Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/MBE

MBE, 19(4): 3767–3786.
DOI: 10.3934/mbe.2022173
Received: 10 October 2021
Revised: 08 January 2022
Accepted: 25 January 2022
Published: 10 February 2022

Research article

Fine-grained identification of camera devices based on inherent features

Ruimin Wang[1], Ruixiang Li[1,2,*], Weiyu Dong[1], Zhiyong Zhang[3] and Liehui Jiang[1]

[1] State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou, China
[2] Key Laboratory of Cyberspace Situation Awareness of Henan Province, Zhengzhou, China
[3] Cyberspace Security Key Laboratory of Sichuan Province, Chengdu, China

* **Correspondence:** Email: xxxlrxxx@163.com.

Abstract: Camera devices are being deployed everywhere. Cities, enterprises, and more and more smart homes are using camera devices. Fine-grained identification of devices brings an in-depth understanding of the characteristics of these devices. Identifying the device type helps secure the device safe. But, existing device identification methods have difficulty in distinguishing fine-grained types of devices. To address this challenge, we propose a fine-grained identification method based on the camera devices' inherent features. First, feature selection is based on the coverage and differences of the inherent features type. Second, the features are classified according to their representation. A design feature similarity calculation strategy (FSCS) for each type of feature is established. Then the feature weights are determined based on feature entropy. Finally, we present a device similarity model based on the FSCS and feature weights. And we use this model to identify the fine-grained type of a target device. We have evaluated our method on Dahua and Hikvision camera devices. The experimental results show that we can identify the device's fine-grained type when some inherent feature values are missing. Even when the inherent feature "missing rate" is 50%, the average accuracy still exceeds 80%.

Keywords: device identification; feature similarity calculation; fine-grained identification; inherent features; Internet of Things

1. Introduction

Camera devices are typical pieces of Internet of Things (IoT) equipment ubiquitously deployed in cyberspace and play an important role in maintaining safety in daily life and industrial operations.

However, camera devices are usually technically heterogeneous and geographically dispersed, and it is time-consuming for devices owners to test which cameras are online and whether they are functioning normally. Camera devices are becoming attractive targets of cyber-attacks lack of security management. In 2016, up to 1.5 million webcams were attacked by the Mirai malware. The compromised webcams launched a large-scale distributed denial-of-service (DDoS) attack against some high-profile network infrastructure, causing paralysis of half of the U.S. internet. Therefore, accurately identifying the fine-grained type of device is of great significance for asset management. Conducting cyberspace resource surveying and mapping [1], assessing the impact of equipment vulnerabilities [2], and improving the effectiveness of network device governance [3] also requires understanding the type of devices.

Precise identification of camera devices is helpful for asset management, vulnerability assessment, and patch upgrading and thus is the groundwork for security management. Concurrently, there are two main kinds of methods for device identification, traffic-based, and web search-based methods—described as follows.

Traffic-based identification approaches can be classified into two main areas: active probing and passive monitoring.

Active Traffic Probing. A server sends a request message to a remote host using a specific application protocol (such as HTTP, FTP, POP3, etc.) via an IP address and extracts features from the response. Features match with a pre-established device fingerprint to establish the identification of the device. Shodan [4] is the world's first search engine for networked devices based on this technology. It uses Nmap [5] to periodically perform port scans on approximately 600 million devices around the world and returns banner information through processing to identify specific devices. Scholars at the University of Michigan used the self-developed Zmap [6] scanning tool to build the Censys [7] system to search for devices. In addition, ZoomEye, FoFa, and Oshadan are examples of the implementation of active probing [8], which has strong scalability. For new device types, a fingerprint is added to the fingerprint database to establish identification.

This method can customize the request packet to obtain a specific response message text, feature data are easy to obtain. However, with the enhancement of security protection strategies, more and more devices no longer respond to request messages. In this case, the response data is not obtained and the device identification is invalid.

Passive Traffic Monitoring. This technique monitors network traffic without sending any messages. First, it extracts various characteristics such as protocol parameters, packet fingerprints, or communication patterns from the traffic. Second, it establishes an appropriate identification model leveraging machine learning theory is established. Finally, it identifies the device types using the identification model.

Y. Meidan et al. [9] took a quadruple composed of a source IP, destination IP, port number, and flag bits (SYN, FIN, ACK, etc.) in the TCP session data as characteristics. They used several machine learning approaches to classify and identify IoT devices.

Miettinen M et al. [10] extracted a total of 23 features (e.g., the protocol types of each layer of the packet, source, and destination port number) from network traffic. They built the SENTINEL system by using the random forest method to classify IoT devices.

Arunan Sivanathan et al. [11] extracted features such as flow duration, port number, domain name, and cipher suite. They built a multistage classification model based on naive Bayes and random forest methods.

Cheng et al. [12] took the differences among the device file headers in traffic as features and used classification algorithms such as a backpropagation neural network [13], support vector machine [14], and k-nearest neighbors [15] to identify device types.

Yang et al. [16] took advantage of the characteristics of device network protocols at different open systems interconnection layers and neural network algorithms to generate the fingerprints of IoT devices for device identification.

Arunan et al. [17] used traffic characteristics obtained at the network level for IoT device classification. They presented insights into the underlying network traffic characteristics using statistical attributes such as activity cycles, port numbers, signaling patterns, and cipher suites. The work developed a multistage machine learning-based classification algorithm to identify specific IoT devices.

Sakthi et al. [18] proposed GTID, a wireless device identification technique based on traffic measurement metrics using the Ping and iPerf tools. They collect the interarrival time (IAT) of messages from the switch as features to create a unique and reproducible device signature and use artificial neural networks (ANNs) for classification.

This type of method circumvents the problem of non-responsive messages due to security policies, but it also limits the acquisition of high-value features and reduces the accuracy of device classification and identification.

Web-search-based. Web-search-based device identification method acquires knowledge of IoT devices—i.e., IoT devices and related vendors, products, and models—to build a device information repository or establish annotation rules.

Zou et al. [19] proposed an IoT device recognition framework based on web searches, which identified the brand and model of IoT devices by matching their protocol banners with a product attributes database established by crawling specific electronic business websites.

Feng [20] proposed an ARE engine that extracts relevant terms from the response data as search query crawl websites. They used an association algorithm to generate rules for IoT device annotations based on vendors, products, and models.

Agarwal et al. [21] developed a tool named WID that captures the web pages of IoT devices and performs device type classification by analyzing the source code of the web pages.

This type of approach does not need manual construction of fingerprints or training data to identify device types, but the accuracy relies on the reliability of Internet resources and requires the design of very complex rules for matching attribute information, which limits the accuracy and recall of the algorithm.

In addition, there is a device type identification method based on clock deviation [22–27]. This type of method uses the deviation that still exists after each device is calibrated by the network time protocol (NTP) to classify IoT devices. However, the clock deviation of a device is very difficult to measure, and it is difficult to distinguish the difference in the clock deviation among devices from the same manufacturer. This method of identifying the specific type of device can cause large errors.

The above methods have high accuracy in identifying coarse-grained device types. However, it is a challenge to accurately identify fine-grained types of devices. We propose a fine-grained camera device identification method based on the inherent features of devices. Inherent features represent characteristics of the device itself, such as device-specific parameters, geometric shapes, physical properties, and technical parameters. There are few research studies based on inherent features for classifying and identifying devices.

We believe that inherent features can be used to effectively identify the fine-grained device. When distinguishing a device, our method determines the weight of each feature based on the coverage and differences of the inherent features. We develop a feature similarity calculation strategy (FSCS) based on the manifestation of an inherent feature value. Based on each weight and the FSCS, we build a device identification model to establish the fine-grained identification of a device type. The detailed contribution of work is given below:

1) We propose a new feature weight determination strategy. The method proposed in this paper selects features based on the coverage of inherent attributes. The greater the feature differences among different types are, the better the distinction of the features. For the inherent selection characteristics, we calculate a feature entropy value based on the feature differences, use the feature entropy value to determine a feature weight, and provide a feature weight determination strategy based on a theoretical foundation.

2) We design a more reasonable feature similarity calculation rule. The inherent characteristics are classified into "phrase type", "numerical value", "interval type", and "collection type" according to the expression form. Similarity calculation rules are developed for each type of inherent characteristic. Compared with a strategy that uses a single calculation rule to calculate all types of similarity, the classification similarity calculation rule is more concise, and the calculation cost is lower.

3) We construct a fine-grained camera equipment type identification model based on inherent characteristics. Based on the feature weights and similarity calculation rules, we design a device type identification model using the idea of weighted average. This model can recognize fine-grained types of target devices with high accuracy even when some inherent feature values are missing, and it has good applicability in a real environment.

The rest of the paper is as follows: Section 2 introduces the detailed methodology and steps of the method. The rationality and reliability analysis of the method is given in Section 3. Section 4 presents the experimental results and analysis. In the end, Section 5 concludes our paper and discusses future work.

2. Methodology

In this section, we explain our proposed method in detail. We first present the basic framework, and we explain the notations of our algorithm. Then, we analyze the FSCS. Finally, we build the device identification model. Our proposed basic framework is shown in Figure 1, which includes three phases described as follows.

Extracting the inherent features. For data about the inherent attributes (e.g., size, weight, parameter, etc.) of a device $f_{i,n}$ in various types of equipment, we select the inherent attributes with large differences and wide coverage as the inherent features.

Building the identification model. First, we design the calculation rules of feature similarity using the manifestation of the inherent features. Then, the feature weights are determined from the information entropy of the features in the dataset. Finally, we construct the identification model according to the calculation rules of feature similarity and the weights of the features.

Identifying the device type. Based on the inherent features of the target device, the similarity between the target device and known devices is calculated to identify the type of the target device, and the device model with the largest similarity value is considered as the model of the target device.

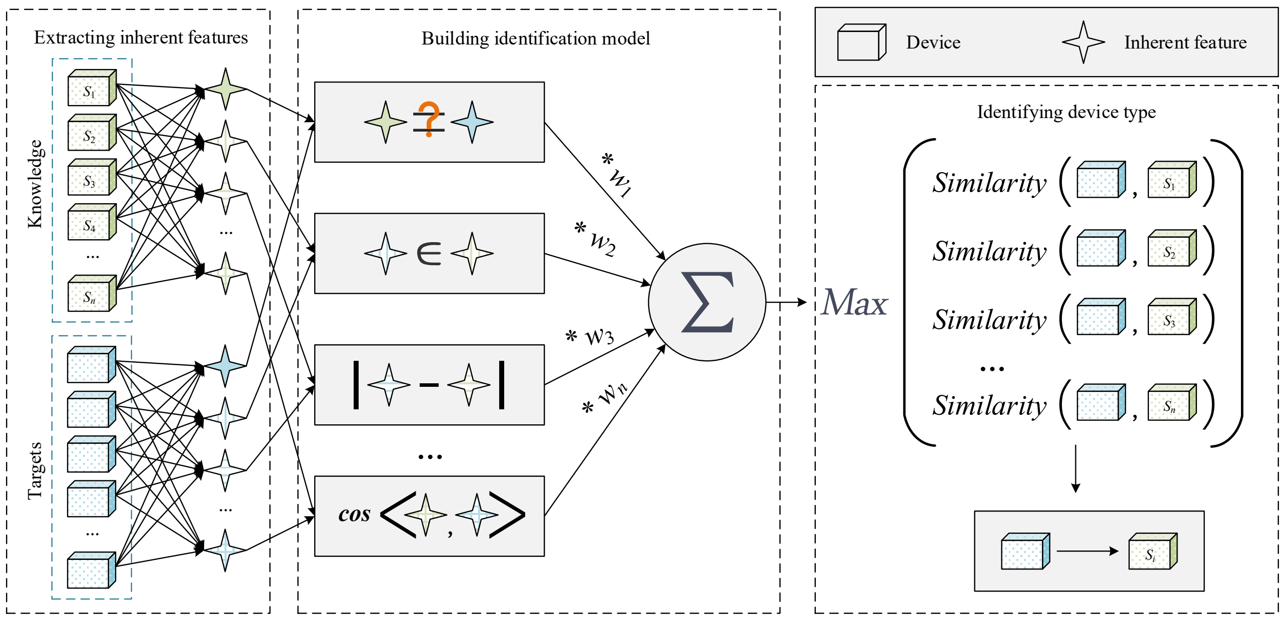

Figure 1. Framework of fine-grained device identification based on inherent features.

2.1. Notations

The notations used to describe our proposed approach and the means of evaluating them are summarized below.

f: An inherent feature of the device. $f_{i,n}$ is the nth feature of the ith device.

F: A collection of all inherent features of the device, and the inherent features of the ith device are $F_i = \{f_{i,1}, f_{i,2}, \dots, f_{i,n}\}$

T_i: The type of device I. T_{no} indicates the unknown device type; that is, if the type of device i is unknown, then $T_i = T_{no}$.

dev_i: A two-tuple composed of the inherent features and type of device i, $dev_i =< F_i, T_i >$.

D_{base}: A knowledge set defined as the collection of all known types of devices, $D_{base} = \{<F_{base,1}, T_{base,1} >, < F_{base,2}, T_{base,2} >, \dots, < F_{base,n}, T_{base,n} >\}$, $F_{base,n}$ is the feature set of the nth device in the D_{base} database and $T_{base,n}$ is the type of the nth device in the D_{base} database. $T_{base,i} \neq T_{no} (1 \leq i \leq n)$.

D_{target}: The target set is defined as the set consisting of all types of devices to be identified, $D_{target} = \{< F_{target,1}, T_{no} >, < F_{target,2}, T_{no} >, \dots, < F_{target,s}, T_{no} >\}$.

$Similarity(f_{i,k}, f_{j,k})$: The similarity of the k-th inherent feature between F_j and F_i. At this time, $f_{i,k}$ is called the benchmark feature, and $f_{j,k}$ is the target feature (note: $Similarity(f_{i,k}, f_{j,k})$ is not necessarily equal to $Similarity(f_{j,k}, f_{i,k})$).

$Similarity(F_i, F_j)$: The similarity of the feature set F_i and F_j. At this time, F_i is called the benchmark feature set, and F_j is the target feature set (note: $Similarity(F_i, F_j)$ is not necessarily equal to $Similarity(F_j, F_i)$).

2.2. Feature Similarity Calculation Strategy (FSCS)

Different inherent features are usually expressed in different forms. For example, the size of a device is usually "length × width × height" or "bottom radius × height", and the shape of a device is usually "square", "cylindrical" or "spherical". Therefore, it is necessary to design different feature similarity calculation rules for the features of different expressions. Analyzing the manifestation of the inherent features of the camera devices resulted in the division of the manifestation of the inherent features into four types: "phrase type", "numerical value", "interval type" and "collection type". We design similarity calculation rules for these four manifestations.

For "phrase type", when the feature phrases are the same, the feature similarity is 1; otherwise, it is 0. Considering the k-th feature of the inherent feature vectors F_{base} and F_i as a "phrase type" feature, the similarity calculation strategy between the target feature $f_{i,k}$ and the benchmark feature $f_{base,k}$ is shown in Eq (1).

$$Similarity\left(f_{base,k}, f_{i,k}\right) = \begin{cases} 1, & if \;\; f_{base,k} = f_{i,k} \\ 0, & if \;\; f_{base,k} \neq f_{i,k} \end{cases} \tag{1}$$

For the "numerical type", the size of the feature value and the measurement error ε of the feature value are comprehensively considered. If the kth feature of the inherent feature vectors F_{base} and F_i is a "numerical type" feature, we can obtain the similarity calculation strategy between the target feature $f_{i,k}$ and the benchmark feature $f_{base,k}$ described as Eq (2).

$$Similarity\left(f_{base,k}, f_{i,k}\right) = \begin{cases} 1 - \dfrac{\left|f_{base,k} - \left(f_{i,k} + \varepsilon\right)\right|}{\max\left(f_{base,k}, f_{i,k}\right)}, & if \;\; f_{base,k} > f_{i,k} \\ 1, & if \;\; f_{base,k} = f_{i,k} \\ 1 - \dfrac{\left|f_{base,k} - \left(f_{i,k} - \varepsilon\right)\right|}{\max\left(f_{base,k}, f_{i,k}\right)}, & if \;\; f_{base,k} < f_{i,k} \end{cases} \tag{2}$$

For "interval type", the calculation strategy is designed based on an interval inclusion relationship. If the kth feature of the inherent feature vectors F_{base} and F_i is an "interval type" feature, then the similarity calculation strategy between the target feature $f_{i,k}$ and the benchmark feature $f_{base,k}$ is constructed as Eq (3).

$$Similarity\left(f_{base,k}, f_{i,k}\right) = \begin{cases} 1, & if \;\; f_{i,k} \subseteq f_{base,k} \\ 0, & if \;\; f_{i,k} \not\subset f_{base,k} \end{cases} \tag{3}$$

For "collection type", the benchmark feature is used as a reference. The target feature and the benchmark feature are vectorized, and the cosine similarity is used to measure the feature similarity. If the kth feature of the inherent feature vectors F_{base} and F_i is a "collection type" feature, where $f_{base,k} = \left(e_{base,1}, e_{base,2}, \cdots, e_{base,n}\right)$ represents the benchmark feature and $f_{i,k} = \left(e_{i,1}, e_{i,2}, \cdots, e_{i,m}\right)$ represents the target feature, suppose the vectorization of the benchmark feature is expressed as $V_{base,k} = \left(a_{base,1}, a_{base,2}, \cdots, a_{base,n}\right)$, and the vectorization of the target feature is expressed as $V_{i,k} = \left(b_{i,1}, b_{i,2}, \cdots, b_{i,n}\right)$. Then, the vectorization process is as follows: for $V_{base,k}$, $\forall r \in Z^+$, $1 \le r \le n$, then $a_{base,r} = 1$; for $V_{i,k}$, $\forall r \in Z^+$, $1 \le r \le n$. If $e_{base,r} \in f_{i,k}$, then $b_{i,r} = 1$. After vectorization, the similarity calculation strategy is expressed as Eq (4).

$$Similarity\left(f_{base,k}, f_{i,k}\right) = \frac{\sum_{r=1}^{\|f_{i,k}\|} b_{i,r}}{\sqrt{\left\|f_{base,k}\right\| \times \sum_{r=1}^{\|f_{i,k}\|} b_{i,r}^2}} \tag{4}$$

where $b_{i,r}$ is the vectorization result of the *r*-th feature of the *i*-th device.

2.3. Identification model

According to the differences in the inherent features, an entropy value of a feature is calculated as the weight value. By using the FSCS combined with the inherent feature weights, the device identification model is constructed. The determination strategy of the inherent feature weight is as follows.

Given a feature, for each distinct value v of the feature, let v be the number of occurrences of *v* in the knowledge set, and let *f* = *v/N* be the frequency of occurrence of *v*, where *N* is the size of the knowledge set. If we approximate *f* as the probability of *v*, we can calculate the entropy of the feature as Eq (5).

$$H_k = -\sum p_i \bullet \log\left(p_i\right) \tag{5}$$

where p_i represents the probability function of the *i*th value of the feature f_k.

After calculating the entropy $H_1, H_2, \cdots, H_n$ of all features, the weight value $w_1, w_2, \cdots, w_n$ of each feature is calculated as Eq (6).

$$w_i = \frac{H_i}{\sum_{k=1}^{n} H_k} \tag{6}$$

According to the similarity calculation strategy and the weight of each feature, a feature set similarity model is constructed as Eq (7).

$$Similarity\left(F_{base}, F_i\right) = \sum_{k=1}^{\|F_{base}\|} Similarity\left(f_{base,k}, f_{i,k}\right) \times w_k \tag{7}$$

The identification model is used to calculate the similarity between the target device and the known devices. In the process of device type identification, the device type with the largest similarity is considered the device type of the target device, so the device type identification is as Eq (8).

$$T_i = T_{base,k} \left| Similarity\left(F_{base,k}, F_i\right) = \max\left\{ Similarity\left(F_{base,j}, F_i\right), < F_{base,j}, T_{base,j} > \in D_{base} \right\}\right. \tag{8}$$

3. Method analysis

In this section, we'll analyze the effectiveness of the inherent feature based device identification and the rationality of the FSCS.

3.1. Effectiveness analysis of inherent feature-based device identification

Devices of different models are usually different in inherent features. For example, as shown in Table 1, camera devices of different models are different in some inherent features such as appearance. size, weight. and so on. So, it's sound and feasible to distinguish devices by their inherent features. We should avoid identifying devices just by one single inherent feature since the measurement process of the value of an inherent feature may introduce deviation, as many as inherent features should be used instead.

Table 1. Differences in some inherent attributes of camera devices.

Camera device model	Appearance	Size (mm)	Weight (g)	Imaging component (inches)	Compression rate (bps)
Dahua DH-IPC-HDBW2230R-AS	Dome	φ122 × 89	406	CMOS,1/2.7	6K-8 M
Dahua DH-IPC-HDBW4636R-AS	Dome	φ122 × 89	444	CMOS,1/2.9	73K-10 M
Dahua DH-IPC-HDBW4833R-ZAS	Dome	φ122 × 88.9	494	CMOS,1/1.8	16K-14.75 M
Dahua DH-IPC-HFW4433F-ZAS	Bullet	186 × 87 × 85	772	CMOS,1/3.0	8K-10 M
Hikvision DS-2CD1311D	Dome	φ127 × 97.5	570	CMOS,1/3.0	32K-16 M
Hikvision DS-2CD3345P1	Dome	φ127.3 × 103.7	340	CMOS,1/3.0	32K-16 M
Hikvision DS-2CD3646F	Barrel	191.4 × 97.9 × 93.5	1260	CMOS,1/2.7	32K-8 M
Hikvision DS-2CD3726F	Dome	153.3 × 153.3 × 111.6	840	CMOS,1/2.7	32K-8 M

Table 1 shows that almost all devices of different types have different values of the "weight" inherent attribute, but it should be noted that in the actual classification process, the value of the "weight" inherent attribute is measured, and deviations are prone to occur during the measurement process. Using only the "weight" attribute as an inherent feature to identify devices results in large errors. It is necessary to use multiple inherent features to identify a camera device. Therefore, the use of multiple inherent features in this method can more effectively identify a device.

3.2. Rationality analysis of the FSCS

There are 4 types of inherent features: "phrase", "numerical", "interval" and "collection", and each has a corresponding FSCS respectively.

A. Phrase

The inherent features of the "phrase" type are described by one or more words. For example, the inherent feature of the shape of a camera device may be described by words like "dome", "bullet", or "barrel". The FSCS of inherent features of the "phrase" type is determined by whether the phrases are the same, so it's reasonable to use Eq (1) to calculate the similarity of two features of the "phrase" type.

B. Numerical

For inherent features of the "numerical" type, the similarity depends on numerical differences, and the smaller the numerical difference between two features is, the larger the similarity. Suppose that the *k*th feature of the vectors F_{base} and F_i are both of "numerical" type, a naive calculation strategy for the similarity between the target and benchmark feature would be expressed as Eq (9).

$$Similarity\left(f_{base,k}, f_{i,k}\right) = \begin{cases} 1, & if\ f_{base,k} = f_{i,k} \\ \dfrac{1}{\left|f_{base,k} - f_{i,k}\right|}, & if\ f_{base,k} \neq f_{i,k} \end{cases} \tag{9}$$

Note that the impact of $\left|f_{base,k} - f_{i,k}\right|$ is relevant to the absolute value of $f_{base,k}$ or $f_{i,k}$, so Eq (9) should be revised as Eq (10).

$$Similarity\left(f_{base,k}, f_{i,k}\right) = 1 - \frac{\left|f_{base,k} - f_{i,k}\right|}{\max\left(f_{base,k}, f_{i,k}\right)} \tag{10}$$

Practically, values of inherent features of the "numerical" type are acquired by measurement. Due to the precision of measurement tools and human errors during the measurement process, there might be a deviation between the real value and measured value of a feature. So, we introduce an error tolerance ε into Eq (10) and obtain Eq (2).

C. Interval

For inherent features of the "interval" type, the similarity is determined by the relationship between intervals. If the interval value of the benchmark feature is the same as the one of the target feature, the benchmark feature and the target feature are considered the same, and vice versa. Thus, suppose that the *k*th feature of the inherent feature vectors F_{base} and F_i are both of "interval" type, a naive calculation strategy for the similarity between the target feature $f_{i,k}$ and the benchmark feature $f_{base,k}$ would be described as Eq (11).

$$Similarity\left(f_{base,k}, f_{i,k}\right) = \begin{cases} 1, & if\ f_{i,k} = f_{base,k} \\ 0, & if\ f_{i,k} \neq f_{base,k} \end{cases} \tag{11}$$

Practically, the interval of a benchmark feature is usually complete, and the interval of a target feature is often a subset of the interval of the benchmark feature, so it might be a fault to calculate similarity using Eq (11). We think it's reasonable to use Eq (3) for evaluating the similarity of features of the "interval" type. When a target feature interval is a subset of the benchmark feature interval, the two features are similar, and vice versa. Note that when an interval value is numeric (denoted by *a*), the numeric value *a* should be regarded as an interval [*a*, *a*].

D. Collection

Features of the "collection" type contain multiple elements, and the similarity between features can be determined by the inclusion relationship between collections, or by cosine similarity between vectors after collection vectorization.

The calculation strategy of cosine similarity is considered to be more accurate than the strategy of inclusion relationship. Taking the "image resolution" feature as an example, suppose that the benchmark and target features are $f_{base} = \{a \times b, c \times d\}$, $f_1 = \{a \times b\}$ and $f_2 = \{a \times b, c \times d\}$, when using calculation strategy of inclusion relationship, the similarity values of f_1 vs. f_{base} and f_2 vs. f_{base} are both 1. however, when using calculation strategy of cosine similarity, the similarity value of f_1 vs. f_{base} is 0.5 and value of f_2 vs. f_{base} is 1. Obviously, the calculation strategy of cosine similarity gets a smaller error and is more reasonable.

However, "interval type" features are difficult to vectorize due to the continuous elements, so it is not suitable to use the cosine similarity to measure feature similarity. In contrast, the elements in the "collection type" feature are discrete, and the vectorization process is simple. It is undoubtedly a more reasonable choice to construct an FSCS based on the cosine similarity method with smaller errors. The above analysis shows that the inherent FSCS adopted in this method is reasonable.

4. Experiment

In this section, we conduct two experiments: 1) when the feature sets are complete, a device type identification experiment verifies the feasibility of the method; 2) when features are missing, a device type identification experiment verify the effectiveness of the method.

4.1. Dataset

We select 40 types of fine-grained devices each from Dahua and Hikvision camera products as the knowledge set of the experiment. The specific device types are shown in Table 2, while each device has a number for the convenience of recording and presentation.

Table 2. Device list.

No.	Device model	No.	Device model
T01	Dahua DH-IPC-HDBW1020R	T21	Hikvision DC-2CD2T45
T02	Dahua DH-IPC-HDBW2130R-AS	T22	Hikvision DS-2CD1225
T03	Dahua DH-IPC-HDBW2230R-AS	T23	Hikvision DS-2CD1311D
T04	Dahua DH-IPC-HDBW4636R-AS	T24	Hikvision DS-2CD3310F-I
T05	Dahua DH-IPC-HDBW4833R-ZAS	T25	Hikvision DS-2CD3345P1
T06	Dahua DH-IPC-HDPW4233-PT-SA-0360B	T26	Hikvision DS-2CD3410FD-IW
T07	Dahua DH-IPC-HFW4433F-ZAS	T27	Hikvision DS-2CD3646F
T08	Dahua DH-IPC-HFW4631K-AS	T28	Hikvision DS-2CD3726F
T09	Dahua DH-IPC-HFW8841K-ZRL-DS	T29	Hikvision DS-2CD3935FWD-IWS
T10	Dahua DH-IPC-PDBW4638-B270	T30	Hikvision DS-2CD3942F-I
T11	Dahua DH-CA-DW48	T31	Hikvision DS-2CD3A20F-IS
T12	Dahua DH-CA-FW19M-IR8	T32	Hikvision DS-2CD3T25-I5
T13	Dahua DH-HAC-HDW2208E	T33	Hikvision DS-2CD3T56WD-I5
T14	Dahua DH-IPC-EW4431-ASW	T34	Hikvision DS-IPC-E22H-IW
T15	Dahua DH-IPC-HDBW4243E-ZFD	T35	Hikvision DS-IPC-T12-I
T16	ACTi ACM-3001	T36	TP-LINK TL-IPC223
T17	ACTi ACM-5001	T37	TP-LINK TL-IPC313K-W10
T18	ACTi ACM-3511	T38	TP-LINK TL-IPC43KZ
T19	ACTi ACM-5601	T39	TP-LINK TL-IPC546H
T20	ACTi ACM-7411	T40	TP-LINK TL-IPC646P-A4

For each device, the device shape (f_1), size (f_2), weight (f_3), imaging component (f_4), resolution (f_5), minimum illumination (f_6), lens parameters (f_7), compression code rate (f_8), electronic shutter time (f_9) and ambient temperature (f_{10}) are inherent features for device identification. The different inherent characteristic values of the different device types are shown in Table 3

Table 3. Different intrinsic feature values of different fine-grained device types.

No.	f_1	f_2 (mm)	f_3 (g)	f_4 (inch)	f_5	f_6 (Lux)	f_7	f_8 (bps)	f_9 (s)	f_{10} (℃)
T01	Dome	122×89	400	CMOS, 1/4.0	{704×576;1280×720}	0.4;0.22;	{6,F2.6,37;3.6,F2.5,59;2.8,F2.5,70.5}	5K->5M	1/3->1/10^{-5}	-30->60
T02	Dome	122×89	462	CMOS, 1/3.0	{704×576;1280×960;1280×720;704×480}	0.01;0.001;0	{8,F2.5,35.5;6,F2.1,47;3.6,F2.0,72.2;2.8,F2.0,91.7}	12K->6M	1/3->1/10^{-5}	-40->60
T03	Dome	122×89	406	CMOS, 1/2.7	{704×576;1920×1080;704×480}	0.01;0.001;0	{8,F2.2,42;6,F2.0,55;3.6,F2.0,90;2.8,F2.0,115}	6K->8M	1/3->1/10^{-5}	-40->60
T04	Dome	122×89	444	CMOS, 1/2.9	{2688×1520;704×576;2592×1944;1280×720;3072×2048;704×480;2560×1440}	0.002;0.0002;0	{6,F2.5,47.34;3.6,F2.2,70;2.8,F2.0,99}	73K->10M	1/3->1/10^{-5}	-30->60
T05	Dome	122×88.9	494	CMOS, 1/1.8	{704×576;1920×1080;3840×2160;704×480}	0.002;0.0002;0	{12,F2.8,45.5;3.5,F1.9,110}	16K->14.75M	1/3->1/10^{-5}	-40->55
T06	Dome	124.6×82.3	265	CMOS, 1/2.8	{704×576;1920×1080;1920×10801;704×480}	0.002;0.0002;0	{3.6,F1.6,87}	12K->10M	1/3->1/10^{-5}	-20->50
T07	Bullet	186×87×85	772	CMOS, 1/3.0	{704×576;1280×720;704×480;2592×1520;2560×1440}	0.001;0.0001;0	{13.5,F3.13,27.7;2.7,F1.6,104.2}	8K->10M	1/3->1/10^{-5}	-30->60
T08	Bullet	194×96×89	640	CMOS, 1/2.9	{2688×1520;704×576;2592×1944;1280×720;3072×2048;704×480;2560×1440}	0.002;0.0002;0	{6,F2.5,47.34;3.6,F2.2,70;2.8,F2.0,99}	73K->10M	1/3->1/10^{-5}	-40->60
T09	Bullet	221×109×101	1080	CMOS, 1/1.8	{704×576;1920×1080;3840×2160;1704×576;704×480}	0.01;0.001;0	{2.7,F1.2,111}	32K->8M	1/3->1/10^{-5}	-30->60
T10	Dome	285.1×100.8	2600	CMOS, 1/2.8	{704×576;1920×1080;1280×720;1280×960;704×480}	0.002;0.0002;0	{12,F2.7,44;2.7,F1.8,105}	16K->8M	1/3->1/10^{-5}	-30->60
T11	Dome	113.6×85.4	300	CCD, 1/1.3	{976×582}	0.001;;0	{2.8,F1.2,0}	16K->8M	1/50->1/10^{-5}	-30->60
T12	Bullet	194.4×96.6×89.5	400	CMOS, 1/3.0	{1280×720}	0.01;;	{3.6,F1.2,0}	16K->8M	1/25->1/10^{-5}	-30->60
T13	Dome	110×95	440	CMOS, 1/2.8	{1920×1080;1280×720}	0.001;;	{3.6, F1.2,0}	16K->8M	1/1->1/30000	-30->60

Continued on next page

No.	f_1	f_2 (mm)	f_3 (g)	f_4 (inch)	f_5	f_6 (Lux)	f_7	f_8 (bps)	f_9 (s)	f_{10} (℃)
T14	Dome	126×37.7	500	CMOS, 1/3.0	{2688×1520;704×576;1920×1080;1280×720}	0.001;;	{1.6, F1.4,180}	14K->40M	1/3->1/10^{-5}	-10->50
T15	Dome	159.1×117.9	925	CMOS, 1/2.8	{704×576;1920×1080;704×480}	0.002;0.0002;0	{35,F1.8,13;13.5,F3.05,31;7,F1.4,36;2.7,F1.6,110}	16K->8M	1/3->1/10^{-5}	-40->60
T16	Dome	130×99	350	CMOS, 1/3.0	{320×240;160×120;640×480}	;;	{4.2,F1.8,0}		1/5->1/2000	0->40
T17	Bullet	67×55	400	CMOS, 1/3.0	{320×240;160×120;640×480}	;;	{4.2,F1.8,0}		1/5->1/2000	0->50
T18	Dome	130×99	380	CMOS, 1/3.0	{320×240;160×112;640×480;1280×1024;1280×720}	;;	{3.3,F1.6,0}		1/5->1/2000	-10->45
T19	Bullet	67×55×129.5	400	CMOS, 1/3.0	{640×480;1280×720;1280×1024}	;;	{4.2,F1.8,0}		1/5->1/2000	0->50
T20	Dome	151.69×114.9	1040	CMOS, 1/3.0	{640×480;1280×720;1280×1024}	;;	{3.3,F1.4,0}		1/10->1/2000	-30->50
T21	Barrel	90×85×169	550	CMOS, 1/3.0	{2560×1440}	0.1;;	{1.68,F2.0,180}	32K->16M	1/3->1/10^{-5}	-30->60
T22	Bullet	100.5×88.1×157.3	600	CMOS, 1/4.0	{1280×720}	0.01;;	{12,F1.2,26;8,F1.2,39;6,F1.2,50;4,F1.2,76;2.8,F1.2,95}	32K->8M	1/25->1/10^{-5}	-30->60
T23	Dome	127×97.5	570	CMOS, 1/3.0	{1280×960}	0.01;;	{16,F1.2,18;12,F1.2,22;8,F1.2,33;6,F1.2,50;4,F1.2,69;2.8,F1.2,90}	32K->16M	1/25->1/10^{-5}	-30->60
T24	Dome	114.6×89.4	670	CMOS, 1/3.0	{1280×960}	0.01;;	{4,F2.0,75.8}	32K->8M	1/3->1/10^{-5}	-30->60
T25	Dome	127.3×103.7	340	CMOS, 1/3.0	{2560×1440}	0.01;;	{4,F1.2,69;2.8,F1.2,90}	32K->16M	1/3->1/10^{-5}	-10->40
T26	Bullet	66×139.1×70.6	400	CCD, 1/3.0	{1280×920}	0.02;;	{4,F2.0,75.8}	32K->16M	1/25->1/10^{-5}	-25->60
T27	Barrel	191.4×97.9×93.5	1260	CMOS, 1/2.7	{2560×1440}	0.005;;	{2.8,F1.2,105}	32K->8M	1/3->1/10^{-5}	-30->60

Continued on next page

No.	f_1	f_2 (mm)	f_3 (g)	f_4 (inch)	f_5	f_6 (Lux)	f_7	f_8 (bps)	f_9 (s)	f_{10} (℃)
T28	Barrel	153.3×153.3×111.6	840	CMOS, 1/2.7	{1920×1080}	0.002;;	{2.7,F1.2,103}	32K->8M	1/3->1/10^{-5}	-30->60
T29	Dome	119.9×41.2	600	CMOS, 1/2.8	{1600×1200;1280×960;2048×1536}	0.005;;0	{1.16,F2.2,180}	32K->16M	1/3->1/10^{-5}	-10->40
T30	Dome	119.9×41.2	600	CMOS, 1/3.0	{2048×1536}	0.01;;0	{1.6,F1.6,186}	32K->8M	1/25->1/10^{-5}	-10->40
T31	Barrel	134×116.1×293	2000	CMOS, 1/2.8	{1920×1080}	0.01;;0	{25,F1.2,12.4;16,F1.2,19.2;12,F1.2,24.6;8,F1.2,40;6,F1.2,52;4,F1.2,85}	32K->8M	1/3->1/10^{-5}	-30->60
T32	Bullet	194.04×93.85×89.52	1000	CMOS, 1/2.7	{1920×1080;1280×960;1296×732;1280×720}	0.01;;	{4,F1.2,80}	32K->8M	1/3->1/10^{-5}	-30->60
T33	Bullet	93.85×93.52×194.1	690	CMOS, 1/2.7	{704×576;2560×1920;640×480;1280×720;2560×1536;1920×1280;352×288}	0.005;;	{6,F1.2,55;4,F1.2,83}	32K->8M	1/3->1/10^{-5}	-30->60
T34	Barrel	175×89×75	340	CMOS, 1/2.7	{1920×1080}	0.01;;	{8,F1.2,43;6,F1.2,54.4;4,F1.2,89.1;2.8,F1.2,114.8}	32K->8M	1/3->1/10^{-5}	-30->50
T35	Dome	110×93.2	350	CMOS, 1/2.7	{1920×1080}	0.01;;	{8,F1.2,43;6,F1.2,54.5;4,F1.2,91;2.8,F1.2,114.8}	32K->8M	1/3->1/10^{-5}	-30->60
T36	Dome	113×113×87	268	CMOS, 1/2.7	{1920×1080}	0.1;0.1;0	{4,F2.1,0;6,F2.1,0}	64K->8M	1/25->1/10^{-5}	-30->60
T37	Barrel	173×83.4×84.2	390	CMOS, 1/3.0	{1280×960}	0.1;0.1;0	{2.8,F2.1,0;4,F2.1,0;6,F2.1,0}	64K->4M	1/25->1/10^{-5}	-10->60
T38	Dome	120×120×129	328	CMOS, 1/2.7	{2304×1296}	0.1;;0	{3.35,F2.4,0}	64K->2M	1/25->1/10^{-5}	-30->50
T39	Barrel	173×83.4×84.2	390	CMOS, 1/2.7	{2560×1440}	0.01;;	{4,F1.6,0;6,F1.6,0;8,F1.6,0;12,F1.6,0}	256K->6M	1/25->1/10^{-5}	-30->60
T40	Ball	230×177×188	930	CMOS, 1/3.0	{2560×1440}	0.002;;0	{4,F1.6,0}	64K->6M	1/25->1/10^{-5}	-30->60

In Table 3, f_1 and f_4 are "phrase type" inherent features; f_2, f_3, and f_6 are "numerical value" inherent features; f_8, f_9 and f_{10} are "interval type" inherent features; and f_5 and f_7 are "collective type" inherent features. The elements in f_6 represent the minimum illumination of the color camera, black-and-white camera, and infrared camera. For example, in the f_6 value of T01, "0.4; 0.22; -" indicates that the lowest illuminance of the T01 color camera is 0.4 lux, the minimum illuminance of the black-and-white camera is 0.22 lux, and the infrared camera is not supported. Each set element of f_7 includes the focal length, aperture, and horizontal field of view. For example, in the f_7 value of T01, "6, F2.6, 37" indicates that the focal length is 6 mm, the aperture is F2.6, and the horizontal field of view is 37 degrees.

From Eqs (5) and (6), we calculate the weight of each feature separately, and the results are $w_1 = 0.09089$, $w_2 = 0.11481$, $w_3 = 0.11294$, $w_4 = 0.09217$, $w_5 = 0.11061$, $w_6 = 0.10381$, $w_7 = 0.11596$, $w_8 = 0.09915$, $w_9 = 0.07682$, and $w_{10} = 0.08285$. In Sections 4.2 and 4.3, we use the calculated feature weights.

4.2. Experiment on complete feature set

To verify the effectiveness of identifying device types using the inherent features, we carry out a device identification experiment with each feature set complete and calculate the similarity between any two devices.

We use the feature weights calculated in Section 4.1, set the error tolerance $\varepsilon=1$ for the "numerical type" features, and calculate the difference between any two devices in {T01, T02..., T20} using Eq (7). Figure 2 shows the confusion matrix of the device similarity.

In Figure 2, "True Type" represents the actual fine-grained type of the device, and "Basic Type" represents the basic type. When "True Type" is "T02" and "Basic Type" is "T01", the similarity is 0.492, which means that the similarity between T02 and T01 is 0.492 in other words, $Similarity\left(F_{base,1}, F_{target,2}\right) = 0.492$.

By using the 10 inherent characteristics described in Section 4.1, Figure 2 shows that the similarity between any two devices is less than 0.85 (the maximum value is 0.842), which can effectively distinguish the devices, in other words, it proves the effectiveness of inherent features in distinguishing devices type. The areas with greater similarity are T1-T15, T16-T20, T21-T25, T36-T40, which indicates that the inherent features of the devices from the same manufacturer are more similar than the devices from different manufacturers.

4.3. Experiment on incomplete features

Our experiment simulates a situation with incomplete device features in an actual environment. We discard some features or elements of collective features with a certain probability and accomplish fine-grained device identification with incomplete features.

The discarding probability of an inherent feature changes from 0 to 1 in increments of 0.01. At each discarding probability value, the target set with inherent missing features is generated using the knowledge set, and the similarity between each feature in the target set and the knowledge set is calculated separately, and the greatest similar device type is taken as the target device type. To ensure arbitrary discarding of probability values, the experiment is repeated 1000 times to obtain the average identification accuracy of each type of device.

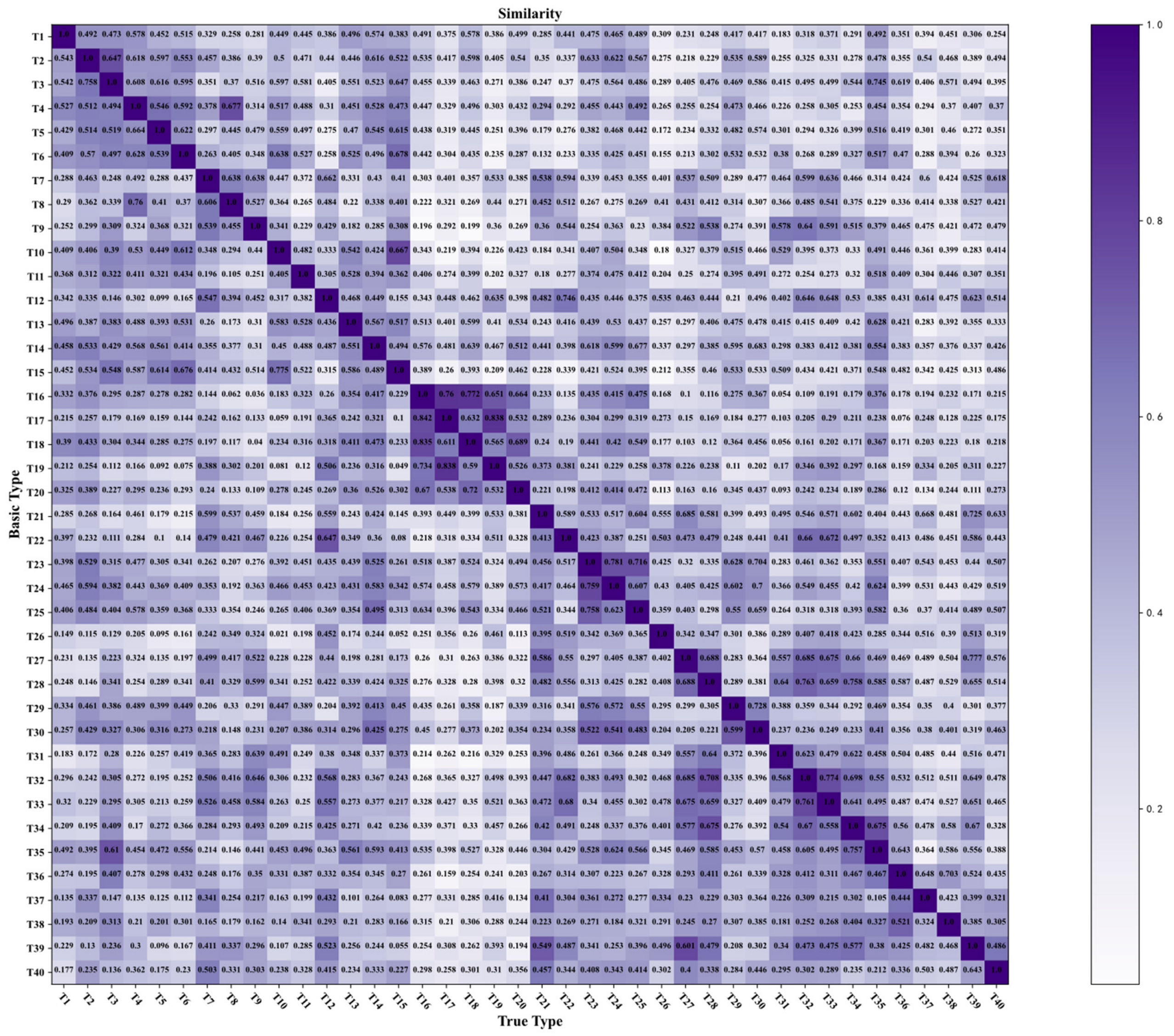

Figure 2. Confusion matrix of device similarities.

The increase of feature discard probability makes the similarity value between fine-grained device types decrease, which is because when a feature value of the target device is empty (i.e., no feature is acquired), the feature similarity between the target device and the knowledge device will be 0. When the target device has fewer non-empty feature values, the similarity value between it and the knowledge set device type will be less. Therefore, we set a similarity threshold in the experiment. When the maximum similarity is less than the threshold, device identification is considered to fail (even if the type corresponding to the maximum similarity is the correct type). When the threshold value is set to 0.5, the graph of the variation between the average recognition accuracy and the discard probability of each type of device is obtained as shown in Figure 3.

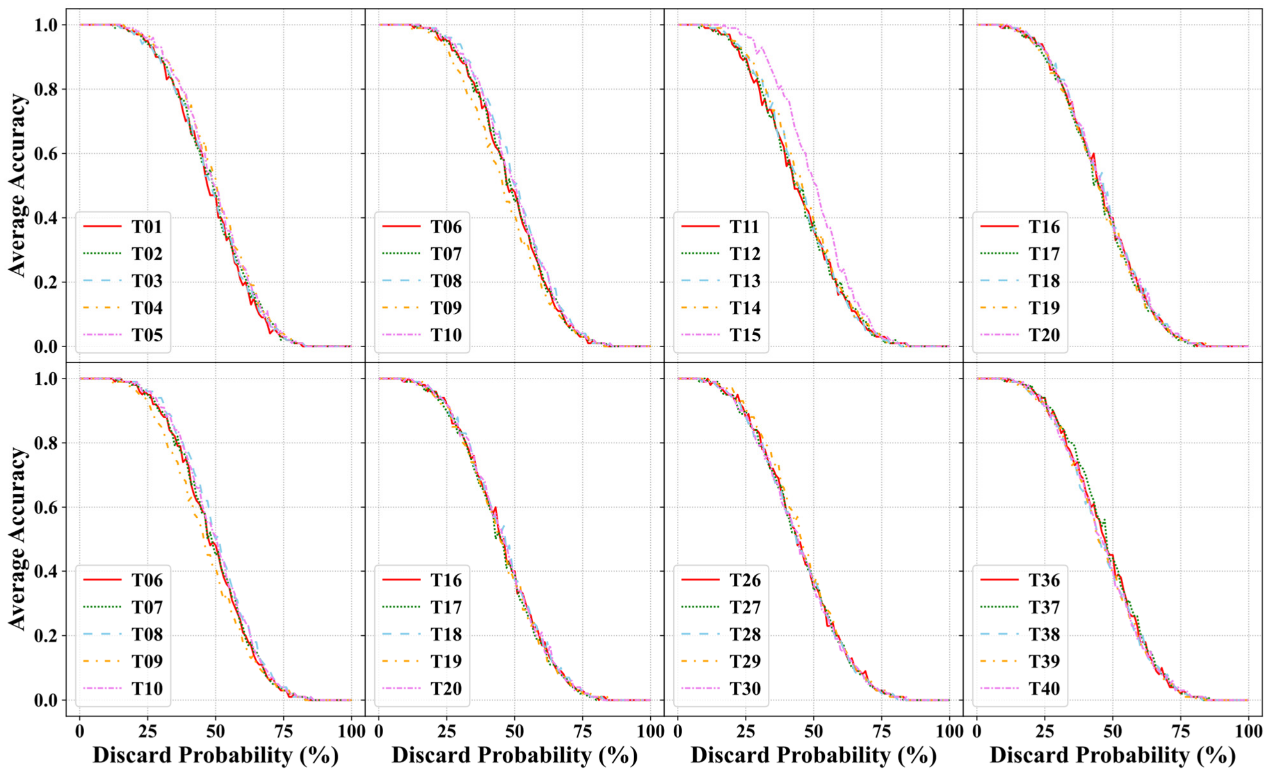

Figure 3. When the threshold value is 0.5, the relationship between the average identification accuracy and discarding probability.

In Figure 3, when the discard probability is less than 0.25, the average identification accuracy of the fine-grained type of the device is greater than 90%. When the discard probability is greater than 0.2, the average identification accuracy decreases rapidly with the increase of discard probability. When the discard probability is 0.4, the average identification accuracy is approximately 60%. When the discard probability is 0.8, the average identification accuracy is close to 0.

An increase in the discard probability inevitably leads to a decrease in the similarity value. While the similarity value of the correct device type decreases, the similarity value between other devices and the target device will also decrease. The strategy of setting the similarity threshold will directly lead to the failure of fine-grained type identification of the device when the discarding probability is large. Figure 4 shows the similarity matrix between the target set devices and the knowledge set devices during the 500th experiment when the discard probability is 0.5.

Figure 4 shows that when the discarding probability of an inherent feature is 0.5, the maximum similarity value (i.e., T10, T19, etc.) is less than 0.5. However, the device type with the maximum similarity value is the correct device type. Therefore, when a feature is missing, a similarity threshold should not be set (or the threshold should be set to 0).

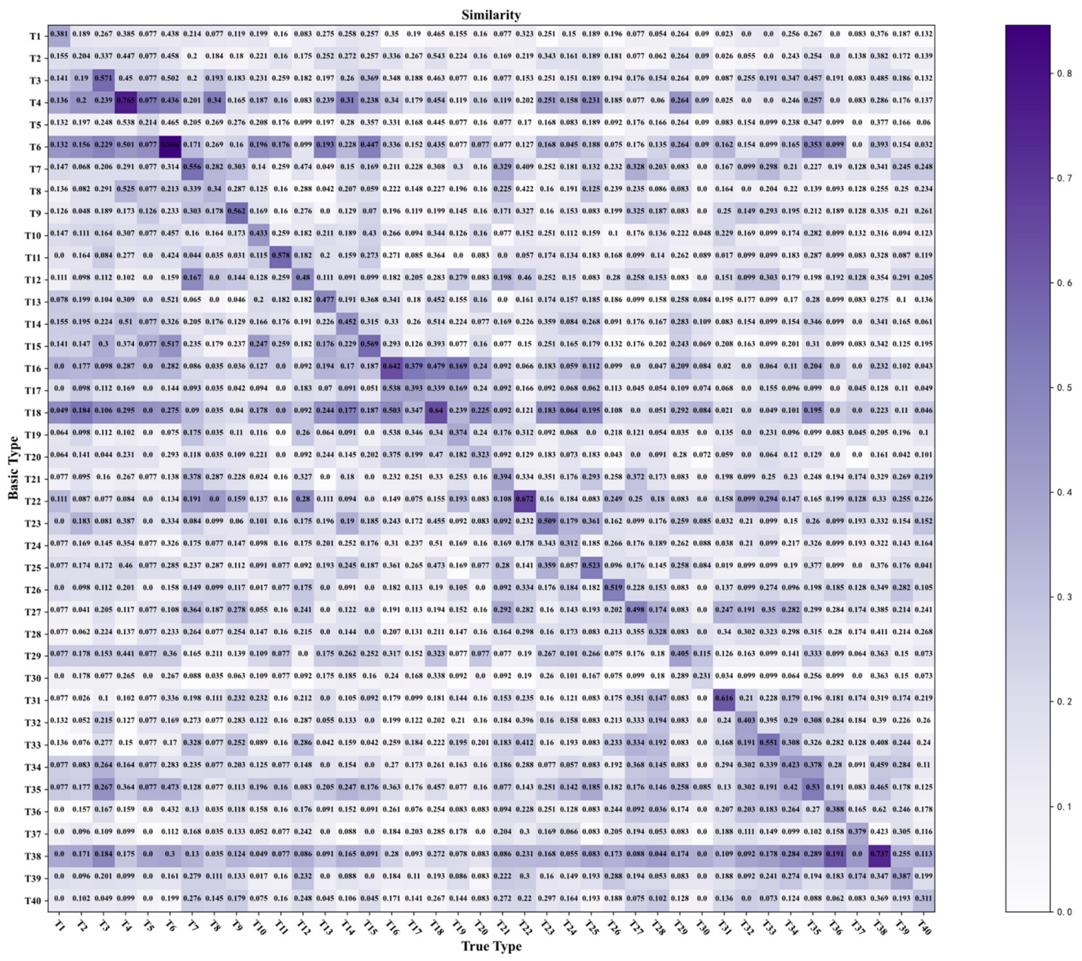

Figure 4. When the discarding probability is 0.5, the 500th result of the device similarity experiment between the target set and the knowledge set.

Figure 5 shows that when there is no similarity threshold, the average identification accuracy of each device fine-grained type is still greater than 80% even if the discarding rate of inherent features is 50%. When the discard rate is 0.8, the average identification accuracy of each device is greater than 35%. This shows that the method proposed in this paper can still identify the fine-grained types of devices effectively when some inherent features are missing.

In this section, the experimental results based on real data show that the method proposed in this paper to identify the fine-grained type of device based on inherent features is feasible and effective. In reality, the more inherent features of the target device are obtained, the more accurate the identification of the target device will be. When the "missing rate" of inherent features is 50%, the average accuracy to identify the fine-grained type of device still exceeds 80%.

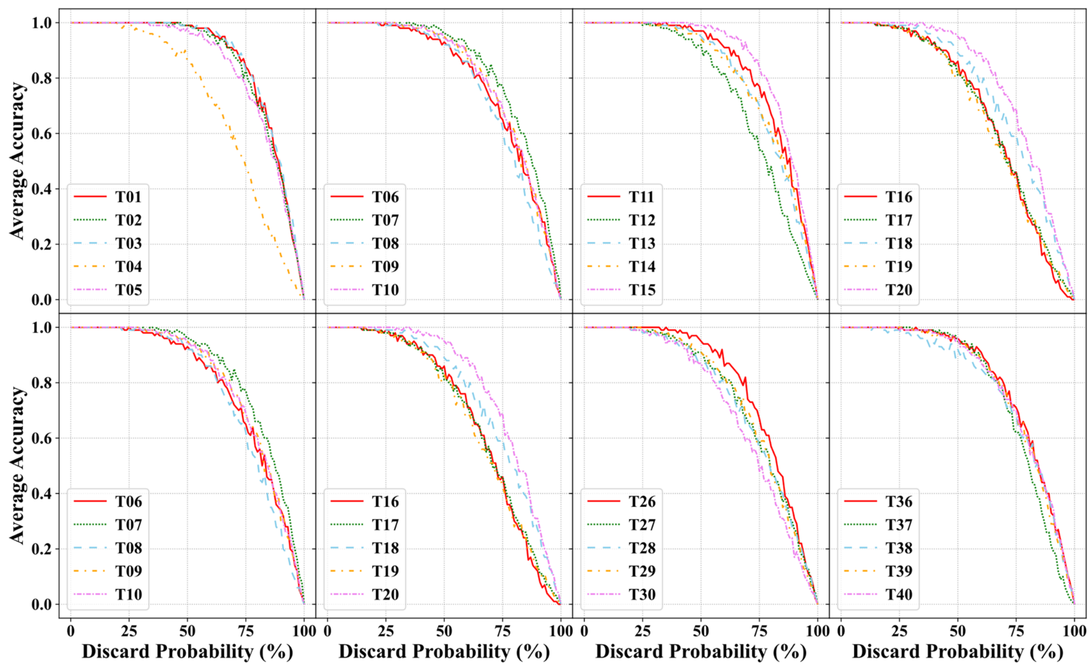

Figure 5. When there is no similarity threshold, the relationship between the average identification accuracy and the discarding probability.

5. Conclusions

This paper develops an inherent feature-based device identification method to solve the current deficiency in fine-grained identification of types of camera devices. The method classifies devices inherent features into 4 types according to their expressive form, establishes a similarity calculation strategy (FSCS) for each type of inherent feature, assigns a weight value derived from information entropy for each inherent feature, and constructs a fine-grained device type identification model combining the FSCS and weight value. Experiment results show that the method proposed in this paper can recognize fine-grained types of devices in the partial absence of inherent features. Especially, when the “absence rate” of inherent features is 50%, the average accuracy is still more than 80%. In future work, finding more effective inherent features and combining them with network features may be an important direction to improve the accuracy of fine-grained device type identification.

Acknowledgments

We would like to thank the reviewers for their positive suggestions, which helped to improve the paper enormously.

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. R. Li, R. Xu, Y. Ma, X. Luo, LandmarkMiner: Street-level network landmarks mining method for IP geolocation, *ACM Trans. Internet Things,* **2** (2021), 1–22. https://doi.org/10.1145/3457409
2. V. F. Taylor, R. Spolaor, M. Conti, I. Martinovic, Robust smartphone app identification via encrypted network traffic analysis, *IEEE Trans. Inf. Forensics Secur.,* **13** (2017), 63–78. https://doi.org/10.1109/TIFS.2017.2737970
3. N. G. B. Amma, S. Selvakumar, R. L. Velusamy, A statistical approach for detection of denial of service attacks in computer networks, *IEEE Trans. Network Service Manage.*, **17** (2020), 2511–2522. https://doi.org/10.1109/TNSM.2020.3022799
4. D. Goldman, Shodan: The scariest search engine on the Internet, 2013. Available from: https://money.cnn.com/2013/04/08/technology/security/shodan/.
5. G. F. Lyon, *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*, Nmap Project, (2009), 5–12.
6. Z. Durumeric, E. Wustrow, J. A. Halderman, ZMap: Fast Internet-wide scanning and its security applications, *USENIX Secur. Symp.*, (2013), 605–620. https://dl.acm.org/doi/10.5555/2534766.2534818
7. Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, J. A. Halderman, A search engine backed by internet-wide scanning, in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security,* (2015), 542–553. https://doi.org/10.1145/2810103.2813703
8. R. Li, M. Shen, H. Yu, C. Li, P. Duan, L. Zhu, A survey on cyberspace search engines, in *China Cyber Security Annual Conference*, Springer, Singapore, (2020), 206–214. https://doi.org/10.1007/978-981-33-4922-3_15
9. Y. Meidan, M. Bohadana, A. Shabtai, J. Guarnizo, ProfilIoT: A machine learning approach for IoT device identification based on network traffic analysis, in *Proceedings of the 32nd ACM Symposium on Applied Computing*, (2017), 506–509. https://doi.org/10.1145/3019612.3019878
10. M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A. Sadeghi, 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), 2511–2514. https://doi.org/10.1109/ICDCS.2017.284
11. A. Sivanathan, D. Sherratt, H. H. Gharakheili, A. Radford, C. Wijenayake, A. Vishwanath, et al., Characterizing and classifying IoT traffic in smart cities and campuses, in *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS*), (2017), 559–564. https://doi.org/10.1109/INFCOMW.2017.8116438
12. W. Cheng, Z. Ding, C. Xu, X. Wu, Y. Xia, J. Mao, RAFM: A real-time auto detecting and fingerprinting method for IoT devices, in *Journal of Physics: Conference Series*, **1518** (2020), 12043–12050. https://dx.doi.org/10.1088/1742-6596/1518/1/012043
13. D. E. Rumelhart, G. E. Hinton, R. J. Williams, Learning representations by back propagating errors, *Nature,* **323** (1986), 533–536. https://doi.org/10.1038/323533a0
14. C. Cortes, V. Vapnik, Support-vector networks, *Mach. Learn.*, **20** (1995), 273–297. https://doi.org/10.1023/A:1022627411411
15. T. Hastie, R. Tibshirani, Discriminant adaptive nearest neighbor classification, *IEEE Trans. Pattern Anal. Mach. Intell.*, **18** (1996), 607–616. https://doi.org/10.1109/34.506411

16. K. Yang, Q. Li, L. Sun, Towards automatic fingerprinting of IoT devices in the cyberspace, *Comput. Networks,* **148** (2019), 318–327. https://doi.org/10.1016/j.comnet.2018.11.013
17. A. Sivanathan, H. H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath, et al., Classifying IoT devices in smart environments using network traffic characteristics, *IEEE Trans. Mobile Comput.*, **18** (2019), 1745–1759. https://doi.org/10.1109/TMC.2018.2866249
18. S. V. Radhakrishnan, A. S. Uluagac, R. Beyah, GTID: A technique for physical device and device type fingerprinting, *IEEE Trans. Dependable Secure Comput.*, **12** (2015), 519–532. https://doi.org/10.1109/TDSC.2014.2369033
19. Y. Zou, S. Liu, N. Yu, H. Zhu, L. Sun, H. Li, et al., IoT device recognition framework based on web search, *J. Cyber Secur.*, **3** (2018), 25–40. https://doi.org/10.19363/J.cnki.cn10-1380/tn.2018.07.03
20. X. Feng, Q. Li, H. Wang, L. Sun, Acquisitional rule-based engine for discovering internet-of-things devices, in *Proceedings of the 27th USENIX Security Symposium*, **148** (2018), 327–341. https://dl.acm.org/doi/10.5555/3277203.3277228
21. S. Agarwal, P. Oser, S. Lueders, Detecting IoT devices and how they put large heterogeneous networks at security risk, *Sensor*, **19** (2019), 4107–4107. https://doi.org/10.3390/s19194107
22. T. Kohno, A. Broido, K. C. Claffy, Remote physical device fingerprinting, *IEEE Trans. Dependable Secure Comput.*, **2** (2005), 93–108. https://doi.org/10.1109/TDSC.2005.26
23. S. J. Murdoch, Hot or not: Revealing hidden services by their clock skew, in *Proceedings of the 13th ACM Conference on Computer and Communications Security*, (2006), 27–36. https://doi.org/10.1145/1180405.1180410
24. S. Zander, S. J. Murdoch, An improved clock-skew measurement technique for revealing hidden services, in *Proceedings of the 17th USENIX Security Symposium*, (2008), 211–225. https://dl.acm.org/doi/10.5555/1496711.1496726
25. D. J. Huang, K. T. Yang, C. C. Ni, W. C. Teng, T. R. Hsiang, Y. J. Lee, Clock skew based client device identification in cloud environments, in *Proceedings of the 26th IEEE International Conference on Advanced Information Networking and Applications*, (2012), 526–533. https://doi.org/10.1109/AINA.2012.51
26. Y. Vanaubel, J. J. Pansiot, P. Merindol, B. Donnet, Network fingerprinting: TTL-based router signatures, in *Proceedings of the 2013 Conference on Internet Measurement Conference*, (2013), 369–376. https://doi.org/10.1145/2504730.2504761
27. T. Kohno, A. Broido, K. C. Claffy, Remote physical device fingerprinting, *IEEE Trans. Dependable Secure Comput.*, **2** (2005), 93–108. https://doi.org/10.1109/SP.2005.18