XAI meets Mobile Traffic Classification: Understanding and Improving Multimodal Deep Learning Architectures

Alfredo Nascita, Antonio Montieri, *Graduate Student Member, IEEE*, Giuseppe Aceto, Domenico Ciuonzo, *Senior Member, IEEE*, Valerio Persico and Antonio Pescapé, *Senior Member, IEEE*

***Abstract*—The increasing diffusion of mobile devices has dramatically changed the network traffic landscape, with Traffic Classification (TC) surging into a fundamental role while facing new and unprecedented challenges. The recent and appealing adoption of Deep Learning (DL) techniques has risen as the solution overcoming the performance of ML techniques based on tedious and time-consuming handcrafted feature design. Still, the black-box nature of DL models prevents its practical and trustful adoption in critical scenarios where the reliability/interpretation of results/policies is of key importance. To cope with these limitations, eXplainable Artificial Intelligence (XAI) techniques have recently acquired the interest of the community. Accordingly, in this work we investigate trustworthiness and interpretability via XAI-based techniques to understand, interpret and improve the behavior of state-of-the-art multimodal DL traffic classifiers. The proposed methodology, as opposed to common results seen in XAI, attempts to provide global interpretation, rather than sample-based ones. Results, based on an open dataset, allow to complement the above findings with domain knowledge.**

***Index Terms*—traffic classification; encrypted traffic; explainable artificial intelligence; deep learning; multimodal learning.**

I. Introduction

INTERPRETING and assessing the behavior of networks is paramount to a wide set of activities and research topics, including—to name a few—network planning and management, network and application security and privacy, social adoption and impact of network technologies, and future progress roadblocks and possibilities. Traffic Classification (TC) is central to these activities, and has witnessed significant recent research activities due to both new challenges—as the rise of mobile traffic—and new powerful tools, e.g. those fueled by Artificial Intelligence (AI) techniques. Indeed, the widespread usage of smartphones has deeply changed the Internet traffic landscape, due to the variety of mobile apps and the ease of installation and automatic update allowed by mobile apps distribution systems ("app stores"). The sum of an expanded and more diverse use base, an increased time and variety of on-line activities, and fast-pace creation, diffusion, and update of apps has resulted in an extremely challenging scenario for TC. A much needed response to such challenges has been recently provided by Deep Learning (DL) techniques, characterized by a fully-automated feature extraction phase that significantly lowers the involvement of human experts in the loop, and a great power in learning (unforeseen) relations among huge volumes of training data, offering significantly better performance with respect to traditional Machine Learning (ML) approaches. These characteristics are making DL the tool of choice to face the highly dynamic nature of modern network traffic. On the other hand, the *black-box* nature of DL techniques impairs the understanding of the reason behind specific classification outcomes, including errors and/or over-optimistic guarantees, and even resilience against intentional adversarial manipulation of traffic to prevent correct identification.

The lack of *interpretability* of the classification models built by DL techniques prevents their applicability to critical scenarios. Indeed, regulatory and standardization bodies in the specific field of telecommunications have considered this issue, highlighting explainability of AI as the basis for the necessary accountability, responsibility, and transparency of processes including AI components [1] such as the "Network Data Analytics Function" [2]. Similarly, for cybersecurity applications the lack of explainability is troubling to the point it can be leveraged to hide attacks [3], besides posing legal and ethical issues. As a consequence, network equipment manufacturers are starting to include *eXplainable Artificial Intelligence* (XAI) in their design [4]. In addition to this, in research contexts where TC is an instrument of analysis and knowledge discovery, the aforementioned lack severely undermines the validity of inferred information and the following analyses and deductions. From a complementary viewpoint, even when interpretability cannot be guaranteed (or it is not required), ensuring the *trustworthiness* of the classification is important for its "safe" usage to end-users. By trustworthiness, in this work we mean to which extent the confidence associated to a given decision by an opaque solution can be deemed reliable—i.e., low (resp. high) confidence in labeling a certain app actually leads to low (resp. high) accuracy in classifying it.[1] Indeed, the latter property is crucial as it informs decisions with impact on user experience and economic efficiency of network management. Finally, the understanding of the behavior of the learned model enables focused performance enhancements,

Manuscript received 12th April 2021; revised 22nd June 2021; accepted 13th July.

The authors are with the Department of Electrical Engineering and Information Technologies (DIETI) at University of Naples Federico II, Italy.

E-mail: a.nascita@studenti.unina.it; {giuseppe.aceto, domenico.ciuonzo, antonio.montieri, valerio.persico, pescape}@unina.it.

[1]We would like to emphasize that the notion of (AI) trustworthiness starkly contrasts with the same term when used in network security contexts.

much more efficient than a less-informed search over the huge hyper-parameters space.

In response to these needs arising from DL adoption, the field of XAI is providing approaches and techniques able to link the outcome of the classification to the structure of the model and the input, shedding light over some aspects of the (former) completely black box. As the adoption of DL is relatively new, even more so in the field of network traffic analysis, it is no wonder that XAI has not yet found significant application to TC as well, despite its acute need: with this work we move an important step in the direction of tackling this open challenge.

To this aim, we perform (i) fine-grained performance evaluation, (ii) behavior interpretation, and (iii) trustworthiness analysis of an enhanced (multimodal) DL architecture for TC. The above analysis includes *different level of granularity or viewpoints*: (a) relative importance of different input modalities, and down to specific parts of each modality, (b) per-class, and (c) per-protocol analysis. Equally important, in the analysis we highlight and quantify the use of encryption in the real-world and recent dataset we base the evaluation on, and interpret the results in the light of this property.

These contributions are the more significant and timely considering the rise of adoption of TLS, gQUIC, and FB-Zero in modern mobile-generated Internet traffic [5, 6].

In detail, the contributions of this paper can be summarized as follows:

- In order to apply the explainability analysis to a practically significant experimental scenario, we design MIMETIC-ENHANCED, a novel architecture for TC operating at *biflow level*. Based on our recently proposed *MultImodal DL-based MobilE TraffIc Classification* (MIMETIC) general framework, the architecture is designed to effectively exploit the heterogeneous nature of the different views of a traffic object, by distilling both intra- and inter-modalities dependence [7]. Our proposal leverages the multimodal paradigm and consists of two branches being fed (i) with the first bytes of the payload of the biflows and (ii) with informative fields extracted from the sequence of the first packets of each biflow. More specifically, we have improved our previous proposal based on the MIMETIC framework from both the architectural and training-procedure viewpoints.
- We systematically evaluate the proposed MIMETIC-ENHANCED TC architecture and compare the achieved performance against state-of-art TC baselines, including MIMETIC [7], App-Net [8], and FS-Net [9].
- We deepen our performance analysis at a finer-grain, inspecting soft classification values to understand (i) behind-the-curtain behavior of TC classifiers when inspecting the traffic of each class and (ii) misclassification patterns possibly imputed to specific protocol usage. Indeed, thanks to a heuristic we propose, we assess how challenging is to classify the traffic of each app, by relating its share of encrypted packets and associated encrypting protocol.
- We evaluate the trustworthiness of the proposed TC architecture through a calibration analysis, in order to assess the reliability of the provided predictions based on the related confidence. This analysis supports further improvements of the proposed MIMETIC-ENHANCED by leveraging *Focal Loss* and *Label Smoothing* techniques to improve the generalization capability of the model (e.g. reducing the excessive confidence associated with predictions and reducing overfitting).
- We investigate the rationale of the working behavior of MIMETIC-ENHANCED, by applying state-of-art XAI tools (i.e. Deep SHAP, capitalizing the integration of Shapley-based importance attribution with deep architectures [10]) to understand input importance in both branches, and within the single branch.
- The experimental campaign is conducted leveraging MIRAGE-2019, a publicly-released human-generated mobile-app traffic dataset, to foster reproducibility [11].

The paper is organized as follows. Section II surveys first attempts of XAI application to networking and TC, positioning our work against related literature. Section III describes the considered XAI-based TC methodology; the dataset employed and the experimental results are discussed in Sec. V; finally, Sec. VI provides conclusions and future perspectives.

II. BACKGROUND AND RELATED WORK

In this section, we first provide the background on (i) interpretability and (ii) trustworthiness of data-driven models (Sec. II-A). Then, we discuss the most related work addressing those problems in computer networks, providing a final focus on XAI for TC (Sec. II-B). Finally, we end the section with a clear positioning of our work w.r.t. the computer network literature described in the second subsection (Sec. II-C).

A. Background: Making Data-Driven AI Models Human-Interpretable and Trustworthy

In line with the unprecedented and growing availability of huge amounts of data in a number of critical knowledge fields (e.g., mobile applications, speech recognition, biological science, Internet of Things) and fostered by the tremendous success in image recognition, enormous incentives exist to use AI as a valuable tool for decision making, cost reduction, risk management etc. However, the adoption of complex AI algorithms inherently leads to the generation of black-box models which result in *untrustworthy behaviors* and undesired (and often even unacceptable) *lack of transparency*. This issue affects complex AI models characterized by a high-dimensionality of the inputs and therefore especially includes cutting-edge DL models.

As a result, in the last years a strong push towards XAI by both governmental entities and the research community was observed, in order to address problems related to transparency, causality, bias, fairness, and safety of the obtained solutions [22]. Investigating the working behavior of already-trained models allows to (i) assess the robustness of the AI system (e.g. to measure artifacts or adversarial perturbations); (ii) evaluate its resilience to drifting data perturbations; (iii) verify legal, safety, and security requirements; (iv) complement human-expertise in decision making and even provide scientists with novel insights embodied in the model.

Table I
RELATED WORKS ADOPTING XAI IN CONJUNCTION WITH VARIOUS NETWORKING PROBLEMS SOLVED VIA DIFFERENT LEARNING TECHNIQUES. THE PAPERS ARE REPORTED IN CHRONOLOGICAL ORDER. ACRONYMS MEANING IS REPORTED AT THE BOTTOM OF THE TABLE.

Paper	Networking Problem	Open	Mobile	Interpret	Trustwrt	XAI Strategy	Paradigm	Technique
K. Amarasinghe *et al.*, 2018, *Proc. IEEE HSI* [12]	Anomaly detection	●	○	●	○	LRP	SL	DNN
Y. Zheng *et al.*, 2018, *Proc. ACM APNet* [13]	Task allocation	—	○	●	○	Saliency Maps, Activation Maximization	UL	DNN
A. Dethise *et al.*, 2019, *Proc. ACM NetAI* [14]	Video bitrate adaptation	●	○	●	○	LIME	RL	A3C (Pensieve)
A. Morichetta *et al.*, 2019, *Proc. ACM Big-DAMA* [15]	Video quality prediction	◐	◐	●	○	LIME	UL	HAC-Ward, HAC-Single, K-means, BIRCH
S. Rezaei *et al.*, 2019, *IEEE Access* [16]	Traffic classification	○	●	●	○	Occlusion Analysis	SL	1D-CNN
G. Aceto *et al.*, 2020, *Elsevier NEUCOM* [17]	Traffic classification	◐	●	○	◐	Calibration Analysis	SL	1D-CNN, RF, MM CNN & BiGRU
C. Beliard *et al.*, 2020, *Proc. IEEE INFOCOM WKSHPS* [18]	Traffic classification	○	○	●	○	Feature Map Visualization, t-SNE	SL	CNN
Z. Meng *et al.*, 2020, *Proc. ACM SIGCOMM* [19]	Global and local network control	●	○	●	○	DT-Distillation, Hypergraph-Distillation, LIME⋆, LEMNA⋆	RL	A3C (Pensieve), DNN (AuTO), GNN (RouteNet)
X. Wang *et al.*, 2020, *Hindawi WCMC* [20]	Traffic classification	●	●	◐	○	Deep SHAP	SL	SDAE, 1D-CNN, LSTM
G. Aceto *et al.*, 2021, *IEEE TNSM* [21]	Traffic prediction	●	●	◐	○	Markovian-Distillation	SL	MC, RFR
This Paper	Traffic classification	●	●	●	●	Calibration Analysis, Calibration Improvement (w/ FL & LS), Deep SHAP	SL	MIMETIC-ENHANCED, SM 1D-CNN⋆, HYBRID⋆, SM MLP-1⋆, FS-Net⋆, MM MIMETIC⋆, App-Net⋆

Open: publicly available data. **Mobile**: mobile-app traffic data. **Interpret**: Interpretability analysis. **Trustwrt**: Trustworthiness analysis.
XAI Strategy: Decision Tree (DT), Focal Loss (FL), Local Explanation Method using Nonlinear Approximation (LEMNA), Label Smoothing (LS), Local Interpretable Model-Agnostic Explanations (LIME), Layer-wise Relevance Propagation (LRP), SHapley Additive exPlanations (SHAP), t-Distributed Stochastic Neighbor Embedding (t-SNE).
Paradigm: Reinforcement Learning (RL), Supervised Learning (SL), Unsupervised Learning (UL).
Technique: Asynchronous Advantage Actor-Critic (A3C), Bidirectional Gated Recurrent Unit (BiGRU), Convolutional Neural Network (CNN), Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH), Deep Neural Network (DNN), Graph Neural Network (GNN), Hierarchical Agglomerative Clustering (HAC), Long Short-Term Memory (LSTM), Markov Chain (MC), MultiLayer Perceptron (MLP), Multimodal (MM), Random Forest Regressor (RFR), Reinforcement Learning (RL), Stacked Denoising AutoEncoder (SDAE), Single-Modal (SM). ● present, ○ lacking, ◐ partial, — not applicable.
⋆ Baselines for performance comparison.

Focusing on **trustworthiness**, a natural desideratum for AI algorithms is to design a black-box model whose outputs adhere to some reliability requirements. In the context of classification, given the probabilistic nature of ML/DL outputs, it is rightful to ask whether the probabilistic outcomes are calibrated, i.e. the confidence actually reflects the reliability of the provided decision. Referring to calibration, current research is oriented toward methods for (a) assessing calibration of designed models and (b) for improving it by making the corresponding ML/DL model close-to-trustworthy. Regarding the former aspect, a set of evaluation metrics for assessing the classification confidence have been proposed, including different notions of calibration [23] (e.g. multi-class, class-wise and confidence), calibration diagrams and related metrics [24], and statistical tests [25] to verify such property. Regarding the latter aspect, recent studies have attempted to design evolved learning procedures to instill calibration, such as Dirichlet-based methods [23], focal loss [26] and label smoothing [27].

Another critical aspect in XAI is represented by the purpose that these methods were created to serve and the ways through which they accomplish this purpose, namely their **interpretability**. Accordingly, four major categories for this class of methods have been identified: (i) methods for explaining complex black-box models, (ii) methods for creating white-box models, (iii) methods that promote fairness and restrict the existence of discrimination, and, lastly, (iv) methods that analyze the sensitivity of model predictions. An especially important separation of interpretability methods is based on the type of algorithm that could be applied: methods whose application is restricted to some specific models (*model-specific*) or approaches which can be virtually applied to every possible ML/DL algorithm (*model-agnostic*). A complementary taxonomy of such methods is based on the *interpretation scale*: *local* methods provide an explanation only for a specific sample, whereas *global* methods attempt to explain the whole model behavior.

Accordingly, a number of *families of approaches* have emerged aiming at shedding light on ML/DL outcomes via *post-hoc explanations*. These can be classified[2] as follows:

- **Occlusion Analysis** [29] is a particular type of perturbation analysis where the effect on the neural network output of occluding patches or individual features in the input is tested. Attribution based on Shapley values (Kernel SHAP, Deep SHAP) can also be seen as an occlusion analysis [10].
- **Interpretable Local Surrogates** aim to replace the decision function with a local surrogate model whose functional form is self-explanatory itself (e.g. a linear or a tree model as used by LIME [30] and LORE [31], respectively).
- **Integrated or Smoothed Gradients** [32] provide an explanation by (a) integrating the $\bigtriangledown f(\boldsymbol{x})$ along some trajectory in input space connecting some root point $\bar{\boldsymbol{x}}$ (a non-informative baseline input) to the data point $\boldsymbol{x}$ to be explained or (b) performing an expectation of $\bigtriangledown f(\boldsymbol{x})$ when some noise-distributional perturbation is applied.

[2]For a detailed discussion regarding different post-hoc explanation techniques we point the interested readers to Samek et al. [28].

- **Layerwise Relevance Propagation (LRP)** [33] explicitly uses the layered structure of the neural network and operates in an iterative manner to produce the explanation.

Next subsection will position recent works on XAI applied to computer network problems in terms of trustworthiness and interpretability.

B. Related Work: Explaining AI Decisions Supporting Computer-Network Problems

A large number of data-driven solutions based on either ML or DL have been recently proposed for several network-related problems such as resource allocation, routing, video rate selection, congestion control [14], traffic classification [18, 20], and traffic prediction [21].

However, general lack of interpretability as well as potential behavioral uncertainties (e.g., in conditions different than those in which the system is trained) and difficulties in integrating domain-specific (hand-crafted) insights into Deep Neural Network models (DNNs) represent the main reasons hindering these solutions and systems to be widely adopted in production environments so far [13, 34].

The applicability of more easy-to-interpret models has been investigated, such as techniques being more interpretable in nature than DL (e.g., those based on either decision trees [35, 36] or Markov models [21, 37]). These techniques represent an alternative walkable path due to their simplicity and interpretability, despite the expected reduced performance and the resulting need for engineering hand-craft features. On the other hand, a first wave of works has provided an initial effort towards the interpretation of data-driven black-box models when applied to computer-networks problems.

Table I summarizes the most-related work and reports the main aspects of interest, and in the following we discuss in detail each listed work.

Meng et al. [19] propose *Metis*, a framework that provides interpretability for networking problems related to **local and global control**. *Metis* performs decision-trees- and hypergraph-based distillation and allows to convert DNN policies to interpretable rule-based controllers for several use cases, supporting design, debugging, deployment, and *ad-hoc* adjustment of DL-based networking systems. Results highlights improved performance or reduced computational requirements in change of limited performance degradation. Zheng et al. [13] inspect the behavior of a DNN trained to minimize average job duration in a **resource allocation** problem. They remark that *saliency map* (showing impact of each input feature on the output) and *inspecting the activation of intermediate neurons in hidden layers* are valuable tools to reveal what features a DNN depends on. The authors also propose a method aimed at improving robustness, training efficiency, and transferability (e.g. avoiding overfitting) by integrating DNNs with domain knowledge. Morichetta et al. [15] investigate explainability for unlabeled data via EXPLAIN-IT methodology: this aims at providing explanation for clustering results attainable in the analysis of QoE in YouTube video streaming. The proposed approach uses clustering results to train a classification model to **predict video quality**, which is then explained through black-box XAI approaches. Thus, the approach overcomes the limitation deriving from the lack of a ground truth which hinders the adoption of supervised quality metrics without incurring the limitations of structured insights achievable with unsupervised metrics. Dethise et al. [14] inspect the behavior of a model based on reinforcement-learning agents and targeting **video bit-rate adaptation**. The investigation is aimed at observing agent's decisions and understanding how individual input features contribute to decisions as well as the broader relationship between feature values and decisions. The study leverages data inspection and visualization and takes advantage of LIME [30]. Focusing on traffic identification and classification, Amarasinghe et al. [12] propose a framework for **anomaly detection** based on DNNs which provides explanations for the detected anomalies. In detail, the framework provides *post-hoc* explanations for DNN predictions (in terms of feature relevance scores) in order to improve users' trust. Input feature relevance is calculated via a method named Layer-wise Relevance Propagation (LRP). Tackling **robustness to adversarial samples**, Sadeghzadeh et al. [38] craft network traffic aimed at fooling DL-based TC methods, focusing separately on perturbating payload, packet size (padding), and flow statistical features (bursts). Their experiments show that DL-based TC is vulnerable to universal adversarial perturbation.

Recently a number of research contributions have investigated **interpretability and trustworthiness aspects in TC**. Beliard et al. [18] demonstrate how basic visualization tools (e.g., to represent the 1D original space, feature projections at intermediate layers, as well as the arc representation of the output confusion matrix) can be helpful in clarifying the inference process in TC tasks based on CNNs. Wang et al. [20] explore DL methods (SDAE, 1D-CNN, and LSTM) for mobile app TC. In addition, the authors uses Deep SHAP [10] to explain the outcomes obtained through 1D-CNN. The analysis focuses on a specific application (WeChat) and is limited to four (representative) outcomes, being the approach sample-dependent (viz. a local explanation method). The analysis (that focuses on both TLS and non-TLS traffic flows) highlights the importance of a number of byte chunks associated with either unknown byte sequences, or the location of specific protocol fields such as TLS `SNI` and `Cipher Suite` extensions. Rezaei et al. [16] consider a CNN model for the classification of mobile-app traffic, which is fed with the header and the payload of the first six packets of a biflow. The occlusion analysis they perform allows to inspect how the proposed architecture can classify SSL/TLS traffic flows, revealing that certain handshake fields can leak information leveraged in the classification process.

More specifically, concerning interpretation and trustworthiness of TC results, in previous work [17, 21] we have preliminary considered both. In Aceto et al. [21], we leverage Markovian distillation for interpreting traffic prediction results. In more detail, we compare Markov Chains and ML concordant/discordant predictions to highlight and interpret ML predictive patterns (focusing on outcome disagreements) by observing Markovian transition probabilities. In Aceto et al. [17], we analyze the reliability of DL-based TC framework

via calibration analysis, although this is not leveraged for any improvement of the architecture.

C. Positioning of Our Work

Like the last works reported in the aforementioned section, we focus on TC and leverage XAI techniques to get insights on the behavior of a DL architecture. Therefore we can compare our work more directly only with Aceto et al. [17], Beliard et al. [18], and Wang et al. [20]. The first work regards a novel TC architecture, whose experimental evaluation demonstrated its better *calibration* compared with the state-of-art (the evaluation has been performed on a different—partially covered by NDA—dataset). However, the present work only focused on analyzing calibration *a-posteriori*, while not explicitly instilling the calibration in the learning process.

Additionally, in the present work, we adopt Deep SHAP to infer the *importance* of a set of inputs for specific samples (a *local* approach), similarly to Wang et al. [20]. Differently from Wang et al. [20], though, we use this sample-dependent information to get to *global* explanations, allowing to interpret (and explain) the general correct behavior of the classifier, also in terms of varying input granularity. Moreover, in Wang et al. [20] the XAI analysis is only limited to a single application, whereas we report results related to 6 protocols (composing the $\approx 99\%$ of biflows of the 41 mobile apps of MIRAGE-2019 dataset) and some exemplifying apps. Differently than in Rezaei et al. [16], our investigation on interpretability leverages Shapley-based importance attribution to inputs rather than occlusion-analysis approaches, thus applying a complementary methodology.

Another significant difference with the cited work regards their lack of *calibration analysis*, that—as will be shown in Sec. V-D and thereafter—is fundamental for *trustworthiness* analysis. More in general, *the calibration step is missing from all the related works* despite its center role in explaining TC performance. While we assess the trade-off between calibration versus performance, Meng et al. [19] evaluate performance versus interpretability (as they adopt a different family of XAI approaches, impacting classification accuracy).

Finally, our adoption of an *open* dataset, focused on human-operated *mobile* apps, for the experimental evaluation sets our work apart from almost the totality of other works (saved for our own previous contribution on traffic prediction).

III. MULTIMODAL DEEP LEARNING–BASED EXPLAINABLE TRAFFIC CLASSIFICATION

In this section, we describe the proposed contribution. Specifically, in Sec. III-A, we introduce MIMETIC-ENHANCED and describe its peculiar and novel characteristics. Then, in Sec.III-B, we introduce the concept of interpretability in DL architectures and describe our approach for interpretability based on Deep SHAP technique. Finally, in Sec. III-C, we motivate the role of trustworthiness and introduce metrics to assess—and techniques to improve—the calibration properties of DL-based TC approaches.

A. Multimodal Deep Learning Traffic Classification

A multimodal DL traffic classifier exploits the multifaceted information extracted from multiple modalities (viz. views) of traffic data, with the aim of capitalizing their heterogeneous nature via *intermediate fusion* [39] of (automatically extracted) features. [3] Herein, we leverage (and then explain via XAI techniques) a novel architecture for multimodal TC, proposing an *enhanced* version of a classifier based on the recently-proposed generic MIMETIC framework [7].

Going into details, the generic MIMETIC framework consists of P different inputs (*modalities*) for each traffic object to be classified. Such DL architecture is first composed by a number of *single-modality* (viz. input-specific) *layers*, allowing to extract in an increasingly-abstract fashion the discriminative features pertaining to the p^{th} view to capitalize intra-modality dependence. On the top of these layers, the abstract features are joined via a merge layer (e.g., concatenation), which channels the modality-specific distilled information toward a joint multimodal representation. Finally, the architecture is completed with a few *shared-representation layers*, distilling features capturing inter-modality dependencies, and the usual softmax layer associated to the mobile TC task considered. In what follows, we describe the MIMETIC-ENHANCED classifier based on the framework previously-recalled.

Based on the results of previous work [7, 8, 9, 40, 41] and to provide a coherent comparison with these baselines (cf. Sec. IV-B), our MIMETIC-ENHANCED classifier operates at *biflow* level, namely the mobile TC task seeks to assign to each biflow a class within the set $\{1, \ldots, L\}$, with L denoting the number of different apps. A biflow (viz. a bidirectional flow) is a traffic object encompassing all the packets sharing the same 5-tuple (i.e. source and destination IP address, source and destination port, and transport-level protocol) in both upstream and downstream directions [42].

In more detail, MIMETIC-ENHANCED consists of $P = 2$ *single-modality* branches for extracting the corresponding intra-modality features. Each branch is fed with one input type chosen among unbiased input data [42] recommended in aforementioned related work, being either the first N_b bytes of transport-layer payload arranged in byte-wise format ($\texttt{PAY}_{\texttt{N}_\texttt{b}}$) or informative fields extracted from the sequence of the first N_p packets ($\texttt{PSQ}_{\texttt{N}_\texttt{p}}$) [4]. Specifically, we consider the first $N_b = 576$ bytes of transport-layer payload ($\texttt{PAY}_{576}$) as the input of the first branch, whereas the informative fields extracted from the sequence of the first $N_p = 12$ packets ($\texttt{PSQ}_{12}$) as the input of the second branch. In the following, the above branches will be simply named PAY-modality and PSQ-modality, respectively.[5]

[3] This peculiar procedure optimizes the less sophisticated *early* (or *data*) *fusion* and *late* (or *score/decision*) *fusion* that are not able to fully exploit the potentiality of multi-modality.

[4] Number of bytes in transport-layer payload, TCP window size (set to zero for UDP packets), inter-arrival time, and packet direction $\in \{-1, 1\}$. We stress that we neither consider IP addresses nor ports as relevant inputs. Indeed, both would imply lack of generalization and lead to inflated results not reflecting those attained in real deployment scenarios [42].

[5] We underline that these choices have been driven by both our past experience [11, 42] and further preliminary analyses (not shown for brevity).

First, to augment the information carried by input data, both branches exploit a *trainable* embedding layer to embed each input element[6] into a vector of dimension $e = 10$. This results in an overall embedding matrix $E \in \mathcal{R}^{N \times e}$ with N denoting the input dimensionality, i.e. equal to 576 and 12 for the PAY- and PSQ-modality, respectively. We emphasize that we have considered both other embedding techniques (e.g., non-trainable one-hot encoding) and other vector dimensions (i.e. $e = 5$ and $e = 30$), obtaining worse TC performance or increased (i.e. computationally unfeasible) training complexity.

Then, in the PAY-modality branch, the related embedding matrix is fed to the following single-modality layers: two 1D convolutional layers—made of 16 and 32 filters, respectively, with kernel size of 25 and unit stride—each followed by a 1D max-pooling layer—with unit stride and spatial extent of 3—and, finally, by one dense layer with 256 neurons. Conversely, the layers of the PSQ-modality branch are a Bidirectional Gated Recurrent Unit (BiGRU)—with 64 units and return-sequences behavior—and one dense layer with 256 neurons. To capture inter-modality dependencies, the abstract features extracted by the single-modality branches are joined via a concatenation layer and fed to a *shared* dense layer—with 128 neurons—before performing the classification through a softmax. All the layers exploit Rectified Linear Unit (ReLU) activations providing better performance than other tested variants (e.g., Scaled Exponential Linear Unit).

MIMETIC-ENHANCED is trained via a two-phase procedure: an independent *pre-training* of each single-modality branch (for 25 epochs) and a subsequent *fine-tuning* of the whole MIMETIC-ENHANCED architecture (for 40 epochs) after freezing the lower single-modality layers (i.e. the 1D convolutional and BiGRU layers). In the former phase, the individual pre-training of the p^{th} single-modality branch is achieved by means of a *softmax stub* [7]. In Sec. III-B, we will discuss how the latter (two) auxiliary outputs are exploited to provide isolated (per-modality) interpretation analysis. Pre-training and fine-tuning minimize respective categorical cross-entropy loss functions [7] via the standard ADAM optimizer [43] (set with a batch size of 50 samples). In more detail, MIMETIC-ENHANCED adopts a learning-rate scheduler that halves the learning rate every 5 epochs, starting with a value of $2 \cdot 10^{-3}$ and 10^{-3} for pre-training and fine-tuning, respectively. This variable learning rate shows improved TC performance compared with a constant learning rate optimization, employed e.g. in the original MIMETIC classifier (cf. Sec. V-B). Also, to foster regularization and avoid overfitting, we add a dropout of 0.2 at the end of each single-modality branch and after every dense layer, and adopt an early-stopping technique measured on the training[7]. accuracy [46].

B. Interpreting the Working Principle of DL-based Traffic Classifiers via Deep SHAP

For complex models, such as DL architectures, the trained model $f(\cdot)$ *cannot be used for explanation purposes*, as its internal behavior is hard to understand/interpret. Accordingly, the key idea of *interpretability techniques* is to use a simpler explanation model $g(\cdot)$, which is designed to closely-approximate the original model $f(\cdot)$.

In the context of explaining the predictive behavior of DL-based traffic classifiers, the model $f(\cdot)$ is chosen herein as the soft-output associated to the generic i^{th} app, i.e. $p_i(\cdot)$. By doing so, we are able to explain which set of inputs contributes the most to the confidence probability value associated to a given app.

In this paper, we focus on *local methods*, which try to explain the original model $f(\boldsymbol{x})$ when the sample value $\boldsymbol{x}$ is given as input, i.e. *only* in the neighborhood of this peculiar value. Accordingly, the use of local methods in our case corresponds to a *per-biflow explanation*. Specifically, (local) explanation models use the so-called *simplified inputs* $\boldsymbol{x'}$ that map to the original inputs through a mapping function $\boldsymbol{x} = \boldsymbol{h_x}(\boldsymbol{x'})$. Accordingly, local methods try to ensure $g(\boldsymbol{z}^{'}) \approx f(\boldsymbol{h_x}(\boldsymbol{z}^{'}))$ whenever $\boldsymbol{z}^{'} = \boldsymbol{x}^{'}$. The per-sample explanation results based on local methods will be then aggregated (viz. pooled) to reach global explanations, as explained at the end of this subsection. Most of the interpretability techniques assume a peculiar functional form for the explanation model $g(\cdot)$, which is described hereinafter.

Additive Feature Attribution (AFA) Methods: Herein we focus on the (wide) class of explanation models, referred to as *AFA methods*, which are linear functions of binary variables, i.e.

$$g(\boldsymbol{z}^{'}) = \phi_0 + \sum_{m=1}^{M} \phi_m \, z_m^{'} \tag{1}$$

where $\boldsymbol{z}^{'} \in \{0,1\}^M$, M denotes the number of simplified inputs and $\phi_m \in \mathbb{R}$. AFA methods provide an explanation model attributing an "effect" ϕ_m to each input, and summing the effects of all input attributions approximates the original model output $f(\boldsymbol{x})$. The most used interpretability techniques (e.g., LIME, DeepLIFT, etc.) belong to this family of explanation models. Additionally, when the functional form in Eq. (1) is required to satisfy (i) local accuracy (i.e. $g(\boldsymbol{z}^{'}) = f(\boldsymbol{h_x}(\boldsymbol{z}^{'}))$ when $\boldsymbol{z}^{'} = \boldsymbol{x}^{'}$), (ii) missingness, and (iii) consistency properties, there exists a unique AFA solution satisfying them [10]. Such solution coincides with the computation of the well-known *Shapley values*.

Shapley values originate from cooperative game theory [47] and identify the contribution of player m to the payoff $v(\mathcal{P})$

[6]To reduce the training complexity due to the explosion of input dimensionality, in the PSQ-modality we embed only the number of bytes in transport-layer payload.

[7]We highlight that the most common approach requires a validation set to monitor early-stopping. However, due to the class-imbalance typically encountered in mobile TC, some apps in the training set may have a limited number of samples. Hence, using part of the (whole) training set for validation could impair performance associated with minority classes. For this reason, we use early-stopping on training data by monitoring the "knee" of the training accuracy, and exiting when this condition is satisfied. Similar approaches were also recently proposed in [44, 45]

achieved by the overall coalition $\mathcal{P}$. To do so, this method assesses the payoff of *every subset* of cooperating players $\mathcal{S} \subset \mathcal{P}$ and tests the effect of removing/adding the player m to $\mathcal{S}$ on the total payoff $v(\mathcal{S})$ obtained by $\mathcal{S}$ if they cooperate.

When transposing the method to the task of explaining a DL-based model, the input data maps into the players of the cooperative game, whereas the DL architecture output $f(\boldsymbol{x})$ corresponds to the payoff function, and the m^{th} Shapley value equals

$$\phi_m(f, \boldsymbol{x}) \triangleq \sum_{\boldsymbol{z}^{'} \subseteq \boldsymbol{x}^{'}} \rho(\boldsymbol{z}^{'}) \left[f_{\boldsymbol{x}}(\boldsymbol{z}^{'}) - f_{\boldsymbol{x}}(\boldsymbol{z}^{'} \backslash m) \right] \quad (2)$$

where $\rho(\boldsymbol{z}^{'}) \triangleq \frac{|\boldsymbol{z}^{'}-1|!\left(M-|\boldsymbol{z}^{'}|-1\right)!}{M!}$, and we have used the compact notation for the composite function $f_{\boldsymbol{x}}(\cdot) \triangleq f(\boldsymbol{h}_{\boldsymbol{x}}(\cdot))$. Additionally, $\boldsymbol{z}^{'} \backslash m$ denotes the vector $\boldsymbol{z}^{'}$ in which the m^{th} entry has been set to zero, and $|\boldsymbol{z}^{'}|$ denotes the number of non-zero entries of $\boldsymbol{z}^{'}$. The sum is carried out over all $\boldsymbol{z}^{'}$ vectors whose m^{th} component is non-zero. The Shapley value is the weighted mean of all these contributions.

Shapley Values Computation with SHAP: Unluckily, the time required for exact computation of Shapley values *grows exponentially* with the input size M. Indeed, the computation of $f_{\boldsymbol{x}}(\boldsymbol{z}^{'}) = f(\boldsymbol{h}_{\boldsymbol{x}}(\boldsymbol{z}^{'}))$ would require the model to be trained-tested with each subset of the inputs withheld. Conversely, SHapley Additive exPlanation (SHAP) *approximates* these quantities via the following conditional expectation [10]:

$$f(\boldsymbol{h}_{\boldsymbol{x}}(\boldsymbol{z}^{'})) \approx \mathbb{E}\left\{f(\boldsymbol{z}) | \boldsymbol{z}_{\mathcal{S}}\right\} \quad (3)$$

where $\mathcal{S}$ denotes the set of non-zero indices within $\boldsymbol{z}^{'}$. The above approximation thus eliminates the need to retrain the model and allows the above quantity to be computed for all the values of $\boldsymbol{z}'$ in a computationally-efficient fashion (i.e. a sample expectation w.r.t. the withheld inputs on the original model $f(\cdot)$).

The expression for $f(\boldsymbol{h}_{\boldsymbol{x}}(\boldsymbol{z}^{'}))$ is further simplified by assuming (a) statistical independence among the inputs and (b) linearity of the model $f(\boldsymbol{x}) = \sum_{m=1}^{M} w_m x_m + b$ [10]. Indeed, when both these assumptions hold, it can be shown that ϕ_m's are in *closed-form* and equal to:

$$\phi_m(f, \boldsymbol{x}) = w_m \left(x_m - \mathbb{E}\left\{x_m\right\}\right) \quad (4)$$

The aforementioned simplification for ϕ_m's is capitalized as described next.

From DeepLIFT to Deep SHAP: DeepLIFT represents a recursive prediction explanation method for DL architectures [33], which attributes to each input x_m a value $C_{\Delta x_m \Delta o}$ that represents the effect of that input being set to a reference value as opposed to its original value. This means that for DeepLIFT, the mapping $\boldsymbol{x} = \boldsymbol{h}_{\boldsymbol{x}}(\boldsymbol{x}')$ converts binary values into the original inputs, where 1 (resp. 0) indicates that an input takes its original (resp. reference) value. The reference value, though being a user-defined parameter, is typically chosen to be an uninformative background value for the m^{th} input. The input effects obtained via DeepLIFT satisfy a "summation-to-delta" property, namely $\sum_{m=1}^{M} C_{\Delta x_m \Delta o} = \Delta o$, where $o = f(\boldsymbol{x})$ is the model output, $\Delta o = f(\boldsymbol{x}) - f(\boldsymbol{r})$, $\Delta x_i = (x_i - r_i)$ and $\boldsymbol{r}$ is the reference input.

DeepLIFT capitalizes a *linear composition rule* for the calculation of the $C_{\Delta x_m \Delta o}$'s, which is based on the linearization of the non-linear components of a neural network, such as (soft)max, products, or divisions. The back-propagation rules of the original DeepLIFT proposal (defining how each component is linearized) were however *heuristically* chosen [33].

Clearly, the above explanation model fits within the functional form of AFA methods described via Eq. (1), when setting $\phi_0 = f(\boldsymbol{r})$. Also, since DeepLIFT is an AFA method that satisfies (i) local accuracy and (ii) missingness, Shapley values represent the *unique* attribution solution that satisfies consistency (iii). Accordingly, the adaptation of DeepLIFT to become an analogous compositional approximation of SHAP values (i.e. using output expectation as a reference value and resorting to explicit Shapley equations, described in Eq. (4), for consistent linearization), leads to *Deep Shap* [10], a fast approximation algorithm for SHAP values which capitalizes the connectionist nature of DL architectures.[8]

Therefore, this approach will be used to assess the importance of inputs selected from the raw traffic data (e.g., related to PAY or PSQ modalities) of a given biflow in classifying the mobile app generating it. The details are provided in what follows.

Local Explanation of DL-based Multimodal Traffic Classifiers: Based on the aforementioned discussion, we interpret the SHAP value ϕ_m as the *importance value* of the m^{th} input x_m (belonging to the raw traffic data $\boldsymbol{x}$) in forming the confidence p_i associated to the i^{th} app for the biflow tested. We recall that, since $\phi_m \in \mathbb{R}$ (i.e. values can be also negative), the meaning of importance values should be interpreted as follows: positive (resp. negative) values increase (resp. decrease) the confidence in the i^{th} app with respect to its average value. Indeed, the sum of the SHAP values equals the considered soft-output value ($p_i(\boldsymbol{x})$) minus the so-called *base output*. The latter represents the average of the same soft-output obtained in correspondence of the samples associated to the background set, i.e. $\mathbb{E}\{p_i\}$.

In this work, for each biflow, we will focus on explaining the soft-output associated to the predicted app $\hat{p}(\boldsymbol{x}) = \max_{i=1,\ldots,L} p_i(\boldsymbol{x})$, as this represents the most relevant (and highest) output in the TC process.

The additive form of Shapley values (as this method belongs to the class of AFA methods) allows for readily evaluation of the importance which can be attributed to non-overlapping subset of inputs. Such analysis well suits assessing to which degree the *different modalities* (as a whole) of a multimodal DL traffic classifier contribute to the interpretation of the predicted outcome (see e.g. later analysis in Sec. V-E). To do so, we denote the input subsets associated with the P modalities considered as $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_P$, where $\boldsymbol{x} = \bigcup_{p=1}^{P} \boldsymbol{x}_p$. Then, to quantify the importance of the p^{th} modality to multimodal TC effectiveness, we resort to a *pooled* SHAP value $\phi_{\mathcal{M}_p}$. This term represents the *importance value* of

[8] We highlight that since Deep SHAP relies on an approximate computation of Shapley values, the local accuracy property $f(\boldsymbol{x}) = g(\boldsymbol{x})$ may be not satisfied with perfect equality. Still, experimental results reported in Secs. V-E, V-F and V-G show a *negligible discrepancy*. For instance, regarding Sec. V-E, this is quantified as a mean absolute percentage error (between original and explanation models) equal to 0.86% and 0.84% for the explanation of MIMETIC and MIMETIC-ENHANCED, respectively.

the input subset $\boldsymbol{x}_p$ (corresponding to the p^{th} modality) in classifying the biflow associated to the whole input $\boldsymbol{x}$ with the label $\hat{\ell}$. The aforementioned value is simply obtained as $\phi_{\mathcal{M}_p} \triangleq \sum_{m \in \mathcal{M}_p} \phi_m$, where $\mathcal{M}_p$ denotes the index set associated to the p^{th} input subset within $\boldsymbol{x}$. Clearly, it holds $\sum_{p=1}^{P} \phi_{\mathcal{M}_p} = \sum_{m=1}^{M} \phi_m = \left(p_{\hat{\ell}}(\boldsymbol{x}) - \mathbb{E}\left\{p_{\hat{\ell}}(\boldsymbol{x})\right\}\right)$. As a result (and similarly to the unpooled case), a positive (resp. negative) $\phi_{\mathcal{M}_p}$ increases (resp. decreases) the confidence in the predicted app $\hat{\ell}$ with respect to its average value.

In a complementary fashion, to focus *exclusively on a given modality* and quantify the corresponding importance contribution of *individual inputs* within each, we proceed as follows. Specifically, we consider the *stub output* associated to the p^{th} modality as our $f(\cdot)$, since the latter function (viz. architecture branch) *only depends* on $\boldsymbol{x}_p$. By doing so, we are able to focus *solely* on the behavior of the p^{th} single-modality branch, i.e. before the combined effect of intermediate fusion achieved by the shared(-representation) layers (see later analyses in Secs. V-F and V-G). This procedure thus isolates the interacting effect of other modalities on the p^{th} modality, and allows *per-modality interpretation*. The (isolated) importance values associated to the input subset $\boldsymbol{x}_p$ (feeding the p^{th} modality) are represented by the SHAP values $\phi_m^{(p)}$, where $m = 1, \ldots, |\mathcal{M}_p|$.

The above methodology for interpreting the behavior of multimodal DL-based traffic classifiers will be applied to both MIMETIC and MIMETIC-ENHANCED in Secs. V-E–V-G. Still, we remark that the proposed interpretability approach is quite general and can be virtually applied to any multimodal DL-based mobile TC architecture, especially those fitting within the MIMETIC framework [7].

From Local to Global Explanations: First of all, we notice that a soft-output can assume a range of different values. Accordingly, the absolute importance of the m^{th} input may differ from sample to sample. For this reason, our proposed *global explanation* approach assumes the preliminary calculation of *normalized* SHAP values, obtained by dividing each SHAP value by their overall sum, namely

$$\widetilde{\phi}_m \triangleq \frac{\phi_m}{\sum_{m=1}^{M} \phi_m} = \frac{\phi_m}{\sum_{m=1}^{M} \left(p_\ell(\boldsymbol{x}) - \mathbb{E}\left\{p_\ell(\boldsymbol{x})\right\}\right)} \tag{5}$$

Considering $\widetilde{\phi}_m$ (as opposed to ϕ_m), allows focusing on the relative *importance* of each input (indeed, for each sample, the sum of the importance values equals one). By doing so, we are able to draw out importance measures which do not depend on the peculiar architecture confidence (generally higher or lower) by aggregating to obtain global explanation over different samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$. Additionally, as in [20], we solely focus on the aggregation of correctly-classified samples: this choice allows focusing on the correct behavior of a given DL-based traffic classifier, allowing to interpret its counter-intuitive (while right) decisions *a posteriori*.

In detail, to obtain the global explanations pertaining to different granularities, we will consider the following views of aggregation: (i) over the whole MIRAGE-2019, (ii) related to a given protocol (e.g., TLS), and (iii) focused on a given app (e.g., Spotify). We would remark that no implicit assumption about the nature of the protocols is made: the proposed explanation approach is *protocol-agnostic*. Indeed, it can investigate input importance also when: (i) the protocol conveying the traffic is unknown; (ii) the protocol conveying the traffic is not open (proprietary protocols); (iii) the protocol adopts encryption; (iv) the protocol fields have variable length. Moreover, our input importance analysis can be leveraged—without further assumptions on the protocol—for assessing the robustness to adversarial attacks (or even detect them, as done in [48]): we leave this to future work.

C. Calibration in Deep Learning

Other than the accuracy of the considered DL-based traffic classifiers, it is of great importance to evaluate their *reliability* [49]. Such a property is fundamental in many critical scenarios and constitutes a building block of XAI since it assesses the degree of trustworthiness in providing TC outputs with high confidence (or not).

Metrics to assess calibration: Formally speaking, given an input sample $\boldsymbol{x}$ to the DL-based traffic classifier under analysis, we will analyze the trustworthiness of the whole confidence vector (i.e. $\boldsymbol{p}(\boldsymbol{x}) = p_1(\boldsymbol{x}), \ldots, p_L(\boldsymbol{x})$) and of the confidence associated to the predicted app (i.e. $\hat{p}(\boldsymbol{x}) = \max_{i=1,\ldots,L} p_i(\boldsymbol{x})$). In what follows, we introduce a graphical visualization and three metrics to assess calibration, with the first two being related to a weaker definition of calibration, and the third to a stronger one [23].

Indeed, a *miscalibrated* classifier returns excessively optimistic or pessimistic confidence outputs associated to its decisions. On the other hand, a *confidence-calibrated* classifier is such that for each sample, the confidence in the predicted app $\hat{p}$ equals $\Pr\left\{\hat{\ell} = \ell | \, \hat{p}\right\}$, where ℓ is the true app and $\hat{\ell}$ is the predicted label. That is, having e.g. 80% confidence leads to 80% accuracy. To visualize the above property for varying $\hat{p}$, we use the so-called *reliability diagrams* [24, 50], which show the accuracy as a function of the confidence (i.e. $\Pr\left\{\hat{\ell} = \ell | \, \hat{p}\right\}$ vs. $\hat{p}$). The obtained diagram is usually compared with the ideal $\Pr\left\{\hat{\ell} = \ell | \, \hat{p}\right\} = \hat{p}$ identity line. Hence, a perfectly-calibrated classifier entails a reliability diagram corresponding to the identity function. These diagrams are evaluated by partitioning the predictions into M equally-spaced bins (with extent $1/M$) and computing the accuracy of each bin. Let B_m be the set of evaluated samples such that the confidence associated to the predicted app falls within the interval $I_m \triangleq (\frac{m-1}{M}; \frac{m}{M}]$, the corresponding bin accuracy equals:

$$\mathrm{acc}(B_m) = \frac{1}{|B_m|} \sum_{n \in B_m} 1(\hat{\ell}_{(n)} = \ell_{(n)}) \tag{6}$$

where $\ell_{(n)}$ and $\hat{\ell}_{(n)} \triangleq \arg\max_{i=1,\ldots,L} p_i(n)$ are the true and predicted labels for the n^{th} sample, respectively. Confidence values range in $[1/L\,,\,1]$, where L is the number of classes of the mobile TC task. Accordingly, the starting-point of the confidence interval actually coincides with $1/L$.

To obtain a concise metric of the deviation from perfect calibration, we integrate the above diagrams with the *Expected Calibration Error (ECE)*, defined as

$\mathbb{E}_{\hat{p}}\left\{\left|\Pr\left\{\hat{\ell}=\ell|\,\hat{p}\right\}-\hat{p}\right|\right\}$, and the *Maximum Calibration Error (MCE)*, defined as $\mathrm{MCE} \triangleq \max_{\hat{p}}\left|\Pr\left\{\hat{\ell}=\ell|\,\hat{p}\right\}-\hat{p}\right|$. The former metric represents the expected absolute deviation between the confidence and the confidence-conditional accuracy, whereas the latter is the maximum absolute deviation from the identity line. The two aforementioned metrics can be (approximately) calculated as

$$\mathrm{ECE} \approx \sum_{m=1}^{M} \frac{|B_m|}{N} \left|\mathrm{acc}(B_m) - \mathrm{conf}(B_m)\right| \tag{7}$$

$$\mathrm{MCE} \approx \max_{m=1,\ldots,M} \left|\mathrm{acc}(B_m) - \mathrm{conf}(B_m)\right| \tag{8}$$

respectively. The above expressions are based on the overall number of tested samples N and the averaged confidence within bin B_m, obtained as:

$$\mathrm{conf}(B_m) = \frac{1}{|B_m|} \sum_{n \in B_m} \hat{p}_{(n)} \tag{9}$$

where $\hat{p}(n) \triangleq \max_{i=1,\ldots,L} p_i(n)$ denotes the predicted confidence of the n^{th} sample.

The ECE and MCE metrics only consider the confidence in the predicted app, while ignoring the other scores in the softmax distribution. A *stronger* definition of calibration requires the probabilities of all the classes in the softmax distribution to be calibrated, namely to have p_i equal to $\Pr\{\ell = i|\, p_i\}$ for $i = 1, \ldots, L$, i.e. having 80% confidence for the i^{th} app leads to 80% probability of observing that app. A concise metric which relies on the above stronger calibration definition is the *Class-Wise Expected Calibration Error (CW-ECE)* [26], defined as $\frac{1}{L}\sum_{i=1}^{L}\mathbb{E}_{p_i}\left\{\left|\Pr\left\{\ell=i|\,p_i\right\}-p_i\right|\right\}$. Such a metric is evaluated as the class-wise sum $\text{CW-ECE} = \frac{1}{L}\sum_{i=1}^{L}\text{CW-ECE}_i$, where:

$$\text{CW-ECE}_i \approx \sum_{m=1}^{M} \frac{|B_{m,i}|}{N} \left|\varrho(B_{m,i}) - \mathrm{conf}(B_{m,i})\right| \tag{10}$$

In the latter definition, $B_{m,i}$ denotes the set of samples whose prediction for the i^{th} app p_i falls within the m^{th} bin, and $\mathrm{conf}(B_{m,i})$ (resp. $\varrho(B_{m,i})$) the corresponding bin-averaged confidence probability (resp. proportion of samples labeled as the i^{th} app).

Techniques to Improve Calibration: We now describe two relevant techniques to improve calibration in DL-based architectures. According to the literature, all the methods to ensure a better calibration of a classifier involve a modification of the training process and/or of the training loss.

- *Label Smoothing* can be interpreted as a form of loss regularization [27] to improve the generalization ability of models and reduce an excessively high prediction confidence. Such a technique encompasses, during the training phase, that the cross-entropy (CE) loss minimizes the prediction w.r.t. a *smoothed* one-hot representation of the ground truth $\ell_{(n)}$, calculated as

$$\boldsymbol{t}_{\mathrm{ls}}(n) = (1-\alpha)\,\boldsymbol{t}(n) + \frac{\alpha}{L}\,\mathbf{1}_L \tag{11}$$

where $\boldsymbol{t}(n) \triangleq \begin{bmatrix} t_1(n) & \cdots & t_L(n) \end{bmatrix}^T$ denotes the usual *one-hot* representation of the label $\ell_{(n)}$. The smoothing parameter α defines the amount of uncertainty enforced on the ground truth, with $\alpha \to 0$ collapsing to the usual non-smoothed CE-based training procedure.

- *Focal Loss* is a suitably-defined loss function for classification tasks, and is one viable solution to deal with class imbalance. However, in a recent work [26] it has been demonstrated its successful application in improving calibration, capitalizing its implicit (entropy-based) regularization properties. The explicit expression of the focal loss for the generic n^{th} sample is

$$\mathcal{L}_{\mathrm{FL}}(n) \triangleq -\sum_{i=1}^{L} t_i(n)\,(1-\hat{p}_i(n))^{\gamma}\,\log(\hat{p}_i(n)) \tag{12}$$

The overall loss is then obtained as $\mathcal{L}_{\mathrm{FL}} \triangleq \sum_{n=1}^{N}\mathcal{L}_{\mathrm{FL}}(n)$. In this case, the parameter γ controls the impact of such a loss, with $\gamma \to 0$ collapsing to the usual CE loss.

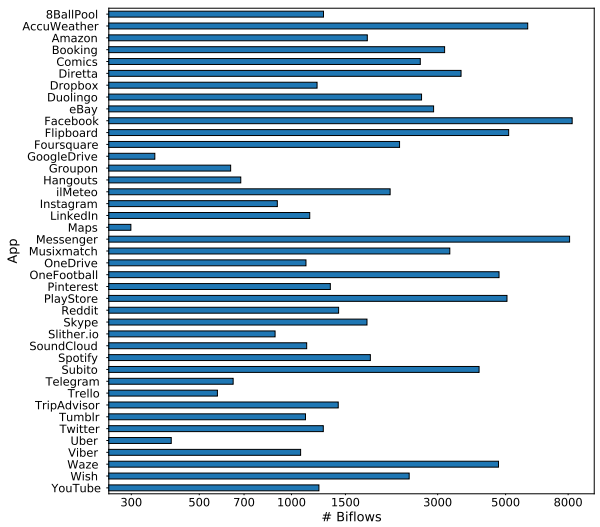


Figure 1. Number of per-app biflows in the MIRAGE-2019 dataset.

Table II
CLEARTEXT MESSAGES BY PROTOCOL.

Protocol	**Cleartext Messages**
FB-Zero[a]	`CHLO`, `SNOM`, `SCFG` (*heuristic*)
gQUIC[b]	`CHLO`, `REJ`
SSL[c] / TLS[d]	`CHLO`, `SHLO`, `Certificate`, `Server Key Exchange`, `Server Hello Done`, `Client Key Exchange`, `Change Cipher Spec`
STUN[e]	*all packets*
HTTP[f]	*packets containing HTTP headers*

[a] https://engineering.fb.com/2017/01/27/android/building-zero-protocol-for-fast-secure-mobile-connections
[b] https://tools.ietf.org/html/draft-tsvwg-quic-protocol-02
[c] RFC 7568: https://tools.ietf.org/html/rfc7568
[d] RFC 8446: https://tools.ietf.org/html/rfc8446
[e] RFC 5389: https://tools.ietf.org/html/rfc5389
[f] RFC 7235: https://tools.ietf.org/html/rfc7235

IV. Experimental Setup

This section describes the experimental setup considered in this work. Specifically, in Sec. IV-A a brief description of the dataset MIRAGE-2019 is provided. Then, in Sec. IV-B, the considered DL-based traffic classification baselines (used for comparative performance and interpretation analysis) are introduced.

A. Dataset Description

The whole experimental campaign in this study leverages the publicly released MIRAGE-2019 mobile-app traffic dataset[9,10] to foster reproducibility. The dataset was collected by more than 280 experimenters at the ARCLAB laboratory of the University of Napoli "Federico II" during the period May 2017–May 2019, exploiting the MIRAGE architecture [11]. The facility allows to collect mobile-traffic in PCAP format, which is enriched with ground-truth information that labels each biflow with the corresponding Android-package name. During the collection, the experimenters have mimicked the typical usage of each app by testing most-common functionalities (e.g., service registration and login, habitual interactions and use cases, etc.). Latest app versions available at time of capture were used. Overall, MIRAGE-2019 encompasses the traffic generated by 41 Android apps belonging to 16 categories according to the Google Play Store[11]. The dataset contains more than 4600 traffic traces, each generated in a capture session of $[5, 10]$ minutes, 91778 TCP, and 4589 UDP labeled biflows. Three distinct devices were used. For the sake of brevity, the following experimental evaluation focuses on the traffic captured with the Xiaomi Mi5.

Figure 1 details the number of biflows for each app, spanning from 299 (for `Maps`) to 8246 (for `Facebook`). Nevertheless, for 32 apps out of the 41 considered, MIRAGE-2019 contains more than 1000 biflows per app. We remark that, albeit roughly the same time has been allotted for the usage of each application during the traffic capture time, the number of biflows significantly depend on the specific app considered: this constitutes a real-world and challenging scenario for the interpretability analyses provided herein.

B. Traffic Classification Baselines

This section describes state-of-the-art baselines considered in the following interpretability analyses and compared with MIMETIC-ENHANCED. All single-modal baselines are fed with either $\texttt{PAY}_{\texttt{N}_\texttt{b}}$ or $\texttt{PSQ}_{\texttt{N}_\texttt{p}}$ (see Sec. III-A for more details), both normalized within $[0, 1]$. On the other hand, similarly to MIMETIC-ENHANCED, multimodal baselines are fed with both types of input data [17].

Unless stated otherwise, all DL architectures are trained to minimize a categorical cross-entropy loss function, and exploit the Adam optimizer [43] and the early-stopping technique to prevent overfitting. Specific training parameters of each baseline (e.g., learning rate, number of epochs, batch size, etc.) are set based on the recommendation provided in the respective original studies.

Single-Modal DL Traffic Classifiers: We have selected the best single-modal DL classifiers according to extensive evaluations performed in our previous works [7, 42]. In more detail, we consider the *1D-CNN* fed with $\texttt{PAY}_{784}$ [40] and the *2D-CNN + LSTM* (named *HYBRID* hereinafter) fed with $\texttt{PSQ}_{20}$ [41].

Additionally, we evaluate the *Flow Sequence Network (FS-Net)* [9], having as input the IP packet size of the first $N_p = 12$ packets ($\texttt{PS}_{12}$). FS-Net consists of (i) an embedding layer (with a 128-dimensional vector), (ii) an encoder followed by (iii) a decoder (both a 2-layer BiGRU with 128 units) whose output is given to (iv) a reconstruction layer (with softmax activation), (v) a dense layer (with SeLu activation) combining encoder and decoder outputs, and (vi) a classification layer (with softmax activation). After each intermediate layer, a dropout of 0.3 is employed to avoid overfitting. For FS-Net, the loss function is the weighted sum of reconstruction and classification categorical cross-entropy losses.

Finally, to investigate the performance of "shallow" learning, we consider a MultiLayer Perceptron encompassing one hidden dense layer with 100 neurons (named *MLP-1* hereinafter) trained with either $\texttt{PAY}_{784}$ or $\texttt{PSQ}_{20}$.

Multimodal DL Traffic Classifiers: MIMETIC-ENHANCED is compared also with two multimodal baselines recently proposed in related literature. The first baseline is the original MIMETIC architecture we have previously proposed [7]. MIMETIC consists of two modalities using $\texttt{PAY}_{576}$ and $\texttt{PSQ}_{12}$ input type, respectively. The former is made up of two 1D convolutional layers (with 16 and 32 filters, kernel size of 25, and ReLu activation), each followed by a 1D max-pooling layer, and a dense layer (with 256 neurons). The latter comprises a BiGRU (with 64 units) and a dense layer (with 256 neurons). The intermediate representations of the two branches are concatenated and fed to a shared dense layer (with 128 nodes). A dropout of 0.2 is used at the end of each branch and after each dense layer. Similarly to MIMETIC-ENHANCED, the original MIMETIC is trained via a two-stage procedure based on pre-training and fine-tuning phases. It is worth noting that MIMETIC and MIMETIC-ENHANCED classifiers, despite derived from the same general multimodal framework [7], significantly differ from both the architectural and training-procedure viewpoints. In more detail, in MIMETIC-ENHANCED, we better represent the information carried from both input types—and consequently its explainability degree—by exploiting a trainable embedding layer in both modalities. Such an embedding layer changes the dimension of the input vector fed to the subsequent layers, and introduces new parameters to be learned. Moreover, MIMETIC-ENHANCED employs a novel training procedure based on an adaptive learning-rate that is halved every 5 epochs, demonstrating a beneficial effect on training with respect to simpler constant learning-rate adopted in MIMETIC (see Sec. III-A). Finally, we have also applied a calibration procedure (based on different approaches, namely focal loss and label smoothing) explicitly devised and optimized for the training phase of MIMETIC-ENHANCED (see

[9] http://traffic.comics.unina.it/mirage/

[10] https://ieee-dataport.org/open-access/mirage-mobile-app-traffic-capture-and-ground-truth-creation

[11] https://play.google.com/store/apps

Sec. III-C).

The second baseline is the multimodal *App-Net* architecture [8] which also has two modalities with $\texttt{PAY}_{200}$ and $\texttt{PS}_{12}$ as input, respectively. The first branch encompasses an embedding layer performing one-hot encoding of the input and two 1D convolutional layers (with 128 filters, kernel size of 15, and ReLu activation), each followed by a 1D max-pooling layer. The second branch consists of an embedding layer (with a 128-dimensional vector) and two BiLSTM layers (with 128 units). Lastly, the intermediate features are fused via a learned weighted element-wise addition. Here too, a dropout of 0.5 is used at the end of each branch. Unlike other baselines, *App-Net* employs the stochastic gradient descent optimizer with a variable learning rate which is halved every 3 epochs.

ML-based Traffic Classifiers: Additionally, we compare MIMETIC-ENHANCED with a Decision Tree (DT) being a simpler and (more) interpretable ML-based model. We underline that the interpretability degree of the DT is *strongly dependent* on (a) the feature set (i.e. the input dimensionality) and (b) the tree depth: the more complex the DT, the less interpretable its decision rules. Indeed, the number of leaf nodes increases exponentially with the tree depth (i.e. by 2^{depth}). Therefore, to foster a fair comparison, we feed the DT with three different feature sets: (i) $\texttt{PAY}_{576}$, (ii) flattened $\texttt{PSQ}_{12}$, and (iii) $\texttt{PAY}_{576}+\texttt{PSQ}_{12}$, where "+" denotes a flattened concatenation.

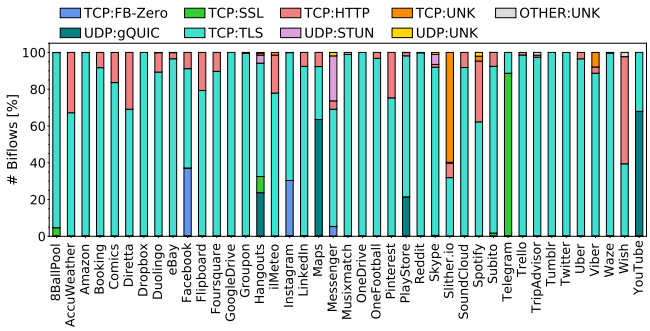
C. Implementation Details

To foster reproducibility, in addition to the public release of the MIRAGE-2019 dataset, we provide specific implementation details on the whole experimental workbench. All the APIs refer to `Python (3.7)` programming language. Specifically, we exploited the DL models provided by `Keras` (https://keras.io) and `TensorFlow 2` (https://www.tensorflow.org/), and ML models provided by `scikit-learn` (https://scikit-learn.org/) to implement, test, and calibrate the traffic classifiers described herein. Also, we leverage the original Deep SHAP implementation available at https://github.com/slundberg/shap. Data pre- and post-processing have been performed mainly by means of `numpy` (https://numpy.org/) and `pandas` (https://pandas.pydata.org/) libraries. Finally, the graphical data representation has been obtained using `matplotlib` (https://matplotlib.org/) and `seaborn` (https://seaborn.pydata.org/) libraries.

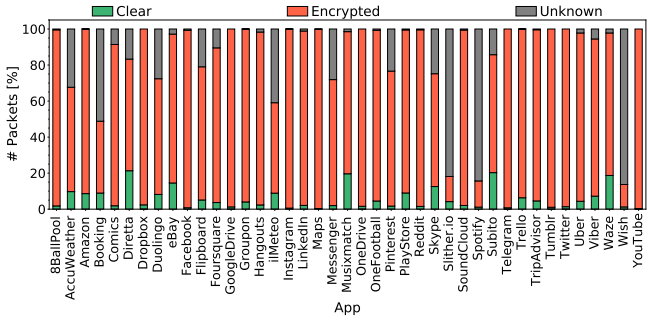
All the experiments refer to the same hardware architecture, that is an OpenStack virtual machine with 16 vCPUs and 32 GB of RAM, and Ubuntu 16.04 (64 bit) operating system, running on a physical server with 2 × Intel(R) Xeon(R) E5-4610v2 CPUs @ 8 × 2.30 GHz and 64 GB of RAM. Regarding the evaluation of (per-epoch) training complexity of DL approaches, we highlight that all the execution times have been computed via `time.process_time()` to take into account only the actual runtime on the CPUs (i.e. in a sequential fashion).

V. EXPERIMENTAL EVALUATION

This section reports and discusses the experimental evaluation performed in this work. The latter is based on a *stratified ten-fold cross-validation* that keeps stable the share of per-app samples among the folds and allows a TC performance evaluation on the whole dataset. Hence, for each considered metric we report the average—and, when needed, the standard deviation—of the evaluations on the ten folds. More specifically, in Sec. V-A, a fine-grained characterization of MIRAGE-2019 dataset is provided. Then, in Sec. V-B, we compare the proposed MIMETIC-ENHANCED against the considered baselines in terms of TC performance. A first in-depth analysis on the soft-output of our proposal is provided and discussed in Sec. V-C. The trustworthiness of MIMETIC-ENHANCED predictions (against the baselines) is quantified (and improved) in Sec. V-D. Finally, the interpretability of MIMETIC-ENHANCED TC-results is discussed in Secs. V-E, V-F and V-G, focusing on relative contribution of each modality, and per-modality investigation of these inputs.



(a) Protocol distribution in terms of biflows. *UNK* stands for *unknown*, *SSL* stands for *undetected version of SSL/TLS*.



(b) Heuristic results: clear, encrypted, and unknown packets.

Figure 2. MIRAGE-2019 per-app traffic characterization.

A. Dataset Characterization

In this section, we report the results of the dissection of the app traffic in terms of adopted protocols, based on the well-known `tshark` library.[12] We also discuss how we infer the nature (clear vs. encrypted) of the payload at per-packet granularity (based on identification of known protocol messages).

Figure 2a depicts the distribution of protocols in terms of biflows for each app within the dataset. For each biflow, we

[12]In detail, `PyShark` (https://kiminewt.github.io/pyshark) and `Scapy` (https://kiminewt.github.io/pyshark) were employed.

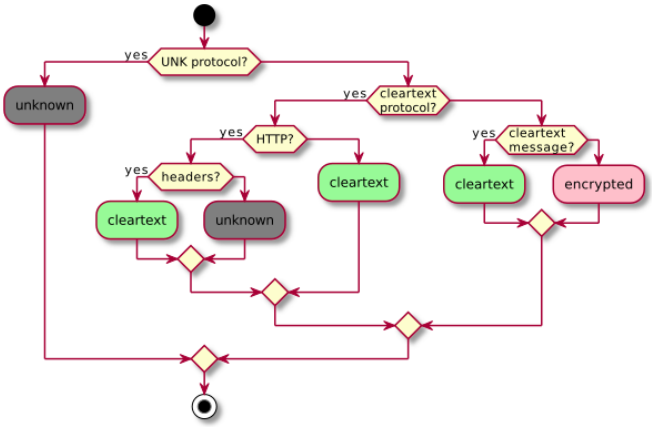


Figure 3. Proposed heuristic to identify the encrypted portion of app traffic in MIRAGE-2019.

consider the highest protocol in the stack the dissector is able to interpret. For the sake of clarity, in the following we refer to the protocol classes in our dataset reporting the TCP/IP stack portion from transport layer and the one above, in the form *L4:L5*.

Firstly, we can notice that the *Transport Layer Security (TLS)* protocol (`TCP:TLS`) constitutes $\approx$ 80% of overall biflows and each app has 15% TLS biflows, at least.

Inspecting the $\approx$ 76k `TCP:TLS` biflows in the dataset, 94% employ the 1.2 version *or above*[13] , while the remaining 6% the `1.0` version (in fact, only 3 TLS 1.1 biflows can be found). Analyzing the latter 6% in more depth, 95% of these biflows are associated with the sole `Waze` app, while the remaining part is shared by `eBay`, `Twitter`, `Tumblr`, `TripAdvisor`, `Subito`, `OneDrive`, `Flipboard`, and `Facebook`. Notably, TLS 1.0 constitutes 93% of TLS `Waze` biflows, while for the other mentioned apps, this protocol accounts for smaller shares (less than 0.05% of the biflows for all the mentioned apps but `Facebook`, for which TLS 1.0 represents 5% of the TLS traffic in terms of biflows).

According to the `tshark` protocol dissection, 19 apps have at least one biflow marked as *Secure Sockets Layer (SSL)* protocol (which conveys $\approx$ 85% of the biflows for `Telegram`, in particular).[14] We highlight that this naming choice is due to the default labeling `tshark` adopts when the dissector is unable to detect the specific version of SSL/TLS. We have chosen to follow the same convention for coherence with the usage of `tshark` response as ground truth for protocol type, and report this *undetected version of SSL/TLS* as `TCP:SSL`.

Other prominent protocols in the dataset are *Facebook Zero (FB-Zero)* and *Google Quick UDP Internet Connection (gQUIC)* (`TCP:FB-Zero` and `UDP:gQUIC`, respectively). As expected, the former is used only by *Facebook Inc.'s* apps (i.e. `Facebook`, `Instagram`, and `Messenger`), whereas the latter by *Alphabet Inc.'s* ones (i.e. `Maps`, `PlayStore`, `Hangouts`, and `YouTube`). For these apps, up to $\approx$ 37% (for `Facebook`) and $\approx$ 63% (for `YouTube`) of the biflows utilize FB-Zero or gQUIC, respectively. Notably, 32 out of 41 apps present HTTP biflows (`TCP:HTTP`), with the highest shares ($>$ 30% of biflows) exhibited by `Wish`, `Spotify`, `AccuWeather`, and `Diretta`. For a limited number of biflows, the dissecting process could not go further the identification of the presence of the transport-layer headers with `TCP:UNK` and `UDP:UNK` accounting for $\approx$ 0.7% and $\approx$ 0.2% of the overall biflows. A more limited portion of the biflows employ STUN protocol (`UDP:STUN`), which represent the $\approx$ 24% of `Messenger` biflows. Finally, $<$ 0.1% of the biflows in the dataset do not match any of the mentioned protocols and are grouped in `OTHER:UNK`.

The distribution of protocols shown in Fig. 2a also provides hints about the nature of the traffic: protocols that natively support payload encryption (i.e. TLS, SSL, FB-Zero, and gQUIC) can be easily distinguished from those that do not (i.e. HTTP, STUN, TCP, UDP). It is worth to notice that: (i) protocols supporting payload encryption also dictate for portions of the biflows to be conveyed in cleartext (e.g., messages that establish the initial secure connection); (ii) applications may exchange encrypted data through protocols that do not support payload encryption (e.g. when content encryption is enforced at application level). To provide a different viewpoint with respect to protocols, i.e. to identify the encrypted portion of the app traffic at a finer granularity, now *we focus on L4 payload*. To count the encrypted packets we adopt a conservative approach that is based on packet dissection to detect application-layer messages and takes advantage of known protocol characteristics to classify a packet as *clear*, *encrypted*, or *unknown*, according to an heuristic described as flow chart in Fig. 3. Details about the messages considered to be in clear for each protocol are reported in Tab. II. On the other hand, STUN only uses cleartext.[15] Besides such cleartext messages, the payload of `TCP:FB-Zero`, `UDP:gQUIC`, `TCP:SSL`, and `TCP:TLS` biflows is therefore considered to be encrypted. Note that for biflows marked as `TCP:SSL`, usually the packet dissector does not provide detailed information on the message type, thus we could not infer the related packet nature. Moreover, from version 1.3 the TLS standard mandates an opaque *content type* header value corresponding to generic "application data", no longer revealing the actual TLS message type (now encrypted). Differently, concerning HTTP, only the packet containing HTTP headers is considered to be in clear, while the packets delivering the body of the message are marked as *unknown*. We report the results of the heuristic applied to the dataset in Fig. 2b, which depicts the percentage of *clear*, *encrypted*, and *unknown* packets for each app. As expected, this information is strongly related to the (per-biflow) protocol distribution associated to each app: cleartext traffic constitutes $\approx$ 5–10% of per-app packets which correspond to the beginning of `TCP:FB-Zero`, `UDP:gQUIC`, `TCP:SSL`, `TCP:HTTP`, and

[13]The TLS version 1.3 standard (RFC4886) mandates the usage of cleartext *protocol version* field *with the same value of TLS version 1.2*, therefore the `tshark` dissector we use as a ground truth cannot tell apart these two versions.

[14]Most of `Telegram` traffic is labeled by `tshark` as "SSL Continuation Data" despite the (open source) `Telegram` client supports TLS1.3 .

[15]STUN protocol leverages TLS (DTLS) for encryption (RFC7350), therefore `UDP:STUN` is in cleartext.

Table III
PERFORMANCE METRICS AND DEFINITIONS.

Metric	Definition
Accuracy	$\text{acc} = \frac{\sum_{i=1}^{L} tp_i}{tp_i+tn_i+fp_i+fn_i}$
F-measure	$\text{F-meas} = \frac{\sum_i^L \frac{\text{prec}_i \cdot \text{rec}_i}{\text{prec}_i+\text{rec}_i}}{L}$
G-mean	$\text{G-mean} = \frac{\sum_i^L \sqrt{\text{spec}_i \cdot \text{rec}_i}}{L}$
Recall	$\text{rec}_i = \frac{tp_i}{tp_i+fn_i}$
Precision	$\text{prec}_i = \frac{tp_i}{tp_i+fp_i}$
Specificity	$\text{spec}_i = \frac{tn_i}{tn_i+fp_i}$

True Positives, True Negative, False Positives, and False Negatives are denoted with tp, tn, fp, and fn, respectively.
The i subscript refers to the i^{th} class. L is the total number of classes.

Table IV
COMPARISON OF MIMETIC-ENHANCED ACCURACY, F-MEASURE, AND G-MEAN WITH STATE-OF-THE-ART BASELINES.

Traffic Classifier	*Accuracy [%]*	*F-measure [%]*	*G-mean [%]*
MIMETIC-ENHANCED	**92.14 (± 0.28)**	**91.47 (± 0.27)**	**95.36 (± 0.19)**
MIMETIC [7]	88.89 (± 0.25)	87.78 (± 0.45)	93.10 (± 0.46)
App-Net [8]	88.25 (± 0.21)	87.60 (± 0.38)	92.91 (± 0.25)
FS-Net [9]	87.19 (± 0.33)	86.23 (± 0.56)	92.38 (± 0.41)
1D-CNN [40]	83.97 (± 0.54)	82.68 (± 0.84)	90.29 (± 0.39)
MLP-1 ($\texttt{PAY}_{784}$)	80.12 (± 0.64)	78.79 (± 0.69)	87.70 (± 0.36)
HYBRID [41]	79.10 (± 0.83)	76.23 (± 1.09)	85.97 (± 0.73)
MLP-1 ($\texttt{PSQ}_{20}$)	59.47 (± 0.61)	54.51 (± 0.87)	70.68 (± 0.76)
DT ($\texttt{PAY}_{576} + \texttt{PSQ}_{12}$)	87.12 (± 0.45)	86.06 (± 0.48)	92.49 (± 0.28)
DT ($\texttt{PSQ}_{12}$)	85.28 (± 0.32)	83.35 (± 0.42)	90.99 (± 0.27)
DT ($\texttt{PAY}_{576}$)	84.10 (± 0.36)	83.02 (± 0.46)	90.81 (± 0.31)
MIOBB	**+ 3.25 (± 0.27)**	**+ 3.68 (± 0.37)**	**+ 2.26 (± 0.31)**

Results are in the format *avg. (± std.)* obtained over the 10-folds and refer to the 41-app mobile TC task.
Overall best classifier is highlighted in boldface.
Baselines are grouped based on their type (multimodal DL, single-modal DL, and ML-based DT) and ranked according to TC performance.
The last row shows the Maximum Improvement Over the Best Baseline (MIOBB) [%], namely MIMETIC-ENHANCED improvement over MIMETIC.

`TCP:TLS` biflows and to `UDP:STUN`. communications. In fact, notable exceptions are `Spotify` and `Wish` for which a remarkable amount of `TCP:HTTP` traffic is observed and `Slither.io` and `Telegram` that present several `TCP:UNK` and `TCP:SSL` biflows, respectively.

When performing protocol breakdown, the following analyses will omit `TCP:UNK`, `UDP:UNK`, and `OTHER:UNK` as the related amount of traffic in MIRAGE-2019 is negligible. The analyses will focus on `TCP:FB-Zero`, `UDP:gQUIC`, `TCP:SSL`, `TCP:TLS`, `UDP:STUN`, and `TCP:HTTP` whose traffic will be simply referred to as `FB-Zero`, `gQUIC`, `SSL`, `TLS`, `STUN`, and `HTTP`, respectively, for the sake of brevity.

B. Performance Comparison with State-of-the-art Baselines

In this section, we analyze in depth the performance of MIMETIC-ENHANCED against the considered baseline architectures. Table IV summarizes wrap-up results in terms of the typical classification performance metrics, whose definition is briefly reported in Tab. III. In addition to the well-known *Accuracy* (i.e. the fraction of correctly classified biflows over the total number of biflows), we also leverage per-class measures, namely *F-measure* and *G-mean*, which concisely take into account recall, precision, and specificity. Specifically, for the evaluation of multi-class traffic classifiers, we employ their arithmetically-averaged (viz. macro) versions.

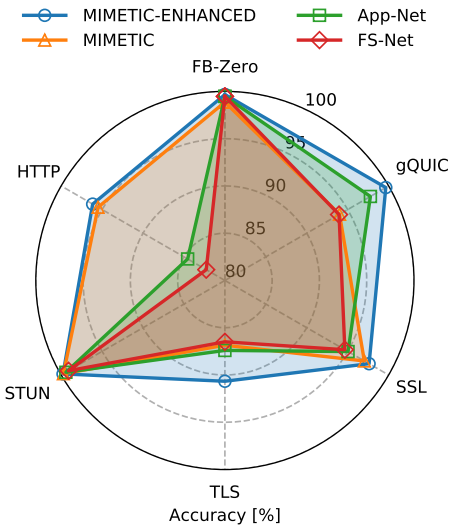


Figure 4. Accuracy of MIMETIC-ENHANCED and best baseline traffic classifiers divided by protocol.

Experimental results witness that MIMETIC-ENHANCED *performs better than all the considered baselines according to all the performance metrics*. Indeed, it guarantees remarkable improvements with respect to the best performing baseline (that is MIMETIC in all the cases), reporting performance improvements of $+3.25\%$, $+3.68\%$, and $+2.26\%$ in terms of Accuracy, F-measure, and G-mean, respectively. More in general, the performance-rank of the architectures confirms that multimodal approaches (i.e. MIMETIC-ENHANCED, MIMETIC, and App-Net) are able to achieve better—and typically less variable—performance. On the other hand, simpler DT performs almost on par with single-modal DL-based traffic classifiers when fed with the most complex $\texttt{PAY}_{576}+\texttt{PSQ}_{12}$ feature set, but has worse performance than *all* multimodal DL approaches, including MIMETIC-ENHANCED. Interestingly, when limiting the DT maximum-depth to 10 (corresponding to 1024 leaf nodes) with the aim of retaining its interpretability, we incur a significant decrease of performance down to 41.37%, 39.00%, and 48.62% in terms of Accuracy, F-measure, and G-mean, respectively, in the best configuration (i.e. with $\texttt{PAY}_{576} + \texttt{PSQ}_{12}$ feature set as input).

To shed light on where this improvement sits, we break the performance results down on the different protocols, as reported in Fig. 4. For brevity, we focus on the four best-performing architectures and we only report the analysis of the *Accuracy*, as same trends and observations can be devised by the inspection of the other metrics. Investigating the classification accuracy obtained by MIMETIC-ENHANCED for app traffic conveyed by different protocols, we can notice

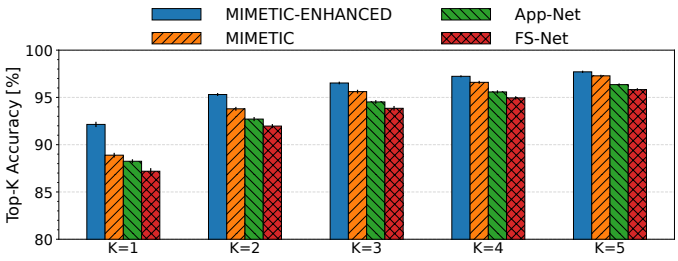


Figure 5. Top-K accuracy of MIMETIC-ENHANCED and best baseline traffic classifiers. Error bars report average ± standard deviation.

that our architecture highlights different performance on varying protocols: (i) `FB-Zero`, `gQUIC`, and `STUN` biflows are classified with $\approx 99\%$ accuracy; (ii) lower accuracy values are obtained for `SSL`, `HTTP`, and `TLS`, with `TLS` traffic being the most challenging to be classified by MIMETIC-ENHANCED (attaining however 90.66% accuracy). Interestingly, this does not hold for all the architectures (e.g., `HTTP` traffic specifically challenges App-Net and FS-Net). On the other hand, `TLS` traffic is classified with accuracy values not higher than 87.40% by the other architectures.

According to the protocol breakdown for the different architectures, we can assert that MIMETIC-ENHANCED *achieves better performance than the baselines for all the protocols*. While the improvements of classification accuracy for `FB-Zero`, `SSL`, `STUN`, and `HTTP` traffic appear limited ($\leq 1\%$ w.r.t. the best performing baseline per-protocol), remarkable accuracy enhancement can be observed for `gQUIC` and `TLS` traffic ($+4.65\%$, w.r.t. MIMETIC and $+3.81\%$ w.r.t. App-Net, respectively).

Finally, focusing on complexity, our results show that the benefits in terms of TC performance obtained by MIMETIC-ENHANCED come at a *limited complexity increase* of the training phase. Indeed, an increment of per-epoch training-phase runtime limited to $\approx 40\%$ is observed w.r.t. MIMETIC. This is paired with the *number of trainable parameters* of the two architectures which do not differ significantly, being $0.98M$ and $0.96M$ for MIMETIC-ENHANCED and MIMETIC, respectively. Still, MIMETIC-ENHANCED shows *a training-phase runtime more than* $5.5\times$ *and* $3.2\times$ *lower* than App-Net ($1.77M$ trainable parameters) and FS-Net ($6.39M$ trainable parameters), respectively[16].

C. Fine Grained Analysis based on Soft-Outputs

This section extends the performance analysis drawn in Sec. V-B providing a finer-grained study of the output of the considered DL architectures by inspecting the *soft-outputs* of the classifiers, rather than their sole final decision. Before focusing on the class-wise trends and patterns highlighted by the soft-outputs provided by MIMETIC-ENHANCED, we compare the performance of the four best architectures—according to the TC performance reported in Tab. IV—in terms of the *Top-K Accuracy*. This metric relaxes the constraints imposed in the accuracy analysis and defines a classification event as *correct* if the actual app falls within the top-K predicted labels ($K < L$ is a free parameter). Accordingly, the Top-K Accuracy provides hints about the general behavior of soft-outputs (actual classes not chosen as the most likely output are still considered correct when lying within the top-K). Note that when $K = 1$ the metric results into the standard accuracy already discussed, and that the larger the value chosen for K, the higher the Top-K Accuracy, by construction.

Figure 5 presents the Top-K Accuracy obtained for $K \in [1, 5]$. While the performance gain of MIMETIC-ENHANCED w.r.t. the baselines decreases as K increases, MIMETIC-ENHANCED *outperforms the baselines for all the values of* K *considered*. Notably, for $K = 3$ both MIMETIC-ENHANCED and MIMETIC achieve $> 95\%$ Top-K Accuracy, while the best single-modal baseline (i.e. FS-Net) attains this result with $K = 5$.

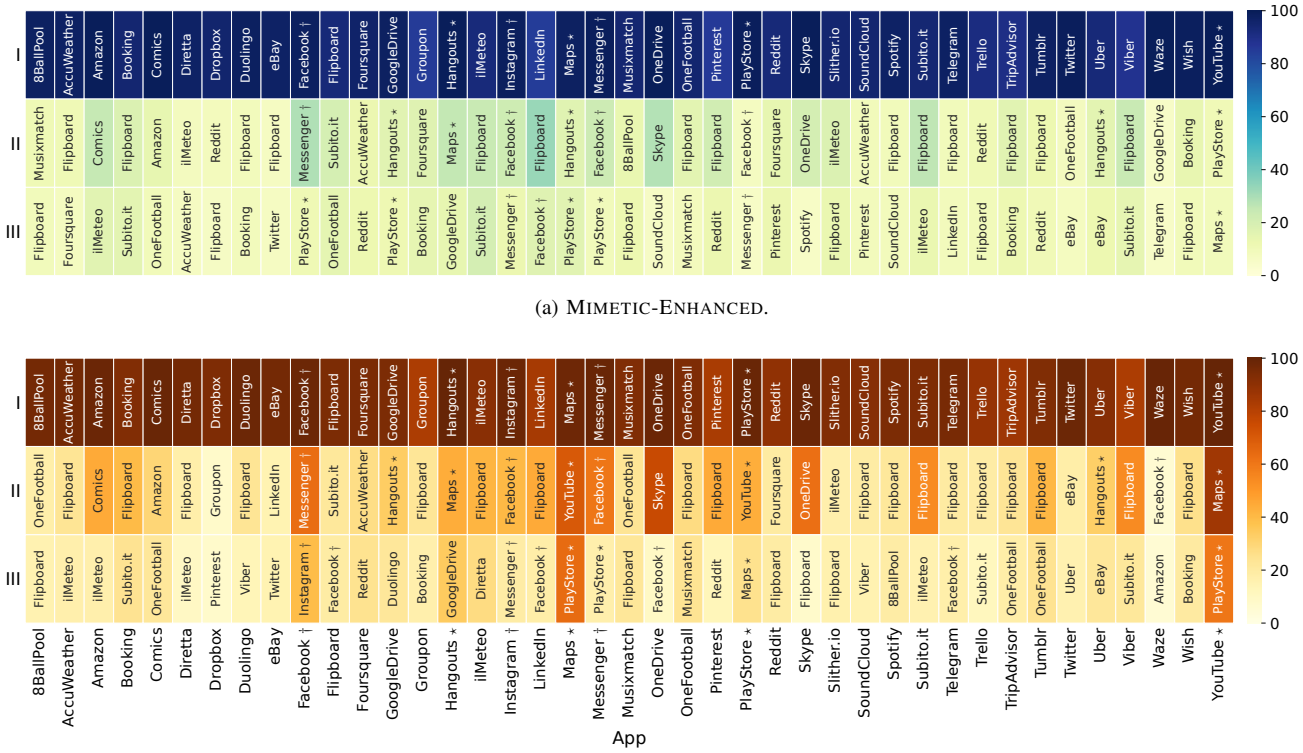
Hence, in the following analysis, we focus on the *Top-3 Accuracy* which aims at deepening the soft-output behavior of the best-performing DL architectures. In more details, for each app (actual class) we have computed the most recurring top-3 predictions based on the soft-output vectors provided by the architectures. The results of this analysis are reported in Figs. 6a and 6b for MIMETIC-ENHANCED and MIMETIC, respectively (details for App-Net and FS-Net are omitted for brevity). The values in the heatmaps represent the percentage of occurrence of the predicted apps that most-likely fall among the top-3 predictions for each actual class.

In line with the high TC-performance presented in Tab. IV and Fig. 5, for both MIMETIC-ENHANCED and MIMETIC, the correct label is always the most frequent in the top-3 (i.e. it is ranked I). Notably, *for* 36 *out of* 41 *apps,* MIMETIC-ENHANCED *shows an occurrence of this most frequent (I) correct prediction higher than* MIMETIC. Major discrepancies (w.r.t. MIMETIC) are observed for `Viber` ($+5.79\%$) and `TripAdvisor` ($+4.15\%$). These results suggest that for a consistent number of samples of certain apps (e.g., the above-mentioned `Viber` and `TripAdvisor` along with `Groupon` and `LinkedIn`), the actual label does fall in the top-3 for MIMETIC-ENHANCED, but not for MIMETIC. More in general, with the exception of these notable cases, the trends shown for the occurrences of the most recurring label (I) in the top-3 are qualitatively the same for the four architectures (with App-Net and FS-Net showing lower values than MIMETIC-ENHANCED and MIMETIC in almost all the cases).

On the other hand, comparing the second and the third most-likely predicted labels (II and III in Fig. 6, respectively) provided by MIMETIC-ENHANCED and MIMETIC, they often do not match. However, some predicted labels represent the second or the third choice for most of the apps. For instance, `Flipboard` appears for the $\approx 49\%$ of apps in II or III for MIMETIC-ENHANCED and for the $\approx 56\%$ for MIMETIC.

Interestingly, the frequency of occurrence of labels in II and III is remarkably lower for MIMETIC-ENHANCED (30% at most) than MIMETIC (whose values reach peaks of more than 60% for 7 out of 41 apps). These results highlight that

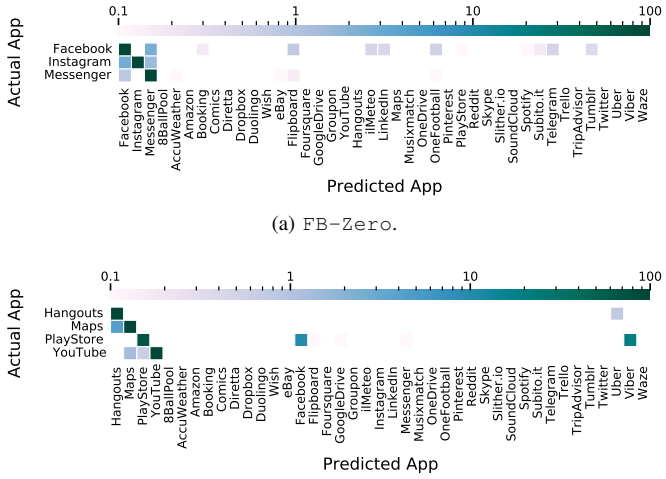
[16]This apparently counterintuitive outcome can be justified considering the more complex elementary DL layers constituting the App-Net architecture compared to FS-Net (and also w.r.t. MIMETIC and MIMETIC-ENHANCED), namely larger embedding vectors, BiLSTM instead of simpler BiGRU, and more complex trainable merge layer.



(a) MIMETIC-ENHANCED.

(b) MIMETIC.

Figure 6. Most recurring top-3 predictions of MIMETIC-ENHANCED (a) and MIMETIC baseline (b) traffic classifiers. '⋆' and '†' highlight apps (also) using gQUIC and FB-Zero protocol, respectively.



(a) FB-Zero.

(b) gQUIC.

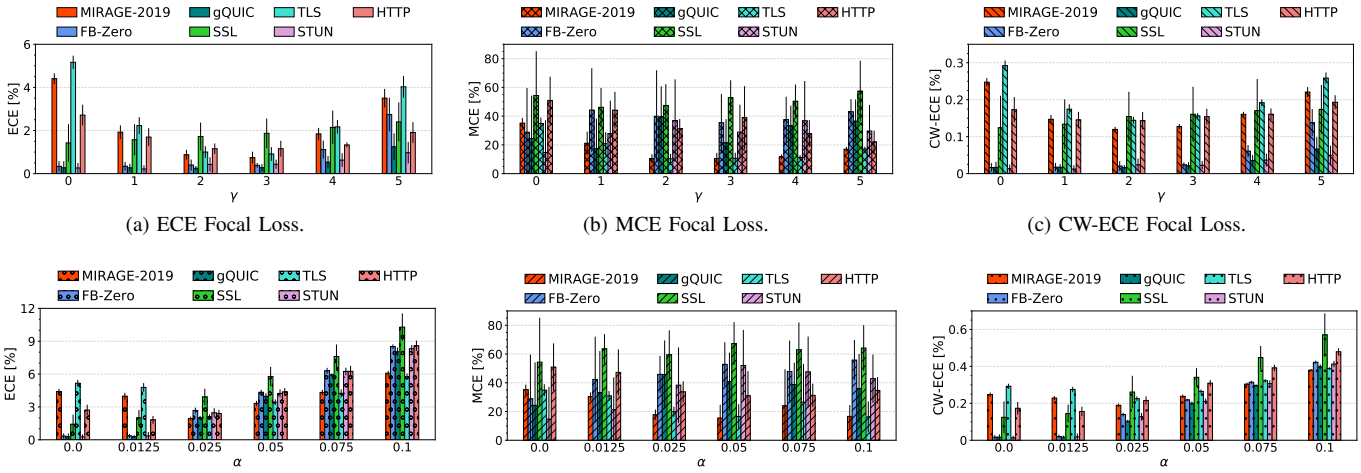
Figure 7. Confusion matrices of apps using FB-Zero (a) and gQUIC (b) protocols. Note that the log scale is used to evidence small errors.

prediction stability for MIMETIC-ENHANCED mostly relates to top-1 predictions, but does not extend to the second and third choices of the classifier. Also, inspecting the soft-output behavior of MIMETIC, its higher values for II and III most-frequent predictions are associated with evident app clusters of the same provider and sharing the same underlying protocols e.g., Instagram, Facebook, and Messenger (predicted among the top-3 for Instagram and Facebook) using FB-Zero, or Maps, PlayStore, YouTube, and Hangouts sharing gQUIC. These same patterns are also present but less evident for MIMETIC-ENHANCED.

Indeed, to further deepen our analysis of fine-grained behavior of MIMETIC-ENHANCED, we have investigated the error patterns of MIMETIC-ENHANCED in classifying the apps that (also) carry traffic through FB-Zero and gQUIC. To this aim, Figs. 7a and 7b depict the confusion matrices for the former group of apps (i.e. Facebook, Instagram, and Messenger) and for the latter (i.e. Hangouts, Maps, PlayStore, and YouTube), respectively. We recall that confusion matrices allow to clearly highlight misclassification patterns with a higher concentration where predicted classes equal the actual ones implying better TC performance. We can notice that most of the misclassifications happen within the same "protocol-group", that is wrongly assigning to a biflow a label of an app using the same protocol. This clusterization is particularly evident in Fig. 7a, where Facebook and Messenger are confused with each other, while Instagram is misclassified with both of them. Interestingly, Fig. 7a exhibits a slightly different behavior for the apps using gQUIC. In this case, the misclassifications are less frequent within the same protocol-group (with the notable exception of YouTube and Maps), with PlayStore more frequently misclassified as Facebook or Viber.

D. Calibration Analysis

We now focus on assessing the trustworthiness of MIMETIC-ENHANCED in terms of the calibration metrics defined in Sec. III-C, namely the (*i*) ECE, (*ii*) MCE and



(a) ECE Focal Loss. (b) MCE Focal Loss. (c) CW-ECE Focal Loss.

(d) ECE Label Smoothing. (e) MCE Label Smoothing. (f) CW-ECE Label Smoothing.

γ	*Accuracy [%]*	*F-measure [%]*	*G-mean [%]*
0	92.14 (± 0.28)	91.47 (± 0.27)	95.36 (± 0.19)
1	92.33 (± 0.25)	91.68 (± 0.34)	95.44 (± 0.21)
🏆 2	92.43 (± 0.31)	91.83 (± 0.35)	95.50 (± 0.19)
⚖ 3	92.21 (± 0.21)	91.61 (± 0.20)	95.39 (± 0.15)
4	92.14 (± 0.20)	91.55 (± 0.30)	95.36 (± 0.16)
5	91.90 (± 0.33)	91.32 (± 0.41)	95.24 (± 0.21)
⚖ − 🏆	− 0.21 (± 0.27)	− 0.21 (± 0.30)	− 0.10 (± 0.14)

(g) TC-performance Focal Loss.

α	*Accuracy [%]*	*F-measure [%]*	*G-mean [%]*
0	92.14 (± 0.28)	91.47 (± 0.27)	95.36 (± 0.19)
0.0125	92.22 (± 0.28)	91.52 (± 0.26)	95.37 (± 0.18)
⚖ 0.025	92.39 (± 0.27)	91.77 (± 0.35)	95.41 (± 0.22)
0.05	92.44 (± 0.36)	91.90 (± 0.44)	95.44 (± 0.28)
0.075	92.47 (± 0.24)	91.88 (± 0.30)	95.40 (± 0.19)
🏆 0.1	92.58 (± 0.26)	92.01 (± 0.31)	95.48 (± 0.21)
⚖ − 🏆	− 0.19 (± 0.35)	− 0.24 (± 0.42)	− 0.07 (± 0.26)

(h) TC-performance Label Smoothing.

Figure 8. Calibration sensitivity analysis of MIMETIC-ENHANCED in terms of ECE (a, d), MCE (b, e), and CW-ECE (c, f) when varying γ for focal loss (top row) and α for label smoothing (bottom row). Tables 8g and 8h show related TC performance using 🏆 to indicate the best Accuracy, F-measure, and G-mean [%] and ⚖ to indicate the best-calibrated configuration. The last row quantifies the TC-performance difference [%] for each metric between the best-calibrated and the best-performing configuration (⚖ − 🏆).

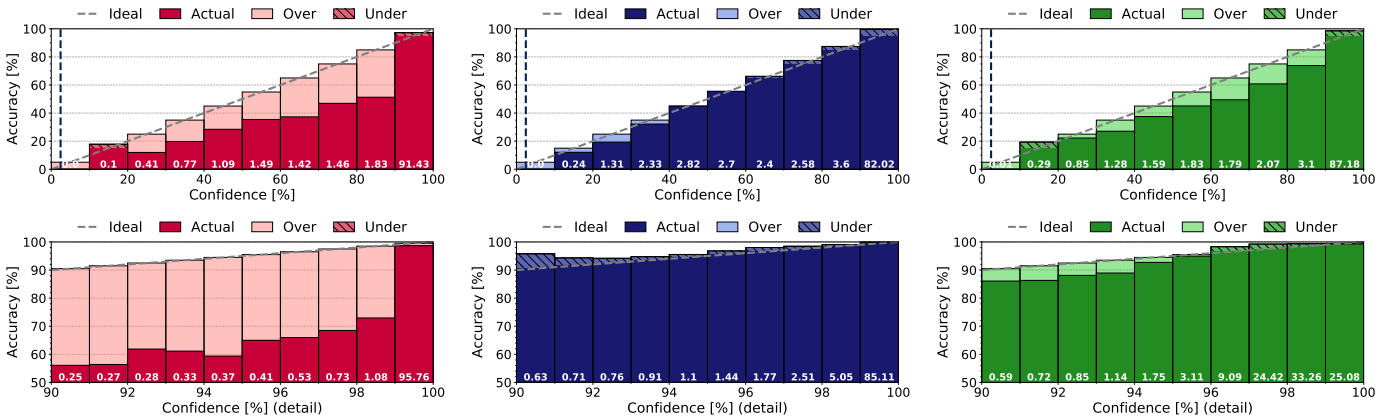
Table V
CALIBRATION PERFORMANCE OF MIMETIC-ENHANCED WITHOUT AND WITH FOCAL LOSS (FL) AND LABEL SMOOTHING (LS) OPTIMIZATIONS AND OF RELATED BASELINES IN TERMS OF ECE, MCE, AND CW-ECE.

Traffic Classifier	*ECE [%]*	*MCE [%]*	*CW-ECE [%]*
MIMETIC-ENHANCED	4.41 (± 0.25)	35.16 (± 3.55)	0.25 (± 0.01)
MIMETIC-ENHANCED w/ FL	**0.74 (± 0.28)**	**10.47 (± 3.96)**	**0.13 (± 0.01)**
MIMETIC-ENHANCED w/ LS	1.94 (± 0.18)	17.71 (± 3.64)	0.19 (± 0.01)
MIMETIC [7]	1.97 (± 0.33)	12.44 (± 3.44)	0.17 (± 0.01)
MIMETIC [7] w/ FL	3.42 (± 0.37)	18.24 (± 2.40)	0.25 (± 0.01)
MIMETIC [7] w/ LS	1.67 (± 0.21)	10.44 (± 2.03)	0.18 (± 0.01)
App-Net [8]	4.16 (± 0.21)	24.03 (± 2.54)	0.25 (± 0.01)
FS-Net [9]	6.29 (± 0.30)	30.07 (± 2.07)	0.35 (± 0.01)
1D-CNN [40]	12.78 (± 0.47)	52.55 (± 1.74)	0.67 (± 0.02)
MLP-1 ($\texttt{PAY}_{784}$)	9.54 (± 0.49)	27.64 (± 2.02)	0.54 (± 0.03)
HYBRID [41]	8.54 (± 0.56)	22.18 (± 1.98)	0.50 (± 0.03)
MLP-1 ($\texttt{PSQ}_{20}$)	2.90 (± 0.50)	8.80 (± 2.94)	0.43 (± 0.04)
DT ($\texttt{PAY}_{576} + \texttt{PSQ}_{12}$)	12.88 (± 0.45)	44.19 (± 25.38)	0.63 (± 0.02)
DT ($\texttt{PSQ}_{12}$)	14.72 (± 0.33)	57.17 (± 16.37)	0.72 (± 0.02)
DT ($\texttt{PAY}_{576}$)	15.89 (± 0.36)	48.83 (± 17.26)	0.78 (± 0.02)

Results are in the format *avg. (± std.)* obtained over the 10-folds.
Overall best-calibrated classifier is highlighted in boldface.

(*iii*) CW-ECE. Specifically, we first focus on investigating the calibration of MIMETIC-ENHANCED by varying relevant parameters in focal loss (γ) and label smoothing (α) techniques. The calibration sensitivity analysis is depicted in Fig. 8, by reporting both the results on (*a*) the whole whole MIRAGE-2019 dataset (in red) and (*b*) grouped by protocol. Since both techniques modify the training process, for each of the considered values also the corresponding achieved Accuracy, F-measure, and G-mean are reported in complementary Tabs. 8g and 8h.

By looking at the trend of the three calibration metrics w.r.t. γ and α over MIRAGE-2019, it is first apparent that ECE and CW-ECE share a unimodal trend. A slight different observation holds for the MCE (which accounts for the extreme-valued calibration discrepancies), whose trend appears to be less sensitive to α (and less obvious) when label smoothing is considered. The optimal values in terms of the three considered calibration metrics are obtained for $\gamma = 3$ and $\alpha = 0.025$ for focal loss and label smoothing, respectively, when considering the TC performance obtained on the whole MIRAGE-2019 dataset. Focusing on TC calibration *per protocol*, there is a similar behavior for `TLS` biflows, representing the highest quota in the dataset considered (and also the most used encryption protocol for mobile traffic). Conversely, when considering `SSL`, there seems to be a *detrimental* effect of both focal loss and label smoothing on ECE and CW-ECE, and a weak-dependence of the MCE on these tuning parameters. Interestingly, by matching calibration with TC performance, the parameter ensuring the optimal calibration for each training variant ($\gamma = 3$ and $\alpha = 0.025$) *does not coincide* with the highest DL-based TC effectiveness observed for MIMETIC-



ECE [%]	*MCE [%]*	*CW-ECE [%]*
4.41 (± 0.25)	35.16 (± 3.55)	0.25 (± 0.01)
Accuracy [%]	*F-measure [%]*	*G-mean [%]*
92.14 (± 0.28)	91.47 (± 0.27)	95.36 (± 0.19)

(a) MIMETIC-ENHANCED.

ECE [%]	*MCE [%]*	*CW-ECE [%]*
0.74 (± 0.28)	10.47 (± 3.96)	0.13 (± 0.01)
Accuracy [%]	*F-measure [%]*	*G-mean [%]*
92.21 (± 0.21)	91.61 (± 0.20)	95.39 (± 0.15)

(b) MIMETIC-ENHANCED w/ Focal Loss.

ECE [%]	*MCE [%]*	*CW-ECE [%]*
1.94 (± 0.18)	17.71 (± 3.64)	0.19 (± 0.01)
Accuracy [%]	*F-measure [%]*	*G-mean [%]*
92.39 (± 0.27)	91.77 (± 0.35)	95.41 (± 0.22)

(c) MIMETIC-ENHANCED w/ Label Smoothing.

Figure 9. Reliability diagrams of MIMETIC-ENHANCED (a) and MIMETIC-ENHANCED with label smoothing (b) and focal loss (c) in their best configurations (i.e. with $\gamma = 3$ and $\alpha = 0.025$, respectively). Confidence is divided in 10 bins, and it is $\geq 1/L$ (vertical dashed line), with L being the number of classes. Over and under gap represent an over-confident (optimistic) and under-confident (pessimistic) miscalibration pattern, respectively. The number at the bottom of each bar reports the percentage of samples within the corresponding bin. The bottom row shows a $10\times$ finer grained representation of the last bin (i.e. $[90, 100]$). Synthetic *ECE*, *MCE*, and *CW-ECE* along with TC-performance measures are also reported for each case.

ENHANCED ($\gamma = 2$ and $\alpha = 0.1$). Still, as shown in the last row of Tabs. 8g and 8h, the relative loss *is always* $< 0.25\%$ (in absolute value) for all the three considered TC-effectiveness metrics.

We then delve into the calibration details of these two variants of MIMETIC-ENHANCED, using focal loss and label smoothing, and optimized w.r.t. tuning parameters to achieve the optimal calibration. To appreciate the calibration effect of these, they are compared with a naive (non-calibrated) version of MIMETIC-ENHANCED. The comparison is performed in terms of the relevant calibration diagrams associated to the ECE (i.e. taking care of the calibration performance on the probability associated to the predicted class), reported in Fig. 9. Given the high performance achieved by MIMETIC-ENHANCED, most of the classified biflows are predicted with confidence values $\in [90, 100]\%$: hence, the bottom row shows a $10\times$ zoom of the diagrams for the aforementioned bin. For readers' convenience, the corresponding TC-performance metrics and calibration metrics for the three counterparts are reported in the bottom-wise tables.

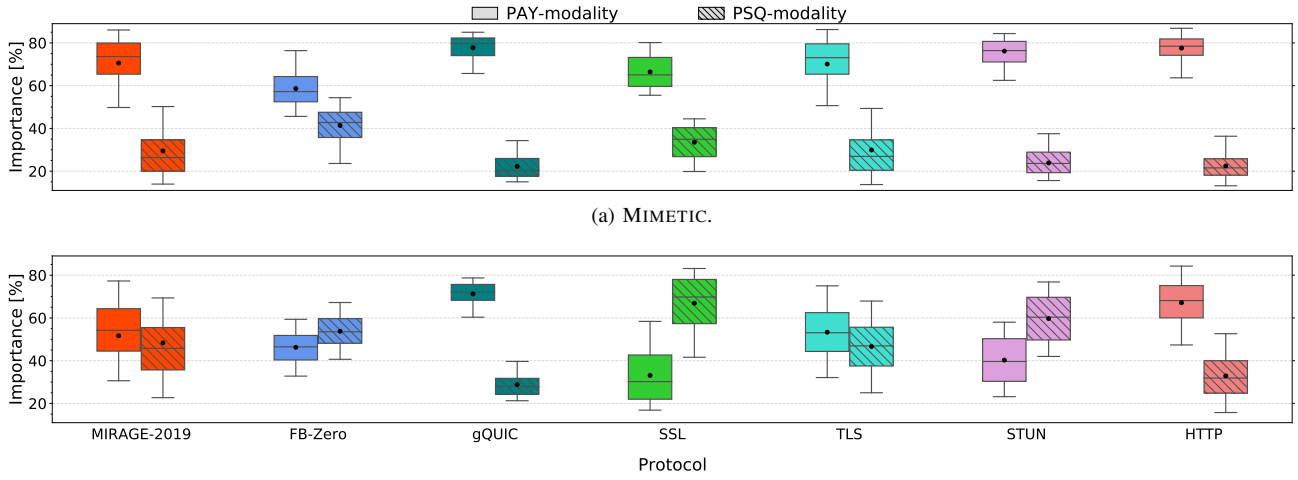
First of all, it is apparent the non-perfect calibration of MIMETIC-ENHANCED, when both considering the whole confidence range and also focusing on the bin $\in [90, 100]\%$. Conversely, the calibration effect of the (optimized) focal loss and label smoothing is evident *for all the values of the predicted confidence*, as also witnessed by the concise calibration metrics. For instance, the adoption of focal loss and label smoothing is able to achieve a $6\times$ and $2.2\times$ ECE reduction with respect to the uncalibrated case, respectively (similar loss cut is observed for MCE and CW-ECE).

Differently, focusing on the relative calibration performance achieved by these two methods, the focal loss achieves the best calibration performance in terms of all the three considered metrics. This can be also appreciated by direct comparison of the calibration diagrams. Indeed, looking at the whole prediction range, the calibration diagram of focal loss is closer to the ideal $accuracy = confidence$ line. Also, focusing on the $\in [90, 100]\%$ bin, a similar reasoning applies for MIMETIC-ENHANCED with focal loss, but with the latter having a slighter under-confident behavior (w.r.t. label smoothing) in the range $\in [90, 95]\%$.

Finally, Tab. V summarizes the results of the calibration analysis extended to all the baselines considered. More specifically, for both MIMETIC-ENHANCED and the best-performing baseline (i.e. MIMETIC), we report the results attained with the best-calibrated configurations with focal loss ($\gamma = 3$ and $\gamma = 1$ for MIMETIC-ENHANCED and MIMETIC, respectively) and label smoothing ($\alpha = 0.025$ for both architectures), and also without exploiting any calibration technique.

Recalling the classification performance in Tab. IV the reported results highlight the existing trade-off between the accuracy of the provided predictions and the related confidence: in its native formulation, MIMETIC-ENHANCED exposes $\approx 2\times$ calibration error with respect to MIMETIC, witnessing that the achieved TC performance comes at the cost of reduced trustworthiness in the provided outcomes and thus calling for methods to improve it. On the other hand, all the baselines report calibration errors higher than MIMETIC-ENHANCED with either focal loss or label smoothing (e.g., up to more than $17\times$ and $6\times$, respectively, in terms of ECE). It is worth noticing that among considered baselines the least calibrated approach is the (naturally interpretable) DT that tends to be significantly *over-confident* in its predictions.

Such an outcome thus remarks the utility of focal loss and label smoothing to improve calibration performance, with their usefulness being more limited when considering an approach



(a) MIMETIC.

(b) MIMETIC-ENHANCED.

Figure 10. Modality contributions of MIMETIC (a) and MIMETIC-ENHANCED (b). Importance $\widetilde{\phi}_{\mathcal{M}_p}$ for both PAY-modality and PSQ-modality is reported for both the whole dataset (with red color) and grouped by protocol (with different colors). Whiskers show 5^{th} and 95^{th} percentiles.

which is already well-calibrated in its native variant. For instance, when considering MIMETIC, only label smoothing is able to attain a calibration improvement (up to -2% in terms of MCE), whereas MIMETIC with focal loss is even less calibrated than its native formulation. Interestingly, experimental observations witnessed also that the use of either focal loss or label smoothing, regardless of whether they provide a remarkable calibration improvement (as for MIMETIC-ENHANCED) or not (as for MIMETIC), leads to an extremely-limited TC-accuracy decrease, namely $< 0.25\%$ and $< 0.33\%$ loss for MIMETIC-ENHANCED (cf. Tabs. 8g and 8h) and MIMETIC[17], respectively.

E. Modality Contribution to Correct TC Decisions

In Fig. 10, we investigate the (relative) contribution that each traffic modality (PAY and PSQ) gives to TC, focusing on MIMETIC (Fig. 10a) and MIMETIC-ENHANCED (Fig. 10b). The analysis is obtained by evaluating the pooled SHAP values $\widetilde{\phi}_{\mathcal{M}}$ for each modality. The corresponding distributions (shown via the boxplots) are then obtained by selecting the correctly-classified tested samples of either (a) the whole MIRAGE-2019 (in red) or (b) the single protocol (other colors).

Focusing on MIMETIC, it is apparent that PAY modality *always* contributes with higher importance values: the median is $\approx 80\%$ for all the considered protocols except for FB-Zero, where it is $\approx 60\%$. For this protocol, the PSQ modality has higher values (median is $\approx 40\%$).

A different situation is observed for MIMETIC-ENHANCED (cf. Fig. 10b): the importance of PSQ-modality increases (thanks to the embedding layer used for PL values) and the resulting distributions are wider than the previous ones. For FB-Zero, SSL and STUN biflows, the aforementioned modality overcomes the other (the median is $\approx 70\%$ for SSL). For gQUIC and HTTP in both architectures, the values associated with PSQ modality are always small (the median is $\approx 20\%$ and $\approx 30\%$, respectively). Conversely, for TLS the two modalities contribute almost equally to the correct TC decision.

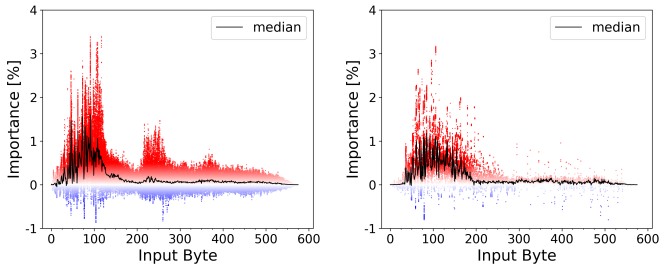
F. SHAP PAY-modality

In this section, we analyze the relative importance of inputs (transport-layer payload bytes) associated to the PAY-modality, based on Deep SHAP (cf. Sec. III-B). Specifically, we focus on providing global explanations for MIMETIC-ENHANCED, which relies on the first $N_b = 576$ bytes of each biflow. For our proposal, we provide *global explanations* pertaining to both (a) per-protocol and (b) per-app aggregation.[18] The corresponding results are reported in Figs. 11 and 12, respectively. In all the plots, sample-wise positive and negative SHAP values are highlighted with red and blue colors, respectively. Also, for completeness, the median importance value of each byte (over different samples) is reported as a solid black line. This allows highlighting regions which are more consistently influential for predictions and those having low/high variability. We remark that a similar analysis (not shown) has been carried out also for MIMETIC, since it uses the same input type for PAY-modality. Still, experimental results have highlighted less appreciable explanation patterns, due to the absence of the "focusing effect" of the embedding layer not present in the MIMETIC architecture.

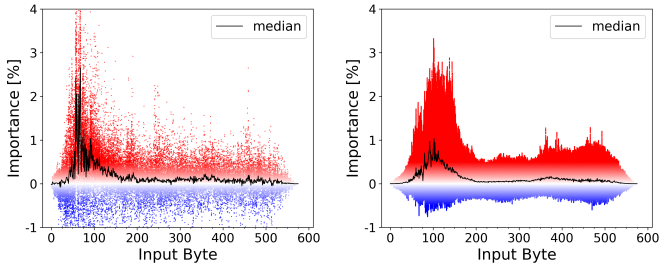
Concerning FB-Zero (Fig. 11a), we observe *two* regions attaining high per-sample positive values: one covers the first 100 bytes, whereas the other corresponds with bytes in the interval $[200, 300]$. Looking at the median of the distributions obtained for each byte, we see that such value is however low for bytes associated to the second region. This is attributed to a limited number of FB-Zero biflows having this profile.

[17]Specifically, MIMETIC experiences a performance drop on Accuracy, F-measure, and G-mean of -0.23 ($\pm$ 0.31), -0.10 ($\pm$ 0.39), and -0.11 ($\pm$ 0.33) with focal loss ($\gamma = 1$), and of -0.30 ($\pm$ 0.24), -0.32 ($\pm$ 0.47), and -0.08 ($\pm$ 0.42) with label smoothing ($\alpha = 0.025$), respectively.

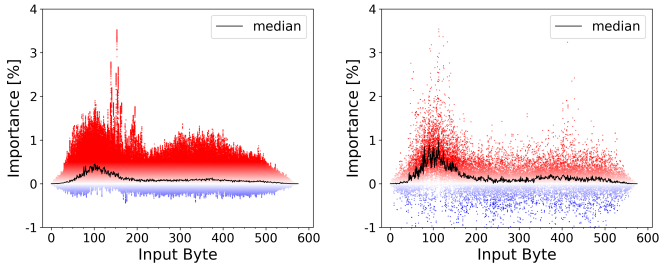
[18]We remark that also per-protocol aggregation refers to the 41-app mobile TC task. In other terms, the two analyses differ only in the subset of the testing samples considered.



(a) FB-Zero. (b) gQUIC

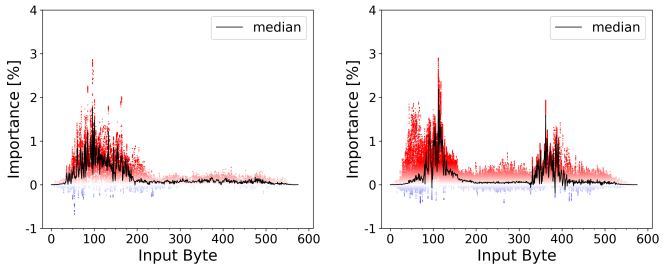


(c) STUN. (d) TLS.



(e) HTTP. (f) SSL.

Figure 11. Importance $\widetilde{\phi}_m$ of the payload-byte inputs (identified by their position) for PAY-modality of MIMETIC-ENHANCED grouped by protocol.



(a) YouTube. (b) Diretta.

Figure 12. Importance $\widetilde{\phi}_m$ of the payload-byte inputs (identified by their position) for PAY-modality of MIMETIC-ENHANCED for exemplifying apps.

Specifically, the aforementioned biflows contain SCFG packets, and the second region is aligned to the relevant fields for this packet, such as server certificates.

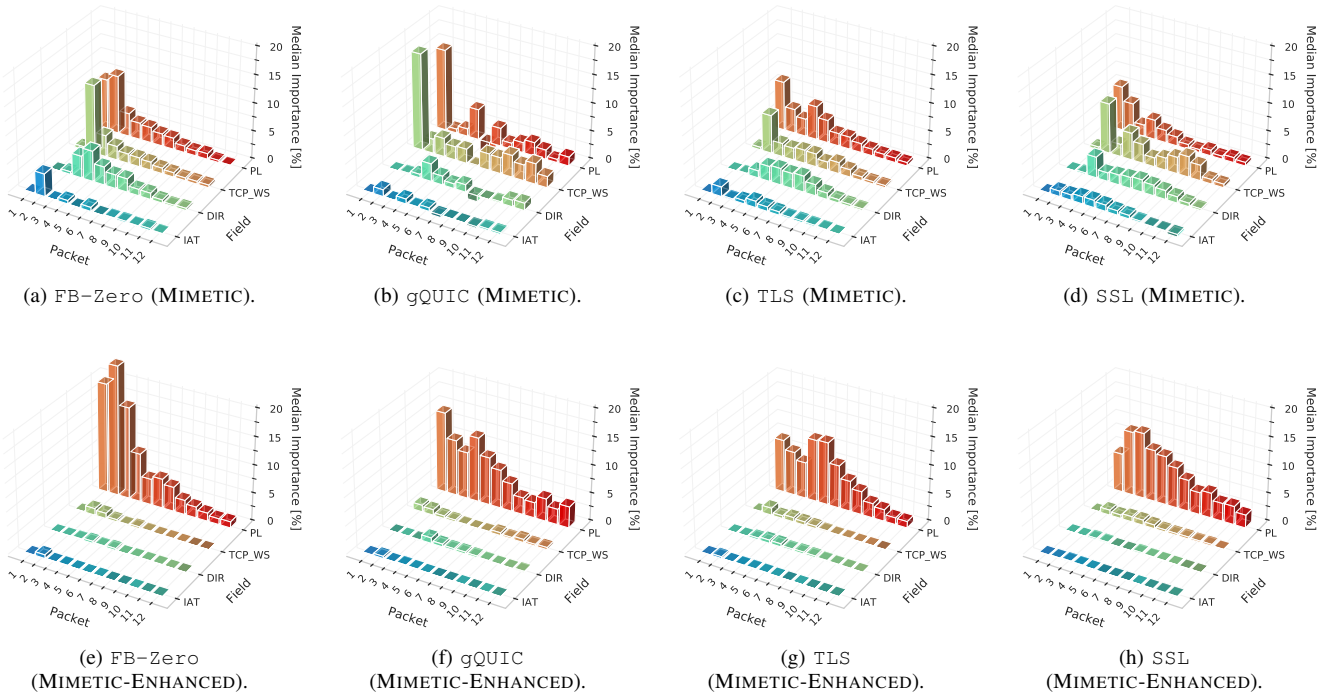
By observing the importance plot for gQUIC (Fig. 11b), it is interesting to observe an initial region ($\approx$ 200 bytes) whose values are significantly higher than the following ones. The above importance values can be attributed to the fact that gQUIC biflows start with a CHLO packet, having a fixed payload length of 1350 bytes. This amount is filled with padding bytes starting approximately from position 200. This is the reason why bytes after this position have minimal importance values. The first 200 bytes, instead, are a list of tags employed in the packet, whose values are listed in the final part of the packet, not observed due to the truncation of the payload input type to $N_b = 576$ bytes. Analogous observations can be drawn for the STUN protocol (Fig. 11c), since the resulting profile is very defined and highlights that the first bytes are given very-high values. Conversely, for the profiles of the other three protocols (i.e. TLS, SSL, and HTTP in Figs. 11d, 11f, and 11e, respectively), consistently-influential regions in the byte sequences are less evident: higher values are associated with the first bytes, but with the median behavior (viz. black line) reporting values $< 1\%$. In particular, concerning TLS, when comparing the distribution of the position of the SNI field (starting at byte 107 and spanning over 20 bytes, on average), we can observe that the bytes exposing higher importance values dramatically match those encoding this field.

Finally, in Fig. 12, we analyze the global explanations of PAY-modality associated to single apps, focusing on some remarkable examples that we discuss in the following. Overall, the distributions of $\widetilde{\phi}_m$ report higher-importance values (positive red points) in correspondence of the initial bytes of the payload ($<$ 200). For instance, the profile obtained for Youtube (Fig. 12a) has similar characteristics to those described for gQUIC biflows (Fig. 11b). On the other hand, in other cases, in spite of the presence of peak values for the first bytes, the median (black line) does not follow the same trend, according to the mixture of protocols for a given app. This behavior is exemplified, for instance in Fig. 12b (corresponding to the Diretta app), while the distribution reports high values for the very first bytes (imputable to HTTP biflows, cf. Fig. 11e), the median does not follows the same trend (because of the impact of TLS). Indeed, beside the initial peak (centered around byte 100) the median highlights an influential region around bytes 300–400. This witnesses the importance of considering a value for $N_b > 400B$, i.e. not limited to the very first bytes of the payload.

G. Shap PSQ-modality

Similarly to the previous section, here we investigate the importance of inputs associated with the PSQ-modality of MIMETIC-ENHANCED. This branch of the architecture is fed with 4 header fields extracted from the first 12 packets of each biflow, namely inter-arrival time (IAT), direction (DIR), TCP window size (TCP_WS), and transport-layer payload length (PL). Results are compared with analogous ones obtained by MIMETIC.

Considering the importance associated to these fields for each biflow, Fig. 13 shows the median importance values for each element of the 4×12 matrix used as input for this modality. In detail, the figure reports the per-protocol breakdown[V-F] considering the most recurrent ones in the dataset (i.e. FB-Zero, gQUIC, TLS, and SSL), for both MIMETIC (a, b, c, d) and MIMETIC-ENHANCED (e, f, g, h).



(a) FB-Zero (MIMETIC). (b) gQUIC (MIMETIC). (c) TLS (MIMETIC). (d) SSL (MIMETIC).

(e) FB-Zero (MIMETIC-ENHANCED). (f) gQUIC (MIMETIC-ENHANCED). (g) TLS (MIMETIC-ENHANCED). (h) SSL (MIMETIC-ENHANCED).

Figure 13. Importance $\widetilde{\phi}_m$ of the header-field inputs (identified by the respective packet) for PSQ-modality of MIMETIC (top row) and MIMETIC-ENHANCED (bottom row) grouped by protocol.

Focusing on the former, it is evident how the header fields of the first packets of the biflows (i.e. from the 1^{st} to the 5^{th}) play a crucial role in identifying the apps. In detail, for gQUIC (Fig. 13b), the very first packet is essential to correctly classify the biflows using this protocol: PL always assumes the same value (1350 bytes) and the TCP_WS is set to 0 because the transport-layer protocol is UDP. Concerning FB-Zero (Fig. 13a), the second packet of the biflows is the most important one (high importance for PL, TCP_WS, and IAT), and interestingly the DIR of the subsequent packets (particularly from the 3^{rd} to the 8^{th}) helps in predictions. The second packet exposes higher importance, on average, also for TLS (Fig. 13c) and SSL (Fig. 13d). More in general, while the elements with higher importance values change with both the protocol as well as the header field, PL and TCP_WS show median importance values higher than other fields, overall.

Moving to the results of MIMETIC-ENHANCED, Figs. 13e–h highlight that the distribution of field importance changes dramatically, with a single feature (i.e. the PL) becoming remarkably more important than the others. This result can be justified by the introduction of the embedding layer for PL in MIMETIC-ENHANCED (see Sec.III-A). In addition, the packets after the first ones also play a more important role exposing higher relative importance values. On the other hand, the major importance of PL related to the specific initial packets of the biflows still holds in some of the cases: the 1^{st} packet and the 2^{nd} packet for both FB-Zero (Fig. 13e) and gQUIC (Fig. 13f) still expose the highest importance.

Finally, we would underline that the complementary analysis performed on a per-app basis (whose results are not shown for brevity), highlights analogous patterns, remarking the major importance of PL for MIMETIC-ENHANCED w.r.t. MIMETIC and—more in general—of the header fields of the first packets of the biflows.

VI. CONCLUSIONS AND FUTURE DIRECTIONS

In this work, we focused on interpreting the behavior of mobile TC performed with DL approaches, and providing design insights toward performance improvement. With the tools of explainable artificial intelligence we analyzed an evolved *multimodal-DL approach*, named MIMETIC-ENHANCED. Evaluated on a publicly available recent dataset of mobile-app traffic (consisting of 41 apps, mostly using 6 different protocols), the proposed architecture was shown to outperform in all the cases a number of relevant DL-based TC single-modal baselines proposed in literature [9, 40, 41], as well as most recent multimodal proposals, e.g. [7, 8]. Also, fine-grained performance analysis revealed that MIMETIC-ENHANCED provides a *more consistent* soft-output outcome (in terms of top-K accuracy), and incurs misclassifications *mostly confined* within the same protocol-group. Our proposal was then investigated in terms of *trustworthiness* (i.e. how reliably we can trust its confidence prediction, via calibration) and *interpretability* (i.e. the underlying rationale which makes them work effectively, via SHAP-based techniques). In the former case, the *calibration* by focal loss achieves a $6\times$ reduction with respect to the uncalibrated case, obtaining a significant gain in *trustworthiness*. Regarding *interpretability*, a *global explanation* has been obtained for each modality (payload-based and packet-sequence-based), quantifying the

importance of each, and highlighting how the payload still retains high importance, despite the vast majority of encrypted traffic.

Future avenues of research will include *(a)* taking advantage of the complementarity of different XAI approaches and extending the proposed interpretability analysis by implementing (for the payload-modality) an occlusion analysis based on the results of the importance analysis, as well as by performing the importance analysis based on the position of specific header fields rather than on byte position; *(b)* the investigation of trustworthiness and interpretability for the analysis and improved design of more challenging multi-task TC problems; *(c)* the robustness assessment of multimodal DL-based traffic classifiers to adversarial attacks; *(d)* the in-depth investigation of focal loss and label smoothing impact on a larger pool of DL architectures; *(e)* the design of (natively) self-explainable DL traffic classifiers; *(f)* the design of self-explainable local decisions; and *(g)* the design of lightweight architectures originated from XAI techniques.

ACKNOWLEDGMENTS

This work is partially funded by the Italian Research Program "PON AIM *Attraction and International Mobility*, Azione I.2 Linea 1, *Mobilità dei Ricercatori*" (Codice proposta attività AIM1878982-2 CUP E56C19000330005).

REFERENCES

[1] V. Dignum, "Responsible artificial intelligence: designing ai for human values," *ITU Journal: ICT Discoveries, Special Issue*, vol. 1, pp. 1–8, 2017.

[2] L. Frost, T. B. Meriem, J. M. Bonifacio, S. Cadzow, F. da Silva, M. Essa, R. Forbes, P. Marchese, M.-P. Odini, N. Sprecher, C. Toche, and S. Wood, "Artificial Intelligence and future directions for ETSI," European Telecommunications Standards Institute (ETSI), White Paper 34, Jun 2020. [Online]. Available: https://www.etsi.org/images/files/ETSIWhitePapers/etsi_wp34_Artificial_Intellignce_and_future_directions_for_ETSI.pdf

[3] M. Taddeo, T. McCutcheon, and L. Floridi, "Trusting artificial intelligence in cybersecurity is a double-edged sword," *Nature Machine Intelligence*, vol. 1, no. 12, pp. 557–560, 2019.

[4] R. Inam, A. Terra, A. Mujumdar, E. Fersman, and A. V. Feljan, "Explainable AI—how humans can trust AI," Ericsson, White Paper, Apr 2021. [Online]. Available: https://www.ericsson.com/en/reports-and-papers/white-papers/explainable-ai--how-humans-can-trust-ai

[5] A. Razaghpanah, A. A. Niaki, N. Vallina-Rodriguez, S. Sundaresan, J. Amann, and P. Gill, "Studying TLS usage in Android apps," in *13th ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, 2017, pp. 350–362.

[6] D. Madariaga, L. Torrealba, J. Madariaga, J. Bermúdez, and J. Bustos-Jiménez, "Analyzing the adoption of QUIC from a mobile development perspective," in *ACM Workshop on the Evolution, Performance, and Interoperability of QUIC (EPIQ)*, 2020, p. 35–41.

[7] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapè, "MIMETIC: mobile encrypted traffic classification using multimodal deep learning," *Elsevier Computer Networks*, vol. 165, p. 106944, 2019.

[8] X. Wang, S. Chen, and J. Su, "Automatic mobile app identification from encrypted traffic with hybrid neural networks," *IEEE Access*, vol. 8, pp. 182 065–182 077, 2020.

[9] C. Liu, L. He, G. Xiong, Z. Cao, and Z. Li, "FS-Net: A flow sequence network for encrypted traffic classification," in *IEEE Conference on Computer Communications (INFOCOM)*, 2019, pp. 1171–1179.

[10] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *30th Conference on Neural Information Processing Systems (NeurIPS)*, 2017, pp. 4765–4774.

[11] G. Aceto, D. Ciuonzo, A. Montieri, V. Persico, and A. Pescapé, "MIRAGE: Mobile-app traffic capture and ground-truth creation," in *4th International Conference on Computing, Communications and Security (ICCCS)*, Oct 2019, pp. 1–8.

[12] K. Amarasinghe, K. Kenney, and M. Manic, "Toward explainable deep neural network based anomaly detection," in *11th International Conference on Human System Interaction (HSI)*, 2018, pp. 311–317.

[13] Y. Zheng, Z. Liu, X. You, Y. Xu, and J. Jiang, "Demystifying deep learning in networking," in *2nd ACM Asia-Pacific Workshop on Networking (APNet)*, 2018, p. 1–7.

[14] A. Dethise, M. Canini, and S. Kandula, "Cracking open the black box: What observations can tell us about reinforcement learning agents," in *ACM Workshop on Network Meets AI & ML (NetAI)*, 2019, p. 29–36.

[15] A. Morichetta, P. Casas, and M. Mellia, "EXPLAIN-IT: Towards explainable AI for unsupervised network traffic analysis," in *3rd ACM CoNEXT Workshop on Big DAta, Machine Learning and Artificial Intelligence for Data Communication Networks (Big-DAMA)*, 2019, p. 22–28.

[16] S. Rezaei, B. Kroencke, and X. Liu, "Large-scale mobile app identification using deep learning," *IEEE Access*, vol. 8, pp. 348–362, 2020.

[17] 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.

[18] C. Beliard, A. Finamore, and D. Rossi, "Opening the deep pandora box: Explainable traffic classification," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2020, pp. 1292–1293.

[19] Z. Meng, M. Wang, J. Bai, M. Xu, H. Mao, and H. Hu, "Interpreting deep learning-based networking systems," in *ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*, 2020, p. 154–171.

[20] X. Wang, S. Chen, and J. Su, "Real network traffic collection and deep learning for mobile app identification," *Hindawi Wireless Communications and Mobile Computing*, 2020.

[21] G. Aceto, G. Bovenzi, D. Ciuonzo, A. Montieri, V. Persico, and A. Pescapé, "Characterization and prediction of mobile-app traffic using Markov modeling," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 907–925, 2021.

[22] H. Hagras, "Toward human-understandable, explainable AI," *IEEE Computer*, vol. 51, no. 9, pp. 28–36, 2018.

[23] M. Kull, M. Perello-Nieto, M. Kängsepp, H. Song, P. Flach *et al.*, "Beyond temperature scaling: Obtaining well-calibrated multiclass probabilities with dirichlet calibration," in *32th Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[24] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *22nd International Conference on Machine learning (ICML)*, 2005, pp. 625–632.

[25] D. Widmann, F. Lindsten, and D. Zachariah, "Calibration tests in multiclass classification: A unifying framework," in *33th Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[26] J. Mukhoti, V. Kulharia, A. Sanyal, S. Golodetz, P. H. Torr, and P. K. Dokania, "Calibrating deep neural networks using focal loss," in *34th Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[27] R. Müller, S. Kornblith, and G. Hinton, "When does label smoothing help?" in *32th Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[28] W. Samek, G. Montavon, S. Lapuschkin, C. J. Anders, and K. R. Müller, "Explaining deep neural networks and beyond: A review of methods and applications," *Proceedings of the IEEE*, vol. 109, no. 3, pp. 247–278, 2021.

[29] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Springer European Conference on Computer Vision (ECCV)*, 2014, pp. 818–833.

[30] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you?: Explaining the predictions of any classifier," in *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2016, p. 1135–1144.

[31] R. Guidotti, A. Monreale, S. Ruggieri, D. Pedreschi, F. Turini, and F. Giannotti, "Local rule-based explanations of black box decision systems," *arXiv preprint arXiv:1805.10820*, 2018.

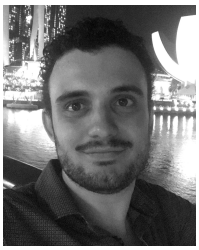
[32] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *34th PMLR International Conference on Machine Learning (ICML)*, 2017, pp. 3319–3328.

[33] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *34th PMLR International Conference on Machine Learning (ICML)*, 2017, pp. 3145–3153.

[34] K. Ismailaj, M. Camelo, and S. Latré, "When deep learning may not be the right tool for traffic classification," in *6th IEEE/IFIP International Workshop on Analytics for Network and Service Management (AnNet)*,

2021, pp. 1–6.
[35] T. Bakhshi and B. Ghita, "On internet traffic classification: A two-phased machine learning approach," *Journal of Computer Networks and Communications*, 2016.
[36] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapè, "Traffic classification of mobile apps through multi-classification," in *IEEE Global Communications Conference (GLOBECOM)*, 2017, pp. 1–6.
[37] M. Korczyński and A. Duda, "Markov chain fingerprinting to classify encrypted traffic," in *IEEE Conference on Computer Communications (INFOCOM)*, 2014, pp. 781–789.
[38] A. M. Sadeghzadeh, S. Shiravi, and R. Jalili, "Adversarial network traffic: Towards evaluating the robustness of deep-learning-based network traffic classification," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1962–1976, 2021.
[39] D. Ramachandram and G. W. Taylor, "Deep multimodal learning: A survey on recent advances and trends," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 96–108, 2017.
[40] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," in *IEEE International Conference on Intelligence and Security Informatics (ISI)*, 2017, pp. 43–48.
[41] 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.
[42] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Mobile encrypted traffic classification using Deep Learning: Experimental evaluation, lessons learned, and challenges," *IEEE Trans. Netw. Service Manag.*, vol. 16, no. 2, pp. 445–458, 2019.
[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference for Learning Representations (ICLR)*, 2015.
[44] D. Duvenaud, D. Maclaurin, and R. Adams, "Early stopping as nonparametric variational inference," in *Artificial Intelligence and Statistics*. PMLR, 2016, pp. 1070–1077.
[45] M. Mahsereci, L. Balles, C. Lassner, and P. Hennig, "Early stopping without a validation set," *arXiv preprint arXiv:1703.09580*, 2017.
[46] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.
[47] L. S. Shapley, "A value for n-person games," *Contributions to the Theory of Games*, vol. 2, no. 28, pp. 307–317, 1953.
[48] G. Fidel, R. Bitton, and A. Shabtai, "When Explainability Meets Adversarial Learning: Detecting Adversarial Examples using SHAP Signatures," in *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2020, pp. 1–8.
[49] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *34th PMLR International Conference on Machine Learning (ICML)*, 2017, pp. 1321–1330.
[50] M. H. DeGroot and S. E. Fienberg, "The comparison and evaluation of forecasters," *Journal of the Royal Statistical Society: Series D (The Statistician), 32(1-2), 12-22*, 1983.

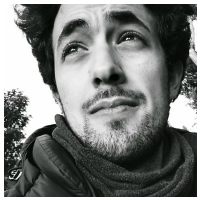
Alfredo Nascita is a Postgraduate Early-Stage Researcher at DIETI, University of Napoli Federico II. He received his M.S. Laurea Degree in Computer Engineering from the same University in March 2021. His research interests include traffic classification, machine and deep learning, and explainable artificial intelligence.



Antonio Montieri (GSM'18) is a Postdoctoral Researcher at DIETI of the University of Napoli Federico II. He has received his Ph.D. degree in Information Technology and Electrical Engineering in April 2020 from the same University. His work concerns network measurements, (encrypted and mobile) traffic classification, traffic modeling and prediction, and monitoring of cloud network performance. Antonio has co-authored 29 papers in international journals and conference proceedings.

Giuseppe Aceto is an Assistant Professor at University of Napoli Federico II, where he received his PhD in Telecommunication Engineering. His research concerns network performance and censorship, both in traditional networks and SDN, and ICTs applied to health. He received the best paper award at IEEE ISCC 2010, and 2018 Best Journal Paper Award by IEEE CSIM.

Domenico Ciuonzo (S'11-M'14-SM'16) is an Assistant Professor at University of Napoli Federico II. He holds a PhD from University of Campania L. Vanvitelli (IT) and, from 2011, he has held several visiting researcher appointments. Since 2014 he is editor of several IEEE, IET and ELSEVIER journals. He is the recipient of two Best Paper awards (IEEE ICCCS 2019 and ELSEVIER COMPUTER NETWORKS 2020), the 2019 Exceptional Service award from IEEE Aerospace and Electronic Systems Society, and the 2020 Early-Career Technical Achievement award from IEEE Sensors Council for sensor networks/systems. His research interests include data fusion, Internet of Things, network analytics and machine learning.



Valerio Persico is an Assistant Professor at DIETI, University of Napoli Federico II, where he received the PhD in Computer and Automation Engineering in 2016. His work concerns network measurements, cloud-network monitoring, and Internet path tracing and topology discovery. He has co-authored more than 30 papers within international journals and conference proceedings.

Antonio Pescapé (SM'09) is a Full Professor of computer engineering at the University of Napoli Federico II. His work focuses on measurement, monitoring, and analysis of the Internet. He has co-authored more than 200 conference and journal papers, he is the recipient of a number of research awards. Also, he has served as an independent reviewer/evaluator of research projects/project proposals co-funded by a number of governments and agencies.