used to collect network traffic in nTAP and RaspAP.

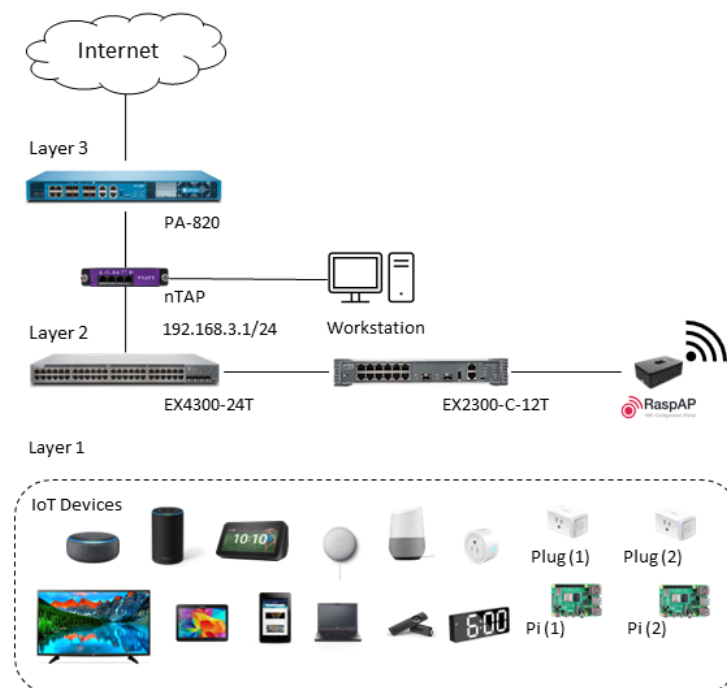


Figure 4. Experimental network testbed.

Data were collected between 31 March 2022 and 9 May 2022. Multiple datasets were collected during different periods during the experiment. *tcpdump* was used on both the RaspAP and the nTAP data to collect network traffic. Data collected through RaspAP is used for ML training. Furthermore, data collected through nTAP is used to validate the ML classifier on the network perimeter. Approximately 150 GB and 200 GB of data were collected from RaspAP and nTAP, respectively.

3.7. Feature Extraction

Network traffic from IoT devices is collected through tools such as *tcpdump* and Wireshark. The network traffic is saved in pcap files. An open-source tool, CICFlowMeter, extracts network features from pcap files [29,30]. CICFlowMeter can generate bidirectional flows and calculate time-related features in both the forward and backward directions. Originally, CICFlowMeter was created to identify malicious traffic that might contain malware. The features that could be extracted from each traffic flow include flow duration, total forward packets, and total backward packets. In addition to the flow features, flow ID, source IP, source port number, destination IP, destination port number, protocol, and

timestamps are also recorded for each flow. After feature extraction, CICFlowMeter creates a CSV file with all the features for each pcap file. In this study, we have developed a tool, *CICFlowMeter++* by enhancing the original CICFlowMeter. CICFlowMeter++ can extract 233 features from a TCP flow and includes many new features from [10] for comparison. In addition to these 233 features, 9 more features, including source device, destination device, source medium, destination medium, source state, destination state, source manufacturer, destination manufacturer, and flow device, are also added to the CSV file. A total of 242 features are available in the CSV file.

Our preliminary testing and results indicated that ML was ineffective in identifying a device if the device did not generate sufficient data for training. Furthermore, similar devices, e.g., Amazon Echo, Amazon Echo Dot, and Amazon Echo Show, or the same type of devices, e.g., K Smart Plug 1 and 2, also present challenges for device identification. After removing three devices that did not meet the criteria and combining similar devices, eight devices remained in the dataset. Table 1 shows the number of TCP flows for each device in the dataset.

Table 1. Datasets generated by the devices in the experiments (* combined TCP flows from similar devices).

Device	Dataset
Amazon Echo Show	47,804
Lenovo Chromebook	9307
Google Nexus Tablet	9388
K Smart Plug *	79,160
Raspberry Pi *	19,481
ZMI Smart Clock	7185
Amazon Smart Plug	5227
Samsung Smart TV	94,976

4. Results and Discussions

Using the approach presented in Section 3, the performance of the unsupervised ML approach for device identification is studied. We also compare the supervised ML approaches for device identification using the same dataset. For the unsupervised approach, a maximum of 10,000 samples for each of the eight devices are selected for training from the training subset. Exactly 1000 samples for each of the eight devices are selected for testing from the testing subset.

4.1. Feature Selection

A total of 242 features are available after data processing. Using FA and PCA, the number of features used by the model decreased from 242 to 100. Since the PCA model has transformed these 100 features, these features can be seen as combinations of the original features. Thus, the 100 features used for unsupervised ML are not directly mapped to any original features.

4.2. Clustering

Based on the elbow point method for identifying K values for K-Means clustering, each device had varying values of K used. Table 2 shows the number of clusters used for each device in the clustering step.

Table 2. K values identified for each device's model for K-Means clustering.

Device	Number of Clusters
Amazon Echo Show	400
Lenovo Chromebook	100
Google Nexus Tablet	300
K Smart Plug	60
Raspberry Pi	300
ZMI Smart Clock	70
Amazon Smart Plug	70
Samsung Smart TV	100

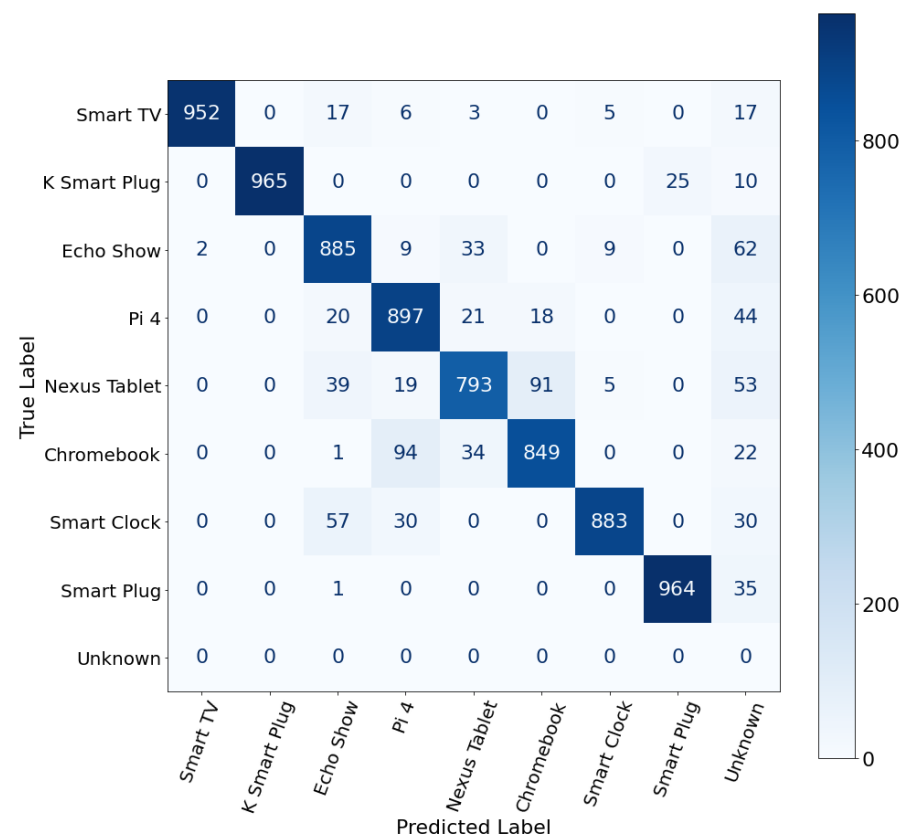
4.3. Device Identification

Two approaches, DBSCAN, and 1% Dropoff, are used for threshold creation. Table 3 shows the accuracies from both approaches. As shown in Table 3, the testing accuracy from using 1% Dropoff for threshold creation is slightly higher than when using the DBSCAN method.

Table 3. Distance threshold methods and their results.

Metric	DBSCAN	1% Dropoff
Macro Precision	0.821	0.828
Macro Recall	0.777	0.799
Macro F1 Score	0.797	0.813

Figure 5 shows the confusion matrix for device identification. As shown in Figure 5, unsupervised ML shows excellent accuracy values when identifying the Smart TV, K Smart Plug, and Amazon Smart Plug devices. The accuracy for identifying the Nexus Tablet device is not ideal.

**Figure 5.** Confusion matrix utilizing 1% dropoff.

4.4. Unsupervised ML vs. Supervised ML

We evaluate the supervised ML approach for device identification using the same dataset, as shown in Table 1. The testing dataset is divided into two datasets, i.e., the training dataset and the testing dataset, using an 80/20 split. The Random Forest classifier is used to evaluate the importance of the features for device identification. We further evaluate six supervised ML classifiers for device identification, including AdaBoost, Decision Tree, K-Nearest Neighbor, Logistic Regression, Random Forest, and LinearSVC. Our evaluation shows that the AdaBoost with 200 features achieves the best testing accuracies for device identification. Table 4 shows the accuracy values for the eight IoT devices from the AdaBoost with 200 features. Furthermore, Table 5 shows the precision, recall, f1 score, and accuracy values from the proposed unsupervised approach.

Table 4. Supervised ML vs. Unsupervised ML.

Device	Supervised ML	Unsupervised ML
Amazon Echo Show	92.4%	88.5%
Lenovo Chromebook	87.5%	84.9%
Google Nexus Tablet	100.0%	79.3%
K Smart Plug	91.0%	96.5%
Raspberry Pi	97.5%	89.7%
ZMI Smart Clock	98.8%	88.3%
Amazon Smart Plug	90.4%	96.4%
Samsung Smart TV	99.1%	95.2%

Table 5. Unsupervised ML.

Device	Precision	Recall	F1 Score	Accuracy
Amazon Echo Show	0.87	0.89	0.88	88.5%
Lenovo Chromebook	0.89	0.85	0.87	84.9%
Google Nexus Tablet	0.90	0.79	0.84	79.3%
K Smart Plug	1.00	0.96	0.98	96.5%
Raspberry Pi	0.85	0.90	0.87	89.7%
ZMI Smart Clock	0.98	0.88	0.93	88.3%
Amazon Smart Plug	0.97	0.96	0.97	96.4%
Samsung Smart TV	1.00	0.95	0.97	95.2%

As shown in Table 4, supervised ML generally provides better accuracies in device identification than unsupervised ML. For certain devices, e.g., K Smart Plug and Amazon Smart Plug, the unsupervised ML performs better than the supervised ML approach.

Our results show that both supervised and unsupervised ML could be used for device identification. Although supervised ML methods provide better accuracy values for device identification in this environment, the supervised paradigm requires labeled datasets that may not be attainable. In scenarios where labeling is not possible, our results further indicate that an unsupervised approach to device identification may still be a viable option. Additionally, the one-class nature that is inherent to our method allows for separate device classifiers to be added and removed from an ensemble model without the need for retraining the entire model. This modular property suggests that an unsupervised approach may be preferred in dynamic networks where devices frequently join and leave a network. Consequently, a hybrid approach including both supervised ML and unsupervised ML can be considered.

5. Conclusions and Outlook

This paper studies unsupervised ML for device identification. Our unsupervised ML approach employs a series of one-class classifiers that each includes five steps, i.e., pre-processing, feature selection, clustering, threshold creation, and predicting. The presented approach was applied to a dataset that includes eight devices. The obtained results show

reasonable accuracies for these eight devices during our testing. Our analysis indicates that unsupervised ML may have similar challenges as supervised ML in identifying similar devices or devices of the same type. However, an unsupervised approach may provide additional benefits, such as scalability in dynamic networks and the removal of labeling processes not found in supervised methods, to the challenge of IoT device classification. Potential avenues for future work include utilizing a hybrid approach, including both supervised ML and unsupervised ML approaches for device identification, and studying how ML-assisted approaches perform in untrusted environments and in real-time traffic environments.

Author Contributions: Conceptualization, B.P.R. and Y.W.; methodology, C.K.; software, C.K.; validation, C.K.; formal analysis, C.K.; investigation, C.K., T.S., and C.F.; resources, C.K.; data curation, C.K.; writing—original draft preparation, C.K., T.S., and C.F.; writing—review and editing, B.P.R. and Y.W.; visualization, C.K.; supervision, B.P.R. and Y.W.; project administration, B.P.R. and Y.W.; funding acquisition, B.P.R. and Y.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research is partially supported by the National Centers of Academic Excellence in Cybersecurity (NCAE-C).

Data Availability Statement: The data presented in this study are not publicly available due to proprietary tools used in the research.

Conflicts of Interest: The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Al-Fuqaha, A.; Guizani, M.; Mohammadi, M.; Aledhari, M.; Ayyash, M. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 2347–2376. [\[CrossRef\]](#)
2. Bhattarai, S.; Wang, Y. End-to-End Trust and Security for Internet of Things Applications. *Computer* **2018**, *51*, 20–27. [\[CrossRef\]](#)
3. Muthusamy Ragothaman, K.N.; Wang, Y. A Systematic Mapping Study of Access Control in the Internet of Things. In Proceedings of the 54th Hawaii International Conference on System Sciences, Kauai, HI, USA, 5–8 January 2021; pp. 7090–7099. Available online: <http://hdl.handle.net/10125/71474> (accessed on 25 February 2023).
4. Pal, S.; Hitchens, M.; Varadharajan, V. Modeling Identity for the Internet of Things: Survey, Classification and Trends. In Proceedings of the 12th International Conference on Sensing Technology (ICST), Limerick, Ireland, 4–6 December 2018; pp. 45–51. [\[CrossRef\]](#)
5. Koo, J.; Kim, Y.G. Interoperability of device identification in heterogeneous IoT platforms. In Proceedings of the 2017 13th International Computer Engineering Conference (ICENCO), Cairo, Egypt, 27–28 December 2017; pp. 26–29. [\[CrossRef\]](#)
6. Ning, H.; Zhen, Z.; Shi, F.; Daneshmand, M. A Survey of Identity Modeling and Identity Addressing in Internet of Things. *IEEE Internet Things J.* **2020**, *7*, 4697–4710. [\[CrossRef\]](#)
7. Jøsang, A.; Fabre, J.; Hay, B.; Dalziel, J.; Pope, S. Trust requirements in identity management. In *Proceedings of the Australasian Workshop on Grid Computing and e-Research-Volume 44*; Australian Computer Society, Inc.: Darlinghurst, NSW, Australia, 2005; pp. 99–108.
8. Alpár, G.; Batina, L.; Batten, L.; Moonsamy, V.; Krasnova, A.; Guellier, A.; Natgunanathan, I. New directions in IoT privacy using attribute-based authentication. In Proceedings of the ACM International Conference on Computing Frontiers, Como, Italy, 16–19 May 2016; pp. 461–466.
9. Cameron, K. The laws of identity. *Microsoft Corp.* **2005**, *12*, 8–11.
10. Meidan, Y.; Bohadana, M.; Shabtai, A.; Guarnizo, J.D.; Ochoa, M.; Tippenhauer, N.O.; Elovici, Y. ProfilloT: A Machine Learning Approach for IoT Device Identification Based on Network Traffic Analysis. In Proceedings of the Symposium on Applied Computing, Marrakech, Morocco, 4–6 April 2017; pp. 506–509.
11. Wang, Y.; Rimal, B.P.; Elder, M.; Maldonado, S.I.C.; Chen, H.; Koball, C.; Ragothaman, K. IoT Device Identification Using Supervised Machine Learning. In Proceedings of the 2022 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 7–9 January 2022; pp. 1–6. [\[CrossRef\]](#)
12. Sivanathan, A.; Gharakheili, H.H.; Sivaraman, V. Inferring IoT Device Types from Network Behavior Using Unsupervised Clustering. In Proceedings of the 2019 IEEE 44th Conference on Local Computer Networks (LCN), Osnabrueck, Germany, 14–17 October 2019; pp. 230–233.
13. Marchal, S.; Miettinen, M.; Nguyen, T.D.; Sadeghi, A.-R.; Asokan, N. AuDI: Toward Autonomous IoT Device-Type Identification Using Periodic Communication. *IEEE J. Sel. Areas Commun.* **2019**, *37*, 1402–1412. [\[CrossRef\]](#)

14. Bhatia, R.; Benno, S.; Esteban, J.; Lakshman, T.V.; Grogan, J. Unsupervised Machine Learning for Network-Centric Anomaly Detection in IoT. In Proceedings of the 3rd ACM CoNEXT Workshop on Big Data, Machine Learning and Artificial Intelligence for Data Communication Networks, Orlando, FL, USA, 9 December 2019; pp. 42–48.
15. Zhang, S.; Wang, Z.; Yang, J.; Bai, D.; Li, F.; Li, Z.; Wu, J.; Liu, X. Unsupervised IoT Fingerprinting Method via Variational Auto-encoder and K-means. In Proceedings of the ICC 2021—IEEE International Conference on Communications, Montreal, QC, Canada, 14–23 June 2021; pp. 1–6. [\[CrossRef\]](#)
16. Sikeridis, D.; Rimal, B.P.; Papapanagiotou, I.; Devetsikiotis, M. Unsupervised Crowd-Assisted Learning Enabling Location-Aware Facilities. *IEEE Internet Things J.* **2018**, *5*, 4699–4713. [\[CrossRef\]](#)
17. Liu, X.; Abdelhakim, M.; Krishnamurthy, P.; Tipper, D. Identifying Malicious Nodes in Multihop IoT Networks Using Diversity and Unsupervised Learning. In Proceedings of the 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, USA, 20–24 May 2018; pp. 1–6. [\[CrossRef\]](#)
18. Liu, Y.; Wang, J.; Li, J.; Niu, S.; Song, H. Machine Learning for the Detection and Identification of Internet of Things Devices: A Survey. *IEEE Internet Things J.* **2022**, *9*, 298–320. [\[CrossRef\]](#)
19. Safi, M.; Dadkhah, S.; Shoeleh, F.; Mahdikhani, H.; Molyneaux, H.; Ghorbani, A.A. A Survey on IoT Profiling, Fingerprinting, and Identification. *ACM Trans. Internet Things* **2022**, *3*, 1–39. [\[CrossRef\]](#)
20. Meidan, Y.; Bohadana, M.; Shabtai, A.; Ochoa, M.; Tippenhauer, N.O.; Guarnizo, J.D.; Elovici, Y. Detection of unauthorized IoT devices using machine learning techniques. *arXiv* **2017**, arXiv:1709.04647.
21. Aksoy, A.; Gunes, M.H. Automated IoT Device Identification using Network Traffic. In Proceedings of the IEEE ICC, Shanghai, China, 20–24 May 2019; pp. 1–7.
22. Hamad, S.A.; Zhang, W.E.; Sheng, Q.Z.; Nepal, S. IoT Device Identification via Network-Flow Based Fingerprinting and Learning. In Proceedings of the 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), Rotorua, New Zealand, 5–8 August 2019; pp. 103–111.
23. Bekerman, D.; Shapira, B.; Rokach, L.; Bar, A. Unknown malware detection using network traffic classification. In Proceedings of the 2015 IEEE Conference on Communications and Network Security (CNS), Florence, Italy, 28–30 September 2015; pp. 134–142. [\[CrossRef\]](#)
24. Kassambara, A. Practical Guide to Cluster Analysis in R: Unsupervised Machine Learning. 2017; Volume 1. Available online: <http://www.sthda.com/english/> (accessed on 25 February 2023).
25. Jolliffe, I.T. *Principal Component Analysis for Special Types of Data*; Springer: Berlin/Heidelberg, Germany, 2002.
26. Lloyd, S. Least squares quantization in PCM. *IEEE Trans. Inf. Theory* **1982**, *28*, 129–137. [\[CrossRef\]](#)
27. Schubert, E.; Sander, J.; Ester, M.; Kriegel, H.P.; Xu, X. DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN. *ACM Trans. Database Syst.* **2017**, *42*, 1–21. [\[CrossRef\]](#)
28. RaspAP. RaspAP: Simple Wireless AP Setup & Management for Debian-Based Devices. Available online: <https://github.com/RaspAP> (accessed on 25 February 2023).
29. Lashkari, A.H.; Draper-Gil, G.; Mamun, M.S.I.; Ghorbani, A.A. Characterization of tor traffic using time based features. In Proceedings of the ICISSP, Porto, Portugal, 19–21 February 2017; pp. 253–262.
30. Draper-Gil, G.; Lashkari, A.H.; Mamun, M.S.I.; Ghorbani, A.A. Characterization of encrypted and vpn traffic using time-related. In Proceedings of the 2nd International Conference Information Systems Security and Privacy, Rome, Italy, 19–21 February 2016; pp. 407–414.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.