AMDetector: Detecting Large-scale and Novel Android Malware Traffic with Meta-learning

Wenhao Li[1,2], Huaifeng Bao[1,2], Xiao-Yu Zhang[1,2(✉)], and Lin Li[1,2]

[1] Institute of Information Engineering,
Chinese Academy of Sciences, Beijing 100093, China
{liwenhao,baohuaifeng,zhangxiaoyu,lilin}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100093, China

Abstract. In the severe COVID-19 environment, encrypted mobile malware is increasingly threatening personal privacy, especially those targeting on Android platform. Existing methods mainly focus on extracting features from Android Malware (DroidMal) by reversing the binary samples, which is sensitive to the deduction of the available samples. Thus, they fail to tackle the insufficiency of the novel DoridMal. Therefore, it is necessary to investigate an effective solution to classify large-scale DroidMal, as well as to detect the novel one. We consider few-shot DroidMal detection as DoridMal encrypted network traffic classification and propose an image-based method with meta-learning, namely AMDetector, to address the issues. By capturing network traffic produced by DroidMal, samples are augmented and thus cater to the learning algorithms. Firstly, DroidMal encrypted traffic is converted to session images. Then, session images are embedded into a high dimension metric space, in which traffic samples can be linearly separated by computing the distance with the corresponding prototype. Large-scale and novel DroidMal traffic is classified by applying different meta-learning strategies. Experimental results on public datasets have demonstrated the capability of our method to classify large-scale known DroidMal traffic as well as to detect the novel one. It is encouraging to see that, our model achieves superior performance on known and novel DroidMal traffic classification among the state-of-the-arts. Moreover, AMDetector is able to classify the unseen cross-platform malware.

Keywords: Malware Detection · Network Security · Meta-learning · COVID-19 · Privacy Security.

1 Introduction

With the prosperity of mobile applications, it becomes increasingly intractable to protect the security and privacy of mobile users [3]. The explosion of Android malware (DroidMal) is challenging the effectiveness and capability of detection systems. Previous works managed to detect DroidMal by reversing engineering and focused on the extraction of the static features in binary [30]. By analyzing

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_33

the binary of DroidMal, static features are considered as the representation of DroidMal, including CFG [15] (Call Flow Graph), static API calls [6], permissions, etc. Furthermore, the development of dynamic analysis offers an auxiliary way to detect DroidMal [26, 25]. Dynamic features such as API call chains and system operations [16] can be included, by dynamically executing and hooking DroidMals in sandbox [19]. Overall, DroidMal detection methods based on source code analysis have achieved eye-catching performance after continuous optimization. However, it is generally not an easy task to obtain DroidMal samples, especially those from the emerging families. Moreover, advanced obfuscation techniques are applied to reinforce DroidMal and tremendously weaken the performance of DroidMal detectors [27].

Detecting the network traffic that generated by DroidMal is proved to be a more straightforward and efficient approach to tackle DroidMal [2]. DroidMal analysis problems are converted to the identification of malware network traffic, which skillfully tackles the insufficiency of samples and the diversity of obfuscation. Conventional methods with static IP/Port/Payload matching cannot satisfy the rapidly growing malicious traffic encryption, especially under the COVID-19 environment [24]. Therefore, many works equip machine learning algorithms, including Random Forest, Decision Tree, SVM, etc, to detect DroidMal network traffic [20]. Although such methods dramatically develop the detection systems, they suffer from the overwhelming demands on feature engineering and prior expert knowledge. Thus, they are not suitable to identify large-scale DroidMal traffic. The explosion of deep learning offers another solution to classify DroidMal traffic. Combining feature engineering and classification algorithm, the end-to-end models based on deep learning achieve superior performance compared with machine-learning-based ones [10]. In general, payloads in network traffic are automatically extracted by Deep Neural Networks (DNN) and fed to a Multi-layer perceptron (MLP) classifier [28]. Although such methods perform outstandingly on plaintext, they fail to detect encrypted malware traffic thus reach their limitations.

In general, three challenges remain unsolved in DroidMal traffic classification. **1)** It is difficult to attain emerging DroidMal and the corresponding network traffic, which leads to few-shot scenario in DroidMal Detection [4]. **2)** Conventional detectors mostly rely on pattern matching with plaintext, which are countered by the encryption techniques. **3)** Conventional techniques reach their limitations when tackling novel DroidMal. To address the issues, we propose a session-image-based model based on meta-learning, which can classify large-scale DroidMal encrypted traffic and detect unknown DroidMal traffic categories. First, we convert encrypted DroidMal network traffic to session images. Then, images are embedded into high dimension metric space based on meta-learning. Therefore, samples can be linearly classified by computing the distance between prototypes on high dimension separatable space.

Our contributions can be briefly summarized as follows:

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_33

1. We consider DroidMal detection as encrypted traffic classification to tackle the lack of DroidMal samples and propose AMDetector, a DroidMal-image model based on meta-learning to classify large-scale and novel DroidMal.
2. AMDetector is capable to classify large-scale DroidMal encrypted traffic while detecting unknown DroidMal categories and achieves superior performance compared with the state-of-the-arts.
3. It is encouraging to see that AMDetector attains the ability to perform cross-platform detection when classifying malware traffic generated by Windows.

The rest of the paper is organized as follows. In Section 2, we define the problems we focus on. In Section 3, AMDetector is described in detail. In Section 4, compare AMDetector with the state-of-the-art baselines and analyze the experimental results. Finally, we make a conclusion of our work and present the future enhancement.

2 Preliminaries

In this section, we first give the definition of DroidMal encrypted traffic classification. Then, we briefly introduce the meta-learning strategy on DroidMal encrypted traffic classification, which consists of the detection on known and novel DroidMal.

2.1 Problem Definition

We consider DroidMal detection as DroidMal encrypted traffic classification based on meta-learning. By running each DroidMal sample in sandbox, network traffic generated by the DroidMal can be captured with the help of network sniffer. Initially, DroidMal traffic dataset is separated into training set, validation set and testing set. First, we train our proposed model with training set with meta-learning strategy. Then, after each training epoch, model is evaluated on validation set to validate the effectiveness. Finally, a well-trained model is tested with testing set in order to verify the robustness and generalization. Overall, we aim to attain a well-trained model that gains the ability to classify large-scale known and novel DroidMal encrypted traffic precisely through a few testing samples.

We propose a traffic-based model with meta-learning to classify large-scale known and novel DroidMal encrypted traffic. Specifically, raw traffic of DroidMal is represented as image in the first part of our model. Then, images are embedded into high dimension metric space through convolutional mapping model. By computing the prototypes for each category, distance between samples and the corresponding prototype is obtained and contributes to the classification of known and novel DroidMal traffic.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_33

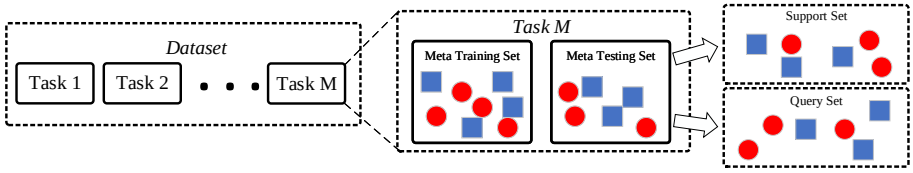


Fig. 1. Framework of meta-learning. The whole dataset is divided into M meta tasks, which contain meta training set and meta testing set. A support set and a query set are included in a meta set.

2.2 Traffic-based Android Malware Classification with meta-learning

In conventional machine learning (ML) and deep learning (DL) methods, models usually focus on a specific task. In the field of DoridMal encrypted traffic, a basic task is to train a classifier that can predict the labels of testing samples. Specifically, we train a classify with a training set $Set_D = \{(x_1, y_1), ..., (x_n, y_n)\}$ where x_n and y_n refer to traffic sample and the corresponding label, respectively. After optimizing the parameters, a well-trained classifier can determine the labels of the input samples. Generally, a conventional ML or DL classifier demands a great amount of available samples to converge, where $|Set_D|$ can be up to tens of thousands. Especially, when $|Set_D|$ is small, we can consider the task as a few-shot scenario.

Due to the scarce emerging DroidMal, capturing sufficient traffic samples can be a difficult task. Motivated by the previous work on few-shot learning [23], we employ meta-learning strategy to tackle the DoridMal traffic classification. In meta-learning, classifier learns to accomplish several meta tasks rather than focus on a specific one. For example, in a N-way-K-shot strategy, training set is split into meta tasks that consist of N categories with K samples. Demonstrated as Figure 1, M meta tasks are separated. Each task consists of a meta training set and a meta testing set. In a single epoch, a meta batch is randomly picked up and considered as the meta tasks. Specifically, meta training set is used to optimize the model while the testing set is for validation. Theoretically, meta-learning classifier can be transferred to an unseen dataset with the help of its learning-to-learn strategy, which is capable to detect novel DroidMal traffic.

In the field of DroidMal encrypted traffic detection, we consider meta-tasks as multi-class classification. In general, an N-Way-K-Shot meta task consists of $N \times K$ samples. As shown in Figure 1, a meta set is separated into a support set and a query set, which in our proposed model, the support set is for prototype computation while the query set is to optimize our model with a distance loss function.

We evaluate on a DoridMal encrypted dataset with 42 classes of DroidMal to classify large-scare known DroidMal traffic. Therefore, N classes with K samples are randomly selected from 42 classes when constructing a N-Way-K-Shot meta task in training process. During testing, the model can classify 42 classes of

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_33

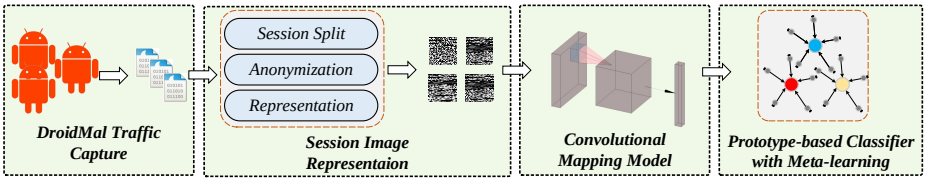


Fig. 2. Main framework of AMDetector.

DroidMal where $N = 42$ (AndMal2017 we evaluated on contains 42 classes of DroidMal).

In the task of novel DroidMal encrypted traffic detection, O of 42 classes are randomly chosen to be the novel categories, where O is the number of novel classes. During training, N classes are picked up from $42-O$ classes to construct a meta task. Then, the model can classify known classes while detecting the novel ones where $N = 42$.

3 DroidMal Detection Based on Traffic Image with Meta-learning

The main structure of our proposed DroidMal detecting model (AMDetector) is shown in Figure 2. Our model consists of Session Image Translation, a Convolutional Mapping Model and a Prototype-based Classifier. In the following sections, we will exhaustively introduce each part of our model.

3.1 Session Image Translation

In Session Image Translation, we aim to translate binary traffic data produced by DroidMal to session-based images to enhance the representation of DoridMal. In general, raw DroidMal traffic data are converted to 28×28 gray-scale images during this section.

Session Split: A unique traffic flow is constructed by abstracting a five-tuple, which formed as {Source IP, Destination IP, Source Port, Destination Port, Protocol}. During a time period, a session is the combination of two opposites flows that generated between two hosts. In real-world network, traffic is captured by sniffers and stored in traffic files (PCAP format), which are composed of diverse sessions. In this section, unique sessions are extracted separately from raw PCAP files and stored in discrete PCAPs, which contain a single session.

Anonymization: The irrelevant fields of network packets are masked during anonymization. Data captured from real-world network adheres to Open System Interconnection (OSI) reference model. Therefore, transmission information such as MAC address, IP address and TCP/UDP port, are included in every single packet. However, it is not appropriate to learn from these strong-relevant fields

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_33

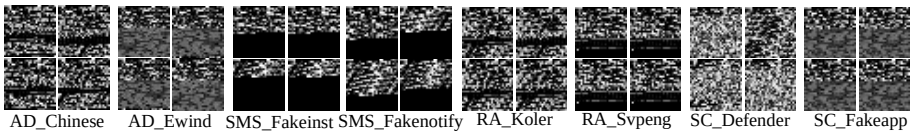


Fig. 3. Visualization of Session Images. Eight classes of DroidMal are illustrated, in form of {Family}_{subfamily}, e.g. `AD` stands for the family of `Advertisement`

that can specify the labels in certain dataset but may change in different network environment, which leads to overfitting and limits the generalization of our model. Thus, such learning-irrelevant fields are masked by padding with 0x00.

Session Image Representation: In Session Image Representation, PCAPs that store a single session are converted to gray-scale image. Specifically, the first 784 bytes of a PCAP file are extracted and construct a 28×28 feature image. If a PCAP is smaller than 784 bytes, 0x00 will be padded to 784 bytes. Intuitively, binary with 0x00 and 0xFF are represented as a black pixel and a white pixel in session image, respectively.

Theoretically, the front part of a session contains the majority of connection features, thus reflects the intrinsic characteristics of a session to the most extent [5]. Meanwhile, translation of session image caters the input of CNN, as well as enhances the ability of representation. Shown as Figure 3, we demonstrate part of the session images from different categories. We argue that different types of session images present discrimination obviously while the ones from the same class show high consistency, which proves the rationality of the session images.

3.2 Convolutional Mapping Model with Meta-Learning

Convolutional mapping model embeds session images into high dimension metric space. In general, the mapping model promises to embed the original less-separated images into a high dimensional linearly separable spatial. Convolutional mapping model with meta-learning is intrinsically distinct from the conventional classifiers. Rather than classifies directly, convolutional mapping model gains the power to learn how to learn by accomplishing meta classification tasks. Theoretically, it is not essential to construct a complicated mapping model when employing meta-learning strategy, which effectively prevents overfitting when tackling few-shot problem and enhances the generalization and transferability.

With N-Way-K-Shot strategy, $N \times K$ session images with the size of 28×28 are fed into our model and embedded into 64-dimensional vectors. A concatenation of 4 convolutional sequences is concluded in our mapping model. Specifically, each sequence consists of a convolutional layer with a kernel of 3×3 windows, a batch normalization, and an element-wise rectified-linear non-linearity (ReLU) activation function. In order to improve the generalization of our model, max-pooling is applied before outputs. Finally, session images of 28×28 from meta tasks are embedded into 64-dimensional metric spatial.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_33

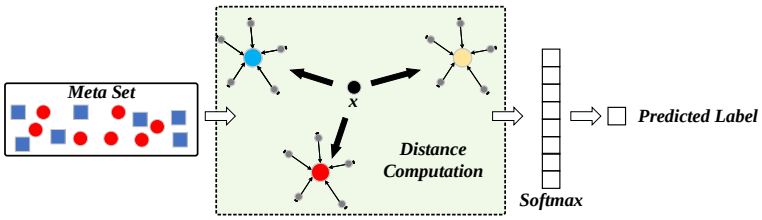


Fig. 4. Meta set is separated into support set and query set. Prototypes are attained by aggregating support set while query is used to compute the distance. Then, distance between each prototype is fed into Softmax and the predicted label is obtained.

3.3 Prototype-based Classifier

Conventional DoridMal traffic classifiers attain the probability of each class by applying $Softmax$ directly. Then, the sample is classified into the category with the highest probability. However, conventional classification schema with $Softmax$ is not suitable for our N-Way-K-Shot meta learning, in which N is indefinite during training and testing. Inspired by Prototypical Nets [18], we propose a prototype-based classifier to adapt flexible N and K dynamically. Support set is used to compute prototypes of each class while query set is to optimize the model by computing the distance between prototypes.

Prototype Computation with Support Set An N-Way-K-Shot meta task is separated into a support set and query set, which share the same classes of DroidMal. Prototypes of each class are obtained according to **Step 3** in **Algorithm 1**. For example, in a 5-Way-10-Shot meta task, a support set and a query set are randomly separated. Both of them contain 5 classes of DoridMal that embedded by convolutional mapping model. Then, 5 prototypes are attained by averaging the embedded vectors of each class in support set.

Model Optimization with Aggregation of Query Set By computing the distance between vectors in query set and the corresponding prototypes, loss is obtained and is used to optimize convolutional mapping model, which endows the mapping model with the capability to aggregate the DoridMal of same class in high dimension metric spatial. Specifically, Euclidean distance is computed between vectors in query set and the corresponding prototypes. Loss is obtained by **Step 5** in **Algorithm 1**.

In summary, the training and testing process of our proposed Prototype-based Classifier is described as **Algorithm 1**.

3.4 Detecting Known and Novel Android Malware Traffic

Employed with different strategies of meta-learning, AMDetector can classify large-scale known DoridMal encrypted traffic and the novel ones. Figure 4 illustrates the process of classification.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_33

Algorithm 1 Training and Testing Framework of Prototype-based Classifier

Input:
The meta set D.
The mapping model $f(x)$
Unlabel sample x

Output:
Training: Mean of Loss L, Testing: Predicted Label $\hat{y}$ of x

1: Embedding samples in D, obtaining $f(D)$.
2: Randomly dividing $f(D)$ into Support Set D_S and Query Set D_Q .
3: Compute prototypes $\mathbf{P}$ of each class from Support set D_S with n classes, where $\mathbf{P} = \{P_N | P_N = \frac{1}{|D_N|} \sum_{x_i \in D_N} x_i, N = 1, 2, ..., n\}$ and $D_N \subseteq D_S$, where D_N refers to the samples with same label.

Training:

4: Computing the distance $Dst(D_Q, \mathbf{P})$ between samples from D_Q and the corresponding Prototype $P_i \in \mathbf{P}$.
5: Updating Loss by computing

$$L \leftarrow L + \frac{1}{|D_Q|} \left[Dst(f(x), \mathbf{P}) + \log \sum \exp(-Dst(f(x), \mathbf{P})) \right]$$

where $x \in D_Q$.

Testing:

6: Computing distance $Dst_x(x, \mathbf{P})$ between x and prototypes $\mathbf{P}$, where $Dst_x(x, \mathbf{P}) = \{Dst(x, P_N) | P_N \in \mathbf{P}, N = 1, 2, ..., n\}$.
7: Obtaining the predicted label $\hat{y}$ of x, where $\hat{y} = argmin(Dst_x)$.
8: **return** Training: L, Testing: $\hat{y}$

Known Android Malware Traffic Classification In the field of known DoridMal encrypted traffic classification, label set in testing is the subset of that in training, where $Labels(Testing) \subseteq Labels(Training)$. Assuming that the number of classes in training set is $|Labels(Training)|$, the number in testing set $|Labels(Testing)| \leq |Labels(Training)|$. Figure 4 illustrates the process of classification. With N-Way-K-Shot detection strategy, N is set to $|Labels(Testing)|$ and $K = |$Testing Set$|$. Then, by calculating the distance between the query set and the prototype of the meta task in the testing set, the sample is classified into the category with the smallest distance between the corresponding prototype.

Novel Android Malware Traffic Detection AMDetector is capable to classify large-scale DoridMal encrypted traffic precisely, as well as to detect the novel DoridMal traffic, even if the number of the available samples is small. Assuming that the available dataset is Set_U and the known set is Set_K, novel set $Set_N = Set_U - Set_K$, where $Set_K \cap Set_N = \varnothing$. Figure 4 demonstrates the framework of novelty detection. Model is trained with Set_K, applied with N-Way-K-Shot training strategy. During testing, N is set to $|Labels(Set_N)|$. The classification process is similar to that when classifying the known.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_33

4 Experiment and Result

In this section, we evaluate our proposed model on two public datasets to verify the rationality and effectiveness of AMDetector. First, we briefly introduce the two datasets we use. Then, we describe the experimental setting and the state-of-the-arts, which are considered as baseline. Finally, we discuss the experimental results on known and novel DroidMal encrypted traffic classification in detail.

4.1 Experiment Preparation

Dataset Organization We evaluated our model on two datasets, including AndMal2017 [12] and USTC2016 [22]. Specifically, AndMal2017 consists of 42 classes in 4 families of encrypted traffic generated by DoridMal. USTC2016 contains traffic produced by 10 classes of malware and 10 benign software.

AndMal2017 contains 42 classes of DroidMal from 4 major Android malware families, including Adware, Ransomware, Scareware and SMS Malware, of each contains around 10 classes of DroidMal. Encrypted traffic data is captured by running DroidMal in sandbox, which is used to represent the corresponding binary sample. After session image translation, around 2,000 images can be obtained from each class, which extends the insufficient binary samples of DroidMal.

USTC2016 is the second dataset included in our experiment, which consists of traffic data generated by 10 classes of malware and 10 classes of benign software from Windows. To verify the generalization of AMDetector, USTC-MW is regenerated by regrouping 10 classes of Windows malware traffic from the original dataset. USTC-MW contains 10 classes of malware traffic, including Cridex, Geodo, Htbot, Miuref, etc. After image translation, 1,000 samples from each class contribute to USTC-MW.

Evaluation Metrics We compare AMDetector against the state-of-the-arts with four standard evaluation metrics that is commonly used in the task of classification, including Overall Accuracy (OA), Precision (Pr), Recall (Re) and F1-score (F1).

4.2 Experiment Design

Baseline Evaluation AMDetector is compared against the state-of-the-art baselines for DoridMal traffic classification, including XGBoost-based method [17], RandomForest-based method [7], FlowPrint [8], conversation-level Features-based method [1], CNN-based method [21], GAN-based method [14], SiameseNet [11], FS-Net [13], TripletNet [9] and RBRN [29]. Note that part of the baselines cannot detect novel DroidMal, so they will be absent from evaluations of novelty detection (Section 4.4 and 4.5).

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_33

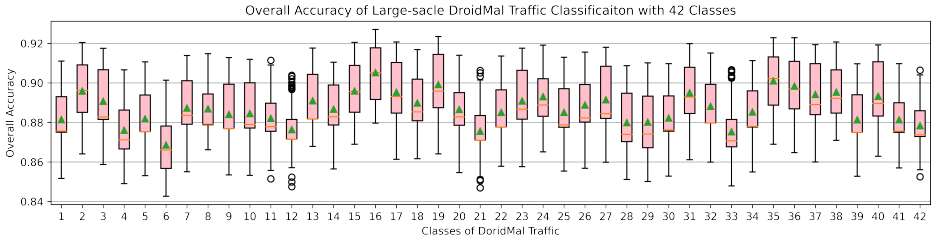


Fig. 5. Results on Large-scale DoridMal Traffic Classification with 42 classes.

Table 1. Comparison Results on Large-scale DroidMal Traffic Classification with 42 Known Classes.

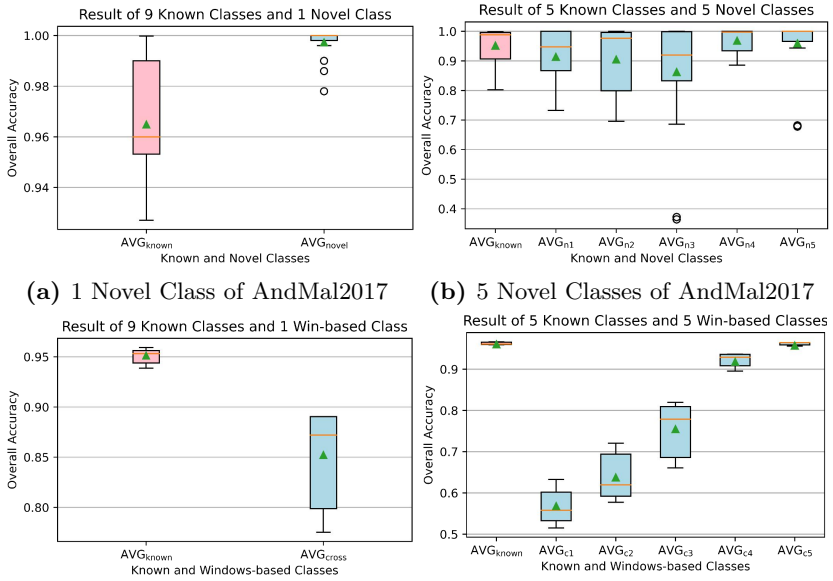
Methods	**OA**	**Pr**	**Re**	**F1**
XGBoost [17]	23.50	25.98	22.78	22.31
Random Forest [7]	27.43	27.22	26.39	26.46
FlowPrint [8]	20.26	21.77	19.84	19.61
Conversation Features [1]	66.71	39.71	41.09	40.38
CNN [21]	75.36	75.36	73.97	74.65
GAN [14]	66.52	67.60	66.44	67.02
FS-Net [13]	56.41	58.64	57.74	58.19
RBRN [29]	45.71	45.19	44.68	44.93
SiameseNet [11]	82.77	85.06	82.88	83.96
TripletNet [9]	67.33	67.76	69.93	68.83
AMDetector	**88.34**	**88.62**	**90.55**	**89.58**

Experimental Setting We first evaluate on AndMal2017 to classify large-scale encrypted traffic from known DroidMal. Then, novelty detection on unknown DoridMal encrypted traffic is performed on the same dataset. Finally, cross-platform classification is evaluated on USTC-MW. Specifically, we employed Adam as the optimizer of our model and trained for 100 epochs with learning rate of 0.01. The experiments were performed using the following hardware and software platforms: Intel i7-9750 @2.6GHz, 16GB RAM, NVIDIA GeForce RTX2060; Windows 10, CUDA 10.1, and PyTorch 1.0.1.

4.3 Classification on Large-scale Android Malware Encrypted Traffic

We evaluate the following well-designed experiments to validate the rationality and advancement of AMDetector. First, we evaluate on AndMal2017 to classify large-scale DroidMal encrypt traffic (42 classes detection). Specifically, 5-Way-10-Shot strategy is employed with a 42-classses training set when optimizing the model. During testing, N is set to 42 to evaluate large-scale classification. Note that N is set to the number of classes included in testing set to predict the testing data integrally. Table 1 illustrates the classification results on 42 classes

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_33



(a) 1 Novel Class of AndMal2017 **(b)** 5 Novel Classes of AndMal2017

(c) 1 Novel Class of USTC-MW **(d)** 5 Novel Classes of USTC-MW

Fig. 6. (a) and (b): Results of Novelty Detection on AndMal2017. (c) and (d): Results of Cross-platform Detection on USTC-MW.

of DoridMal traffic, of which our model achieves superior performance compared against the state-of-the-arts. Also, Figure 5 shows the results of 50 independent evaluations. It is not an easy task for conventional Softmax-based classifiers to classify large-scale of DroidMal traffic directly. However, AMDetector can maintain eye-catching performance, supported by the superiority of meta-learning strategy.

4.4 Novel Android Malware Encrypted Traffic Detection

In order to validate the ability to classify novel classes, we evaluate AMDetector on the regenerated DroidMal2017. Note that only the baselines that support novelty detection will be evaluated in this part. Two experiments are performed in this part, including 1-way and 5-way detection. In 1-way novelty detection, we randomly splintered 1 class out of the 42 classes of AndMal2017. Specifically, the number of classes in training set N_{known} is set to 41 and the number of classes in novel set N_{novel} is 1. During detection phase, 9 classes of known classes are randomly picked up to reconstruct a testing set, which consists of 9 known classes and 1 novel class. Similarly, in 5-way novelty detection, N_{konwn} and N_{novel} are set to 37 and 5, respectively. During detection, testing set that contains 5 known classes and 5 novel classes is obtained. Figure 6a, 6b and Table 2 show the results after repetitive evaluation, where $N_novel = 1$ and 5, respectively. AVG_{known} refers to the average OA of the known classes while

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_33

Table 2. Novel DroidMal Traffic Classification, where the $N_{novel} = 1, 5$ respectively.

Mode	$N_{novel} = 1, N_{known} = 9$				$N_{novel} = 5, N_{known} = 5$			
	OA	Pr	Re	F1	OA	Pr	Re	F1
CNN [21]	78.15	78.14	68.81	73.18	48.75	45.89	38.73	42.01
GAN [14]	72.07	72.48	45.32	55.77	42.09	43.19	25.87	32.36
SiameseNet [11]	92.76	93.33	92.76	93.04	89.26	90.00	89.26	89.63
Flowprint [8]	87.63	88.93	87.66	88.29	84.52	85.61	86.01	85.81
FS-Net [13]	92.11	93.41	93.69	93.55	91.87	90.99	91.42	91.20
TripletNet [9]	96.03	96.06	96.03	96.04	93.86	94.00	93.86	93.93
AMDetector	**99.77**	**99.80**	**99.83**	**99.81**	**95.06**	**95.32**	**95.95**	**95.63**

Table 3. Cross-platform Malware Traffic Classificaiton, where the $N_{cross} = 1, 5$ respectively.

Mode	$N_{cross} = 1, N_{known} = 9$				$N_{cross} = 5, N_{known} = 5$			
	OA	Pr	Re	F1	OA	Pr	Re	F1
CNN [21]	57.32	81.09	31.93	45.82	10.92	34.73	10.72	16.38
GAN [14]	41.59	58.31	27.99	37.82	5.10	17.93	6.54	9.58
SiameseNet [11]	91.15	92.07	91.16	91.61	64.57	61.63	64.57	63.07
Flowprint [8]	82.71	83.06	83.75	83.40	71.85	72.82	72.09	72.45
FS-Net [13]	85.13	84.62	84.97	84.79	75.14	76.47	76.25	76.36
TripletNet [9]	89.89	90.47	89.87	90.18	69.08	68.25	69.08	68.66
AMDetector	**93.59**	**93.24**	**95.17**	**94.19**	**81.59**	**82.08**	**83.23**	**82.65**

AVG_{c_n} refers to the average OA of the $n-th$ novel class. It is encouraged to see that AMDetector performs superiorly when detecting diverse novel classes. Meanwhile, AMDetector can achieve stable performance on known and novel classes classification synchronously.

4.5 Cross-Platform Malware Detection

In order to verify the generalization of AMDetector, AMDetector is trained with AndMal2017 and tested with USTC-MW. Note that only the baselines supporting novelty detection will be evaluated in this part. Following the same principle in Section 4.4, we randomly select 1 or 5 classes from USTC-MW and 9 or 5 classes from AndMal2017 to construct testing set, respectively. Figure 6c, 6d and Table 3 illustrate the results, where N_{novel} is set to 1 and 5, respectively. The results prove that our well-trained model can transfer to classify the unseen malware traffic, even if the traffic is generated from malware on Windows.

5 Conclusion

In this paper, we convert the few-shot Android malware detection to the problem of encrypted network traffic classification and proposed AMDetector, a ses-

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_33

sion image-based model with meta-learning to tackle the above issues. First, DoridMal encrypted traffic is split with session and transferred to gray-scale images. Then, after embedding the images into a high dimension metric spatial, a prototype-based classifier is applied to separate the samples linearly. Well-designed experiments are evaluated on 2 public datasets and verify the efficiency and generality of our model when classifying large-scale DoridMal traffic, which is superior to the state-of-the-arts. Moreover, our model is capable to detect and classify the encrypted traffic that is unseen in training set, even if it is generated by the malware from different Operation Systems.

6 Acknowledgment

This work was supported by the National Natural Science Foundation of China (Grant U2003111, 61871378).

References

1. Abuthawabeh, M., Mahmoud, K.: Enhanced android malware detection and family classification, using conversation-level network traffic features. Int. Arab J. Inf. Technol. (2020)
2. Arora, A., Garg: Malware detection using network traffic analysis in android based mobile devices. In: 2014 Eighth International Conference on Next Generation Mobile Apps, Services and Technologies (2014)
3. Arshad, S., Shah: Android malware detection & protection: a survey. International Journal of Advanced Computer Science and Applications (2016)
4. Bai, Y., Xing: Unsuccessful story about few shot malware family classification and siamese network to the rescue. In: Proc. of ICSE (2020)
5. Celik, Z.B., Walls: Malware traffic detection using tamper resistant features. In: MILCOM 2015-2015 IEEE Military Communications Conference (2015)
6. Chan, P.P., Song: Static detection of android malware by using permissions and api calls. In: Proc. of ICML (2014)
7. Chen, R., Li: Android malware identification based on traffic analysis. In: International Conference on Artificial Intelligence and Security (2019)
8. van Ede, T., Bortolameotti: Flowprint: Semi-supervised mobile-app fingerprinting on encrypted network traffic. In: Proc. of NDSS (2020)
9. Hoffer, E., Ailon, N.: Deep metric learning using triplet network (2014)
10. Hou, S., Saas: Deep4maldroid: A deep learning framework for android malware detection based on linux kernel system call graphs. In: 2016 IEEE/WIC/ACM International Conference on Web Intelligence Workshops (WIW) (2016)
11. Jmila, H., Khedher: Siamese network based feature learning for improved intrusion detection. In: Proc. of ICONIP (2019)
12. Lashkari, A.H., Kadir: Toward developing a systematic approach to generate benchmark android malware datasets and classification. In: 2018 International Carnahan Conference on Security Technology (ICCST) (2018)

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_33

13. Liu, C., He, L., Xiong, G., Cao, Z., Li, Z.: Fs-net: A flow sequence network for encrypted traffic classification. In: IEEE INFOCOM 2019-IEEE Conference on Computer Communications (2019)
14. Liu, Z., Li: Efficient malware originated traffic classification by using generative adversarial networks. In: 2020 IEEE Symposium on Computers and Communications (ISCC) (2020)
15. Onwuzurike, L., Mariconti: Mamadroid: Detecting android malware by building markov chains of behavioral models (extended version). TOPS (2019)
16. Peiravian, N., Zhu, X.: Machine learning for android malware detection using permission and api calls. In: Proc. of ICTAI (2013)
17. Sharan, A., Radhika, K.: Machine learning based solution for detecting malware android applications. Machine Learning (2020)
18. Snell, J., Swersky: Prototypical networks for few-shot learning. In: Proc. of NeurIPS (2017)
19. Spreitzenbarth, M., Freiling: Mobile-sandbox: having a deeper look into android applications. In: Proceedings of the 28th Annual ACM Symposium on Applied Computing (2013)
20. Tang, Z., Wang, Q., Li, W., Bao, H., Liu, F., Wang, W.: Hslf: Http header sequence based lsh fingerprints for application traffic classification. In: International Conference on Computational Science (2021)
21. Wang, W., Zhu: Malware traffic classification using convolutional neural network for representation learning. In: 2017 International Conference on Information Networking (ICOIN) (2017)
22. Wang, W., Zhu, M.: End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In: 2017 IEEE International Conference on Intelligence and Security Informatics, ISI 2017, Beijing, China, July 22-24, 2017 (2017)
23. Wang, Y., Yao: Generalizing from a few examples: A survey on few-shot learning. ACM Computing Surveys (2020)
24. Wang, Z., Fok, K.W., Thing, V.L.: Machine learning for encrypted malicious traffic detection: Approaches, datasets and comparative study. Computers & Security (2022)
25. Wong, M.Y., Lie, D.: Intellidroid: A targeted input generator for the dynamic analysis of android malware. In: NDSS (2016)
26. Yan, L.K., Yin, H.: Droidscope: Seamlessly reconstructing the {OS} and dalvik semantic views for dynamic android malware analysis. In: USENIX12 (2012)
27. Yang, W., Kong: Malware detection in adversarial settings: Exploiting feature evolutions and confusions in android apps. In: Proc. of ACSA (2017)
28. Yuan, Z., Lu: Droiddetector: android malware characterization and detection using deep learning. Tsinghua Science and Technology (2016)
29. Zheng, W., Gou, C., Yan, L., Mo, S.: Learning to classify: A flow-based relation network for encrypted traffic classification. In: Proc. of WWW (2020)
30. Zhu, H.J., You: Droiddet: effective and robust detection of android malware using static analysis along with rotation forest model. Neurocomputing (2018)

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_33