A Group-Based IoT devices Classification through Network Traffic Analysis based on Machine Learning Approach

Avewe BASSENE, Bamba GUEYE

Université Cheikh Anta Diop, Dakar, Senegal

Abstract. With the rapid growth of the Internet of Things (*IoT*), the deployment, management, and identification of *IoT* devices that are connected to networks become a big concern. Consequently, they emerge as a prominent challenge either for mobile network operators who try to offer cost-effective services tailored to *IoT* market, or for network administrators who aim to identify as well reduce costs processing and optimize traffic management of connected environments. In order to achieve high accuracy in terms of reliability, loss and response time, new devices real time discovery techniques based on traffic characteristics are mandatory in favor of the identification of *IoT* connected devices.
Therefore, we design *GBC_IoT*, a group-based machine learning approach that enables to identify connected *IoT* devices through network traffic analysis. By leveraging well-known machine learning algorithms, *GBC_IoT* framework identifies and categorizes *IoT* devices into three classes with an overall accuracy equals to roughly 99.98%. Therefore, *GBC_IoT* can efficiently identify *IoT* devices with less processing overhead compared to previous studies.

Keywords: Internet of Things, network traffic characteristics, machine learning algorithms.

1 Introduction

Recent long range radio transmissions enable affordable *IoT* solutions for under-served areas [1] [2]. According to Africa, smart cities and/or smart territory enable new opportunities for tackling urban explosion challenge.

Smart cities were originally designed to solve urban planning and sustainable development problems according to northern countries cities. Leveraging *IoT* networks in Africa can facilitate transport mobility, reduce energy consumption and offer optimal and innovative solutions for waste management and sanitation. Recently, Africa has registered many smart city projects that plan to develop by rapidly reducing the technological divide that affects the continent [3]. Recent works center on the deployment of *IoT* communication solutions for low-income developing countries [1] [2] [4].

Former works have mainly focused to deploy low cost networks which are mandatory within these under-served countries. Therefore, smart cities or territories are equipped by a lot of devices that need to be deployed and managed efficiently. Thus, it is useful to monitor *IoT* traffic characteristics in order to propose performance levels that meet current realities requirement. For instance, realities regarding to Quality of Service (*QoS*).

To meet this challenge, connected network devices type should be known. It is worth noticing that existing identification and classification proposals are based on unitary approaches and these methods are not suitable for wider *IoT* network such as smart cities or territories, which support an important number of devices. Consequently, approaches that are suitable for selected agglomerations that host heterogeneous *IoT* devices should be defined.

In fact, devices categorization can provide an adequate level of *QoS* by containing multicast traffic (unnecessary broadcasting) and reducing their impact on other applications. It also allows administrators to scale their networks according to appropriate performance levels in terms of reliability, loss and latency necessary for environmental, health or security applications. Furthermore, it can avoid congestion by adapting traffic load according to each devices category requirement.

In order to provide an efficient classification model, we seek to monitor either traffic features that as substantive as possible within network topology (physical or logical), or features which have already shown their effectiveness in previous works [5], [8], [9], [12].

Therefore, the *GBC_IoT* framework contributions are as follows:

- According to traffic generated by devices, it enables to accurately classify devices with respect to their behavior
- It takes into account the presence of a huge and heterogeneous *IoT* smart-Network devices
- It avoids classification based on no reliable traffic features often done by previous works such as *DNS* queries [5], *TCP* sessions length [6] [7] and traffic Active volume [8].

The rest of this paper is organized as follows: Section 2 reviews the related work on this field. We present in Section 3 our experimental setup. A grouped-based device classification approach with Machine Learning techniques is performed in Section 4. Section 5 discusses our contributions according to previous works. Finally, Section 6 concludes our paper.

2 Related Works

Former works like [8], [9], [12], [13] propose methods to identify *IoT* device based on various features embedded in generated traffic. Nevertheless, they have not take into account the impact of existing uncertainties linked to traffic: from features selection to end user behaviour.

Authors in [6] and [7] use *ML* approaches to identify *IoT* devices from network traffic analyzing. They aim to decide device reliability to join a secure network. The targeted features are optimal length of *TCP* sessions. We do not consider this approach, since features such as session length and intersession duration are strongly dependent on fixed timeout value which can be arbitrarily assigned to individual schemes. In addition, obtaining complete TCP flow requires waiting for the end of the session to be able to extract all the functionalities from flow. Devices such as *Nest security camera* TCP sessions can sometimes last several hours or days [5].

Motivated by privacy concerns, Apthorpe et al., illustrated how an Internet service provider (ISP) can infer the type of connected IoT device by traffic analysis [11]. Nevertheless, they only use a limited, stable and rather "predictable" number of device in terms of traffic (hub, camera and sleep monitor). Proposed approach also depends on a single feature: domain of DNS queries. Nevertheless, devices such as *Amazon Echo* and *Triby Speaker* deal with several home automation devices and this diversity gives them strong dynamic network behavior. These devices can communicate with an increasing number of Internet nodes which the designer cannot define in advance [15].

Miettinen et al. present a method to identify connected IoT device type in order to constrain vulnerable devices communications [12]. They use a wide variety of features extracted during configuration phase. However, this approach can not be used if device configuration phase is missed.

An identification based on MAC addresses is not efficient since potential attacker can usurp MAC addresses of compromised IoT device [16]. Indeed, although MAC addresses can be used to identify the manufacturer of a particular device, to our knowledge, there is no established standard for identifying the brand or type of a device based on its MAC address. Some approaches focus on identifying specific hardware characteristics or drivers (e.g. [17]) to recognize IoT device. Notwithstanding, such approaches are not efficient, since the same hardware or driver components can be deployed in wide variety of devices.

The authors of [5], [15] proposed a mixed characteristics in order to identify IoT devices. According to performance concern, ML approach is used to find the minimum number of packets necessary for early identification of Internet traffic in [18].

Nevertheless, previous works with respect to overall accuracy (99.9% in [5], 99% in [12] and [9], 95% in [8]), do not carry out fine IoT devices characterization that takes into account error related to the choice of features and the unpredictability linked to different factors such as applications, materials and traffic load [19]. In fact, according to working days and hours, autonomous communication protocols, devices add/remove or frequent firmware updating are other factors that alter network behavior. The entire dataset needs to be trained each time these last two cases (add/remove or update) occur in the environment in order to take account these changes.

The work done in [10] is one of the first large-scale study to explore the nature of $M2M$ traffic. It compares $M2M$ and traditional smartphone traffic from different perspectives: time variations, mobility, network performance, generated traffic volume, etc. The authors of [10] do not take into account the complexity (huge quantity of data) of the "new" IoT devices existing on the market today.

The works in [8] and [9] use *ML* techniques to characterize *IoT* devices within a smart network. However, they consider more features and they do not take into account privacy concerns. Traffic active volume is used in [8] to cluster *IoT* attributes and it cooperates in *IoT* devices classification. However, for device such as "*Netatmo Welcome*", we see that traffic Active volume feature is dependent on working days/times and used protocols, therefore, this attribute cannot be considered as a reliable classification indicator.

3 Experimental setup

3.1 Dataset description

The used datasets come from daily traffic captured in a campus network set up as a "Smart environment". These traces are proposed by A. Sivanathan et al., [9] and are formed by a set of 28 *IoT* devices. Traffic is captured and stored on the Internet via "*TP Link Archer C7*" Gateway flashing with the "*OpenWrt*" firmware.

Table 1. Iot devices composing overall traces classified by types.

Devices	**Types**
HP Printer, PIX-STAR Photo-frame, Hello Barbie	**Others**
Chromecast, Triby Speaker	**Multimedia**
Smart Things, Amazon Echo	**Hub**
Belkin Wemo switch, TP-Link Smart plug, iHome, Belkin wemo motion sensor	**Switches & Triggers**
Withings Smart scale, Blipcare Blood Pressure meter, Withings Aura smart sleep sensor	**Healthcare**
NEST Protect smoke alarm, Netatmo weather station	**Air quality sensors**
LiFX Smart Bulb, Philips Hue lightbulb	**Bulbs**
Belkin NetCam, Ring doorbell, August doorbell camera, Canary camera, TP-Link Day Night Cloud camera, Samsung SmartCam, Dropcam, Insteon Camera, Withings Smart Baby Monitor, Netatmo Welcome	**Cameras**

Devices that make up this environment and on which our study will focus are presented in Table 1. On this day, traces are available for free download at *http://149.171.189.1.* A more detailed explanation of the environment configuration and the data collection method is presented in [12].

3.2 Dataset preprocessing

A first approach is to use the *PCAP* files of daily captured traces of September 27^{th} called *Dataset*-1, September 30^{th} called *Dataset*-2, and October 2^{nd} 2016 called *Dataset*-3. We then made an analysis at two levels: Packet levels with the "*tShark*" utility and "*CICFLOWMETER*" [18] for flow level. "*CICFLOWMETER*" is a bidirectional network traffic flow generator tool written in *Java*. It offers more flexibility in the choice of studied features, add of new features and increased control of the flow delay time. It is free and public accessible, it analyses *PCAP* files larger than 100 *MB* and provides a *CSV* file with visual analysis report of 84 network traffic features. Here is what motivates his choice.

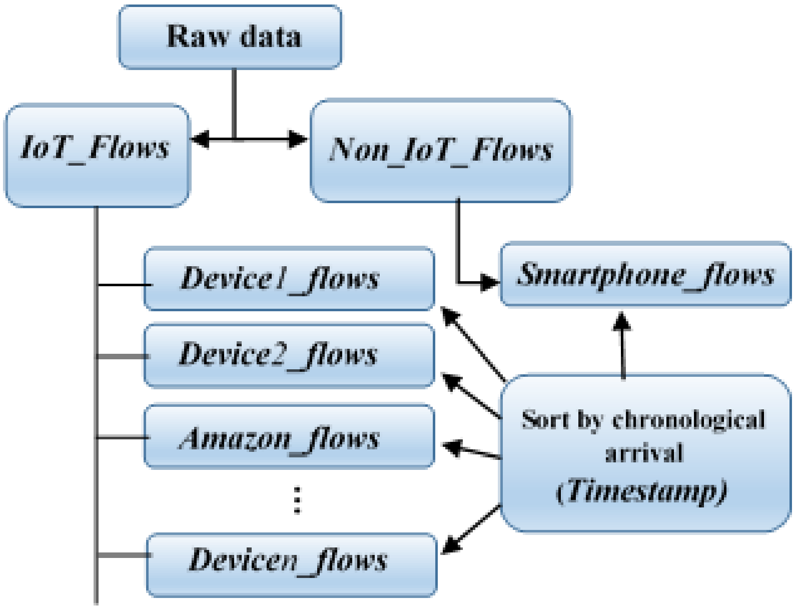

Fig. 1. Datasets preprocessing steps.

Raw network traffic is preprocessed to extract bidirectional flows. Flows is described by set of features such as *flow duration*, *total size of packets* (bidirectional), *active/idle time*, *source/destination IP*, packets *inter-arrival time*, *protocol* and *ports*, etc. It gives an overview of a coexistence of two types of devices ("*IoT/non IoT*") in the same environment.

We define "*IoT*" device as being a device intended to perform a specific task, unlike "*non IoT*" device such as: laptops, desktops or smartphones. With the help of *MAC* addresses, we split traffic by type of devices and by entity (unique device). We then separate each portion of obtained traffic. It facilitates labelling both unique and groups of devices data. For instance, label "*IoT_Flows*" groups together *PCAP/CSV* files containing traces coming from the network traffic generated only by *IoT* devices. While "*Amazon_Flows*" consists of *PCAP/CSV* files with flows coming from the network traffic generated by the *Amazon Echo* device. Finally, we used the *Timestamp* feature to order flows in chronological arrival. Fig. 1 illustrates the different steps it order to extract flows. For the rest of our work, data analysis and visualization are performed using the "*PYTHON*" scikit-learn libraries in *Jupyter* Notebook and *t-SNE*.

We propose a classification approach which studies bidirectional flows characteristics for each of the 28 testbed devices. For this end, 6 *ML* algorithms are used (*ANN*, *k-NN*, *DT*, *GBN*, *RF*, *SVM*). It aims to classifying devices according to category to which their behaviors are closest. Finally, algorithms that reach the greatest precision are those chosen as best suited to our model.

Fig. 2 describes our approach. The choice of these 6 algorithms is motivated by the fact that it is significant to have an overview of all aspects of the studied datasets. Indeed, in a test environment, each of these algorithms has these strengths and weaknesses based on different types of scenarios: For example, *DT* is too sensitive to small changes in the training dataset [20], while *SVM* and *RF* are insensitive to noise or overtraining, and thus, has ability to deal with unbalanced data. This is a good choice when different datasets are processing. According to *k-NN* classifier, while it is less sensitive with imbalanced training sample data, the training sample size had a strong impact on the accuracy of classification [21]. Moreover, the choice of a good value of k can be beneficial. All these above factors mentioned (including parameters choice) must be take into account to hope for a good performance level in various processed datasets.

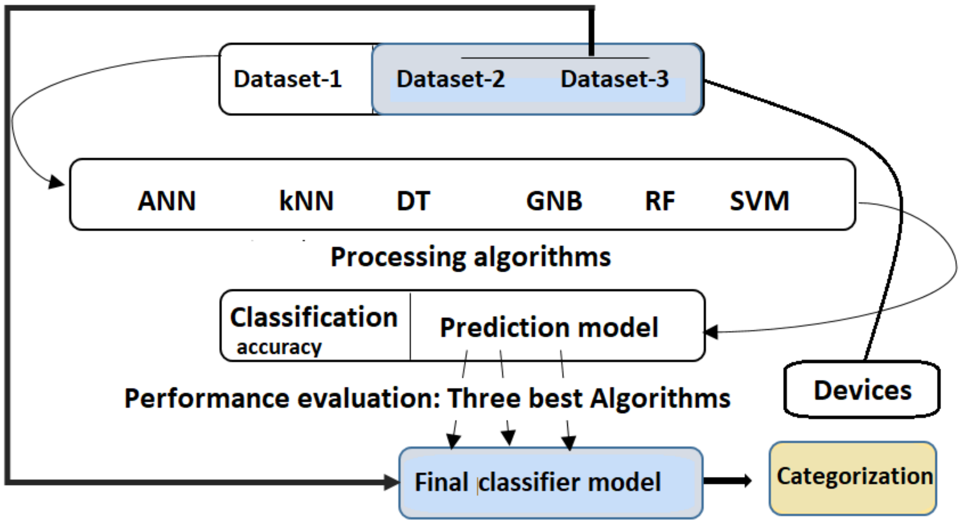

Fig. 2. Experimental processing approach.

3.3 Testbed architecture

We use *Dataset*-1 to train a classifier with algorithms above. It is the one that present less missing values which can affect the training process. We then randomly splits it into training (90% of total instances) and validation (10%) sets.

Our model is obtained by considering reached accuracies values. The classification process is repeated 10 times to tuned hyper parameters for most of the 6 algorithms. Afterwards, we considered *ANN*, *RF*, *SVM* algorithms that best perform. Finally, independent new test datasets are collected (*Dataset*-2, *Dataset*-3) from spanning days (October 27^{th} & October 30^{th}), that have not been seen before. These datasets are used for devices categorization purposes. Devices are categorized two times from each dataset to get their final class. The best classification accuracy value is adapted. This value depends on both the combined results of algorithms and on the best reached performance according to the final datasets (*Dataset*-2 and *Dataset*-3).

4 Experimental results

4.1 Leveraging hybrids features

Generally, two features combined are able to carry more traffic identification information than a single one [22]. The term "hybrid features" refers to a group of two or more traffic features in the hope of obtaining high precision in traffic identification. Features such as *packet size* and *inter-arrival times* have proven to be effective in many *IoT* devices or applications identification works [5], [22], [23].

A proposed model is based on hybrid features. As seen above, a precise identification of *IoT* device based on only one feature is difficult if not impossible because of traffic's various interfering factors. Hybridization lets us show specific shared behaviors in group of given devices. Table 2 illustrates an overview of the list of devices that share common characteristics.

According to Table 2, each index $G1$, $G2$ and $G3$ constitutes a particular group based on exposed fingerprints. Firstly, we observe that some devices share functionalities that we characterize as stationary; These devices use a local gateway

Table 2. List of devices sharing common characteristics.

Index	Devices
G1	Smart Things, Withings Smart Baby Monitor, Netatmo weather station, Withings Smart scale, Withings Aura smart sleep sensor, HP Printer
G2	Netatmo Welcome, TP-Link Day Night Cloud camera, Samsung SmartCam, Drop-cam, Insteon Camera, Belkin Wemo switch, TP-Link Smart plug, iHome, Belkin wemo motion sensor, NEST Protect smoke alarm, Blipcare Blood Pressure meter, Lightbulbs, LiFX Smart Bulb, PIX-STAR Photo-frame, Nest Dropcam
G3	Triby Speaker, Amazon Echo

as a DNS resolver and communicate with a single DNS server throughout their activities (network lifetime). In fact, considering factors can distinguish their traffic. Most of these devices communicate with their associated DNS server via a range of source ports. In fact, only 20% of them use specific source ports number. For instance, "*Netatmo weather station*" uses local gateway address as a resolver and communicates with a single DNS server "*netcom.netatmo.net*" with destination port 25587. "*Withings Smart scale*" receives DNS responses from the server "*scalews.withings.net*" through the local gateway. This group of devices with similar behavior is listed as index $G1$ in Table 2.

The index $G2$ group devices class displaying behaviors depending on generated traffic destination; external (destination outside the local network (internet)) or internal (see Table 2). It constitutes our second observation. Externally, these devices communicate with their $STUN$ servers via arbitrary IP addresses and a range of port numbers between TCP 3205 and TCP 4603. Internally, they use $mDNS$ as name resolution protocol through UDP port 5353. The $mDNS$ protocol is intended to resolve host names to IP addresses in small networks with no local DNS server.

According to second observation, the fingerprints exposed by the external communications protocols ($STUN$) and local $mDNS$ allows to determine this second group of devices. Due to the instability of this devices, it cannot be precisely defined because, device such as *Amazon Echo* deals with several home automation devices (*Belkin Wemo*, *SmartThings*, *Insteon*, etc.). This gives them a dynamic network behavior. [15] shows that these devices can communicate with an increasing number of internet nodes which the designer cannot define in advance. Indeed, $G3$ index with respect to Table 2 illustrates such situation.

Furthermore, thanks to a careful observation of traffic, we were able to obtain a first distinction object for studied IoT devices. This distinction is based on the use of a hybrid model and allows us to define the presence of three categories of device in our traces. However, this is far from sufficient since, despite observations on which Table 2 is based, we find factors that remain similar between devices of different groups. For instance, a couple of devices, tagged by index $G1$ and $G2$ in Table 2, use the source port 49153 in their external $HTTP$ communications (*iHome*, *Withings Smart scale*, *Samsung SmartCam*), while others use it to communicate internally (*Belkin Wemo switch*, *Belkin wemo motion sensor*). *Amazon Echo* distinctly shares the same NTP server with *LiFX Smart Bulb* and *Insteon Camera*. An invariant port number is used in $XMPP$ communications (TCP 5222) for *Withings Aura smart sleep sensor*, *HP Printer* and *Samsung SmartCam*.

This similarity calls for a better refinement of our model in order to dispel the confusion between devices. For that, focus on characteristics more suitable to the devices traffic footprint, thus important traffic features are extracted first to avoid overfitting in data as described in next section.

Since sensors behavior is very application specific, we therefore believe that total size of sent or received packets (*TotLen_Pkts*) is a common communication feature that can be assigned to any device with a sensor. In addition, *TotLen_Pkts*

Table 3. Best selected features and its weight.

Features	**weight**
Bwd Pkt Len Max	0.01165
Bwd Pkt Len Std	0.00036
Flow Byts/s	0.00056
Flow Pkts/s	0.00648
Flow IAT Mean	0.00879
Flow IAT Max	0.00827
Flow IAT Min	0.00321
Fwd Header Len	0.00539
Bwd Header Len	8e-05
Fwd Pkts/s	0.01853
Pkt Len Max	0.00195
Pkt Len Mean	0.01459
Pkt Len Std	0.02289
Pkt Len Var	0.02309
Pkt Size Avg	0.05113
Init Bwd Win Byts	0.08353

and *inter-arrival times* (IAT) have proven their effectiveness in numerous works and specifically in [24], authors have shown that packets from an early stage of Internet flow can contain enough information for traffic classification.

4.2 Traffic features selection

We maintain total packet size (*TotLenBwdPkts*) and inter-arrival time (*FwdIATMean*) features and apply dimensionality reduction techniques to the rest of traffic. It aims to find not only the most important features after those already chosen but also to exclude redundancies into the data and classifiers overtraining. This makes data easily interpretable and increases computational performance. These techniques also avoid an arbitrarily revoking of features which may prove to be relevant in the definition of a specific group of devices. We use the *"RandomForestRegressor"* class of scikit-learn to get 16 from 80 features extracted before. Table 3 illustrates the obtained set of features as well as associated weight.

In fact, results presented in Table 3 illustrate that the total number of received bytes within an initial time window (*Init Bwd Win Byt*), as well average packets size (*AvgPacketSize*), variance packet length (*Pkt Len Var*), and standard deviation packet length (*Pkt Len Std*) are the most suitable features according to device category detection. According to the remainder of our analysis, we have grouped these 4 best features above to those already maintained (*TotLen-BwdPkts FwdIATMean*) and then we empirically reduce this number in order to obtain better performance with a minimum number of features, i.e. we repeated testbed processing by reducing the number of features and writing down the observed classification accuracy values.

Our experimental results reach overall accuracy of 99.98% with a minimum of 3 features (*TotLen Bwd Pkts*, *Fwd IAT Mean*, *Init Bwd Win Byts*). The final features that best describing traffic behavior are then used to form six different *ML* classifiers in order to predict group (class) to which each *IoT* device in the dataset belongs (strongly close). Three classes are defined; C$_{WMS}$, C$_{AEC}$ and C$_{SSC}$ to represent devices whose traffic behavior is closest to that of the following *IoT* devices traffic in the respective order: *Wemo motion sensor* (*WMS*), *Amazon Echo* (*AEC*) and *Samsung SmartCam* (*SSC*). The choice of these classes is based on the similarity observed in Table 2 and on the fact that these devices traffic admit features of almost all the environment equipment.

Belkin Wemo Motion Sensor (*WMS*) is a motion detector that combines traffic with that of several devices; switches, bulbs and multimedia (Belkin wemo switch, switching on and off a television, a bulb, a hi-fi system, a radiator, a fan, etc.). *Amazon Echo* (*AEC*) is a connected speaker working with home automation equipment; *i.e.* Air quality sensors (Temperature display, temperature regulator), heater, WiFi IP Surveillance camera, LED bulb, Healthcare: Omron connected blood pressure monitor (Monitoring of morning hypertension, Detection of irregular heartbeats, Detection of body movement, visceral fat, *BMI* and metabolism rest). *Samsung SmartCam* (*SSC*): HD video camera, two-way talk feature, motion and audio detection, night vision, etc.

In addition, these devices traffic admit less missing data (verify by chronological timestamp values and Pandas *"isna()"* method) among devices in the same index. Artificial Neural Networks (*ANN*), k-Nearest Neighbors (*k-NN*), Decision Tree (*DT*), Naive Bayes (*GNB*), Random Forests (*RF*), Support Vector Machine (*SVM*) classifiers are used in our simulations.

We then assess the effectiveness of our classifiers by applying them to an independent new test datasets. We repeat this experiment on new datasets in order to consolidate ours results.

4.3 Data visualization

This section studies *IoT* devices activity defined by the 18 traffic features previously obtained. For each devices category, we extract these features in a new dataset. Differentiation can be noticed by visualizing the proximity between our data points. It enables to get an insight of their reconciliation and to assess the discriminating aspect of proposed model. We want to visualize the role played by selected features and their effectiveness to differentiate traffic generated by each class of devices.

Fig. 3 describes data visualization results using t-Distributed Stochastic Neighbor Embedding (*t-SNE*) algorithm. It should be noted that according to tags illustrated in Fig. 3, "*SSC*" means *Samsung SmartCam*, "*WMS*" means *Wemo motion Sensor*, and "*Aec*" means *Amazon Echo*.

t-SNE is a non-linear dimension reduction technique which is particularly well suited to visualize large datasets. It is widely applied in image processing, *NLP*, genomics data and speech processing. *t-SNE* is an unsupervised machine

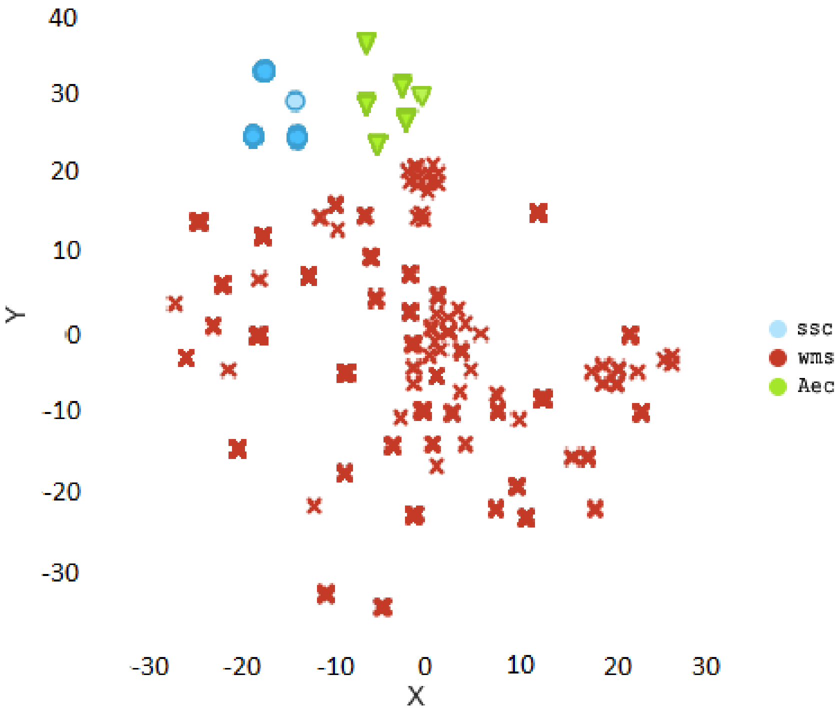

Fig. 3. Data visualization with t-SNE.

learning algorithm that attempts to represent similar data points next to each other while preserving the overall structure of the dataset.

We can note that Fig. 3 depicted data points form visual clusters corresponding to network traffic generated by different categories of IoT devices. Indeed, most data points for same IoT device are close to each other, while data points for different IoT devices are far apart. Our findings exhibit that selected features are effective enough to discern these three devices classes.

4.4 Classification

Classification Results. For classes C$_{WMS}$, C$_{AEC}$ and C$_{SSC}$, devices are classified using following metrics:
Accuracy: means the proportion of correctly classified flows.
Precision: exhibits the ratio $TP/(TP + FP)$; (i) where TP and FP express the number of true positives and false positives.
Recall: illustrates the ratio $TP/(TP+FN)$; (ii) where FN means the number of false negatives. Furthermore, TP/TN are outcomes where the model correctly predicts positive/negative class. Similarly, FP/FN are outcomes where posi-

tive/negative class is incorrectly predicted by the model. A weighted harmonic mean of precision and recall is **F1-Score**.

F1-Score: reaches its best score at 1 and the worst at 0. The obtained formula expressed as follows by Equation 1:

$$F1 - Score = 2 * (Precision * Recall)/(Precision + Recall). \tag{1}$$

Since test datasets varies in size, micro-average is used to computes different metrics in order to assess the overall *precision*, *recall* and $F1 - score$ (Table 4). The macro-average method can be used when you want to know how the system works globally across all datasets. It computes metric independently for each class, then takes the average (so all classes are treated equally), while a micro-average aggregates the contributions of all classes to calculate the average metric. Classifiers that reach the best overall performance are chosen by vote.

Table 4 shows that Artificial Neural Network (ANN), Decision Tree (DT), Random Forest and SVM reach the highest overall precision rate close to 99.9%. The k-Nearest Neighbors (k-NN) algorithm then comes with an overall accuracy of 98.4%. This means that these algorithms are able to uniquely identify a category of IoT device with very high probability. These results once again demonstrate the skill of proposed features to accurately identify devices categories. ANN improvement requires to encode categorical variables label to numerical ones. In addition, we have enough data collected to reach a high precision, which was a limit with work in [5].

Similarly, for each category, we estimate the precision, recall and F1-score individually to show the ability of classifiers to identify each of them (binary decision). Table 5 illustrates obtained results. For each class taken individually, it can be seen that ANN, *DT*, *RF* and SVM algorithms are able to positively verify a class of devices with a high precision (between 89.9% and 100% for ANN, *DT*, *RF*; and between 70.8% and 100% when *k-NN* is included. GNB is amongst algorithms that be downgraded because of his worst performance (overall near 76.3%).

Afterwards, we classify the rest of testbed devices by group. The three best algorithms which offer the high precision are chosen by vote. We then submit samples of each of these 26 devices to our prediction models obtained above from *MLPC.predict()*, *SVC.predict()* and *RFC.predict()* classes. These classes

Table 4. Overall performance on the test set of the different classifiers.

Algorithms	**accuracy**	**micro-avg precision**	**micro-avg recall**	**micro-avg F1 score**
ANN	0.999	0.999	0.999	0.999
k-NN	0.984	0.984	0.984	0.984
DT	0.999	0.999	0.999	0.999
GNB	0.649	0.649	0.649	0.649
RF	0.999	0.999	0.999	0.999
SVM	0.999	0.999	0.999	0.999

Table 5. Precision, Recall and F1-score on the test set for different classifiers.

Algorithms	**Precision**			**Recall**			**F1-score**		
	C$_{SSC}$	C$_{WMS}$	C$_{AEC}$	C$_{SSC}$	C$_{WMS}$	C$_{AEC}$	C$_{SSC}$	C$_{WMS}$	C$_{AEC}$
ANN	1.	1.	1.	1.	1.	.998	1.	1.	1.
k-NN	.797	.974	.984	.708	.998	.957	.935	.989	.968
DT	1.	1.	1.	.996	1.	1.	.999	.997	1.
GNB	.785	.958	.715	1.	.605	.767	.553	.749	.737
RF	1.	1.	1.	.898	1.	1.	.999	1.	1.
SVM	.809	1.	1.	.807	1.	1.	.809	1.	1.

are those used in the prediction phase for respectively ANN, SVM and *RF* algorithms. Table 6 shows classification results. It is worth noticing that:

1. Any device classified as belonging to a single class X by the three algorithms with an accuracy greater than 75% is retained as belonging to X.
2. Device belongs to a class X if predicted by at least two out of three algorithms as being similar to X with overall accuracy greater than or equal to 80%.
3. Any device classified as belonging to the three classes by the three algorithms with any accuracy rate is downgraded.
4. Any classification giving an overall accuracy between 85% and 100% obtained by two out of three algorithms, for device classified twice, the classification of the two algorithms prevails over the remaining one.
5. Any other classification obtained is rejected and the equipment is considered unrecognized.

Case (3) is observed for two devices; *NEST Protect smoke alarm* and Hello Barbie in Table 6. These devices are downgraded and considered as unrecognized. This is due to a lack of collected data. Indeed, over a 24-hour observation period, few data is captured for these devices in the smart environment. Device such as "*Netatmo Welcome*", "*Triby Speaker*" and "*TP-Link Day Night Cloud camera*" belong to case (2). *Netatmo Welcome* is accurately classifier as C$_{AEC}$ with 99.98% by SVM and 86.93% by RF.

Regarding to *Triby Speaker*, it is mapped as C$_{WMS}$ with accuracies of 88% by *SVM*, 95.41% by ANN and 47.8% as C$_{AEC}$ by RF. *TP-Link Day Night Cloud camera* is identified with accuracy near 90.88% as C$_{AEC}$ by *RF*, 99.95% by SVM and as C$_{WMS}$ with 88.18% of precision. "*August Doorbell Cam*" and "*Ring Door Bell*" are also downgraded, because no traffic instance of this devices is observed in dataset. This probably due to their absence (out of service) during the capture phase. Overall, only around 3% devices is not classified for reasons mentioned above. Any other device in Table 6 belongs to case (1).

Performance Evaluation. In order to assess overall performance of our model, we focus on the total number of packets (*Tot Fwd/Bwd Pkts* columns) contained in the first flows generated by each device. We attempt to find optimal number of packets needed to accurately recognize a device. Let p the number

Table 6. Group-based Iot devices classification.

Devices	$\mathbf{C}_{CSS}$	$\mathbf{C}_{WMS}$	$\mathbf{C}_{AEC}$
Dropcam	⊗	✓	⊗
Netatmo weather station	✓	⊗	⊗
LiFX Smart Bulb	⊗	✓	⊗
Triby Speaker	⊗	✓	⊗
TP-Link Day Night camera	⊗	⊗	✓
Withings Aura smart sleep sensor	⊗	✓	⊗
iHome	⊗	✓	⊗
Withings Smart Baby Monitor	⊗	⊗	✓
NEST Protect smoke alarm	⊘	⊘	⊘
PIX-STAR Photo-frame	✓	⊗	⊗
Insteon Camera (Wifi & Lan)	⊗	⊗	✓
Belkin wemo motion sensor	⊗	✓	⊗
Smart Things	⊗	⊗	✓
Belkin Wemo switch	⊗	✓	⊗
TP-Link Smart plug	⊗	✓	⊗
HP Printer	⊗	✓	⊗
Nest Dropcam	⊗	✓	⊗
Amazon Echo	⊗	⊗	✓
Netatmo Welcome	⊗	⊗	✓
Samsung SmartCam	✓	⊗	⊗
Blipcare Blood Pressure meter	⊗	✓	⊗
August Doorbell Cam	–	–	–
Awair air quality monitor	✓	⊗	⊗
Canary Camera	✓	⊗	⊗
Google Chromecast	⊗	⊗	✓
Hello Barbie	⊘	⊘	⊘
Phillip Hue Lightbulb	⊗	✓	⊗
Ring Door Bell	–	–	–

of packets sent and received by each device in chronological order, we rearrange the testbed by varying the value of p from 2 to 10. The Random Forest classifier is used in this empirical qualifier experiment. Indeed, *RF* is one of the trio of algorithms that reaches best accuracy values and is suited for models performance comparison with work in [5].

A small value of p (*e.g.* $p = 1$) seems insufficient to ensure a certain verdict. Fig .4 illustrates the obtained results. We observe that the overall accuracy rate increases according to the number of packets. For $2 \leq p \leq 10$, the accuracy rate goes from 98.81% to 99.98%. It increases by 1.4% between 2 and 3 and by only 0.3% between 3 and 10, 10 maximizes the value of p since no variation is noted next. The accuracy reaches its best rate for $p = 4$ and does not vary regardless to p. We already obtained a suitable precision of 98.81% with only 2 packets received ($p = 2$). However we do not adapt this value since *NEST Protect smoke alarm* and "*Hello Barbie*" are downgraded (Table 6). In fact, flow observed for each of these devices consists of only 2 packets and we see above that the identification of these devices by the three classifiers was uncertain.

Based on obtained results, we reach overall accuracy value near 99.98% with RF for an optimal value of 4 packets. The limited number of devices representative classes explains this high rate because although the number of devices is relatively large, they are finally grouped into only few classes (3). Furthermore, our model offers better performance and less overhead compared to previous work [5]; *GBC_IoT* reaches 99.98% with $p = 4$ while 6 packets are necessary in Shahid et al. model to achieve relatively the same accuracy, Fig. 4.

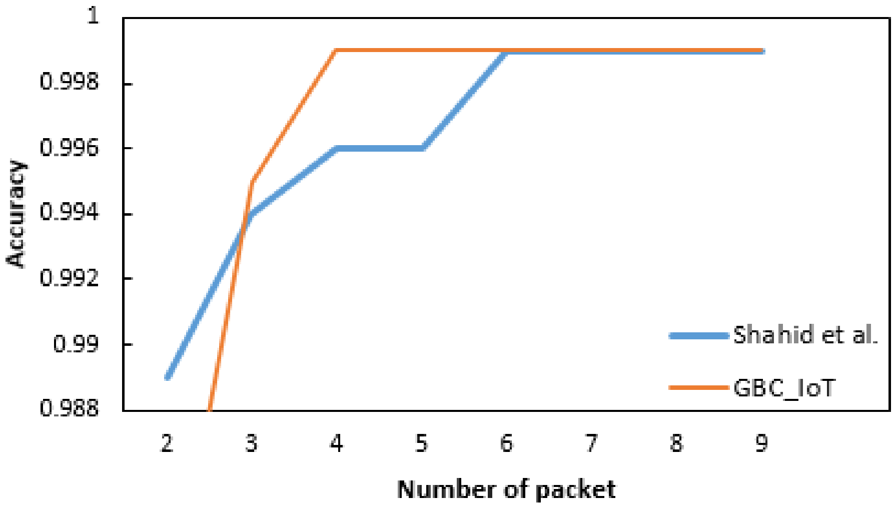

Fig. 4. Overall performance comparison according to number of packet sent and received.

5 Discussion

An unitary classification technique using four features is proposed by Shahid et al. [5] to distinguish traffic from four *IoT* devices. *GBC_IoT* approach is better since it uses fewer features and addresses more *IoT* devices (28 vs. 4). In addition, it gives less computational overhead with an optimal number of packets for device recognition. There is lack of traditional devices (Smartphone, Tablet, PC, etc.) in traffic traces studied in [5], which is not the case in a real *IoT* environment. Thus, a question is: what would be the performance of their model in a real environment which exhibits the behavior of traditional equipment on studied datasets?

Furthermore, in front of huge amount of available *IoT* devices, it becomes difficult to deal with a single classifier for each device as performed in [5]. The same is true for works in [8] [9] [22]. Thus, we propose an optimal and less computational model that train classifiers for groups of devices sharing similar behaviors. The authors in [18] evaluated the effective number of packets in early stage Internet traffic identification. They show this value between 5 and 7. GBC_IoT require just 4 packets to accurately identify IoT device class studied here. A common limitation of works related to IoT (including ours) is the limited number of devices studied due to the lack of publicly available data. However, to deal with a single classifier for every device does not seem to be a good approach. GBC_IoT is more suitable for managing a smart territories scale environment which hosts a large number of devices, since it proposes to classify any device among three instead of 28 or probably more.

GBC_IoT is limited by active classification, it cannot classify additional device that be added to network until it has enough traffic data. It would be interesting to adapt it to a real-time classification model, thereby, it might be adequate for fine group-based traffic QoS management. As solution, we propose an IoT environment based on Software-Defined Network (SDN).

According to this context, the controller with a global view of the network will detect any new added device $app\_profile$ and its GBC_IoT-based class. Thus, data forwarding decision rules will arise from both device application QCI requirements vector and GBC_IoT classification result. Thereby, GBC_IoT should be implemented as a module in the SDN controller. A better alternative approach is to implement it in a hybrid SDN-based IoT network architecture, which is easier to deploy in current networks. However, studies based on our approach must be carried out, trying above all to choose features that are more independent of network structure. A set of features such that there is none that depends on the physical structure of the network would give more scalability to our model.

As summary, the proposed model does not use any of the features mentioned in section 2; *TCP session length* [5], *traffic volume* [10], domain of *DNS queries* [11], MAC addresses or NIC. In addition, GBC_IoT uses less features for a better precision then literature [5], [15]. According to performance concern, GBC_IoT framework gives less computational overhead with a minimum of 4 packets for traffic identification as illustrated by obtained results in Fig. 4. This number is between 5 and 7 in [18] and equals to 6 in [5].

To the best of our knowledge, this is the first work that uses *ML* approach to deal with classification problem relating to both the increasing number and diversity of connected IoT devices and that avoids the use of features that more likely to affect traffic behavior. Indeed, it becomes difficult and resources intensive to deal with a single classifier for every device among thousands of different types of available devices in the market. Thus, a classifier that deals with groups of devices that share similar behaviors is more suitable for this wide variety of equipment in IoT worlds.

6 Conclusion

Nowadays, recognition as well devices integration become a challenging research field with respect to *IoT* networks. Therefore, we proposed a group-based *IoT* devices classification using network traffic characteristics. Visualization using *t-SNE* highlighted the effectiveness of selected features. According different used *ML* algorithms, we achieve an overall accuracy of 99.98%. Furthermore, our model classifies 26 amongst the 28 devices provided by traces collected in [9].

Based on our experimental results, we can conclude that Group-Based *IoT* devices Classification related to passive network traffic analysis can offer great accuracy in devices behavior recognition and is more suitable for smart territories environment that host thousands of different types of devices.

We believe that a group-based classification (active/passive) of *IoT* device according to their traffic characteristic is the more suitable approach. However, continuously update the proposed model is necessary due to regularly network behavior changes over time for different reasons. The integration of *SDN* concepts would be an asset and a good prospect for improving our model, since an instant and adequate model can be implemented according to network real-time state.

References

1. C. Pham, A. Rahim and P. Cousin.: Low-cost, Long-range open IoT for smarter rural African villages. in Proc. IEEE ISC2, pp. 1-6, Trento 2016.
2. M. R. Seye, M. Diallo, B. Gueye and C. Cambier.: COWShED: Communication Within White Spots for Breeders. In Proc. IEEE ICIN, pp. 236-238, France, 2019.
3. Agence Française de Développement (AFD), ASTON, *https://www.afd.fr/fr/actualites/aston-transition-numerique-villes-africaines.*
4. Masinde Muthoni.: IoT Applications that work for the African Continent: Innovation or Adoption?. In Proc. IEEE ICII, pp. 633-638, Porto Alegre, Brazil, 2014.
5. Shahid Mustafizur Rahman, Blanc Gregory, Zhang Zonghua, Debar, Hervé.: IoT Devices Recognition through Network Traffic Analysis. In Proc. IEEE ICBD, pp. 5187-5192, USA, 2018.
6. 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 Proc. SAC, Pages 506–509, Morocco, 2017.
7. 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:1709.04647, 2017.
8. A. Sivanathan et al.: Characterizing and classifying IoT traffic in smart cities and campuses. In Proc. IEEE INFOCOM WKSHPS, pp. 559-564, Atlanta, GA, 2017.
9. A. Sivanathan, H. Habibi Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath, and V. Sivaraman.: Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics. In Proc. IEEE Transactions on Mobile Computing, vol. 18, no. 8, pp. 1745-1759, 1 Aug., 2019.

10. M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang and J. Wang.: Large-Scale Measurement and Characterization of Cellular Machine-to-Machine Traffic. In Proc. IEEE/ACM Transactions on Networking, vol. 21, no. 6, pp. 1960-1973, Dec., 2013.
11. N. Apthorpe, D. Reisman, N. Feamster.: A Smart Home is No Castle: Privacy Vulnerabilities of Encrypted IoT Traffic. In arXiv:1705.06805, 2017.
12. M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A. Sadeghi, S. Tarkoma.: Iot sentinel: Automated device-type identification for security enforcement in iot. In Proc. ICDCS,pp. 2177-2184, Atlanta, GA, 2017.
13. Anderson Blake, McGrew David.: Identifying Encrypted Malware Traffic with Contextual Flow Data. In Proc. AISec, pp. 35-46, 2016.
14. A. Sivanathan, D. Sherratt, H. H. Gharakheili, V. Sivaraman and A. Vishwanath.: Low-cost flow-based security solutions for smart-home IoT devices. In Proc. IEEE (ANTS), pp. 1-6, Bangalore, 2016.
15. Hamza Ayyoob, Ranathunga Dinesha, Habibi Gharakheili Hassan, Benson Theophilus, Roughan Matthew, Sivaraman V.: Verifying and Monitoring IoTs Network Behavior using MUD Profiles. In proc. IEEE Transactions on Dependable and Secure Computing, 2020.
16. Yuchun Tang, Yan-Qing Zhang, Zhen Huang.: FCM-SVM-RFE Gene Feature Selection Algorithm for Leukemia Classification from Microarray Gene Expression Data. In Proc. FUZZ, pp. 97-101, Reno, NV, 2005.
17. M. Weiss et al.: Time-Aware Applications, Computers, and Communication Systems. Technical Note (NIST TN) - 1867, Fev., 2015.
18. L. Peng, B. Yang, Y. Chen.: Effective Packet Number for Early Stage Internet Traffic Identification. In Neurocomputing Journal, vol. 156(25), pp. 252-267, 2015.
19. Li Ying, Noseworthy Bob, Laird Jeff, Winters Timothy, Carlin Timothy.: A study of precision of hardware time stamping packet traces. In Proc. IEEE ISPCS, pp. 102-107, Austin, TX, 2014.
20. Prasad A., Iverson L., Liaw A.: Newer classification and regression tree techniques: Bagging and random forests for ecological prediction. In Ecosystems Journal, vol. 9, pp. 181–199, 2006.
21. Thanh Noi P, Kappas M.: Comparison of Random Forest, k-Nearest Neighbor, and Support Vector Machine Classifiers for Land Cover Classification Using Sentinel-2 Imagery. In Sensors journal, vol. 18(1), pp. 18, 2017.
22. W. Linlin, L. Peng, M. Su, B. Yang, and X. Zhou.: On the impact of packet inter arrival time for early stage traffic identification. In Proc. IEEE iThings and IEEE GreenCom and IEEE CPSCom and IEEE SmartData, pp. 510-515, Chengdu, China, 2016.
23. Z. A. Qazi, J. Lee, T. Jin, G. Bellala, M. Arndt, and G. Noubir.: Application-awareness in SDN. In Proc. ACM SIGCOMM, pp. 487–488, NY, USA, 2013.
24. A. Este, F. Gringoli, and L. Salgar elli.: On the Stability of the Information Carried by Traffic Flow Features at the Packet Level. In Computer Communication Review journal, vol 39, pp. 13-18, 2009.