IoT Botnet-Traffic Classification Using Few-Shot Learning

Davide Di Monda
University of Napoli Federico II
Napoli, Italy
IMT School for Advanced Studies
Lucca, Italy
davide.dimonda@unina.it

Giampaolo Bovenzi
University of Napoli Federico II
Napoli, Italy
giampaolo.bovenzi@unina.it

Antonio Montieri
University of Napoli Federico II
Napoli, Italy
antonio.montieri@unina.it

Valerio Persico
University of Napoli Federico II
Napoli, Italy
valerio.persico@unina.it

Antonio Pescapè
University of Napoli Federico II
Napoli, Italy
pescape@unina.it

***Abstract*—The Internet of Things (IoT) is experiencing a constant expansion, embedding connectivity into everyday objects for increased efficiency. Despite this, security vulnerabilities pose a growing concern because IoT devices often lack robust security measures, leaving room for IoT botnet malware action and underlining the critical need for increased IoT security. During the last years, Machine Learning (ML) and Deep Learning (DL) have offered effective tools against IoT attacks, but these solutions struggle with identifying novel threats. In fact, the dynamic nature of IoT ecosystems requires data-driven systems capable of responding promptly to emerging threats, characterized by the limited availability of samples for training.**

In this context, we exploit *Few-Shot Learning (FSL) to effectively identify emerging network attacks within the traffic generated by IoT devices* by performing *botnet-traffic classification*. In detail, FSL enables ML and DL models to recognize and adapt to novel classes of attack traffic with minimal available samples, tackling class imbalance issues between high-frequency and low-frequency attacks (which generate high and low network traffic, respectively). This strategic integration of FSL is crucial in enhancing overall IoT security, providing a proactive approach to handle dynamic and imbalanced scenarios, and ensuring the resilience of interconnected systems. The experimental evaluation is conducted on the publicly available `IoT-23` dataset. The results highlight that the best FSL approach obtains the highest performance figures with just 3 shots, scoring 92% F1-score when discriminating low-frequency botnet malware. Noteworthy, satisfactory performance (up to 93% F1-score) is achieved also in misuse detection, proving the capability to distinguish between legitimate and malicious traffic.

***Index Terms*—Botnet-Traffic Classification, Intrusion Detection, Internet of Things, Deep Learning, Few-Shot Learning, Network Security.**

I. Introduction

The Internet of Things (IoT) represents a technological paradigm shift, providing everyday objects with Internet connectivity to enhance efficiency and functionality. Its diffusion is rapidly reshaping industries, from smart homes to healthcare, fostering a seamless exchange of data.[1]

However, the rapid expansion of the IoT has also brought increased concerns regarding security vulnerabilities [1]. In fact, while the proliferation of IoT devices has increased the amount of generated traffic data that can be capitalized via Big Data infrastructures for defense purposes [2], it has also expanded the attack surface—leading to unauthorized access, data breaches, and even compromise personal privacy. In this context, malicious actors are encouraged to increasingly target IoT devices, continuously discovering and exploiting new vulnerabilities, and developing either new malware or variants of existing ones.[2] A crucial role in this scenario is played by IoT botnet malware, where interconnected devices form a network of compromised units that can be exploited by malicious actors: attackers remotely control the devices for malicious purposes, such as launching large-scale cyberattacks. Accordingly, strengthening IoT security is crucial to prevent the formation and utilization of such botnet malware [3, 4].

During the last decades, *Machine Learning (ML)* and *Deep Learning (DL)* have proven effective in classifying and mitigating attacks against IoT devices by analyzing patterns and anomalies in network traffic. However, their effectiveness is limited when faced with previously unseen attacks. In fact, ML and DL models rely heavily on historical data on which they are trained, thus exhibiting limitation in identifying novel threats that deviate from established patterns, particularly when they are associated with scarce samples available. Furthermore, the dynamic nature of IoT ecosystems and the ever-evolving tactics of cyberattackers pose challenges for traditional models [5]. Therefore, while ML and DL enhance IoT security, *there is a pressing need for adaptive intrusion detection systems that can promptly recognize and respond to*

[1]https://bit.ly/idc-future-of-industry-ecosystems
[2]https://bit.ly/kaspersky-iot-attacks-doubling

979-8-3503-2445-7/23/$31.00 ©2023 IEEE

emerging, underrepresented, and previously unseen threats in a timely manner.

IoT attacks can be categorized as high-frequency and low-frequency attacks, with the former that generate a higher amount of traffic than the latter. For instance, Distributed Denial-of-Service (DDoS) attacks usually flood the network with an overwhelming volume of traffic, often resulting in numerous data samples (viz. high-frequency). On the other hand, establishing Command and Control (C&C) sessions or downloading infected payloads generate fewer amount of traffic (viz. low-frequency), making them harder to detect amid the noise of normal network activity. This imbalance poses a challenge for ML models, as they may prioritize the majority classes and struggle to accurately identify the minority ones, emphasizing the need for robust techniques that can deal with this inherent imbalance and enhance the overall IoT-network security.

Few-Shot Learning (FSL) emerges as a promising solution to address the challenges posed by the lack of samples in IoT traffic data. This paradigm enables ML/DL models to effectively recognize novel attack classes (i.e. unseen during the training of the models) with only a minimal number of examples available for the adaptation (as in the case of low-frequency attacks). This adaptability enhances the overall robustness of the models, making them more adept at handling the intricacies of dynamic and imbalanced IoT-network security scenarios.

In this work, our contributions are:

1) the design, implementation, and validation of FSL solutions for botnet-traffic classification, with the achieved objective of effectively recognizing the traffic generated by IoT botnet malware that are underrepresented, i.e. characterized by low-frequency attacks;
2) the thorough evaluation of design choices concerning the selected network input (i.e. traffic features and number of packets) and FSL hyperparameters, such as the number of shots (i.e. samples) used for training;
3) the extension of the best FSL solution to misuse detection tasks (both multi-class and binary), demonstrating its capability to achieve satisfactory detection performance even in presence of benign (viz. non-attack) traffic.

Following this introduction, Section II details the adopted methodology, covering FSL paradigms, specific FSL approaches, and related input data. In Section III, we report the experimental setup including dataset characteristics, learning hyperparameters, metrics, and implementation details; then, we show the results of four experimental analyses, addressing feature combination, hyperparameter variation, sensitivity analysis, and misuse detection. Section IV outlines relevant studies, comparing them to the investigation conducted in the present work. Finally, Section V concludes by summarizing results and discussing potential future developments.

II. Classifying Botnet Traffic with FSL

In this section, we provide a brief description of methodological aspects pertaining to the integration of FSL into the task of classifying botnet traffic. The *design choices* encompass establishing the *traffic object* to be classified and the associated *input data* fed to the *embedding function*, which is responsible for extracting relevant features. Subsequently, the particular FSL approach must be determined by discerning its underlying learning paradigm, i.e. meta-learning or transfer-learning. Finally, the FSL approaches we adopt are described within the framework of their respective paradigms.

A. Input Data and Embedding Function

In our analysis, we consider the *bidirectional flow* (*biflow*) as traffic object. A biflow aggregates all those packets that share the same 5-tuple, which is defined by source and destination IP addresses, source and destination port numbers, and the transport-level protocol, regardless of the direction of the communication, meaning that source and destination are interchangeable.

Following as proposed in [6, 7], the FSL approaches exploited herein are fed with a set of $M = 6$ features extracted from the first N_p packets of each biflow [8]. These features are: (*i*) the byte count in the network packet, (*ii*) the packet direction (either -1 or 1, representing downstream and upstream, respectively), (*iii*) the TCP window size (0 for UDP packets), (*iv*) the inter-arrival time (i.e. the time elapsed since the arrival of the previous packet), (*v*) the Time-to-Live (TTL), and (*vi*) the TCP flags (encoding the 8-bit representation in its base-10 form). The input data undergo a Min-Max normalization, resulting in a range of values between 0 and 1.

Each FSL approach leverages an embedding function needed to output a feature vector in the embedded space. Herein, we use a well-known DL architecture proposed for IoT-traffic classification in [6]. The embedding function consists of two bidimensional convolutional layers interleaved with max-pooling and batch normalization. For detailed hyperparameters and architecture specifics, refer to [6].

B. Few-Shot Learning Paradigms

1) Meta-Learning: In the context of FSL, meta-learning is employed in conjunction with *episode learning*, which involves training the model using numerous *tasks* (or *episodes*) comprised of few samples. In order to do that, let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{I}$ be a dataset, where x_i represents the input data and $y_i \in \mathcal{Y}$ denotes the corresponding label (e.g., the attack class). Here, I is the number of samples in the dataset, and $\mathcal{Y}$ is the set of possible classes.

The first step consists in splitting the dataset into three subsets, namely, $\mathcal{D} : \{\mathcal{D}_{nf_1}, \mathcal{D}_{nf_2}, \mathcal{D}_f\}$.[3] This division is done in such a way that each subset has a *different label space* from the others—namely $\mathcal{Y}_{nf_1} \cap \mathcal{Y}_{nf_2} = \mathcal{Y}_{nf_1} \cap \mathcal{Y}_f = \mathcal{Y}_{nf_2} \cap \mathcal{Y}_f = \emptyset$, where $\mathcal{Y}_{nf_1}$, $\mathcal{Y}_{nf_2}$, and $\mathcal{Y}_f$ represent the classes of their respective datasets: $\mathcal{D}_{nf_1}$, $\mathcal{D}_{nf_2}$, and $\mathcal{D}_f$.

The subsequent step entails the actual task sampling from one of the three subsets (depending on the specific learning phase as detailed below). This is achieved by considering

[3] In this case, $I_f \ll I_{nf_1}$ and $I_f \ll I_{nf_2}$, where I_f, I_{nf_1}, and I_{nf_2} are the number of samples in $\mathcal{D}_f$, $\mathcal{D}_{nf_1}$, and $\mathcal{D}_{nf_2}$, respectively.

the triplet of values $\langle N, K_s, K_q \rangle$, where: (*i*) N (*N-way*) is the number of classes to be sampled; (*ii*) K_s (*K-shot*) is the number of samples per class (i.e. the *support set*) allocated for model training; and (*iii*) K_q is the number of samples per class (i.e. the *query set*) designated for measuring the model error. The outcome of this process is referred to as an N-*way* K-*shot classification task*. Notably, each so-made episode encompasses limited training data—namely, $N \cdot K_s$—to mimic the operational (viz. inference) scenario.

The model is then trained over several epochs. In each of them, it learns from a wide range of tasks sampled from $\mathcal{D}_{nf_1}$. This phase is referred to as *meta-training*. At the end of each epoch, *meta-validation* is conducted using $\mathcal{D}_{nf_2}$ in the same episodic fashion. Finally, during the *meta-testing* phase, the performance of the trained model is evaluated and averaged over multiple tasks obtained from $\mathcal{D}_f$. It is worth noting that the model is tested on instances belonging to classes different from those used in the training set to evaluate its generalization ability.

2) Transfer Learning: Transfer learning aims to initially train the model on a large dataset and subsequently adapt it to a set of data belonging to a label space different from that of the training data. Therefore, we split the dataset $\mathcal{D} : \{\mathcal{D}_{nf}, \mathcal{D}_f\}$, where $\mathcal{D}_{nf} (= \mathcal{D}_{nf_1} \cup \mathcal{D}_{nf_2})$ encompasses different classes from the ones in $\mathcal{D}_f$, namely $\mathcal{Y}_{nf} \cap \mathcal{Y}_f = \emptyset$, with $\mathcal{Y}_{nf}$ and $\mathcal{Y}_f$ being the classes of $\mathcal{D}_{nf}$ and $\mathcal{D}_f$, respectively. The first learning task (*pre-training*) utilizes a portion of data extracted from $\mathcal{D}_{nf}$ (e.g., via hold-out) to train the model, and two additional splits for validation and testing. The knowledge acquired in this manner is then transferred and fine-tuned in a second adaptation task (*fine-tuning*) using $\mathcal{D}_f$.

Specifically, to align the evaluation of both meta-learning and transfer-learning approaches, the fine-tuning is conducted episodically. This means the model is adapted by leveraging the support set and is evaluated on the query set, repeatedly across numerous episodes.

C. Few-Shot Learning Approaches

Hereinafter, we present the FSL approaches we leverage to perform botnet-traffic classification. In particular, we adopt 3 approaches belonging to the meta-learning paradigm and 5 to the transfer-learning one, covering a wide range of techniques *some of which*—i.e. `Baseline++`, `RFS` (Rethinking Few-Shot), and `Negative Margin`—*have never been employed for (attack-)traffic classification tasks* (to the best of our knowledge, cf. Sec IV).

The approaches that leverage the **meta-learning paradigm** put in place a common strategy to narrow down the hypothesis space[4] through the knowledge gained from abundant data, thus mitigating the risk of overfitting. They exploit a *comparator* that measures the similarity between the embeddings of the support set and those of the query set to perform the classification. The approaches we consider mainly differ on the particular implementation of the comparator. In detail: (*i*) *Matching Networks* (`MatchingNet`) [9] employ a generalized version of nearest-neighbor with Euclidean distance as metric. (*ii*) *Prototypical Networks* (`ProtoNet`) [10] categorize a sample by computing the Euclidean distance between its embedding and each class centroid—obtained from the support set—known as the *prototype*. (*iii*) `MetaOptNet` [11] leverages a linear Support Vector Machine (SVM) as the comparator trained on the support set.

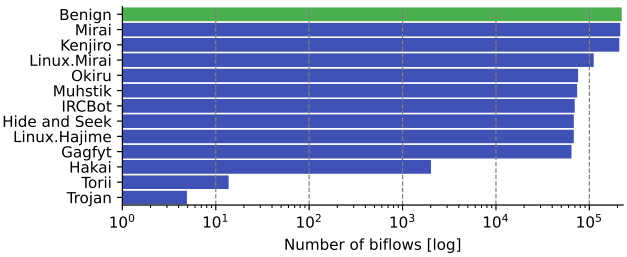


Figure 1: Number of per-class biflows (in log scale) of pre-processed `IoT-23` dataset.

Regarding the **transfer-learning paradigm**, we consider 2 simple and well-known approaches (*Fine-Tuning* and *Freezing*) and 3 that have been proposed in the field of computer vision (`Baseline++`, `RFS`, and `Negative Margin`). *Fine-Tuning* (`FT`) and *Freezing* (`FZ`) share the same pre-training phase on $\mathcal{D}_{nf}$, but they differ in the second task (i.e. the adaptation task involving $\mathcal{D}_f$). Specifically, `FT` adjusts the weights of the entire model, whereas `FZ` freezes the weights of the embedding function, allowing the adaptation only in the last fully-connected layer. `Baseline++` [12] operates similarly to `FZ`, but unlike the latter, it employs a layer that computes the cosine similarity between its weights and feature vectors as a classifier. `RFS` [13] uses sequential knowledge distillation[5] to build an effective model during the pre-training phase. In the second phase, the model weights are frozen. It employs either logistic regression or a nearest-neighbor as a classifier. Lastly, `Negative Margin` [14] replaces the last fully-connected layer with one that computes the cosine similarity, such as `Baseline++`. However, `Negative Margin` differs by subtracting a margin from the cosine similarity. This operation has been proven to improve the generalization ability on the unseen classes.

III. Experimental Setup and Evaluation

A. Experimental Setup

Hereinafter, we describe the experimental setup in terms of (*i*) dataset, (*ii*) FSL setup, (*iii*) evaluation metrics, and (*iv*) implementation details.

[4]The hypothesis space encompasses all potential solutions for a given learning problem. The learning algorithm navigates through this space in search of the optimal model, which translates to finding the most suitable parameters/weights that effectively capture the data and exhibit strong generalization to unseen samples.

[5]In the i^{th} learning cycle, *sequential knowledge distillation* involves using the knowledge of a teacher, where the teacher is the same model from the previous $(i-1)^{th}$ cycle.

1) Dataset: In this study, we evaluate the FSL approaches for botnet-traffic classification using the `IoT-23` dataset [15]. This dataset was collected at the Stratosphere Laboratory of the Czech Technical University over the period 2018–2019. It encompasses 23 PCAP traces along with ground-truth files obtained via *Zeek*. These traces depict heterogeneous IoT network traffic conducted within a controlled IoT setting, utilizing an unrestricted network connection and without the implementation of any defensive measures. Specifically, 20 of these captures originate from a compromised *Raspberry Pi* hosting a specific botnet malware, while the remaining 3 captures represent benign network traffic from real IoT devices (i.e. a *Philips HUE Smart Led Lamp*, an *Amazon Echo Home*, and a *Somfy Smart Doorlock*). The dataset undergoes manual labeling at the biflow level, establishing the connection between malicious flows and the corresponding malicious activities of a botnet malware. Non-malicious traffic is straightforwardly labeled as "benign".

A series of scientific and practical motivations led us to select `IoT-23`. Firstly, `IoT-23` showcases a wide array of botnet traffic with their specific activities. This is of particular interest as it provides a real-world scenario to evaluate the applicability of FSL for botnet-traffic classification. Secondly, this dataset has already been extensively employed in various works [8, 16] within the scientific literature, which promotes the reproducibility of the obtained results.

In contrast to our previous work [8], we have shifted our focus to classifying malicious traffic based on the botnet malware that generated it. Specifically, the available labels in `IoT-23` pertain to the activities conducted by botnet malware (e.g., DDoS, port scanning, file downloading, etc.) [15]. However, in this work, we are interested in identifying patterns of malicious traffic at a higher level—the one of the botnet malware—regardless of the specific action they may be undertaking. This enables the implementation of more fine-grained countermeasures. For instance, defensive action can be taken as soon as a botnet malware is identified, followed by a secondary action based on the particular attack that is underway. Furthermore, it enables the usage of hierarchical architectures that are demonstrated to be particularly effective in the IoT context [17].

Considering the botnet malware as labels, `IoT-23` is characterized by a strong class imbalance problem. In fact, *Mirai* has more than $72M$ biflows, whereas *Trojan*, which is the least-populated class, has only 5 biflows. For this reason, the dataset was downsampled in the following way: (*i*) we group samples of attacks based on the botnet malware that generated it (e.g., Mirai-DDoS, Okiru-DDoS, Trojan-C&C, encompassing all existing botnet-attack combinations); (*ii*) we selected the 70^{th} percentile of the frequency for the various botnet-attack combinations; (*iii*) combinations exceeding this value are downsampled by capping at this threshold—the same is done for the *Benign* class. In other words, we maintain a representative sample of attacks conducted by botnet malware, ensuring good coverage of attack variety without over-representing the most frequent ones. The final result of this pre-processing yields a dataset with $1.2M$ biflows. Figure 1 shows the distribution of the 13 final classes.

2) Few-Shot Learning Setup: The FSL setup is described by detailing the learning setup (in terms of dataset splitting and episode generation) and the configuration of FSL approaches.[6]

FSL Paradigm Setup. As outlined in Section II-B, the `IoT-23` dataset is divided into 3 distinct sets: $\mathcal{D}_{nf_1}$, $\mathcal{D}_{nf_2}$, and $\mathcal{D}_f$. Detailing, $\mathcal{D}_{nf_1}$ comprises the 7 most frequent botnet malware (viz. classes). $\mathcal{D}_{nf_2}$ is composed of the following 3 botnet malware in terms of number of generated biflows. It is worth noting that these 10 botnet malware predominantly involve high-frequency attacks[7], such as DDoS and horizontal port scans. Conversely, the least-populated botnet malware (i.e. *Hakai*, *Torii*, and *Trojan*) are encompassed within $\mathcal{D}_f$. Notably, these malware primarily execute low-frequency attacks like C&C and file downloads. This setup assesses whether FSL approaches successfully generalize to *previously-unseen botnet malware that generate few biflows of completely different attacks* compared to those employed in training.

Once the dataset is split, it is necessary to select the triplet $\langle N, K_s, K_q \rangle$ to generate the episodes (see Sec. II-B). Regarding the meta-learning approaches, we use $\langle N = 3, K_s = 2, K_q = 25 \rangle$ for meta-training episodes, while we use $\langle N = 3, K_s = 2, K_q = 3 \rangle$ for meta-validation and meta-testing[8]. For transfer-learning approaches, $\mathcal{D}_{nf}$ is split into two sets using stratified hold-out: 30% is used for testing, while the remaining 70% is further divided for training (90%) and validation (10%). The fine-tuning phase is conducted in the same way as meta-testing, with episodes characterized by $\langle N = 3, K_s = 2, K_q = 3 \rangle$. The approaches belonging to both learning paradigms are evaluated over 1000 episodes.

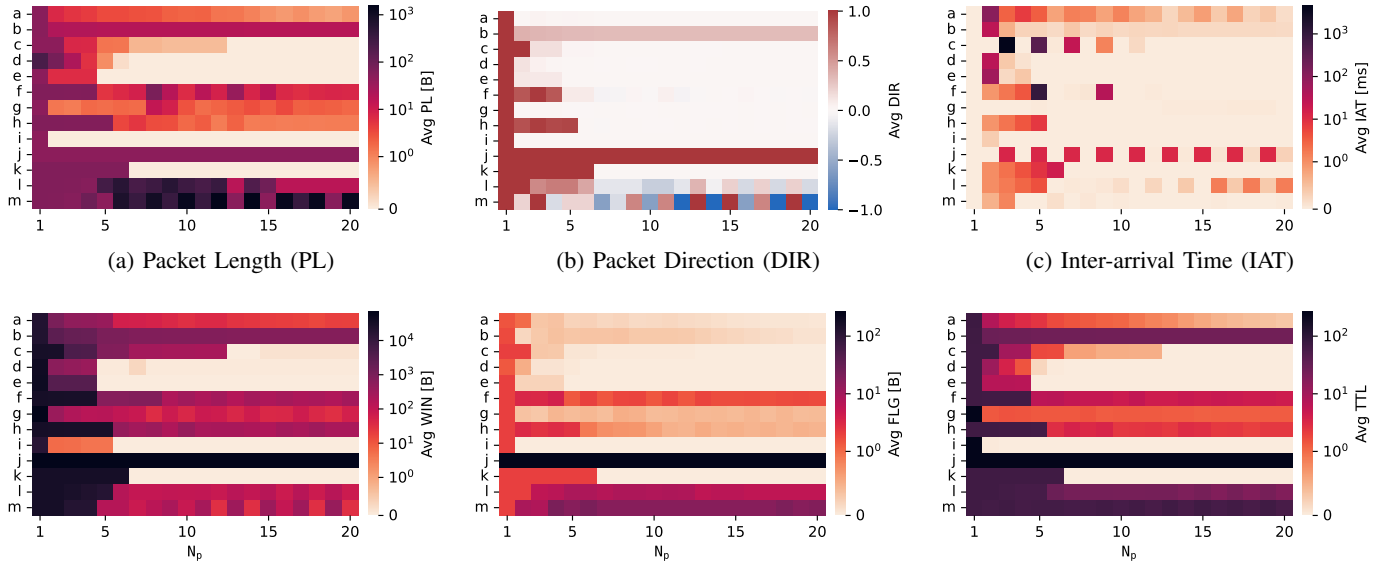
FSL Approaches Setup. The setup of FSL approaches is defined in relation to hyperparameters that are specific to each solution as well as those that are shared among them. As for the **approach-specific hyperparameters**, both `ProtoNet` and `MatchingNet` utilize the Euclidean distance metric. `MetaOptNet` is fine-tuned with a *regularization parameter* set to 0.1 and a maximum of 15 SVM iterations. `RFS` is characterized by $\alpha = \beta = 0.5$, it employs a nearest-neighbor classifier and one *self-distillation cycle*. Lastly, `Negative Margin` is calibrated with a *margin* of -0.1, and with *outer and inner temperatures* set to 30 and 5, respectively.

Regarding the **common hyperparameters**, we use 200 *epochs* and *Adam* optimizer set with 10^{-4} *learning rate*. Transfer-learning approaches, with the exception of `RFS`, employ 50 *inner epochs* and 10^{-3} as *inner learning rate*. To mitigate the overfitting, we exploit a custom *early-stopping*

[6]Setup choices are based on both state-of-the-art results and a preliminary experimental campaign on the validation set, which is not reported here for brevity.

[7]Further detail on the attacks launched by botnet malware can be found here: https://www.stratosphereips.org/datasets-iot23.

[8]Note that for meta-testing, it is not possible to sample more than 5 biflows per episode since 5 is the overall number of samples belonging to the *Trojan* class.



(a) Packet Length (PL) (b) Packet Direction (DIR) (c) Inter-arrival Time (IAT)

(d) TCP Receive Window (WIN) (e) TCP Flags (FLG) (f) Time To Live (TTL)

Figure 2: Characteristics of the biflow contained in `IoT-23` according to the 6 considered informative features (cf Sec. II-A). Values are averaged and transformed with the natural logarithm ($log(1+x)$), with the exception of DIR.

Label encoding: `a` = Benign `b` = Mirai `c` = Kenjiro `d` = Linux.Mirai `e` = Okiru `f` = Muhstik `g` = IRCBot `h` = Hide&Seek `i` = Linux.Hajime `j` = Gagfyt `k` = Hakai `l` = Torii `m` = Trojan

mechanism that monitors both *accuracy and loss on validation set*, it has *minimum delta* of 10^{-3} and *patience* of 20 epochs.

Finally, regarding input data (see Sec. II-A), we use $N_p = 20$ packets, if not stated otherwise. Nevertheless, *we will also show the results of experiments aimed to assess the sensitivity to the number of packets (N_p) and features (M)* in order to identify the best input size for botnet-traffic classification.

3) Performance Metrics: To assess the effectiveness of FSL approaches, we employ the macro F1-score and the silhouette score. The *macro F1-score* is the harmonic mean of precision and recall for each class, which is then averaged across all classes. On the other hand, the *silhouette score* measures how closely a sample aligns with its own cluster in comparison to others. Its values range from -1 (indicating poor similarity) to $+1$ (indicating high similarity). This evaluation is particularly relevant for metric-based approaches (e.g., `MatchingNet`, `ProtoNet`, `RFS`). For both metrics, we show the average per-episode score along with the associated confidence interval, calculated at 95% confidence level, obtained on $\mathcal{D}_f$.

4) Implementation Details: All the analyses are executed on a machine with 12 Intel(R) Xeon(R) CPU E5-2430 v2 @ 2.50GHz and 62GB of memory. We tailored the *PyTorch* implementations of `ProtoNet` and `MetaOptNet` from *learn2learn* framework [18], `MatchingNet` from the code provided in [9], while `Baseline++`, `RFS`, and `Negative Margin` from the *LibFewShot* framework [19]. `FT` and `FZ` are self-built.

B. Experimental Evaluation

In this section, we present the results of our experimental campaign, along with the key take-home messages. Detailing, we (*i*) explore the *impact of selecting diverse features* to construct the input fed to models; (*ii*) perform a *sensitivity analysis assessing the number of packets* from which informative features are extracted; (*iii*) analyze the impact of *leveraging a different number of shots* to classify botnet-malware classes; (*iv*) evaluate the effectiveness of FSL approaches in *solving a misuse-detection task*, specifically including the benign class.

1) Input Features Analysis: This section delves into an examination of input characteristics, focusing on selected features, within the domain of botnet-traffic classification. To this end, this initial analysis involves identifying the *features that contribute most significantly to the performance* of the approaches, while keeping $N_p = 20$, i.e. the first 20 packets of each biflow. We consider the following three combinations of features: (*i*) the full set of features ($M = 6$), as reported in Sec. II-A; (*ii*) a set of $M = 4$ features, namely, direction (DIR), inter-arrival time (IAT), packet length (PL), and window size (WIN) utilized for the classification of mobile-app encrypted traffic in [20]; and (*iii*) a set of $M = 3$ features, obtained by removing IAT from the latter set to further reduce computational overhead. This choice is motivated by the analysis shown in Fig. 2, where the IAT (depicted in Fig. 2c) seems to be less informative compared to the other features. Additionally, the IAT tends to be too sensitive to the specific collection scenario, potentially introducing generalization issues.

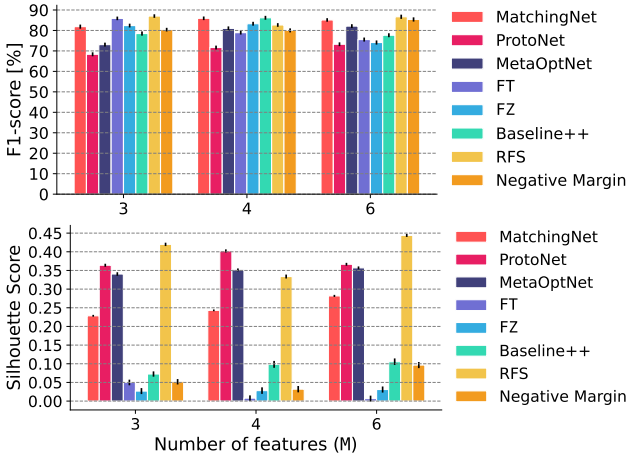


Figure 3: Performance according to F1-score (top) and silhouette score (bottom) for different input features.

Input features encoding: 3 = DIR, PL, WIN 4 = DIR, IAT, PL, WIN
6 = DIR, FLG, IAT, PL, TTL, WIN

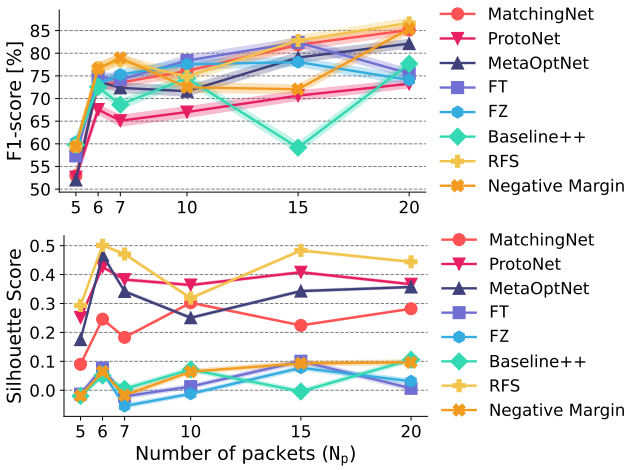


Figure 4: Performance according to F1-score (top) and silhouette score (bottom) when varying the number of packets.

Figure 3 (top) shows the F1-score achieved by the 8 FSL approaches for the 3 sets of input features. Specifically, `RFS` exhibits the best F1-score (87%) when $M = 3$; `MatchingNet` and `Negative Margin` follow close behind. On the other hand, `ProtoNet` stops at 73% F1-score. Overall, the performance for the 3 sets of input are particularly close; this holds true notably for `RFS`, `MatchingNet`, `ProtoNet`, `Negative Margin`, and `Baseline++`, which report small fluctuations. `FT` and `FZ` have degrading performance from $M = 4$ to $M = 6$, unlike `MetaOptNet` that tends to improve.

The silhouette score is shown in Figure 3 (bottom). Notably, we can identify two distinct behaviors: (i) high silhouette score, up to 0.45), exemplified by the meta-learning approaches and `RFS`, and (ii) low silhouette score, up to 0.10, shown by the other transfer-learning approaches. Such results highlight how metric-based techniques, employed by all approaches in (i), positively influence the quality of the embedding space.

Take-home message. *The performance remains relatively consistent across different inputs. This is an interesting result, especially considering that the set with $M = 3$ (i.e. DIR, PL, and WIN) is lighter—requiring no timing mechanisms and extra computations to obtain the IAT—and less dependent on features associated with specific network conditions.*

2) Sensitivity to the Number of Packets: This analysis, complementing that shown in Sec. III-B1, aims to test the *impact of the number of packets (N_p) per biflow*. We consider the full set of features ($M = 6$) as an illustrative example since, as shown before, this choice has no major impact on performance. The goal is to find a *good trade-off between efficiency and performance*—as the number of packets decreases, the model prediction becomes more timely.

In Fig. 4 (top), we observe that the F1-score exhibits a steep rise ($\approx +20\%$) across all approaches when N_p varies from 5 to 6 packets. Such a trend is corroborated by the heatmaps depicted in Fig. 2, where there is high information density in this region (i.e. $N_p \in [1, 6]$). Despite some fluctuations, the rapid increase slows down beyond $N_p = 6$, and the performance figures show a more gradual improvement. Remarkably, at $N_p = 20$, `RFS` is the best-performing, achieving 87% F1-score; `RFS` is closely followed by `Negative Margin` and `MatchingNet`. This result highlights that to achieve the highest performance on low-frequency attacks, it may be necessary to inspect a higher number of packets. Nevertheless, $N_p = 7$ can be selected as a noteworthy compromise between computational efficiency and performance. In fact, the best-performing approach with 7 packets (i.e. `Negative Margin`) shows only a somewhat slight drop (-7.69% F1-score) compared to `RFS` with 20 packets.

Regarding the silhouette score, as shown in Fig. 4 (bottom), similar trends to those observed on the F1-score are highlighted. Specifically, when $N_p = 6$, `RFS` achieves a peak silhouette score of 0.5, indicating an increasing quality of the embedding space. Beyond $N_p = 6$, a plateau is reached, and no significant increases are recorded. Finally, akin to what was observed in the previous analysis (cf. Sec. III-B1), the FSL approaches can be divided into the same two groups—those with high and those with low silhouette scores.

Take-home message. *Optimal performance in detecting low-frequency botnet malware can be achieved by inspecting 20 packets per biflow. However, a notable compromise between computational efficiency and performance can be found with 7 packets, with a slight 7.69% reduction compared to using 20 packets.*

3) Sensitivity to the Number of Shots: In the following, we investigate the performance when considering a *variable number of shots* for the meta-testing/fine-tuning phase. In particular, given that the maximum number of usable biflows per class is 5 (cf. Sec. III-A2 for details), we have defined

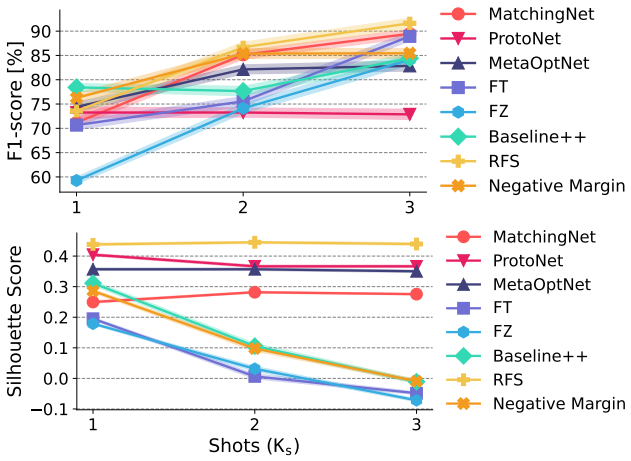


Figure 5: Sensitivity analysis to the number of shots on $\mathcal{D}_f$ in terms of F1-score (top) and silhouette score (bottom).

the following three possible values for K_s : $\{1, 2, 3\}$. The remaining biflows per class, namely K_q : $\{4, 3, 2\}$, are used to form the query set. The input size is fixed and characterized by $M = 6$ and $N_p = 20$.

The experimental results showcase significant improvements in F1-score (see Fig. 5 (top)) across all approaches when passing from $K_s = 1$ to $K_s = 3$, except for `ProtoNet`. In the one-shot task scenario—i.e. $\langle N = 3, K_s = 1, K_q = 4\rangle$—`Baseline++` demonstrates a remarkable F1-score of 78%, followed by `Negative Margin` reaching 76%. For $K_s = 3$, `RFS` significantly outperforms the others with the highest F1-score of 92%; closely behind there are `MatchingNet` and `FT`, both achieving 89% F1-score.

As depicted in Fig. 5 (bottom), the division of approaches into the two groups based on the silhouette score remains consistent with the previous analyses. However, it is worth noting that the gap between the groups is narrower with $K_s = 1$ and tends to broaden as K_s increases to 3. This suggests a degradation in the performance of the embedding function for the approaches utilizing a fully-connected classifier.

Take-home message. *From the sensitivity analysis to the number of shots, it emerges that using only 3 new samples for classifying low-frequency botnet malware leads to high results. In particular, the best-performing approach (i.e. `RFS`) achieves an F1-score of 92%.*

4) Misuse Detection Study: In this section, we conduct a detailed analysis of the results obtained by the best-performing approach, namely `RFS`, configured with the setup $N_p = 20$, $M = 6$, and $K_s = 3$. The primary objective is to validate its effectiveness in performing misuse detection tasks, specifically in distinguishing between low-frequency botnet and legitimate (viz. benign) traffic. To this end, we include the biflows from the benign-traffic class into $\mathcal{D}_f$. Accordingly, the following results are averaged over episodes characterized by a number of ways (N) equal to 4. More specifically, we consider the following traffic classes: *Benign*, *Hakai*, *Torii*, and *Trojan* (the latter three being the classes originally belonging to $\mathcal{D}_f$, see Fig. 1).

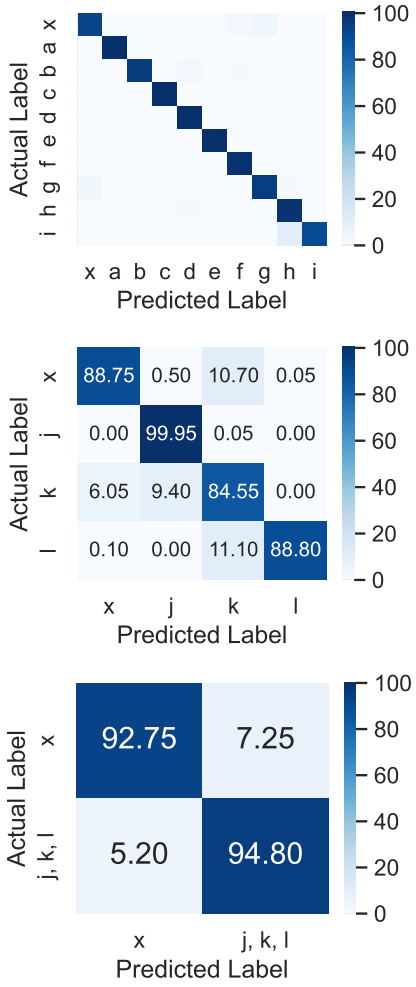


Figure 6: Confusion matrix of `RFS` when tested on benign and high-frequency botnet traffic (i.e. on $\mathcal{D}_{nf}$) (top), benign and low-frequency botnet traffic (i.e. on $\mathcal{D}_f$) (middle), and benign vs. malicious traffic (i.e. on the aggregated binary $\mathcal{D}_f$) (bottom).

Label encoding: `x` = Benign, `a` = Gagfyt, `b` = Hide&Seek, `c` = IRCBot, `d` = Kenjiro, `e` = L.Hajime*, `f` = L.Mirai*, `g` = Mirai, `h` = Muhstik, `i` = Okiru, *j* = *Hakai*, *k* = *Torii*, *l* = *Trojan*
* L stands for Linux

To provide a comprehensive evaluation of `RFS` also on high-frequency botnet and benign traffic, in Fig. 6 (top), we first present the results obtained on $\mathcal{D}_{nf}$. The confusion matrix reveals remarkably high performance, with a 97% F1-score. This confirms the `RFS` ability to effectively differentiate between benign and high-frequency botnet traffic. The only instance of a minor error occurs with *Okiru*, which is occasionally confused with *Muhstik*.

Figure 6 (middle) displays the results of the fine-tuning on $\mathcal{D}_f$. Overall, the performance remains high, recording an F1-score of 89%. Among the four traffic classes, *Torii* tends to be the most frequently confused, particularly with *Hakai* and *Benign*. On the other hand, *Hakai* is the class that is most

accurately classified (99.95%).

Finally, in Figure 6 (bottom), we present the performance in the binary task (i.e. *2-ways 3-shots*) obtained by aggregating the low-frequency botnet-malware classes into a single *Malicious* class. The highest error arises from the confusion between the *Benign* (actual) and the *Malicious* (predicted) class (7.25%). The confusion between *Malicious* (actual) and *Benign* (predicted) is lower (5.20%). Overall, `RFS` achieves a notable F1-score of 93% in effectively distinguishing between *Malicious* (i.e. generated by low-frequency botnet) and *Benign* traffic.

Take-home message. *`RFS` demonstrates strong performance in classifying botnet-generated traffic, achieving a remarkable 97% F1-score in distinguishing benign biflows from those generated by high-frequency botnet malware. Although a minor confusion occurs between low-frequency botnet malware and benign biflows, the overall F1-score remains high at 89%. Additionally, `RFS` offers high performance in a binary classification task (benign vs. malicious), reaching a 93% F1-score. These findings underline the effectiveness and robustness of `RFS` in botnet-traffic classification.*

IV. Related Work

In recent years, the application of FSL in the broad domain of network security has garnered significant attention. This is primarily due to its unique ability to classify traffic associated with previously unseen network attacks (i.e. whose traffic samples are not available when training the ML/DL model), a critical ability in the realm of malware-traffic classification.

Table I summarizes the key aspects of related literature, disclosing a surge in publications utilizing FSL for malware-traffic classification from 2020 onwards. However, we can observe that the prevailing literature mostly employs "classic" FSL approaches (e.g., `MatchingNet` [9], `ProtoNet` [10], `RelationNet` [30], and `MAML` [31]), originally introduced in the field of computer vision around 2017–2018. Indeed, the current state-of-the-art landscape has *yet to adopt the more promising and recent approaches* that have emerged within other research fields. In particular, to the best of our knowledge, none of the existing works leverage the transfer-learning paradigm (column **Transfer-Learning** in Tab. I), with all studies exclusively focusing on meta-learning strategies (column **Meta-Learning** in Tab. I). In fact, it is necessary to expand the scope to legitimate-traffic classification (i.e. outside the network security domain) to find recent works dealing with FSL approaches based on transfer-learning [20, 32]. Moreover, the related work lacks comprehensive comparisons between various FSL approaches (i.e. the comparison is commonly performed only against non-FSL baselines), except for the work in [26].

Detailing on the proposed approaches exploited in the literature (column **FSL Approach** in Tab. I), `MatchingNet` serves as inspiration for Huang et al. [21], who introduces an FSL method based on a gating mechanism. *Gates*—akin to soft-classifiers—facilitate the evaluation of the significance of both known and unknown anomalies in relation to a test instance.

`RelationNet` is another meta-learning approach originally proposed for computer vision tasks, first described in [30]. It works by concatenating embedded training and test samples and feeding them to a relation module, which is a Convolutional Neural Network (CNN) that outputs a similarity score between 0 and 1. Xu et al. [23] also employ `RelationNet` for anomaly detection. Additionally, Zheng et al. [22] utilizes `RelationNet` in conjunction with an *hallucinator* that generates new instances by adding noise to the data.

`ProtoNet` is used by Ouyang et al. [25] for their intrusion detection system designed to counteract malicious attacks against SCADA systems. Similarly, Rong et al. [26] employ a CNN as a feature extractor fed with anonymized traffic data and compare the proposed architecture (based on `ProtoNet`) with `MAML` and `MatchingNet`. Recently, Yang et al. [28] introduce a novel intrusion detection framework called FS-IDS. Their embedding function is fed with input obtained by merging the raw bytes of a network flow (organized as a gray-scale image) and the flow-based statistics processed by an autoencoder. The authors then utilize `ProtoNet` and `RelationNet` as FSL approaches.

Differently, Feng et al. [24], Guo et al. [27], and Lu et al. [29] propose FSL solutions based on *Model-Agnostic Meta-Learning* (`MAML`) [31] to detect anomalous traffic with only few samples. `MAML` is a popular FSL approach that continuously updates the model parameters through a meta-learning process and a fine-tuning phase to quickly adapt to new FSL tasks.

Finally, the employed **Datasets** summarized in the last column of Tab. I, characterize the specific attack-traffic classification task to be faced. To the best of our knowledge, this work is the first that addresses botnet-traffic classification exploiting the capabilities of FSL via both meta-learning and transfer-learning approaches. Moreover, we deepen their performance by investigating different practical configurations (e.g., variable number of biflows, number of packets, and traffic features) and by extending them to misuse-detection tasks.

V. Conclusions and Future Perspectives

In this paper, we aimed to develop a network intrusion detection system capable of effectively classifying *malicious traffic generated by botnet malware executing low-frequency and previously unseen (zero-day) attacks*. To achieve this objective, we adopted *Few-Shot Learning (FSL)* and utilized 8 different approaches, including `MatchingNet`, `ProtoNet`, `MetaOptNet`, `FT`, `FZ`, `Baseline++`, `RFS`, and `Negative Margin`, which belong to two distinct FSL paradigms: meta-learning and transfer-learning. The broad prior knowledge base required by such approaches has been obtained from botnet traffic associated with high-frequency attacks, which provided abundant data for training.

Table I: Related studies focusing on attack-traffic classification using Few-Shot Learning approaches. Papers are arranged chronologically by publication year. The final row provides an overview of the present work. The meaning of acronyms is provided at the bottom of the table.

Paper	**FSL Approach**	**Meta-Learning**	**Transfer-Learning**	**Datasets**
Huang et al. [21], 2020	MN	✓		NSL-KDD
Zheng et al. [22], 2020	RN	✓		ISCX 2012 IDS, ISCX VPN-nonVPN
Xu et al. [23], 2020	RN	✓		ISCX 2012 FS, CIC-IDS2017 FS
Feng et al. [24], 2021	MAML	✓		CICAndMal2017, CIC-IDS2017
Ouyang et al. [25], 2021	PN	✓		SCADA
Rong et al. [26], 2021	PN	✓		MCFP, USTC-TFC, CICInvesAndMal2019, BEN-1, *Self-built*
Guo et al. [27], 2022	MAML	✓		USTC-TFC2016
Yang et al. [28], 2022	PN, RN	✓		CIC-IDS2017
Lu et al. [29], 2023	MAML	✓		*Self-built* combining public datasets
This work	MN, PN, MON, FT, FZ, BL++, RFS, NM	✓	✓	IoT-23

Acronyms: `MatchingNet` (MN), `ProtoNet` (PN), `RelationNet` (RN), `MetaOptNet` (MON), *Model-Agnostic Meta-Learning* (MAML), *Fine-Tuning* (FT), *Freezing* (FZ), `Baseline++` (BL++), *Rethinking Few-Shot* (RFS), *Negative Margin* (NM). ✓ present.

The results highlight several key findings: (*i*) using of a limited set of features (only 3) did not lead to a discernible deterioration of performance; (*ii*) 20 packets per biflow were deemed necessary to achieve optimal performance, but with just 7 packets a good trade-off was achieved, i.e. 7.69% F1-score drop at most; (*iii*) `RFS` attained a satisfactory F1-score of 92% leveraging only 3 new biflows per low-frequency botnet malware, demonstrating its effectiveness in scenarios with extremely limited samples available; (*iv*) `RFS` exhibited high performance in classifying biflows generated by low-frequency (resp. high-frequency) botnet malware and benign traffic, scoring 89% (resp. 97%) F1-score; (*v*) `RFS` demonstrated high proficiency also in binary misuse detection (i.e. benign vs. malicious traffic, with the latter encompassing traffic from different low-frequency botnet malware), attaining an F1-score of 93%.

Moving forward, there are several *promising directions for further research*: (*a*) conducting a *comprehensive deployability study* to assess the practical realization of the proposed intrusion detection approach on real hardware; (*b*) investigating *novel FSL approaches* inspired by the success of `RFS`, e.g., by leveraging knowledge distillation and cluster-based losses to build solid and robust base models; (*c*) developing a *hierarchical FSL approach* that first classifies the botnet-malware type/family and subsequently identifies the specific attack within this botnet-malware family.

Acknowledgements

This work was partially supported by the Italian PNRR MUR "Centro Nazionale HPC, Big Data e Quantum Computing, Spoke9 - Digital Society & Smart Cities" and the "xInternet" project—funded by MUR—within the PRIN 2022 program (D.D.104 -02/02/2022). This manuscript reflects only the authors' views and opinions and the Ministry cannot be considered responsible for them.

The authors would like to thank Mr. Ciro Gallucci, Mr. Lorenzo Pitacoro, and Mr. Luca Tornincasa for their collaboration in the preliminary experiments of this study.

References

[1] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, and N. Ghani, "Demystifying IoT security: An exhaustive survey on IoT vulnerabilities and a first empirical look on internet-scale IoT exploitations," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2702–2733, 2019.

[2] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network intrusion detection for IoT security based on learning techniques," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2671–2701, 2019.

[3] C. Yin, S. Zhang, J. Wang, and N. N. Xiong, "Anomaly detection based on convolutional recurrent autoencoder for IoT time series," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 1, pp. 112–122, 2020.

[4] K. Ferencz, J. Domokos, and L. Kovacs, "Review of Industry 4.0 security challenges," in *IEEE 15th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, 2021, pp. 245–248.

[5] G. Olímpio Jr, L. Camargos, R. S. Miani, and E. R. Faria, "Model update for intrusion detection: Analyzing the performance of delayed labeling and active learning strategies," *Computers & Security*, vol. 134, p. 103451, 2023.

[6] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Network traffic classifier with convolutional and recurrent neural networks for internet of things," *IEEE Access*, vol. 5, pp. 18 042–18 050, 2017.

[7] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Toward effective mobile encrypted traffic classification through deep learning," *Neurocomputing*, vol. 409, pp. 306–315, 2020.

[8] A. Nascita, F. Cerasuolo, D. Di Monda, J. T. A. Garcia, A. Montieri, and A. Pescapè, "Machine and deep learning approaches for IoT attack classification," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2022, pp. 1–6.

[9] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra *et al.*, "Matching networks for one shot learning," *Advances in neural information processing systems*, vol. 29, 2016.

[10] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," *Advances in neural information processing systems*, vol. 30, 2017.

[11] K. Lee, S. Maji, A. Ravichandran, and S. Soatto, "Meta-learning with differentiable convex optimization," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 10 657–10 665.

[12] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. F. Wang, and J.-B. Huang, "A closer look at few-shot classification," *arXiv preprint arXiv:1904.04232*, 2019.

[13] Y. Tian, Y. Wang, D. Krishnan, J. B. Tenenbaum, and P. Isola, "Rethinking few-shot image classification: a good embedding is all you need?" in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*. Springer, 2020, pp. 266–282.

[14] B. Liu, Y. Cao, Y. Lin, Q. Li, Z. Zhang, M. Long, and H. Hu, "Negative margin matters: Understanding margin in few-shot classification," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*. Springer, 2020, pp. 438–455.

[15] S. Garcia, A. Parmisano, and M. J. Erquiaga, "IoT-23: A labeled dataset with malicious and benign IoT network traffic," 2020. [Online]. Available: http://doi.org/10.5281/zenodo.4743746

[16] T. M. Booij, I. Chiscop, E. Meeuwissen, N. Moustafa, and F. T. Den Hartog, "ToN_IoT: The role of heterogeneity and the need for standardization of features and attack types in IoT network intrusion data sets," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 485–496, 2021.

[17] G. Bovenzi, G. Aceto, D. Ciuonzo, V. Persico, and A. Pescapé, "A hierarchical hybrid intrusion detection approach in IoT scenarios," in *GLOBECOM 2020-2020 IEEE global communications conference*. IEEE, 2020, pp. 1–7.

[18] S. M. Arnold, P. Mahajan, D. Datta, I. Bunner, and K. S. Zarkias, "learn2learn: A library for meta-learning research," *arXiv preprint arXiv:2008.12284*, 2020.

[19] W. Li, Z. Wang, X. Yang, C. Dong, P. Tian, T. Qin, H. Jing, Y. Shi, L. Wang, Y. Gao, and J. Luo, "Libfewshot: A comprehensive library for few-shot learning," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 01, pp. 1–18, 2023.

[20] G. Bovenzi, D. Di Monda, A. Montieri, V. Persico, and A. Pescapé, "Few shot learning approaches for classifying rare mobile-app encrypted traffic samples," in *IEEE INFOCOM 2023-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2023, pp. 1–6.

[21] S. Huang, Y. Liu, C. Fung, W. An, R. He, Y. Zhao, H. Yang, and Z. Luan, "A gated few-shot learning model for anomaly detection," in *2020 International Conference on Information Networking (ICOIN)*. IEEE, 2020, pp. 505–509.

[22] W. Zheng, C. Gou, L. Yan, and S. Mo, "Learning to classify: A flow-based relation network for encrypted traffic classification," in *Proceedings of The Web Conference 2020*, 2020, pp. 13–22.

[23] C. Xu, J. Shen, and X. Du, "A method of few-shot network intrusion detection based on meta-learning framework," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3540–3552, 2020.

[24] T. Feng, Q. Qi, J. Wang, and J. Liao, "Few-shot class-adaptive anomaly detection with model-agnostic meta-learning," in *2021 IFIP Networking Conference (IFIP Networking)*. IEEE, 2021, pp. 1–9.

[25] Y. Ouyang, B. Li, Q. Kong, H. Song, and T. Li, "FS-IDS: a novel few-shot learning based intrusion detection system for SCADA networks," in *ICC 2021-IEEE International Conference on Communications*. IEEE, 2021, pp. 1–6.

[26] C. Rong, G. Gou, C. Hou, Z. Li, G. Xiong, and L. Guo, "Umvd-fsl: Unseen malware variants detection using few-shot learning," in *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021, pp. 1–8.

[27] H. Guo, X. Zhang, Y. Wang, B. Adebisi, H. Gacanin, and G. Gui, "Few-shot malware traffic classification method using network traffic and meta transfer learning," in *2022 IEEE 96th Vehicular Technology Conference (VTC2022-Fall)*. IEEE, 2022, pp. 1–5.

[28] J. Yang, H. Li, S. Shao, F. Zou, and Y. Wu, "FS-IDS: A framework for intrusion detection based on few-shot learning," *Computers & Security*, vol. 122, p. 102899, 2022.

[29] C. Lu, X. Wang, A. Yang, Y. Liu, and Z. Dong, "A few-shot based model-agnostic meta-learning for intrusion detection in security of internet of things," *IEEE Internet of Things Journal*, 2023.

[30] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1199–1208.

[31] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International conference on machine learning*. PMLR, 2017, pp. 1126–1135.

[32] I. Guarino, C. Wang, A. Finamore, A. Pescapè, and D. Rossi, "Many or few samples?: Comparing transfer, contrastive and meta-learning in encrypted traffic classification," in *2023 7th Network Traffic Measurement and Analysis Conference (TMA)*, 2023, pp. 1–10.