sensors

MDPI

Article

A Novel Framework for Open-Set Authentication of Internet of Things Using Limited Devices

Keju Huang *, Junan Yang, Pengjiang Hu and Hui Liu

College of Electronic Engineering, National University of Defense Technology, Hefei 230037, China; yangjunan@ustc.edu (J.Y.); pjhu12@nudt.edu.cn (P.H.); liuhui17c@nudt.edu.cn (H.L.)

Abstract: The Internet of Things (IoT) is promising to transform a wide range of fields. However, the open nature of IoT makes it exposed to cybersecurity threats, among which identity spoofing is a typical example. Physical layer authentication, which identifies IoT devices based on the physical layer characteristics of signals, serves as an effective way to counteract identity spoofing. In this paper, we propose a deep learning-based framework for the open-set authentication of IoT devices. Specifically, additive angular margin softmax (AAMSoftmax) was utilized to enhance the discriminability of learned features and a modified OpenMAX classifier was employed to adaptively identify authorized devices and distinguish unauthorized ones. The experimental results for both simulated data and real ADS–B (Automatic Dependent Surveillance–Broadcast) data indicate that our framework achieved superior performance compared to current approaches, especially when the number of devices used for training is limited.

Keywords: Internet of Things; cybersecurity; physical layer identification; deep learning; open-set classification

Citation: Huang, K.; Yang, J.; Hu, P.; Liu H. A Novel Framework for Open-Set Authentication of Internet of Things Using Limited Devices. *Sensors* **2022**, *22*, 2662. https://doi.org/10.3390/s22072662

Academic Editor: Nikos Fotiou

Received: 19 February 2022
Accepted: 29 March 2022
Published: 30 March 2022

1. Introduction

The Internet of Things (IoT), which enables communication and interaction between various devices, promises to transform a wide range of fields. IoT devices primarily transmit information via wireless means, the open nature of which exposes the IoT to cybersecurity threats [1]. One typical cybersecurity threat, identity spoofing, which refers to the action of assuming the identity of some other device, decreases the availability of resources and can be dangerous in critical infrastructures [2]. By using spoofing identities, attackers can gain unauthorized access to internal networks and interfere with communication between authorized devices, which threatens the security of the wireless network. Therefore, the network administrator must identify authorized IoT devices and reject connections from unauthorized devices (Figure 1).

To prevent identity proofing, physical layer authentication (PLA) [3], which is also known as non-cryptographic device identification (NDI) [4], identifies IoT devices based on the physical layer characteristics of their transmitted signals. The feasibility of PLA is based on the fact that the electronic circuits of devices possess specific imperfections that are determined by production and manufacturing processes. PLA is analogous to speaker recognition [5] in the sense that they both concern the characteristics of components that emit signals, regardless of the content of the signals. PLA serves as an effective tool against identity spoofing as it identifies devices using the physical layer characteristics of signals that stem from hardware imperfections, which cannot be counterfeited, in theory. Compared to cryptographic approaches for authentication, PLA does not require sophisticated key management procedures and is hard to deceive. Therefore, PLA has received a lot of attention in the past few years.

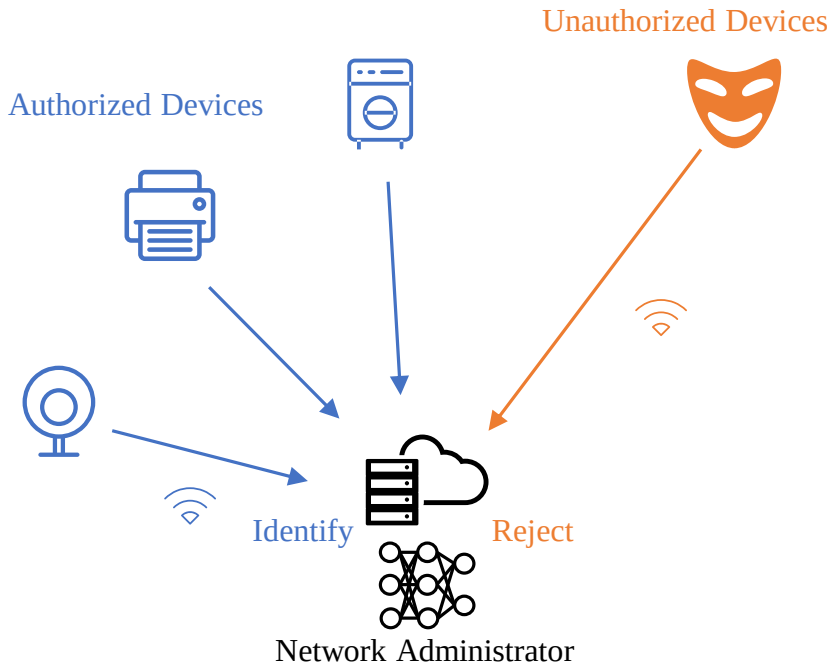


Figure 1. An example of device identification in the IoT.

Existing approaches to PLA can be divided into two categories: hand-crafted features-based approaches and deep learning-based approaches. The approaches that are based on hand-crafted features focus on extracting distinctive features from received signals using expert knowledge, such as I/Q quadrature modulation defects [6], statistics regarding time–frequency energy distribution [7–10], and complexity measurement [11,12]. These approaches require significant prior knowledge and achieve limited accuracy. In response, deep learning-based approaches, which learn to identify devices from data, have been extensively studied recently and have shown prominent advantages [13–18]. While most existing work has focused on classification among a closed set of known devices, the approaches that have been developed are impractical in the prevention of identity proofing since they may simply identify an unseen device as the most similar authorized device. Open-set authentication is more feasible in this scenario as it not only identifies known devices, but also rejects unseen transmitters. Although approaches for the open-set authentication of IoT devices have been proposed in the last two years [19–23], their performances still remain to be improved when the number of authorized devices for training is limited.

In this paper, we propose a novel deep learning framework for the accurate open-set authentication of IoT using limited authorized devices. The framework leverages the strengths of additive angular margin softmax (AAMSoftmax) [24], which promotes the discriminability of features that are learned by neural networks, and OpenMAX [25], which can effectively reject unknown classes. The codes are available at https://github.com/huangkeju/AAMSoftmaxOpenMax (accessed on 19 February 2022). The contributions of this paper are summarized as follows:

- We propose to adopt AAMSoftmax to enhance the discriminability of features that are learned by neural networks, so that the features of unseen devices are distributed away from those of authorized devices;
- We propose a modified OpenMAX method, namely adaptive class-wise OpenMAS, so that it can be combined with AAMSoftmax and unseen IoT devices can be distinguished adaptively based on the features that are learned by neural networks;
- We propose a framework that leverages the strengths of AAMSoftmax and OpenMAX for the open-set authentication of IoT devices. The evaluations of both simulated data and real ADS–B data show that the proposed framework was advantageous

for open-set IoT authentication, especially when the number of devices for training was limited.

The remainder of this paper is organized as follows. Works related to open-set IoT authentication are discussed in Section 2. The problem is formulated in Section 3 and the proposed framework is presented in Section 4. Section 5 presents the performance evaluation and Section 6 concludes the paper. The current literature comparison can be found in Table 1.

Table 1. A comparison to the current literature.

	(Loss Function of) Feature Extractor	Classifier
[19]	GAN	GAN
[20,23]	Angular Softmax	Distance-Based
[22]	Softmax	Disc, DClass, OvA, OpenMAX, Autoencoder
Our work	Additive Angular Margin Softmax	Adaptive Class-Wise OpenMAX

2. Related Works

2.1. Background

The open-set authentication of IoT devices has not been investigated until recently. Generally, the open-set authentication of IoT devices is achieved using a feature extractor and a classifier (Figure 2). The feature extractor is a neural network that outputs the features of the input signals. The feature extractor network is trained to learn features that are discriminative for different devices. The classifier builds boundaries for the learned features of each authorized device at the training stage, so that it can predict whether a sample belongs to an authorized device.

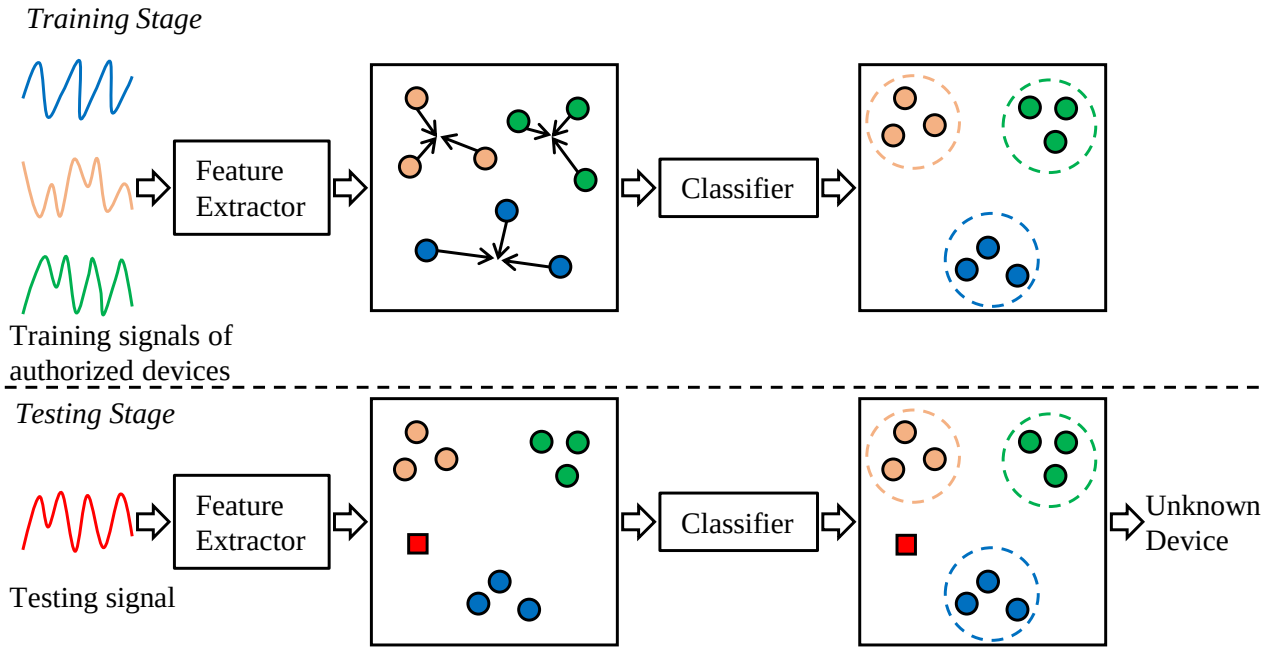


Figure 2. The open-set authentication of IoT devices.

2.2. Feature Extractor

In [19], a generative adversarial network (GAN) was utilized to identify rogue transmitters. However, training a GAN is not trivial and requires a large amount of data. In [23], hypersphere projection was used to learn more separable features in angular space, which was inspired by methods used in face recognition technology [26]. Similarly, the zero-bias layer proposed by [20] also projects features to hypersphere, but with an additional dense layer for faster convergence. In this paper, we propose to learn features not only in angular space, but also with an additive margin, which could further enhance the discriminability of features. The loss function that we adopted, i.e., AAMSoftmax, was first proposed in face recognition technology [24] and has also been applied in speaker recognition technology [27,28]. Moreover, neither [20,23] considered the classifier design for open-set authentication and simply distinguished unseen devices based on distance from authorized devices. However, the distance threshold to reject unseen transmitters can be hard to determine and may influence the final performance. Therefore, we propose to employ a modified OpenMAX classifier that can work with AAMSoftmax and adaptively distinguish unseen devices.

2.3. Open-Set Classifier

In [21], a novel approach was proposed for the outlier detection of Wi-Fi devices and ADS–B signals. A classifying neural network predicts slices of a packet and the statistics of those predictions are then compared to a threshold to determine whether the packet is from an unknown device. Compared to the approach that is based on this threshold, the OpenMAX employed in this paper used an activation vector to incorporate more information, which led to an improved performance [25]. In [22], five different open-set recognition approaches, namely a discriminator (Disc), discriminating classifier (DClass), one-vs.-all classifier (OvA), OpenMAX, and an autoencoder, were compared thoroughly using a Wi-Fi dataset. As Disc and DClass rely on known outlier sets for training, which require more data from additional emitters in practice, they were not considered in our evaluation. Autoencoders were also not considered because they do not make use of labels and usually lead to a moderate performance. Although OvA achieved a better performance than OpenMAX in the experiments of [22], we found that OvA fails when the unseen device is much more similar to one of the authorized devices than the others, which is common when training with limited authorized devices. By contrast, OpenMAX is less sensitive to the number of authorized devices and works more preferably in this setting, which is discussed in detail in Section 4. Furthermore, both [21,22] simply used softmax to train the neural network and did not consider ways to obtain more discriminative features in order to achieve a better performance.

3. Problem Definition

We considered a finite set of devices for authentication given by $\mathcal{A} = \{A_1, A_2, \ldots, A_K\}$, where K denotes the total number of devices. A subset of $\mathcal{A}$, denoted as $\mathcal{A}_\mathcal{T}$, represents the authorized devices. Without loss of generality, we assumed $\mathcal{A}_\mathcal{T} = \{A_1, A_2, \ldots, A_{K_T}\}$, where K_T denotes the number of devices for authorization and $K_T < K$. The dataset for training could be defined as $\mathcal{D}_\mathcal{T} = \{(\boldsymbol{x}_n, y_n)_{n=1}^N\}$, where N denotes the number of training samples. $\boldsymbol{x}_n$ represents the nth sample of the dataset, which contained L complex sampling points, i.e., $\boldsymbol{x}_n \in \mathbb{C}^L$. y_n is the corresponding identity of sample $\boldsymbol{x}_n$ and $y_n \in \{1, 2, \ldots, K_T\}$. A neural network with parameters Ω was trained using $\mathcal{D}_\mathcal{T}$ for open-set authentication. The neural network, which could be viewed as a mapping function $F_\Omega : \mathbb{C}^L \rightarrow \mathbb{R}^{K_T+1}$, mapped the signal $\boldsymbol{x}$ from any device of $\mathcal{A}$ to its prediction score $\boldsymbol{p}$. The ith element of $\boldsymbol{p}$, $p(i)$, indicates the probability that $\boldsymbol{x}$ is from A_i of authorized devices $\mathcal{A}_\mathcal{T}$ when $i <= K_T$ and represents the probability that $\boldsymbol{x}$ is from an unauthorized device when $i = K_T + 1$.

4. Proposed Framework

The neural network F_Ω could be viewed as the combination of a feature extractor G_ϕ and a classifier C_ψ, i.e., $F_\Omega(\boldsymbol{x}) = C_\psi(G_\phi(\boldsymbol{x}))$, where ϕ and ψ denote the parameters of the feature extractor and the classifier, respectively. The feature extractor $G_\phi : \mathbb{C}^L \to \mathbb{R}^M$ transformed signal $\boldsymbol{x}$ into a feature vector of dimension M, while the classifier $C_\psi : \mathbb{R}^M \to \mathbb{R}^{K_T+1}$ mapped the feature vector to the corresponding prediction score $\boldsymbol{p}$. In this paper, we propose to utilize AAMSoftmax to train the feature extractor and use a modified version of OpenMAX as the classifier. The whole training process was divided into two stages (Figure 3). In the first stage, the feature extractor was trained using the AAMSoftmax loss function without the classifier. In the second stage, the OpenMAX classifier was trained using the fixed feature extractor.

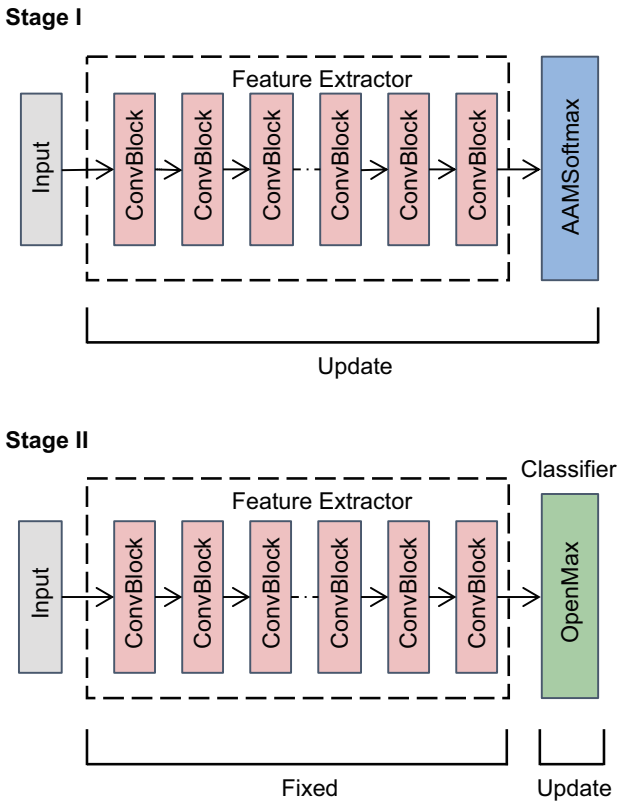


Figure 3. The training stages of the proposed framework. First, the feature extractor is trained by AAMSoftmax. Then, the OpenMAX classifier is updated with the fixed feature extractor.

4.1. Feature Extractor with AAMSoftmax

Typically, neural networks for classification are trained by minimizing the cross entropy between the prediction score and the true label:

$$\mathcal{L} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{K_T}\mathbb{I}(i = y_n)\ln p_n[i], \tag{1}$$

where $\mathbb{I}$ is the indicator function, $\boldsymbol{p}_n$ is the prediction score of sample $\boldsymbol{x}_n$, and $\boldsymbol{f}_n$, i.e., $\boldsymbol{f}_n = G_\phi(\boldsymbol{x}_n)$ denotes the feature vector of $\boldsymbol{x}_n$ extracted by G_ϕ. A vanilla approach to obtaining the prediction score based on $\boldsymbol{f}$ is to use a dense layer followed by the softmax function:

$$p_n[i] = \frac{e^{\boldsymbol{w}_i^\top \boldsymbol{f}_n}}{\sum_{j=1}^{K_T} e^{\boldsymbol{w}_j^\top \boldsymbol{f}_n}}, \tag{2}$$

where $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{K_T}] \in \mathbb{R}^{K_T \times M}$ are the weights of the dense layer. Combining Equations (1) and (2), the loss function with softmax becomes:

$$\mathcal{L}_{Softmax} = -\frac{1}{N}\sum_{n=1}^{N} \ln \frac{e^{\boldsymbol{w}_{y_n}^{\top} \boldsymbol{f}_n}}{\sum_{j=1}^{K_T} e^{\boldsymbol{w}_j^{\top} \boldsymbol{f}_n}}. \tag{3}$$

However, it has been demonstrated that the neural network can easily minimize the softmax loss function by manipulating the norms of the weights and feature vectors [20,23]. More specifically, the neural network may pay more attention to easily classified samples and classes by increasing their norms and neglect samples and classes that are hard to discriminate by decreasing their corresponding norms. Therefore, the feature vectors trained by this approach are not necessarily well separated, especially for devices with similar characteristics, and the performance of open-set authentication may be limited. To address this issue, an approach that was inspired by research in face recognition technology learns features in angular space, i.e., calculates the cross entropy loss after hypersphere projection [23]:

$$\mathcal{L}_A = -\frac{1}{N}\sum_{n=1}^{N} \ln \frac{e^{\boldsymbol{w}'^{\top}_{y_n} \boldsymbol{f}'_n}}{\sum_{j=1}^{K_T} e^{\boldsymbol{w}'^{\top}_j \boldsymbol{f}'_n}}, \tag{4}$$

where

$$\boldsymbol{w}'_i = \frac{\boldsymbol{w}_i}{\|\boldsymbol{w}_i\|}, \tag{5}$$

$$\boldsymbol{f}'_n = s\frac{\boldsymbol{f}_n}{\|\boldsymbol{f}_n\|}, \tag{6}$$

where $\|\cdot\|$ denotes L2 norm and $s > 0$ is a hyperparameter that determines the radius of the hypersphere. A similar approach was proposed by [20], with addition of one dense layer in the feature extractor for faster convergence.

Although learning features in angular space helps to improve performance, this approach only requires the features of different devices to be separable and does not enforce the features of the same device to be compact. As a consequence, the features of samples from authorized devices may occupy most of the angular space, leaving limited space for unseen devices. In this paper, we propose to utilize AAMSoftmax [24], a more advanced method used in face recognition technology, to learn features in angular space with an additive margin, so that learned features are more compact for the same device and more discriminative for different devices. To introduce AAMSoftmax, we first rewrote the loss function $\mathcal{L}_A$ as:

$$\mathcal{L}_A = -\frac{1}{N}\sum_{n=1}^{N} \ln \frac{e^{s\cos\theta_n[y_n]}}{\sum_{j=1}^{K_T} e^{s\cos\theta_n[j]}}, \tag{7}$$

where $\boldsymbol{\theta}_n$ denotes the angle vector of $\boldsymbol{f}_n$, with $\theta_n[j]$ denoting the angle between $\boldsymbol{f}_n$ and $\boldsymbol{w}_j$. The AAMSoftmax loss function was obtained by adding a margin to the angles:

$$\mathcal{L}_{AAM} = -\frac{1}{N}\sum_{n=1}^{N} \ln \frac{e^{s(\cos(\theta_n[y_n]+m))}}{e^{s(\cos(\theta_n[y_n]+m))} + \sum_{j=1, j\neq y_n}^{K_T} e^{s\cos\theta_n[j]}}, \tag{8}$$

where $m > 0$ is a hyperparameter that determines the additive angular margin. As m is equivalent to the geodesic distance margin penalty in the normalized hypersphere, it can enforce an evident gap between different classes. Generally, a larger m value leads to more separable features, but may also make the network more difficult to converge.

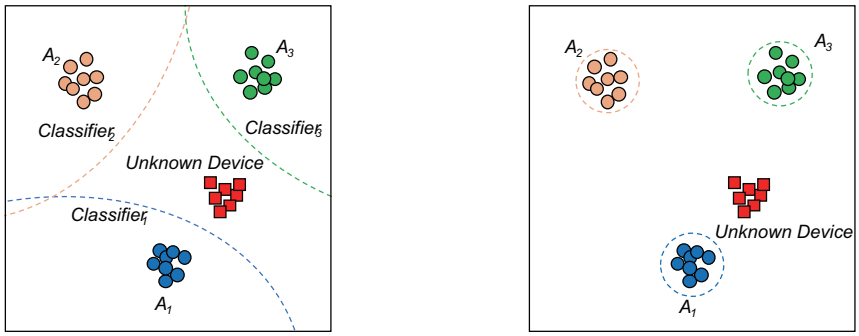
AAMSoftmax can be minimized using a stochastic gradient descent algorithm. This approach provides a simple way to further improve the discriminability of features with negligible computational overheads. After training with AAMSotmax, the learned fea-

tures are supposed to be more compact for the same class and more discriminative for different classes.

4.2. Classifier of Adaptive Class-Wise OpenMAX

With the discriminative features learned by G_ϕ, another issue is to classify the signals of authorized devices and accurately distinguish the signals of unseen devices using only the training signals from authorized devices. Two approaches, namely one-vs.-all (OvA) and OpenMAX, are typically used to achieve this goal [22].

OvA trains one binary classifier network for each authorized device. More specifically, for the authorized device $A_i \in \mathcal{A}_\mathcal{T}$, the binary classifier C_i is trained to predict the probability of the signal transmitted by A_i, with other authorized devices considered as outliers. During inference, signals are predicted as from A_i when C_i outputs the highest probability and the probability is higher than the threshold δ. Otherwise, signals are predicted as from unseen devices when all classifiers output probabilities lower than δ. Although OvA achieves a better performance than OpenMAX in [22], we claim that the performance of OvA can degrade when the characteristics of the unseen device are much more similar to one certain authorized device than others. Unfortunately, this circumstance is unexceptional, especially when the number of authorized devices is limited, as demonstrated in Figure 4.



(**a**) (**b**)

Figure 4. A demonstration of OvA and OpenMAX. Authorized devices are depicted as circles and unknown devices are depicted as squares. The dotted lines depict the boundaries of the classifiers. (**a**) Ova; (**b**) OpenMax.

Contrary to OvA, OpenMAX distinguishes unseen devices by modeling the distributions of the outliers of each authorized device using extreme value theory (EVT), thus OpenMAX is not sensitive to the number of authorized devices. Although OpenMAX was proposed for networks trained with softmax, we adjusted OpenMAX for networks trained by AAMSoftmax with minimal changes. Furthermore, as the original OpenMAX uses the same tail size for all classes, which may lead to performance degradation, we adaptively chose different tail sizes for each class and named the modified approach as adaptive class-wise OpenMAX.

The procedure for training adaptive class-wise OpenMAX is summarized in Algorithm 1. To model the distributions of the tail samples, OpenMAX first computed the feature centers of each class:

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} \mathbb{I}(y_n = k) \boldsymbol{f}_n}{\sum_{n=1}^{N} \mathbb{I}(y_n = k)}, \tag{9}$$

where $\boldsymbol{\mu}_k$ denotes the center of an authorized device A_k, $k = 1, 2, \ldots, K_T$. Then, we calculated the distance distributions of each authorized device and denoted the set of features belonging to device A_k as $\mathcal{S}_k$, i.e., $\mathcal{S}_k = \{\boldsymbol{f}_n | y_n = k\}$. As the feature extractor G_ϕ was trained in angular space using AAMSoftmax, the distance was measured based on cosine similarity:

Algorithm 1 The training algorithm for adaptive class-wise OpenMAX.

Input: Set of extracted features and corresponding labels $\{(\boldsymbol{f}_n, y_n)_{n=1}^{N}\}$, with K_T classes.
Output: mean feature vector of each class $\boldsymbol{\mu}_k$, Weibull model of each class ρ_k.
1: **for** $k = 1$ **to** K_T **do**
2: Compute mean vector of class k, $\boldsymbol{\mu}_k$;
3: Find features belonging to class k,
$\mathcal{S}_k = \{\boldsymbol{f}_n | y_n = k\}$;
4: Fit Weibull model of class k with adaptively chosen tail size γ_k,
$\rho_k = fit(\{d_C(\boldsymbol{f}, \boldsymbol{\mu}_k) | \boldsymbol{f} \in \mathcal{S}_k\}, \gamma_k)$;
5: **end for**
6: **Return** means $\boldsymbol{\mu}_k$ and models ρ_k

$$d_C(\boldsymbol{f}, \boldsymbol{\mu}) = \frac{1}{2}(1 - \cos(\boldsymbol{f}, \boldsymbol{\mu})), \tag{10}$$

$$\cos(\boldsymbol{f}, \boldsymbol{\mu}) = \frac{\boldsymbol{f}^\top \boldsymbol{\mu}}{\|\boldsymbol{f}\| \|\boldsymbol{\mu}\|}. \tag{11}$$

Basically, a larger distance implied that the sample was far from its respective center. The distance distribution of A_k, i.e., $\{d_C(\boldsymbol{f}, \boldsymbol{\mu}_k) | \boldsymbol{f} \in \mathcal{S}_k\}$, was used to fit the corresponding Weibull model, which was then employed to predict the probability of an outlier.

Note that we adaptively chose the tail size for each authorized device instead of pre-defining one tail size for all devices, as proposed in the original OpenMAX. The tail size defines the number of the largest distances for Weibull model fitting. A small tail size may lead to an inaccurate model, while a large tail size may increase the probability of misidentifying signals from authorized devices as unseen signals. Therefore, we adaptively chose the tail size by finding the largest tail size for each authorized device that the fitted model correctly and identified 99% of the training signals.

The inference procedure of adaptive class-wise OpenMAX is listed in Algorithm 2. We denoted the feature of the test sample $\boldsymbol{x}$ as $\boldsymbol{f}$, i.e., $\boldsymbol{f} = G_\phi(\boldsymbol{x})$. Different from the original OpenMAX, where the closed-set prediction score can be obtained by softmax on the activation vector, we computed the closed-set prediction score $\boldsymbol{q}$ by using softmax on the scaled cosine similarity between the feature $\boldsymbol{f}$ and each class center $\boldsymbol{\mu}_k$:

$$q_k = \frac{e^{s \cos(\boldsymbol{f}, \boldsymbol{\mu}_k)}}{\sum_{k=1}^{K_T} e^{s \cos(\boldsymbol{f}, \boldsymbol{\mu}_k)}}, k = 1, 2, \ldots, K_T. \tag{12}$$

The closed-set prediction score $\boldsymbol{q}$ was then revised to obtain the open-set prediction score $\hat{\boldsymbol{q}}$. Another adjustment to our algorithm was that we fixed the the number of top classes to revise in the original OpenMAX, hyperparameter α, as 1 for clarity. The results of the pre-experiment also proved that setting $\alpha = 1$ led to the desirable performance. With $\alpha = 1$, we only needed to consider the class that $\boldsymbol{f}$ was closest to, i.e., $\hat{k} = argmax_k(q_k)$. The trained Weibull model of $A_{\hat{k}}$ produced the probability of $\boldsymbol{f}$ being an outlier of $A_{\hat{k}}$ based on the distance between $\boldsymbol{f}$ and its center: $\omega = \rho_{\hat{k}}(d(\boldsymbol{f}, \boldsymbol{\mu}_{\hat{k}}))$. Then, $\hat{q}_{\hat{k}}$ and $\hat{q}_{K_T+1}$ were revised according to $q_{\hat{k}}$ and ω, with other elements unchanged and the guarantee that $\sum_{k=1}^{K_T+1} \hat{q}_k = 1$. Finally, predictions for open-set authentication could be made with $\hat{\boldsymbol{q}}$.

Algorithm 2 The inference algorithm for adaptive class-wise OpenMAX.

Input: feature $\boldsymbol{f}$ of the test sample.
Require: mean feature vector of each class $\boldsymbol{\mu}_k$, Weibull model of each class ρ_k.
Output: $\hat{\boldsymbol{q}}$, the prediction score of the test sample.
1: Compute the closed-set prediction score $\boldsymbol{q}$;
2: Let $\hat{k} = argmax_k(q_k)$, compute the probability the test sample is an outlier of class $\hat{k}$, $\omega = \rho_{\hat{k}}(d_C(\boldsymbol{f}, \boldsymbol{\mu}_{\hat{k}}))$;
3: Revise the prediction score as $\hat{\boldsymbol{q}}$:
$\hat{q}_k = q_k$, for $k = 1$ to K_T and $k \neq \hat{k}$,
$\hat{q}_{\hat{k}} = (1 - \omega) q_{\hat{k}}$,
$\hat{q}_{K_T+1} = \omega q_{\hat{k}}$;
4: **Return** $\hat{\boldsymbol{q}}$

5. Performance Evaluation

The proposed framework was evaluated using both simulated data and real Automatic Dependent Surveillance–Broadcast (ADS–B) data. The use of AAMSoftmax for training the feature extractor was compared to the use of the conventional softmax and ASoftmax [23], while the OpenMAX classifier was compared to an OvA classifier.

5.1. Evaluation Dataset

5.1.1. Simulated Dataset

The simulated dataset was used as a toy problem. We generated a dataset of eight devices, denoted as $\mathcal{A}^S = \{A_1^S, A_2^S, \ldots, A_8^S\}$. The signals of each device were generated with unique pairs of I/Q gain imbalance parameter G and carrier leakage parameter ξ to simulate physical imperfections [29], as listed in Table 2. Although I/Q gain imbalance and carrier leakage do not cover all of the factors that can lead to emitter impairments, the generated dataset could reflect the properties of different approaches to some extent. The signals were modulated with quadrature phase shift keying (QPSK) with a symbol rate of 64 k symbols/sec, sampled at a frequency of 1024 kHz. We generated 4000 samples for each device, with each sample containing 1024 sampling points, i.e., 64 symbols. We used 3000 samples from each device for training and the remaining 1000 samples from each device for testing.

Table 2. The parameters of the simulated devices.

Devices	A_1^S	A_2^S	A_3^S	A_4^S	A_5^S	A_6^S	A_7^S	A_8^S
G	0.9608	1.0408	0.9608	1.0408	0.9802	1.0202	0.9608	1.0408
$\xi(10^{-2})$	$1+2j$	$1+2j$	$-2-2j$	$-2-2j$	$1+2j$	$1+2j$	1	1

5.1.2. ADS–B Dataset

The real-word ADS–B dataset was provided by [30] and included 107 devices, i.e., $\mathcal{A}^R = \{A_1^R, A_2^R, \ldots, A_{107}^R\}$. There were roughly 400 samples for each device. Each signal was transformed into a 32 by 32 by 3 tensor by the preprocessing approach of [30]. We used 60% of the dataset for training and the remaining 40% for testing.

5.2. Evaluation Using the Simulated Dataset

The backbone of ResNet [31] was employed as the feature extractor for the simulated dataset, which was composed of nine convolutional layers, as shown in Figure 5. The feature extractor took a sample of 1024 I/Q sampling points as input and outputted a feature vector of dimension 512. The networks were trained by softmax and AAMSoftmax, both at learning rate of 10^{-4} for 100 epochs. The batch size for training was set as 100. The hyperparameters of AAMSoftmax were set as $s = 10$ and $m = 1.0$. Then, both feature extractors were used to train an OvA and the OpenMAX classifiers, resulting in four

combinations, namely softmax + OvA, AAMSoftmax + OvA, softmax + OpenMAX, and AAMSoftmax + OpenMAX. The threshold δ of the OvA was set as 0.99 in all combinations.

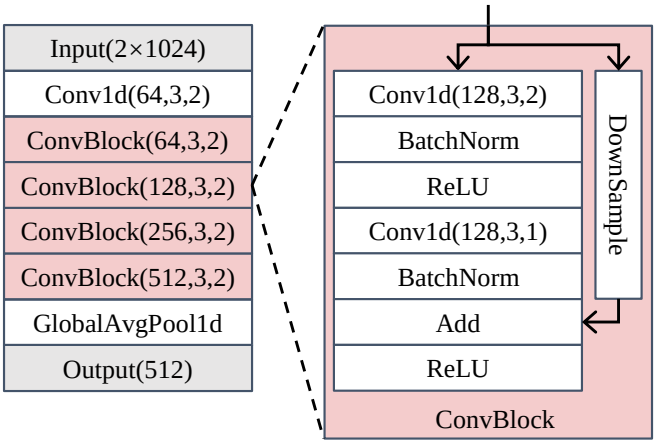


Figure 5. The architecture of the feature extractor for the simulated dataset.

5.2.1. Comparison of Overall Performance

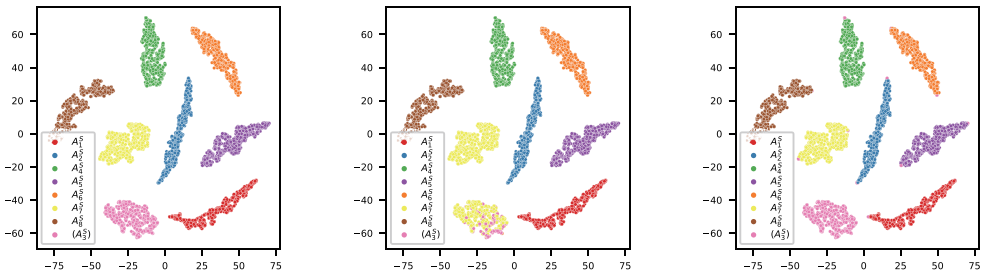
For each combination, we successively chose one device in $\mathcal{A}^S$ to be the unseen device and the remaining seven devices to be the authorized devices. The macro averaged F_1 scores [32] for the different combinations are shown in Table 3. Basically, using the OvA classifier led to a mediocre performance, regardless of whether the feature extractor was trained by softmax or AAMSoftmax, which could have been caused by defects in the OvA classifier. When OpenMAX was combined with softmax, the desirable performance was achieved when A_3^S, A_4^S, A_7^S or A_8^S was chosen as the unseen device. However, the combination of OpenMAX and AAMSoftmax showed a superior performance regardless of the choice of unseen device. One reason for this performance gap could be that the parameters of A_5^S and A_6^S were close to A_1^S and A_2^S, respectively, making them hard to separate when trained by softmax.

Table 3. The F_1 scores of different combinations

Unknown Device	A_1^S	A_2^S	A_3^S	A_4^S	A_5^S	A_6^S	A_7^S	A_8^S
Softmax + OvA	0.8333	0.8333	0.8369	0.8552	0.7376	0.8303	0.8333	0.8331
AAMSoftmax + OvA	0.7842	0.8333	0.8438	0.9391	0.8315	0.8149	0.8326	0.8331
Softmax + OpenMAX	0.8766	0.7308	0.9628	**0.9637**	0.6010	0.8328	0.9451	**0.9741**
AAMSoftmax + OpenMAX	**0.9695**	**0.9629**	**0.9688**	0.9623	**0.9654**	**0.9716**	**0.9658**	0.9679

5.2.2. Comparison of Feature Extractors

We compared the predictions of the OvA and OpenMAX classifiers based on the features learned by AAMSoftmax and with A_3^S as the unseen device. The features obtained by the AAMSoftmax training were visualized using t-SNE [33], with different colors corresponding to the true identities and predictions of different devices, as shown in Figure 6. It was shown that the learned features could properly separate different devices, including the unseen device. However, OvA misclassified most samples of the unseen device A_3^S as A_7^S, the parameters of which were closest to A_3^S, as shown in Table 2. OpenMAX correctly distinguished most samples of the unseen device, although a few samples of other devices were misidentified. This comparison revealed that the OvA classifier was ineffective when the unseen device was more similar to one certain authorized device than others, while OpenMAX was less vulnerable to this condition.



(**a**) (**b**) (**c**)

Figure 6. Visualizations of the true identity and predictions of the classifiers. A_3^S is in brackets to denote that it is the unseen device. (**a**) True identity; (**b**) Predicitions of the OvA; (**c**) Predicitions of OpenMAX.

5.2.3. Comparison of Classifiers

The distributions of the feature distances of softmax and AAMSoftmax were compared to verify the discriminability of the learned features. Assuming A_j^S was chosen as the unseen device and one of the authorized devices A_i^S was chosen as the reference device, three sets of distances, namely intra-distances $\mathcal{U}$, inter-distances $\mathcal{I}$, and open distances $\mathcal{O}$, were calculated as follows:

$$\mathcal{U} = \{d(\boldsymbol{f}_n^S, \boldsymbol{\mu}_i^S) | y_n^S = i\}, \tag{13}$$

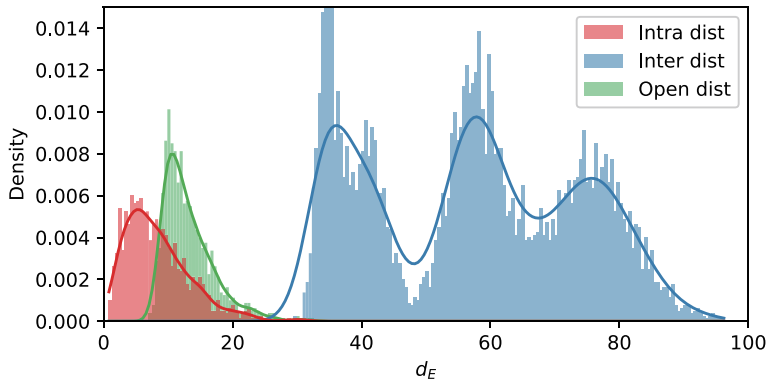
$$\mathcal{I} = \{d(\boldsymbol{f}_n^S, \boldsymbol{\mu}_i^S) | y_n^S \neq i \; and \; y_n^S \neq j\}, \tag{14}$$

$$\mathcal{O} = \{d(\boldsymbol{f}_n^S, \boldsymbol{\mu}_i^S) | y_n^S = j\}, \tag{15}$$

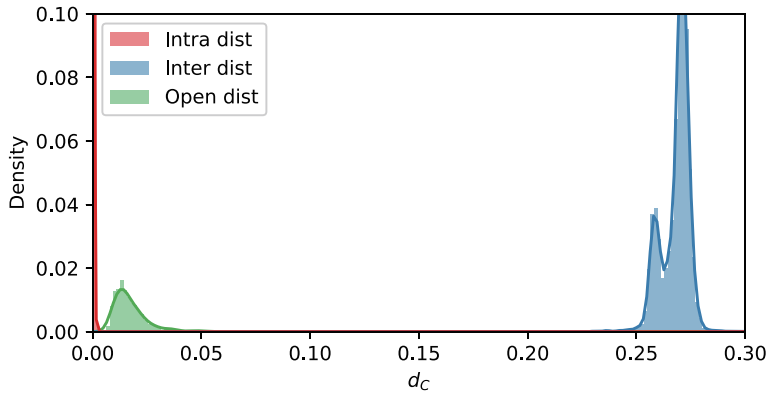
where $\mathcal{U}$ denotes the set of distances between the features of A_i^S and the feature center of A_i^S, $\mathcal{I}$ denotes the set of distances between the features of other authorized devices and the feature center of A_i^S, and $\mathcal{O}$ denotes the set of distances between the features of the unseen device A_j^S and the feature center of A_i^S. For softmax, $d(\boldsymbol{f}, \boldsymbol{\mu}) = d_E(\boldsymbol{f}, \boldsymbol{\mu}) = \|\boldsymbol{f} - \boldsymbol{\mu}\|$, i.e., Euclidean distance, and for AAMSoftmax, $d(\boldsymbol{f}, \boldsymbol{\mu}) = d_C(\boldsymbol{f}, \boldsymbol{\mu})$, i.e., cosine distance. We selected $i = 2$ and $j = 6$ for demonstration, since A_2^S was highly similar to A_6^S. The densities of the distance distributions are shown in Figure 7. Not surprisingly, the intra-distances and inter-distances were distributed apart from each other, regardless of whether the feature extractor was trained by softmax or AAMSoftmax. However, when the feature extractor was trained with softmax, the distribution of open distances was highly overlapped with intra-distances, implying that the features were incapable of accurately separating A_2^S samples from those of the unseen device A_6^S. As for features learned by AAMSoftmax, although the distribution of open distances was still close to that of the intra-distances, they were evidently more separable and demonstrated less overlapping.

5.2.4. Comparison of Different Combinations

The confusion matrix of different combinations is shown in Figure 8, with A_6^S used as the unseen device. The approaches using softmax were inferior at distinguishing the samples of the unseen device A_6^S due to the fact that the features from the unseen device were overlapped with authorized devices, as shown previously. The combination of AAMSoftmax and OvA could also not distinguish the unseen device correctly because the learned features of the unseen device A_6^S were still much closer to A_2^S than to the other authorized devices and the OvA failed in this condition. Only the combination of AAMSoftmax and OpenMAX correctly identified the unseen device, at the cost of a slightly decreased performance concerning authorized devices. The results of the different combinations verified the significance of combining AAMSoftmax with OpenMAX.

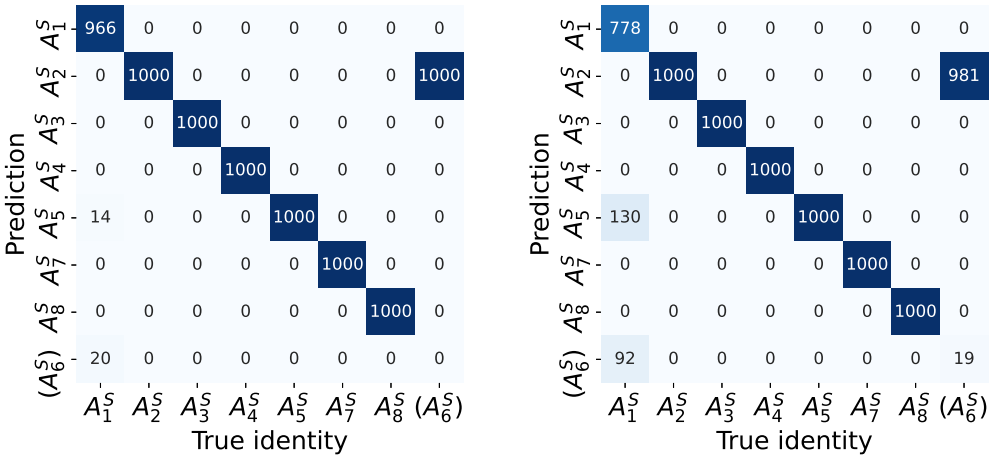


(**a**)



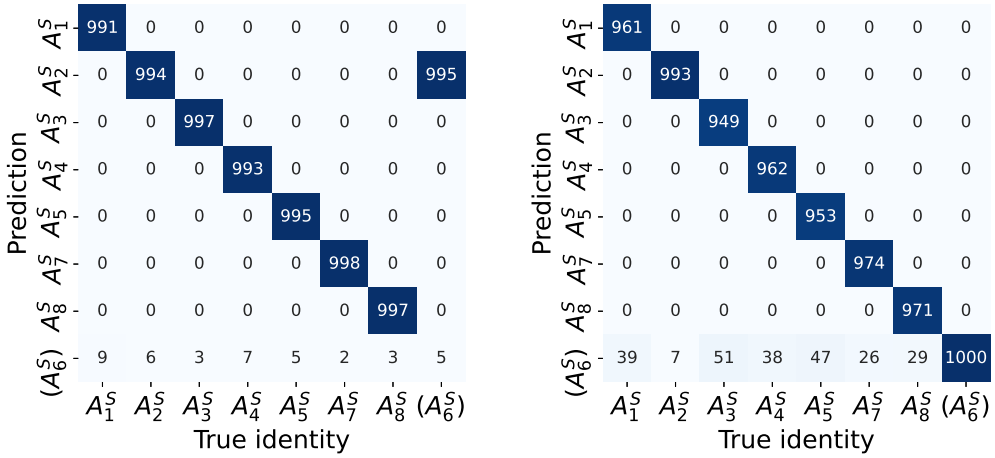
(**b**)

Figure 7. The comparison of the distance distribution of the features. "Intra dist" indicates the distance distribution of the features from the same device with respect to the reference device, "Inter dist" indicates the distance distribution of the features from other authorized devices, and "Open dist" indicates the distance distribution of the features from the unknown device. (**a**) Softmax; (**b**) AAMSoftmax.



(**a**) (**b**)

Figure 8. *Cont.*



(**c**) (**d**)

Figure 8. The confusion matrix of the different combinations. A_6^S is in brackets to denote that it is the unseen device. (**a**) Softmax+OvA; (**b**) AAMSoftmax+OvA; (**c**) Softmax+OpenMAX; (**d**) AAMSoftmax+OpenMAX.

5.3. Evaluation Using the Real ADS–B Dataset

We used the network of [20] as the feature extractor, with the addition of a dropout layer, as shown in Figure 9. The feature extractor took a sample of $3 \times 32 \times 32$ tensor as input and outputted a feature vector of dimension 64. The network was trained by softmax, ASoftmax, and AAMSoftmax, all at learning rate of 10^{-4} for 100 epochs. The batch size for training was set as 128. The hyperparameter of the OvA was set as $\delta = 0.9$ and the hyperparameters of AAMSoftmax were set as $s = 3$ and $m = 1.0$. As the feature extractor ended with a fully connected layer, it was equivalent to the zero-bias layer approach [20] when trained by ASoftmax. Then, all of the feature extractors were used to train the OvA and OpenMAX classifiers, resulting in six combinations. For each combination, we successively selected 5, 10, 21, 32, 43, and 54 devices as the authorized devices and the remaining devices as unseen devices.

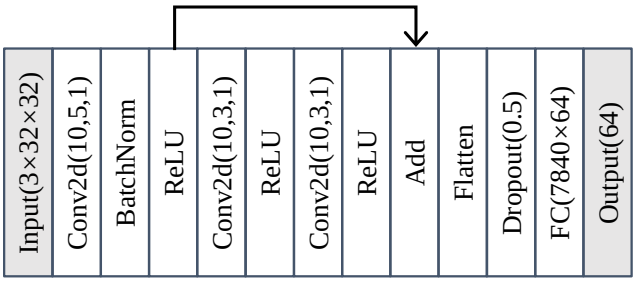
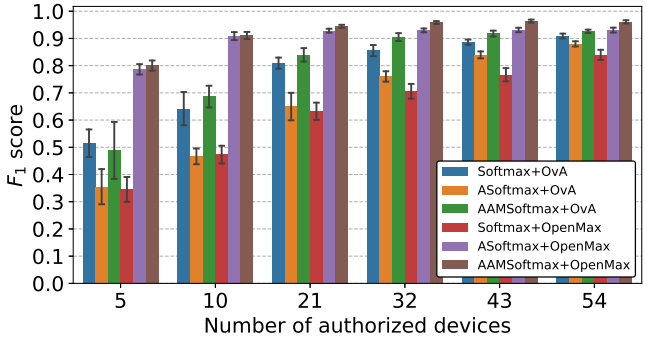


Figure 9. The architecture of the feature extractor for the real ADS–B dataset.

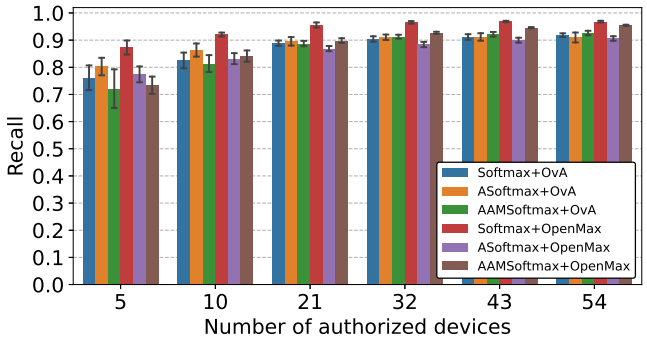
5.3.1. Performance Comparison

The performance of each setting was assessed using the average of ten different runs, with a random selection of authorized devices and a random initialization of network parameters. The results are shown in Figure 10, with the bars indicating the averages of the different runs and the error bars denoting the standard deviation. Figure 10a displays the F_1 scores of the devices, including authorized and unseen devices when using different combinations. All combinations performed better with more authorized devices, implying that the network could learn more discriminative features with more authorized devices, which coincided with research in face recognition technology [34]. Although approaches using the OvA classifier achieved a comparable performance with a large number of

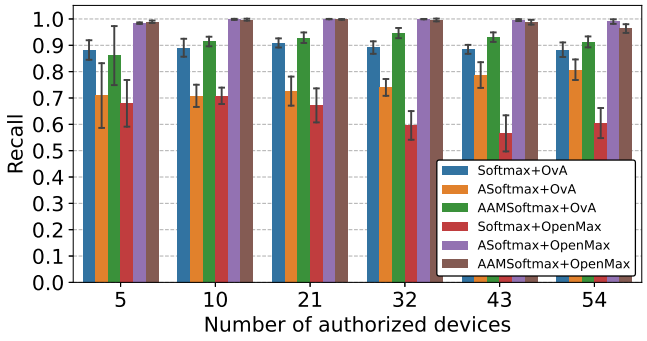
authorized devices, they performed poorly with limited authorized devices regardless of the loss function that was used to train the feature extractor, confirming that the OvA classifier relied heavily on the number of authorized devices, as discussed previously. For approaches using the OpenMAX classifier, softmax + OpenMAX achieved a mediocre performance in all settings, while ASoftmax + OpenMAX and AAMSoftmax + OpenMAX achieved evidently superior performances, with AAMSoftmax + OpenMAX performing modestly better than ASoftmax + OpenMAX. Therefore, the discriminability of extracted features was critical for OpenMAX to perform successfully. By comparing the recall of authorized devices in Figure 10b and the recall of unseen devices in Figure 10c, it is illustrated that AAMSoftmax + OpenMAX was successful at distinguishing unknown devices while satisfactory at identifying authorized devices.



(**a**)



(**b**)

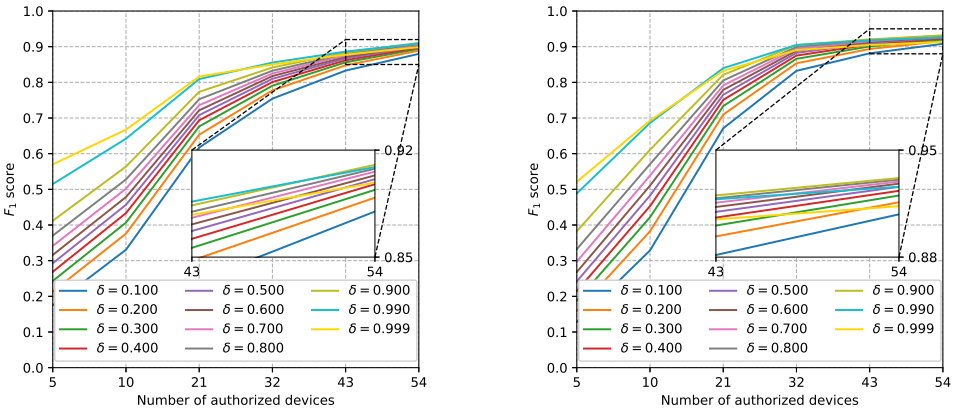


(**c**)

Figure 10. The performances of different combinations. (**a**) F_1 score of all devices; (**b**) Recall of authorized devices; (**c**) Recall of unknown devices.

5.3.2. Effects of Hyperparameters

We first evaluated the effects of δ, the hyperparameter of the OvA classifier. The F_1 scores of Softmax + OvA and AAMSoftmax + OvA with different δ values are shown in Figure 11. When δ increased form 0.1 to 0.9, the performance increased. However, when δ increased to 0.99 and further to 0.999, the performance of a limited number of authorized devices increased but the performance decreased with a large number of authorized devices. Therefore, we claim that 0.99 was the proper value of δ. Nevertheless, the performances with limited authorized devices were mediocre compared to OpenMAX classifier regardless of the value of δ, which further proved the defects of the OvA classifier.



(**a**) (**b**)

Figure 11. The F_1 scores of softmax + OvA and AAMSoftmax + OvA with different δ values. (**a**) Softmax+OvA; (**b**) AAMSoftmax+OvA.

The hyperparameters of AAMSoftmax, i.e., s and m, were also evaluated. The F_1 scores of AAMSoftmax + OpenMAX with different s and m values are shown in Figure 12. Large s/m values could weaken the penalty of feature compactness and lead to less discriminative features, while large m/s values could punish the features too much and make the network hard to converge. Fortunately, AAMSoftmax was tolerant of the hyperparamters to some extent and the desirable performance could be achieved within a certain range of s and m. As AAMSoftmax was equivalent to ASoftmax when $s = 1.0$ and $m = 0.0$, it was evident that AAMSoftmax was better than ASoftmax with appropriately selected s and m values.

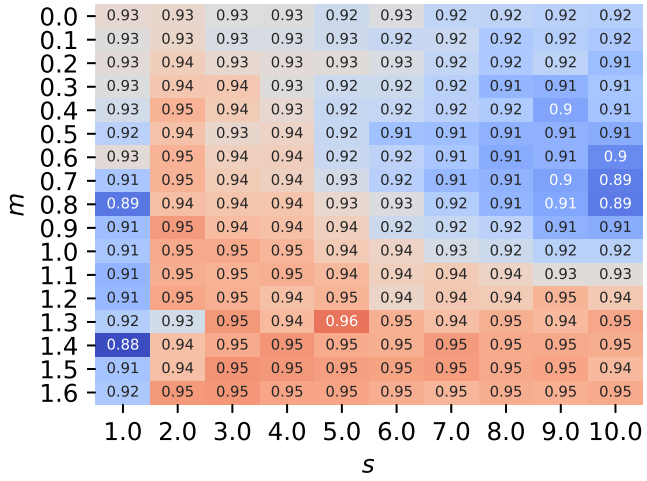


Figure 12. The F_1 scores of AAMSoftmax + OpenMAX when trained by 21 authorized devices with different s and m values.

6. Conclusions

In this paper, we proposed a novel framework for the open-set authentication of Internet of Things using limited authorized devices for training. Our contributions were as follows. Firstly, the AAMSoftmax loss function was utilized to train the feature extractor for more discriminative features. Secondly, an adaptive class-wise OpenMAX classifier was proposed for accurate open-set authentication based on learned features. Thirdly, we verified the superiority of the AAMSoftmax + OpenMAX approach through experiments with both simulated data and real-world ADS–B data. The experiments showed that AAMSoftmax + OpenMAX achieved a better performance than the other approaches, with an F_1 score of 0.91 for the open-set authentication of 107 airborne transponders when only 10 of them were used for training. A remaining challenge of physical layer authentication is the robustness of the learned features [4]; for example, changing wireless channel conditions can degrade authentication accuracy prominently [35]. Our future work will focus on enhancing the robustness of the features.

Author Contributions: Formal analysis, K.H.; Funding acquisition, J.Y.; Resources, H.L.; Software, K.H.; Supervision, J.Y.; Validation, P.H.; Writing—review & editing, K.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Anhui Provincial Natural Science Foundation NO.1908085MF202.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ismail, B.; Patrik, Ö.; Houbing, S. Security of the Internet of Things: Vulnerabilities, attacks, and countermeasures. *IEEE Commun. Surv. Tutor.* **2019**, *22*, 616–644. [CrossRef]
2. Skarmeta, A.F.; Hernandez-Ramos, J.L.; Moreno, M.V. A decentralized approach for security and privacy challenges in the internet of things. In Proceedings of the 2014 IEEE World Forum on Internet of Things (WF-IoT), Seoul, Korea, 6–8 March 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 67–72.
3. Linda, S.; Marco, B.; Ennio, G. Statistical and machine learning-based decision techniques for physical layer authentication. In Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA, 9–13 December 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–6.
4. Yongxin, L.; Jian, W.; Jianqiang, L.; Shuteng, N.; Houbing, S. Machine learning for the detection and identification of internet of things (iot) devices: A survey. *arXiv* **2021**, arXiv:2101.10181.
5. Hansen, J.H.; Hasan, T. Speaker recognition by machines and humans: A tutorial review. *IEEE Signal Process. Mag.* **2015**, *32*, 74–99. [CrossRef]
6. Vladimir, B.; Suman, B.; Marco, G.; Sangho, O. Wireless device identification with radiometric signatures. In Proceedings of the 14th ACM International Conference on Mobile Computing and Networking, San Francisco, CA, USA, 14–19 September 2008; pp. 116–127.
7. Zhang, J.; Wang, F.; Dobre, O.A.; Zhong, Z. Specific emitter identification via Hilbert–Huang transform in single-hop and relaying scenarios. *IEEE Trans. Inf. Forensics Secur.* **2016**, *11*, 1192–1205. [CrossRef]
8. Udit, S.; Nikita, T.; Gagarin, B.; Barathram, R. Specific emitter identification based on variational mode decomposition and spectral features in single hop and relaying scenarios. *IEEE Trans. Inf. Forensics Secur.* **2018**, *14*, 581–591. [CrossRef]
9. Jie, H.; Tao, Z.; Zhaoyang, Q.; Xiaoyu, Z. Communication emitter individual identification via 3D-Hilbert energy spectrum-based multiscale segmentation features. *Int. J. Commun. Syst.* **2019**, *32*, e3833. [CrossRef]
10. Dongfang, R.; Tao, Z. Specific emitter identification based on intrinsic time-scale-decomposition and image texture feature. In Proceedings of the 2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN), Guangzhou, China, 6–8 May 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1302–1307.
11. Guangquan, H.; Yingjun, Y.; Xiang, W.; Zhitao, H. Specific emitter identification for communications transmitter using multi-measurements. *Wirel. Pers. Commun.* **2017**, *94*, 1523–1542. [CrossRef]
12. Yongqiang, J.; Shengli, Z.; Lu, G. Specific emitter identification based on the natural measure. *Entropy* **2017**, *19*, 117. [CrossRef]
13. Kevin, M.; Shauna, R.; George, S.; Bryan, N. Deep learning for RF device fingerprinting in cognitive communication networks. *IEEE J. Sel. Top. Signal Process.* **2018**, *12*, 160–167. [CrossRef]

14. Riyaz, S.; Sankhe, K.; Ioannidis, S.; Chowdhury, K. Deep learning convolutional neural networks for radio identification. *IEEE Commun. Mag.* **2018**, *56*, 146–152. [CrossRef]
15. McGinthy, J.M.; Wong, L.J.; Michaels, A.J. Groundwork for neural network-based specific emitter identification authentication for IoT. *IEEE Internet Things J.* **2019**, *6*, 6429–6440. [CrossRef]
16. Jiabao, Y.; Aiqun, H.; Guyue, L.; Linning, P. A robust RF fingerprinting approach using multisampling convolutional neural network. *IEEE Internet Things J.* **2019**, *6*, 6786–6799. [CrossRef]
17. Tong, J.; Costa, R.B.; Emmanuel, O.; Nasim, S.; Zifeng, W.; Kunal, S.; Andrey, G.; Jennifer, D.; Kaushik, C.; Stratis, I. Deep learning for RF fingerprinting: A massive experimental study. *IEEE Internet Things Mag.* **2020**, *3*, 50–57. [CrossRef]
18. Guanxiong, S.; Junqing, Z.; Alan, M.; Linning, P.; Xianbin, W. Radio Frequency Fingerprint Identification for LoRa Using Deep Learning. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 2604–2616. [CrossRef]
19. Debashri, R.; Tathagata, M.; Mainak, C.; Erik, B.; Eduardo, P. Rfal: Adversarial learning for rf transmitter identification and classification. *IEEE Trans. Cogn. Commun. Netw.* **2019**, *6*, 783–801. [CrossRef]
20. Yongxin, L.; Jian, W.; Jianqiang, L.; Houbing, S.; Thomas, Y.; Shuteng, N.; Zhong, M. Zero-bias deep learning for accurate identification of Internet-of-Things (IoT) devices. *IEEE Internet Things J.* **2020**, *8*, 2627–2634. [CrossRef]
21. Andrey, G.; Zifeng, W.; Tong, J.; Jennifer, D.; Kaushik, C.; Stratis, I. Finding a 'new' needle in the haystack: Unseen radio detection in large populations using deep learning. In Proceedings of the 2019 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN), Newark, NJ, USA, 11–14 November 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–10.
22. Samer, H.; Samurdhi, K.; Danijela, C. Open set wireless transmitter authorization: Deep learning approaches and dataset considerations. *IEEE Trans. Cogn. Commun. Netw.* **2020**, *7*, 59–72. [CrossRef]
23. Renjie, X.; Wei, X.; Yanzhi, C.; Jiabao, Y.; Aiqun, H.; Kwan, N.D.W.; Lee, S.A. A Generalizable Model-and-Data Driven Approach for Open-Set RFF Authentication. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 4435–4450. [CrossRef]
24. Jiankang, D.; Jia, G.; Niannan, X.; Stefanos, Z. Arcface: Additive angular margin loss for deep face recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4690–4699.
25. Bendale, A.; Boult, T.E. Towards open set deep networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 1563–1572.
26. Yandong, W.; Kaipeng, Z.; Zhifeng, L.; Yu, Q. A discriminative feature learning approach for deep face recognition. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 499–515.
27. Zhongxin, B.; Xiao-Lei, Z. Speaker recognition based on deep learning: An overview. *Neural Netw.* **2021**, *140*, 65–99. [CrossRef]
28. Yuheng, W.; Junzhao, D.; Hui, L. Angular Margin Centroid Loss for Text-Independent Speaker Recognition. In Proceedings of the INTERSPEECH, Shanghai, China, 25–29 October 2020; pp. 3820–3824.
29. Boxiang, H.; Fanggang, W. Cooperative Specific Emitter Identification via Multiple Distorted Receivers. *IEEE Trans. Inf. Forensics Secur.* **2020**, *15*, 3791–3806. [CrossRef]
30. Yongxin, L.; Jian, W.; Jianqiang, L.; Shuteng, N.; Houbing, S. Class-Incremental Learning for Wireless Device Identification in IoT. *IEEE Internet Things J.* **2021**, *8*, 17227–17235. [CrossRef]
31. Kaiming, H.; Xiangyu, Z.; Shaoqing, R.; Jian, S. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
32. Juri, O.; Sebastian, B. Macro f1 and macro f1. *arXiv* **2019**, arXiv:1911.03347.
33. Van der Maaten, L.; Hinton, G. Visualizing data using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.
34. Sáez, T.D.; Li, M.; Margaret, H. Face recognition: From traditional to deep learning methods. *arXiv* **2018**, arXiv:1811.00116.
35. Restuccia, F.; D'Oro, S.; Al-Shawabka, A.; Belgiovine, M.; Angioloni, L.; Ioannidis, S.; Chowdhury, K.; Melodia, T. DeepRadioID: Real-time channel-resilient optimization of deep learning-based radio fingerprinting algorithms. In Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing, Catania, Italy, 2–5 July 2019; pp. 51–60.