META MIMETIC: Few-Shot Classification of Mobile-App Encrypted Traffic via Multimodal Meta-Learning

Giampaolo Bovenzi[1], Davide Di Monda[1,2], Antonio Montieri[1], Valerio Persico[1], Antonio Pescapé[1]
[1]University of Napoli Federico II, Napoli (Italy) and [2]IMT School for Advanced Studies, Lucca (Italy)
{giampaolo.bovenzi, davide.dimonda, antonio.montieri, valerio.persico, pescape}@unina.it

***Abstract*—Despite its proven effectiveness in classifying encrypted network traffic, deep learning requires large amounts of labeled data to feed typical data-hungry training processes. Few-shot learning provides means to overcome these limitations, supporting classification tasks related to traffic with few labeled data available. Its extensive investigation in other domains notwithstanding (e.g., computer vision), it has been only preliminarily adopted for classifying encrypted traffic.**

In this work, we design and evaluate META MIMETIC a novel multimodal few-shot learning solution for classifying mobile-app encrypted traffic. The proposal is based on the meta-learning paradigm and introduces enhancements via the adoption of a multimodal feature extractor trained via a novel ad-hoc meta-learning procedure. Since META MIMETIC is orthogonal to the specific few-shot learning approach, in our experimentation, we adapt it to a number of different meta-learning approaches (namely *MatchingNet*, *ProtoNet*, *RelationNet*, *MetaOptNet*, *fo-MAML*, and *ANIL*). We provide an empirical assessment of these approaches, considering the `Mirage-2019` dataset as a test bench. Results show that META MIMETIC represents the best trade-off in terms of performance and complexity in mobile-app traffic classification (up to 91% F1-score) when compared to state-of-the-art solutions. The in-depth analysis of the performance of its components allows us to shed light on the multimodal internal mechanisms and further improve classification performance. Finally, we demonstrate the robustness of our proposal (only $\approx 2\%$ F1-score drop) against the next variations introduced by the TLS 1.3 encryption that may impair the information exploitable by payload-based traffic classifiers.

***Index Terms*—Traffic Classification; Mobile Apps; Encrypted Traffic; Deep Learning; Multimodal; Few Shot Learning; Meta-Learning.**

I. INTRODUCTION

Traffic classification is essential, as it serves as a crucial component in various network management tasks such as resource allocation, billing, accounting, intrusion detection, as well as user activity identification and profiling. Accordingly, it is a constantly evolving research field that requires adaptation to the changing nature of networks and traffic traversing them.

Notably, the classification of mobile-app traffic is of particular interest given the valuable profiling information that can be inferred, which may also imply privacy issues. In fact, it comes with intrinsic challenges given by many apps to discriminate from, frequent app updates, and the presence of common third-party services [1].

Today, traffic classification approaches often use *statistical or time-series features* to address the shortcomings of port-based approaches (applications rely on non-standard or variable port numbers to conceal their traffic) and the challenges due to the increasing adoption of encryption (compromising deep-packet inspection, which may also collide with privacy preservation efforts) [2]. The mentioned features can be employed to feed the training of data-driven *Machine Learning models*. However, the adoption of the latter may not scale well when traffic classes change rapidly over time, as the intervention of domain experts may be required to guide feature extraction. *Deep Learning* (DL) overcomes this limitation via end-to-end techniques, taking advantage of large amounts of labeled data to train supervised models and automatically extracting relevant features from raw data.

Nonetheless, feeding the training of these data-hungry approaches with satisfactorily abundant data is not trivial. Indeed, collecting extensive labeled network-traffic datasets requires huge resources for capturing and segmenting traces and carefully assigning labels, also raising privacy concerns [3, 4]. Furthermore, the fast-evolving nature of traffic exacerbates the problem, resulting in the need for continual collection processes aiming at keeping these large traffic datasets updated. Consequently, the available labeled data are often unbalanced, with limited examples for certain classes, making it challenging to adopt DL techniques that struggle with poor knowledge of some classes (i.e. class imbalance problem).

The goal of *Few-Shot Learning* (FSL) is to properly learn when limited samples are available by capitalizing on the knowledge from related tasks to build a model that can generalize well to new tasks. FSL was first explored in depth in the computer vision domain, leading to a diverse range of available solutions that differ in how the *prior knowledge* is exploited: (i) *Algorithm-based approaches* utilize prior knowledge to modify the search strategy for new parameters [5]; (ii) *Model-based approaches* jointly learn a set of initial tasks that are related to reducing the search space for optimal parameters [6–8]; (iii) *Data-based approaches* utilize prior knowledge to augment the data of the few-shot tasks.

In light of its promising results in other domains, recent studies have explored the application of FSL to address network-traffic classification. However, most networking-related FSL works are based on previous computer-vision solutions, with only minor variations proposed [9–12], and mainly focus on attack-traffic classification [9–11, 13–15].

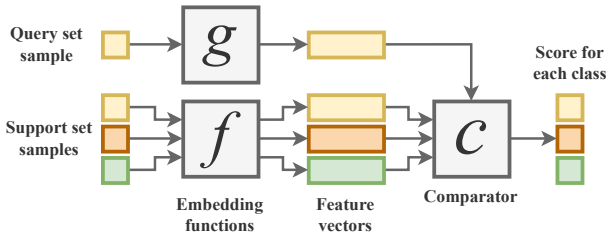


Figure 1: Meta-learning general framework. Support and query samples are compressed via an embedding function (f and g, respectively). Model-based approaches use a comparator c that exploits the embeddings of the support set to classify those of the query set.

In this work, we propose META MIMETIC, a novel solution aiming to improve the performance of FSL for mobile-app encrypted traffic classification. It is based on a *multimodal embedding function* to perform the extraction of the features while capitalizing on its capability to distill informative details *from multiple views* of the same bidirectional flow (i.e. the sequence of its packet fields and payload bytes). The devised solution tackles the non-obvious adaptation of an advanced *multimodal-training procedure* to the general *meta-learning* paradigm. It results in a two-phase training that is agnostically applicable to all the approaches adopting meta-learning.

The *validation of the proposal* considers *six state-of-the-art FSL approaches based on meta-learning*. The proposal is also compared against multiple state-of-the-art single-modal embedding functions, such as 1D- and 2D-Convolutional Neural Networks (1D-CNN and 2D-CNN), also deepening the benefits of advanced multimodal-training strategies against naïve ones. The META MIMETIC *internal mechanisms* are inspected in-depth, providing the analysis of per-modality behaviors and the exploration of few-shot hyperparameters, such as the number of shots, and the capitalization of per-modality embeddings. Finally, we assess *the robustness of the proposed solution* to the adoption of `TLS 1.3`—which extends the encryption also to the TLS handshake—by performing an occlusion analysis of the SNI (Server Name Indication).

The rest of the paper is organized as follows: Sec. II provides the required background and positions our contribution against related works; the proposed META MIMETIC is presented along with meta-learning approaches we evaluated in Sec. III; then, we detail the experimental setup in Sec. IV and discuss the experimental results in Sec. V; conclusions and future perspectives are provided in Sec. VI.

II. BACKGROUND AND RELATED WORK

This section provides a background on the meta-learning paradigm when applied to the few-shot classification task (Sec. II-A). Then, we discuss the most related studies facing traffic classification under few-shot assumptions and frame the present work in relation to them (Sec. II-B).

A. Meta Learning for Few-Shot Learning

Meta-learning refers to the capability of an algorithm of *"learning to learn"*, namely to leverage the accumulated experience to improve the performance on a new task. A meta-learner aims to distill the regularities existing over a distribution of tasks, each having a specific internal structure and objective. Therefore, where the meta-learning goal is building a model that can quickly learn new tasks, FSL aims to build a model that can learn how to distinguish an unknown class when only few samples of that class are available.

To fulfill these goals, meta-learning is combined with *episodic learning*, which consists in organizing the model training in a series of learning tasks each related to a limited number of classes and samples, i.e. N*-way* K*-shot episodes*. In each episode, N classes are randomly sampled (N*-way*) and then two non-overlapping partitions are obtained: (*i*) the *support set* having $N \times K_s$ samples (where K_s defines the K*-shot* setup) and (*ii*) the *query set* having $N \times K_q$ samples.

Formally, the employed dataset $D = \{D_{nf},\ D_f\ \}$ encompasses the samples belonging to two disjoint sets of classes: D_{nf} is used during the training phase of the model and contains the samples from *non-few classes*, while D_f is used during the operational phase and contains the samples from *few-shot classes* (i.e. the least-populated ones). In detail, during the *meta-training*, the model learns from a set of episodes/tasks using the support-set samples and evaluates the error on the query set. Afterward, the operational phase of the model is emulated, assessing its ability to generalize on unknown few-shot classes via a *meta-testing* phase, which leverages an analogous episode-based procedure.

In this work, we exploit meta-learning approaches belonging to the algorithm- and model-based FSL families. For both families, the process of feature extraction is performed by an *embedding function* that maps the samples belonging to a d-dimensional space $\mathbb{R}^d$ into a smaller m-dimensional space $\mathbb{R}^m$ (with $m < d$). In the embedded space, similar samples are closer, whereas dissimilar ones are more easily differentiable.

Algorithm-based approaches use prior knowledge to assist in finding the best set of parameters in the hypothesis space[1] or providing a starting point as close as possible to the optimal hypothesis. In other words, they start from a model already trained on similar tasks and adapt it by fine-tuning the target task. Differently, *model-based approaches* make the hypothesis space smaller via prior knowledge to reduce 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. Figure 1 shows the general framework of meta-learning approaches. Whereas algorithm-based approaches are characterized by the learning procedure for finding optimal parameters, model-based ones differ by the comparator (i.e. the c block in Fig. 1).

B. Related Work

Learning from few samples is a relevant problem that has attracted the interest of researchers from a number of

[1]Hypothesis space represents the range of possible solutions for a given learning problem. The learning algorithm explores this hypothesis space to find the best model, or in other words, the best parameters/weights, that fits the data and generalizes well to unseen samples.

different domains. FSL originated in the field of computer vision [5–8, 16, 17]. Hence, its applications to the networking domain mostly exploit minor variants of solutions designed for computer vision. The recent interest of the networking community in this topic is witnessed by the relevant studies of FSL in this domain [9–15, 18, 19], all dating to 2018–2023.

Most of these works leverage FSL in the context of attack-traffic classification and intrusion detection [9, 10, 13–15]. Notably, all the mentioned studies focus on the problem of classifying *unseen* traffic with just few samples. In other words, we do not consider here studies dealing with FSL that perform the model evaluation on seen classes, i.e. where the main goal is simply mitigating the class imbalance in data.

Most of the works exploit available knowledge via *meta-learning*. Notably, an alternative paradigm to meta-learning, that is *transfer learning*, is considered in [12, 18, 19]. Mostly, the applied FSL approaches belong to *model-based* and *algorithm-based* families.

Going into details, several solutions that exploit meta-learning approaches were recently proposed. Huang et al. [14] developed a solution for network anomaly detection by integrating a gating technique into `MatchingNet` [6]. The gates provide a similarity score that evaluates the importance of known and unknown anomalies against a test instance. Other recent works [9, 11] are based on `RelationNet` [8]. Zheng et al. [9] combines meta-learning with a data-based method, augmenting the training set by adding noise to data. Zhao et al. [11] optimizes the embedding function of the `RelationNet` by leveraging an ensemble of CNNs. While these works have proposed variations from the computer-vision solutions, others have utilized these methods without significant changes, such as Rong et al. [13] that applies the original `ProtoNet` [7] for intrusion detection and Feng et al. [15] that exploits `MAML` [5] to perform anomaly detection.

Regarding the transfer-learning paradigm, Xiao et al. [18] propose a supervised approach based on deep neural networks. The proposed architecture is first pre-trained, then the knowledge on few-shot tasks is acquired by adding new branches. Similarly, Rezaei and Liu [19] propose a semi-supervised solution for encrypted traffic classification, based on a CNN that is first pre-trained using a large unlabeled dataset. Then, this model is further fine-tuned with few labeled flows, with the aim of classifying the few-shot classes they belong to.

We underline that only few studies [10, 12, 13] provide evaluation including more than one FSL approach. Based on this analysis of the literature, only a limited part of related work deals with (encrypted) traffic classification [11, 12, 18, 19]. *To the best of our knowledge, the only paper that focuses on mobile-app traffic is our previous work [12], which is limited to a comparative assessment of FSL approaches with no robustness analysis provided. Moreover, no work considers the adoption of a multimodal feature extractor for FSL.*

III. META MIMETIC

This section describes the methodology we apply to tackle mobile-app traffic classification in a few-shot context by capitalizing on multimodal DL via meta-learning. Firstly, we define the traffic segmentation adopted and the related information fed to the classification architecture (Sec. III-A). Then, in Sec. III-B we present and detail the peculiarities of META MIMETIC, our novel FSL solution leveraging a multimodal embedding function and an innovative advanced way for meta-training. Finally, in Sec. III-C, we introduce the meta-learning approaches we apply META MIMETIC to (belonging to both model-based and algorithm-based FSL families).

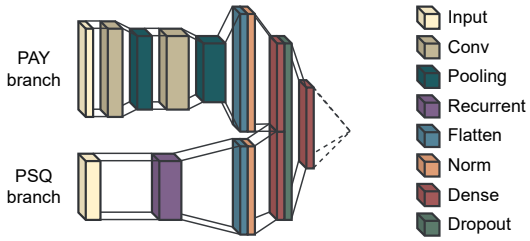


Figure 2: Architectural view of META MIMETIC embedding function. Each branch is fed with the related input type (i.e. `PAY` or `PSQ`).

A. Traffic Object and Network Input

We make use of the *bidirectional flow (biflow)* as the elementary sample of our classification task. More specifically, a biflow is defined as an aggregation of all network packets sharing the same 5-tuple (i.e. source IP, source port, destination IP, destination port, and transport-level protocol) including both directions of communication.

The input data used to feed the FSL approaches exploited herein are organized in two input sets: *packet-field sequences* (`PSQ`) and *payload bytes* (`PAY`). Both `PAY` and `PSQ` input data are normalized within $[0, 1]$ using a Min-Max normalization. We underline that we have chosen the traffic object and input types based on previous works [15, 20, 21] and preliminary validation analyses not shown for brevity.

`PSQ` Input. For each biflow, a set of F informative unbiased fields is extracted from the sequence of the first N_p packets, resulting in an $N_p \times F$ matrix. In particular, the selected $F = 4$ fields are: (*i*) the number of bytes in the transport layer payload; (*ii*) the packet direction (can be -1 or 1); (*iii*) the TCP window size (equal to 0 for UDP packets); (*iv*) the elapsed time since the arrival of the previous packet (i.e. the inter-arrival time).

`PAY` Input. For each biflow, we extract the first N_b bytes (each arranged as an integer ranging from 0 to 255) of the L4-layer payload data. It is worth underlining that the effectiveness of this input is increasingly threatened by ubiquitous traffic encryption. This is especially true for TCP, which has been further secured through the enforcement of TLS 1.3, which encrypts also the small amount of cleartext data present in the TLS handshake of previous TLS versions (e.g., the supported cipher suites and the SNI extension).

B. Capitalizing on Multimodal Embedding Functions

This section describes how we capitalize on *multimodal embedding functions* to improve FSL solutions for mobile-

Algorithm 1 META MIMETIC Meta-Learning Procedure

Require: $\phi_1, \cdots, \phi_M$, $\omega_1^{stub}, \cdots, \omega_M^{stub}$, ϕ, ω

1: **for** modality $m \leftarrow 1$ to M **do** ▷ Pre-training
2: **for each** epoch in *pre-training* **do**
3: metaTraining(ϕ_m, ω_m^{stub})
4: metaValidation(ϕ_m, ω_m^{stub})
5: removeStub(ω_m^{stub})
6: **for** layer $l \leftarrow 1$ to $(L_m - 1)$ **do**
7: freezeLayer(ϕ_m, l)
8: **for each** epoch in *fine-tuning* **do** ▷ Fine-tuning
9: metaTraining($\phi_1, \cdots, \phi_M$, ϕ, ω)
10: metaValidation($\phi_1, \cdots, \phi_M$, ϕ, ω)
11: metaTesting($\phi_1, \cdots, \phi_M$, ϕ, ω) ▷ Evaluation

app encrypted traffic classification. Traffic classifiers based on multimodal DL are known to achieve high performance in non-few contexts and outperform single-modal ones by effectively distilling knowledge from the heterogeneous views of the same traffic object [20–22]. Their success relies on the presence of complex interconnections between single-modal branches that can capture both intra- and inter-modality dependencies. *However, their potentialities may be not fully exploited in FSL when training a multimodal embedding function with the default (naïve, viz. monolithic) meta-learning procedure.*

To fill this gap, we devise a ***novel two-step meta-learning procedure*** *specifically designed for multimodal embedding functions. Detailing, the ad-hoc training involves a pre-training and a fine-tuning phase, both conducted via the episodic meta-learning procedure.* The preliminary pre-training allows the embedding function to separately distill peculiar knowledge from each modality. Then, the fine-tuning optimizes the whole multimodal architecture after the intermediate fusion of single-modality representations. In this way, we can fully capitalize on the multimodal traffic representation exploited. *We name the resulting FSL solution, based on a multimodal embedding function and the related properly-designed ad-hoc meta-learning procedure,* META MIMETIC.

The architecture of the embedding function of META MIMETIC is depicted in Fig. 2 and is made of two branches, each corresponding to a different modality. The first branch is fed with the `PAY` input and is made of two 1D convolutional layers—each followed by a 1D max-pooling layer—and a final dense layer. Differently, the second branch is fed with the `PSQ` input and consists of a Bidirectional Gated Recurrent Unit and one dense layer. The intermediate features extracted by each modality are concatenated and further elaborated via a *shared dense layer* (i.e. the penultimate dark-red layer in Fig. 2) before the model head.[2]

Algorithm 1 details the general procedure devised for META MIMETIC training and testing. We define with ϕ_m the set of trainable parameters of the m^{th} single-modal branch, with $m \in [1, M]$ and M being the number of modalities.[3]

During the *pre-training* phase (lines 1–7), each single-modal branch is updated individually (i.e. without the shared layers). More specifically, the last layer of the m^{th} branch is linked to a stub layer ω_m^{stub} that outputs the soft values. First (line 3), the m^{th} "stubbed" branch is trained in a meta-learning fashion to minimize the classification loss function $\mathcal{L}_m(\cdot)$. Formally,

$$\mathcal{L}_m(\phi_m, \omega_m^{stub}) = \sum_{i=1}^{N \cdot K_q} CE(t_{(i)}, p_{(i)}[\phi_m, \omega_m^{stub}]) \quad (1)$$

being $p_i \triangleq [p_{1,(i)}, \cdots, p_{N,(i)}]$ the vectors of the softmax probabilities for the N classes in the query set, which depend on $(\phi_m, \omega_m^{stub})$ parameters. The goal is to make these probabilities as similar as possible to the one-hot representation of the ground truth $t_{(i)} \triangleq [t_{1,(i)}, \cdots, t_{N,(i)}]$ by minimizing the categorical cross-entropy loss $CE(t, p) \triangleq -\{\sum_{\ell=1} t_\ell \log p_\ell\}$. At the end of the pre-training phase, each ω_m^{stub} is discarded (line 5), and all layers of each single-modal branch are frozen except for the last dense layer (lines 6 and 7, where L_m denotes the number of layers of the m^{th} single-modal branch).

During the *fine-tuning* phase (lines 8–10), the whole META MIMETIC architecture is trained. In detail, the optimization involves the trainable parameters of the non-frozen layers of the single-modal branches $(\phi_1, \cdots, \phi_M)$ and of the shared layer (ϕ) and the model head ω. Similar to Eq. 1, the objective of the meta-training (line 9) is to minimize the classification loss function related to the fine-tuning phase $\mathcal{L}(\cdot)$, which is also based on a categorical cross-entropy:

$$\mathcal{L}(\phi_1, \cdots, \phi_M, \phi, \omega) = \sum_{i=1}^{N \cdot K_q} CE(t_{(i)}, p_{(i)}[\phi_1, \cdots, \phi_M, \phi, \omega]) \quad (2)$$

Finally, META MIMETIC performance is assessed via meta-testing at the end of the fine-tuning phase (line 11).

We recall that meta-training and meta-testing are performed via the episodic meta-learning procedure introduced in Sec. II-A for a given number of epochs. In both pre-training and fine-tuning, in addition to meta-training on D_{nf_1} (lines 3 and 9) and meta-testing on D_f (line 11), we perform meta-validation on D_{nf_2} (lines 4 and 10). This dataset has a disjoint label space from the other two, i.e. $D_{nf} = D_{nf_1} \cup D_{nf_2} \mid D_{nf_1} \cap D_{nf_2} = \emptyset$. Specifically, D_{nf_2} is used (i) to implement an early-stopping procedure, which relies on the accuracy and loss achieved on it, and (ii) to select the model showing the highest performance on it once meta-training is finished.

It is worth highlighting that exploiting a multimodal embedding function enables us to extract features at different granularities. In fact, both the outputs of the single-modal branches (i.e. the per-modality feature vectors) and the shared-representation output can be distilled. Accordingly, we can explore various combinations of these feature vectors, not

[2]The term "model head" is used to refer to the dense layer with softmax/sigmoid activation connected to the last layer of the embedding function.

[3]The specific architecture of META MIMETIC embedding function considered herein consists of two branches/modalities (i.e. $M = 2$), namely the set of trainable parameters of the two single-modal branches are ϕ_1 and ϕ_2.

limiting our study only to the shared-representation one, with the aim of enhancing the latent space representation obtained.

C. FSL Approaches

META MIMETIC is orthogonal to the specific FSL approach used, as long as it exploits an embedding function as a feature extractor. In the following, we introduce the FSL approaches based on meta-learning we deal with in our experimentations.

Regarding *algorithm-based approaches*, we consider `MAML` [5] and `ANIL` [17]. Both learn an initial parameter set via meta-learning and further refine it using knowledge extracted from few-shot classes.

Model-Agnostic Meta-Learning (`MAML`) aims to find a highly adaptable set of parameters by continuously updating the initial meta-learned parameter set based on the performance. *Almost No Inner Loop* (`ANIL`) is a computationally faster version of `MAML`: it applies the parameter updates only to the head of the model.

Conversely, *model-based approaches* differ mainly by the *comparator*, i.e. the similarity measure/mechanism exploited: (*i*) *Matching Networks* (`MatchingNet`) [6] perform a generalized form of nearest-neighbors classification based on Euclidean distance; (*ii*) *Prototypical Networks* (`ProtoNet`) [7] classify a sample via a Euclidean distance function calculated between its embedding and a *prototype*, i.e. a centroid representative of a class; (*iii*) *Relation Network* (`RelationNet`) [8] employs a *relation module* based on a convolutional network that measures the similarity between the embeddings of query and support samples of each class; (*iv*) `MetaOptNet` [16] exploits a *linear Support Vector Machine*, trained on labeled support samples, as the comparator and measures the generalization error on query samples.

IV. EXPERIMENTAL SETUP

This section provides details on the experimental setup we adopted in the following analyses. Hence, we describe the mobile-app traffic dataset we used (Sec. IV-A), the FSL setup we leveraged (Sec. IV-B), the baselines against which META MIMETIC is compared (Sec. IV-C), and the performance metrics we consider (Sec. IV-D).

A. Dataset

We use the public `Mirage-2019` dataset [3] collected at the University of Napoli "Federico II" in 2017–2019 involving more than 300 students and researchers voluntarily participating. `Mirage-2019` includes the traffic of 40 Android apps generated by imitating their common functionalities (e.g., service registration and login, habitual interactions, and use cases).[4] `Mirage-2019` is collected in PCAP format at client-side, leveraging the architecture described in [3]. In addition, for each capture, metadata are saved and exploited to label each biflow with the pertinent Android-package name and generate the ground truth.

[4]https://traffic.comics.unina.it/mirage/mirage-2019.html

`Mirage-2019` contains $92k$ TCP and $5k$ UDP labeled biflows. The number of per-app biflows ranges from 361 to 8246, with 32 out of 40 apps having more than $1k$ biflows. The latter number depends on the peculiarities of each specific app, despite the time that apps are used being roughly the same. Therefore, mobile-app traffic classification carried out on `Mirage-2019` defines a real-world and challenging task.

B. FSL Setup

The first aspect to detail is the *meta-learning setup* in terms of dataset partitioning and episode definition. To define a reasonable few-shot scenario, `Mirage-2019` apps are divided into three disjunct subsets. D_{nf_1} contains the 24 most populous apps, D_{nf_2} the most populous 8 apps besides those in D_{nf_1}, and D_f the 8 remaining least-populous apps.[5] Regarding episode definition, we set $N = 8$ (i.e. the number of apps in D_f) in all meta-learning phases: our goal is solving a classification task where the biflows of 8 apps are available during the operational phase (i.e. at inference time) and by exploiting the latter setup also during meta-training. Conversely, we set K_s and K_q depending on the aim of the analysis. Specifically, for meta-training, we use $K_q = K_s$ (i.e. the same number of biflows for query and support set). For meta-validation and meta-testing, K_s is the same used during meta-training, while we keep $K_q = 100$. This choice aims to improve the coverage and the stability of obtained results.

Additionally, we have tuned the *hyperparameters* based on a preliminary experimental campaign on D_{nf_2}, not reported for brevity. Concerning input data, we set $N_p = 10$ for `PSQ` and $N_b = 512$ for `PAY`. Pre-training and fine-tuning phases of META MIMETIC are conducted for 280 and 440 *epochs*, respectively. In each epoch, the meta-learning procedure is conducted for 100 *episodes*. The optimizer exploits the *Adam* algorithm set with a learning rate of 10^{-4}. Moreover, we use an *early-stopping* mechanism as described in Sec. III-B, with a minimum improvement of 0.01 and patience of 20 epochs.

The implementations of FSL approaches are taken from computer-vision public repositories. In detail, we refer to the well-established GitHub repositories [6, 8] for `MatchingNet` and `RelationNet`, and to the *learn2learn* framework [23] for the other approaches.

C. Comparison Baselines

To define performance baselines for the sake of a comparative evaluation, we resort to different multimodal and single-modal state-of-the-art DL architectures used as alternative embedding functions. These architectures were initially proposed and utilized for traffic classification in both non-few [20, 24, 25] and few-shot scenarios [12].

Firstly, we compare META MIMETIC with a multimodal architecture that has the same structure and inputs as our proposal (depicted in Fig. 2) but is trained in a monolithic fashion, namely not leveraging the novel meta-learning procedure proposed in Sec. III-B. This baseline aims to corroborate

[5]We chose for D_f the apps with less than 1000 biflows. The same number of apps was then selected for D_{nf_2}.

Table I: Complexity (viz. # Trainable Parameters) of embedding functions. 1D-CNN *is* $3\times$ *more complex than* META MIMETIC.

Embedding Function	# Trainable Parameters
META MIMETIC / NAIVE-MM	$821k = 392k^{\dagger} + 356k^{\ddagger} + 64k^{\circ}$
1D-CNN	$3M$
2D-CNN	$530k$

† PAY branch layers; ‡ PSQ branch layers; ∘ shared layers.

the effectiveness of META MIMETIC training procedure in capitalizing on the benefit of a multimodal traffic representation. We refer to such a baseline as NAIVE-MM [12].

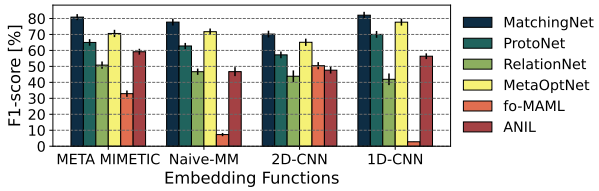
We also consider two single-modal baselines fed with different input types. The first one is a single-modal 1D-CNN originally proposed in [24]. It is made of two 1D convolutional layers, each followed by a 1D max-pooling layer, and is terminated with a dense layer. 1D-CNN is fed with the PAY input. The second is a single-modal 2D-CNN firstly proposed in [25]. It consists of two 2D convolutional layers, each followed by a 2D max-pooling layer and by a batch normalization operation, and ends with a dense layer. According to the 2D nature of the convolutional layer, we feed the 2D-CNN with the PSQ input. *In summary, we consider* 6 META MIMETIC *variants and compare them with* 18 *baselines obtained from the combination between the* 6 *meta-learning approaches and the* 3 *embedding function architectures.*

D. Performance Metrics

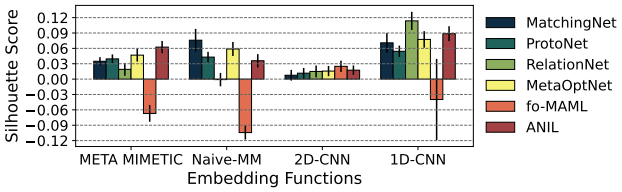
To evaluate the performance of FSL approaches, we use the macro F1-score and the silhouette score. The *macro F1-score* is the harmonic mean of per-class precision and recall arithmetically averaged over apps. The *silhouette score* quantifies how similar a sample is to its own cluster compared to the others. It ranges from -1 (worst) to $+1$ (best). This is paramount since most FSL approaches based on meta-learning behave like a nearest-neighbor classifier in the embedded space. For both metrics, we show the per-episode mean and standard deviation attained on D_f. Specifically, the results are shown as $avg. \pm std.$ over 100 episodes. Finally, to investigate the computational complexity, we also report the number of trainable parameters and the overall training time (all the analyses are executed on a machine with 12 Intel(R) Xeon(R) CPU E5-2430 v2 @ 2.50GHz and 62GB of memory).

V. EXPERIMENTAL EVALUATION

In this section, we show the results of the experimental evaluation conducted. We first report the overall classification performance of all meta-learning approaches when using both single-modal and multimodal embedding functions (Sec. V-A); then we inspect the internal mechanisms of our META MIMETIC proposal with the aim of improving it (Sec. V-B); finally, we perform a robustness analysis of meta-learning approaches against the novel encryption process introduced with TLS 1.3 (Sec. V-C).



(a) F1-score.



(b) Silhouette Score.

Figure 3: Comparison of meta-learning approaches with different multimodal and single-modal embedding functions. META MIMETIC *and* 1D-CNN *obtain comparable performance figures with* MatchingNet.

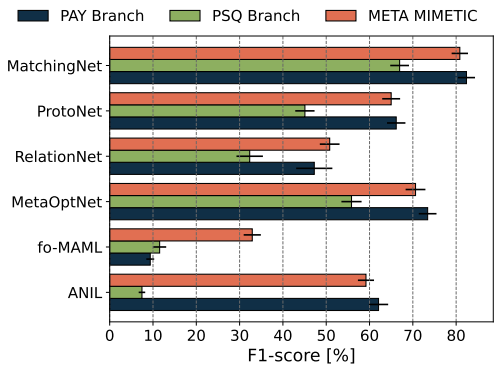
A. Overall Performance

This experimental campaign shows the performance of the various meta-learning approaches considered when leveraging different embedding functions. Particularly, we compare META MIMETIC with the baselines described in Sec. IV-C.

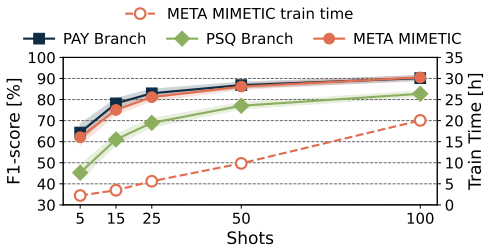
The results in Fig. 3a show that MatchingNet is always the top-performing meta-learning approach regardless of the specific embedding function, followed by MetaOptNet and ProtoNet. Conversely, fo-MAML, ANIL, and RelationNet obtain unsatisfactory F1-scores. Notably, leveraging META MIMETIC leads to an F1-score improvement for the latter worse-performing approaches. Overall, META MIMETIC attains the best performance for 4 out of 6 meta-learning approaches, proving its versatility to different FSL families and paradigms. Also, the comparison of the two multimodal embedding functions shows that all meta-learning approaches perform better with META MIMETIC as opposed to NAIVE-MM, demonstrating the effectiveness of the ad-hoc meta-learning procedure we propose herein.

Focusing on the top-performing MatchingNet, the *highest improvement* is attained with META MIMETIC and 1D-CNN, which both achieve an *F1-score higher than* 80%. Interestingly, MatchingNet *is also the most robust approach w.r.t. the goodness of the embedding space*, which is reported by depicting the silhouette score in Fig. 3b. Indeed, while the other approaches have an F1-score strongly dependent on the related silhouette score, MatchingNet shows high F1-scores also for poor silhouette scores, likely due to its comparator implementing a mechanism simpler than that of other meta-learning approaches (cf. Sec. III-C).

Finally, Tab. I focuses on the computational complexity of considered embedding functions. We can notice that, despite reaching similar performance, 1D-CNN *has a number of trainable parameters more than* $3\times$ *larger than* META MIMETIC, thus possibly introducing deployability issues when



(a) F1-score achieved by varying the meta-learning approach we apply META MIMETIC to.



(b) F1-score and training time using `MatchingNet` for different number of shots. The training time required for META MIMETIC includes pre-training of single-modal branches and later fine-tuning.

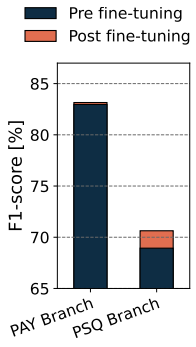
Figure 4: Insights about the multimodal internal architecture by comparing F1-score obtained by `PAY` and `PSQ` branches before fine-tuning and by the overall META MIMETIC architecture after fine-tuning: (a) thoughtful comparison of meta-learning approaches; (b) sensitivity to the number of shots using `MatchingNet`.

dealing with devices with constrained computing resources. In summary, *our proposal* META MIMETIC *proves to be the best trade-off between classification performance and complexity.*

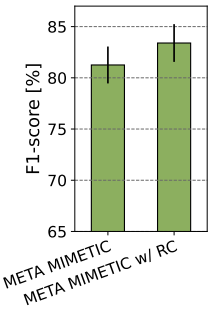
B. Inspection and Enhancement of META MIMETIC *Internal Structure*

Hereinafter, we go into details of the internal mechanisms of META MIMETIC with the goal of deepening and further capitalizing on its multimodal structure. To this aim, we compare the performance of pre-trained single-modal branches (i.e. `PAY` and `PSQ` branches) with that of the fine-tuned multi-modal architecture. As depicted in Fig. 4a, the latter performs approximately on par w.r.t. the `PAY` branch and significantly outperforms the `PSQ` one for all meta-learning approaches. Notably, `fo-MAML` has a $> 20\%$ F1-score improvement with META MIMETIC w.r.t. either single-modal branch.

We deepen the analysis, focusing on the best-performing `MatchingNet` and evaluating the performance of META MIMETIC and its single-modal branches for different values of $K_s \in \{5, 15, 25, 50, 100\}$. Figure 4b highlights that the performance of `MatchingNet` significantly improves for higher values of K_s: it gains up to $+30\%$ F1-score with $K_s = 100$ ($> 90\%$ F1-score with META MIMETIC) compared to the initial value of $K_s = 5$. Given that the number of shots



(a) Comparison of the performance of the single-modal branches of META MIMETIC before and after fine-tuning.



(b) Classification performance of META MIMETIC with and without residual connections.

Figure 5: Deepening of META MIMETIC using `MatchingNet`, by showing (a) the impact of the fine-tuning phase and by evaluating (b) the META MIMETIC variant w/ Residual Connections. Bar charts are zoomed in to better highlight differences in performance.

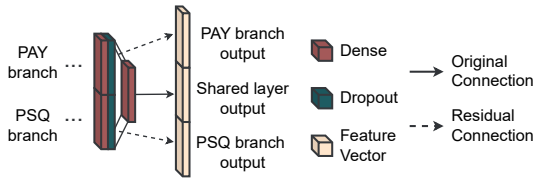


Figure 6: View of the residual connections of META MIMETIC w/ RC. Residual Connections are used to enrich the output of the shared layer by concatenating it with the output of each single-modal branch. Dropouts are used only during the meta-training.

significantly impacts the training time, in Fig. 4b, we report also the total time (in hours) needed to train `MatchingNet` for each K_s. When passing from $K_s = 5$ to $K_s = 100$, we observe that the times grow in a linear way. We identify $K_s = 25$ as the best trade-off between performance and training complexity. At this value, a clear decrease of the steepness of the F1-score curves can be observed, and training times are low and close to the ones registered at 5 and 15 shots, namely circa 5 hours. Therefore, we set such a value as the number of shots in the following experiments.

Once we have assessed the performance of META MIMETIC, we further inspect the benefits deriving from its characteristics and the room for improvements this solution offers. Hence, we investigate both the performance benefits of fine-tuning and the performance advantages achievable by taking advantage of its structure. Such investigations focus on the best-performing `MatchingNet`.

First, experiments show the beneficial impact of the fine-tuning procedure. In detail, Fig. 5a shows the F1-score achieved by single-modal branches before and after fine-tuning. The experimental results highlight that at the end of the fine-tuning phase devised for META MIMETIC, the F1-score of each branch is improved: significantly for the `PSQ` branch ($\approx +3\%$) and marginally for the `PAY` one ($< 1\%$).

Starting from these results, we further capitalize on the

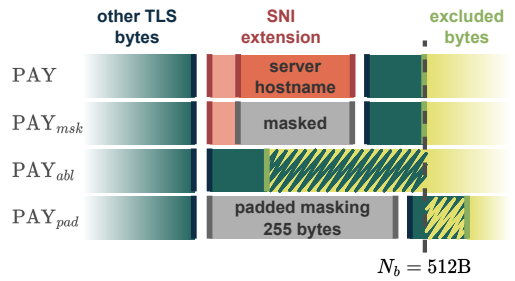


Figure 7: Sketch of *information-removal schemes*. SNI-extension bytes are colored in **red**, the masked ones are colored in **gray**, the other TLS bytes considered in the PAY$_*$ input are colored in **green**, and those excluded from the PAY$_*$ input are in **yellow**. Extra bytes considered in the PAY$_{abl}$ input (w.r.t. PAY) are shown with a **green zigzagged line** and bytes ablated in the PAY$_{pad}$ input (w.r.t. PAY) are shown with a **yellow zigzagged line**.

connectionist philosophy of META MIMETIC by variously merging the features extracted from single-modal and shared layers. Particularly, we investigate the performance of a variant of META MIMETIC (named META MIMETIC W/ RC) that enriches the output of the shared layer by concatenating it with the output of each branch via *Residual Connections (RC)*. The changes that characterize this variant are depicted in Fig. 6, where the outputs of the two single-modal branches are propagated by skipping the shared layers and then concatenated to the shared representation to construct the feature vector. Formally, $\nu \triangleq [v_{1,1}, \cdots, v_{a,1}, v_{1,2}, \cdots, v_{b,2}, v_{1,3}, \cdots, v_{c,3}]$, where a, b, and c are the dimensions of the feature vectors being the outputs of the branches above. Accordingly, Fig. 5b presents the F1-score of META MIMETIC with and without the latter feature concatenation. By leveraging this improved feature concatenation, META MIMETIC W/ RC can outperform its "base" version (up to 2.24% F1-score).

We remark that the versatility of META MIMETIC is a significant point in its favor compared to other FSL solutions exploiting monolithic embedding functions, such as its single-modal branches and state-of-the-art baselines (e.g., 1D-CNN and 2D-CNN), as witnessed by the gain obtained with this "simple" modification. Indeed, such versatility has allowed us to realize *a novel FSL solution that can outperform all its competitors in terms of both classification performance and computational complexity (residual connections do not imply any additional training parameter nor improved training time) by capitalizing on its multimodal embedding function.*

C. Robustness to TLS 1.3 Encrypted Client Hello

When the bytes of the PAY input are carried over a TCP connection with TLS encryption, they can still transport the hostname of the server as cleartext within the Server Name Indication (SNI) extension of the Client Hello (CH) message. The latter is proven to carry rich information that can be leveraged to guess the generating application [26]. However, the recent adoption of *TLS 1.3* may hinder the contribution of SNI to traffic classification due to the *Encrypted Client Hello (ECH) extension*. In fact, this optional mechanism provided by TLS 1.3 results in appending the encryption of the entire CH to the CH itself: in this way, *the actual SNI is transmitted encrypted*, and the outer cleartext SNI (used only for backward compatibility) may no longer be considered relevant.

Accordingly, as shown in Fig. 7, we adopt three further variants of the PAY input for emulating the removal of the SNI bytes information at a growing extent, resulting in the following *information removal schemes*:

- PAY$_{msk}$, where the sole server hostname is *masked* being replaced by random values;[6] this scheme leaves unaltered the SNI-length information that is explicitly set in the SNI extension header.
- PAY$_{abl}$, where the entire SNI is *ablated* from the CH; this scheme introduces extra input bytes (the portion with a green zigzagged line in Fig. 7), thus increasing the information that can be exploited (w.r.t. PAY input), which can partially compensate for the SNI ablation.
- PAY$_{pad}$, where the SNI is first *padded* to 255 bytes as suggested by the related IETF draft[7] and then *masked* with random values[6]; differently from the previous scheme, this reduces the knowledge passed to the model, thus it is expected to cause a worsening of performance. It is worth noting that the discarded bytes—the portion with a yellow zigzagged line in Fig. 7—likely encode the server certificate (along with the name of the certification authority), which may be very informative in diversifying applications.

We apply these information-removal schemes to all the TLS biflows (79% of the overall dataset and 83% of the TCP ones). Table II reports the results obtained when exploiting such PAY-input variants.

First, *all the information-removal schemes result in a drop in performance as expected, with single-modal* 1D-CNN *being the most affected ones, as it exploits the* PAY*-input only*. It is worth underlining that the results presented leveraging the 1D-CNN can be also considered as a proxy to assess the robustness of the PAY branch of META MIMETIC, since the 1D-CNN and the PAY branch have the same input. Going into detail, the masking of the sole SNI (PAY$_{msk}$) degrades only the 1D-CNN performance by 0.92% F1-score. On the other hand, both multimodal solutions based on META MIMETIC are not impacted by this scheme. Such a limited performance drop is likely related to the presence of the SNI-length field, which carries valuable information for the classification task.

Indeed, the above interpretation is confirmed by the largest performance drop obtained when the SNI is completely ablated (PAY$_{abl}$): in this case, both META MIMETIC and 1D-CNN experience a performance drop, but the latter has again the worst degradation (i.e. 1.98% F1-score drop). Surprisingly, META MIMETIC W/ RC slightly improves its performance (+0.78% F1-score): this finding confirms the benefit of having a more complex embedding space that is capable of extracting more discriminative knowledge from the input data.

[6] For each biflow, the random masking is achieved by sampling—with replacement—the required number of bytes from those preceding the SNI extension.

[7] https://datatracker.ietf.org/doc/draft-ietf-tls-esni/

Table II: F1-scores [%] of META MIMETIC (w/ and w/o RC) and 1D-CNN when diverse SNI removal schemes are enforced. The *discrepancy from no information removal scheme (PAY input)* is reported in round brackets. The best F1-scores are reported in **bold**. The standard deviation is always $< 2\%$.

	Information Removal Scheme			
*Embedding Function**	PAY	PAY_{msk}	PAY_{abl}	PAY_{pad}
META MIMETIC W/ RC	**83.38**	**83.58** (**+0.20**)	**84.14** (**+0.76**)	**81.21** (**−2.17**)
META MIMETIC	81.14	81.20 (+0.06)	80.32 (−0.82)	78.02 (−3.12)
1D-CNN	82.06	81.14 (−0.92)	80.08 (−1.98)	72.97 (−9.09)

*2D-CNN is not reported since its PSQ input is not affected by the TLS 1.3 ECH. Nevertheless, 2D-CNN reaches 81.14% F1-Score at most.

Finally, with the most aggressive information removal scheme enforcing both padding and masking (PAY_{pad}), the performance of 1D-CNN decreases by 9.09% F1-score, against the 2.17% drop revealed by META MIMETIC W/ RC: *overall,* META MIMETIC W/ RC *has an F1-score drop* $\approx 4\times$ *lower than* 1D-CNN *and outperforms it by* 8.24% *F1-score.*

VI. CONCLUSIONS AND FUTURE PERSPECTIVES

In this paper, we presented META MIMETIC, a novel multimodal-based FSL solution for mobile-app encrypted traffic classification using meta-learning. META MIMETIC (i) distills informative features from multiple traffic data views, (ii) combines them through intermediate fusion, and (iii) introduces advanced strategies for training single-modal branches and shared layers of the multimodal embedding function, designed for episodic meta-learning.

The experimental evaluation exploited `Mirage-2019`, a publicly-available mobile-app traffic dataset. We considered six meta-learning-based FSL approaches and compared META MIMETIC with NAIVE-MM, 1D-CNN, and 2D-CNN embedding functions. META MIMETIC outperformed all, achieving 81.14% F1-score and 73% fewer trainable parameters (-2.2M) compared to the best competitor. Moreover, META MIMETIC internal mechanisms were investigated, achieving a remarkable 91% F1-score with 100 shots. On this basis, we devised a novel multimodal feature combination technique using concatenation and residual connections, which resulted in $+2.24\%$ F1-score without increasing the computational complexity. Finally, we evaluated META MIMETIC robustness to future TLS evolution, particularly the SNI encryption introduced in TLS 1.3. META MIMETIC with residual connections had an F1-score drop $4\times$ smaller than its best-performing competitor and outperformed it by 8.24% F1-score.

Future work will explore multimodal modeling for transfer learning in FSL, optimize single-modal embedding functions, and apply META MIMETIC to other real-world problems like intrusion detection and malware classification.

ACKNOWLEDGMENTS

This work has been funded in the framework of the Huawei Innovation Lab project on "Network Traffic and AI enabled Network Technologies" by Huawei Technologies France SASU at DIETI, University of Napoli Federico II.

REFERENCES

[1] G. Aceto, et al. Characterization and prediction of mobile-app traffic using markov modeling. *IEEE Transactions on Network and Service Management*, 18(1):907–925, 2021.

[2] E. Papadogiannaki et al. A survey on encrypted network traffic analysis applications, techniques, and countermeasures. *ACM Computing Surveys*, 54(6):1–35, 2021.

[3] G. Aceto, et al. MIRAGE: Mobile-app Traffic Capture and Ground-truth Creation. In *4th IEEE International Conference on Computing, Communications and Security*, pages 1–8, 2019.

[4] C. Wang, et al. AppClassNet: A commercial-grade dataset for application identification research. *ACM SIGCOMM Computer Communication Review*, 52(3):19–27, 2022.

[5] C. Finn, et al. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 1126–1135, 2017.

[6] O. Vinyals, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

[7] J. Snell, et al. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

[8] F. Sung, et al. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208, 2018.

[9] W. Zheng, et al. Learning to classify: A flow-based relation network for encrypted traffic classification. In *Proceedings of The Web Conference 2020*, pages 13–22, 2020.

[10] Z.-M. Wang, et al. A Few-Shot Learning-Based Siamese Capsule Network for Intrusion Detection with Imbalanced Training Data. *Computational Intelligence and Neuroscience*, 2021:7126913, 2021.

[11] Z. Zhao, et al. A Few-Shot Learning Based Approach to IoT Traffic Classification. *IEEE Communications Letters*, 26(3):537–541, 2022.

[12] G. Bovenzi, et al. Few Shot Learning Approaches for Classifying Rare Mobile-App Encrypted Traffic Samples. In *IEEE INFOCOM 2023-IEEE Conference on Computer Communications Workshops*, 05 2023.

[13] C. Rong, et al. UMVD-FSL: Unseen Malware Variants Detection Using Few-Shot Learning. In *2021 International Joint Conference on Neural Networks*, pages 1–8, 2021.

[14] S. Huang, et al. A Gated Few-shot Learning Model For Anomaly Detection. In *2020 International Conference on Information Networking*, pages 505–509, 2020.

[15] T. Feng, et al. Few-Shot Class-Adaptive Anomaly Detection with Model-Agnostic Meta-Learning. In *2021 IFIP Networking Conference*, pages 1–9, 2021.

[16] K. Lee, et al. Meta-learning with differentiable convex optimization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10657–10665, 2019.

[17] A. Raghu, et al. Rapid learning or feature reuse? towards understanding the effectiveness of maml. *arXiv preprint arXiv:1909.09157*, 2019.

[18] Y. Xiao, et al. Common Knowledge Based Transfer Learning for Traffic Classification. In *2018 IEEE 43rd Conference on Local Computer Networks*, pages 311–314, 2018.

[19] S. Rezaei et al. How to achieve high classification accuracy with just a few labels: A semi-supervised approach using sampled packets. *arXiv preprint arXiv:1812.09761*, 2018.

[20] G. Aceto, et al. MIMETIC: Mobile encrypted traffic classification using multimodal deep learning. *Elsevier Computer Networks*, 165:106944, 2019.

[21] P. Lin, et al. A Novel Multimodal Deep Learning Framework for Encrypted Traffic Classification. *IEEE/ACM Transactions on Networking*, 2022.

[22] X. Wang, et al. App-net: A hybrid neural network for encrypted mobile traffic classification. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops*, pages 424–429, 2020.

[23] S. M. Arnold, et al. learn2learn: A library for meta-learning research. *arXiv preprint arXiv:2008.12284*, 2020.

[24] W. Wang, et al. End-to-end encrypted Traffic Classification with one-dimensional convolution neural networks. In *IEEE International Conference on Intelligence and Security Informatics*, pages 43–48, 2017.

[25] M. Lopez-Martin, et al. Network traffic classifier with convolutional and recurrent neural networks for Internet of Things. *IEEE Access*, 5: 18042–18050, 2017.

[26] A. Nascita, et al. Unveiling MIMETIC: Interpreting Deep Learning Traffic Classifiers via XAI Techniques. In *2021 IEEE International Conference on Cyber Security and Resilience*, pages 455–460, 2021.