Fine-Grained Traffic Prediction of Communication-and-Collaboration Apps via Deep-Learning: a First Look at Explainability

Idio Guarino, Giuseppe Aceto, Domenico Ciuonzo, Antonio Montieri, Valerio Persico, Antonio Pescapé
University of Napoli "Federico II" (Italy)
{idio.guarino, giuseppe.aceto, domenico.ciuonzo, antonio.montieri, valerio.persico, pescape}@unina.it

***Abstract*—The lifestyle change originated from the COVID-19 pandemic has caused a measurable impact on Internet traffic in terms of volume and application mix, with a sudden increase in usage of communication-and-collaboration apps. In this work, we focus on four of these apps (`Skype`, `Teams`, `Webex`, and `Zoom`), whose traffic we collect, reliably label at fine (i.e. per-activity) granularity, and analyze from the viewpoint of traffic prediction. The outcome of this analysis is informative for a number of network management tasks, including monitoring, planning, resource provisioning, and (security) policy enforcement. To this aim, we employ state-of-the-art multitask deep learning approaches to assess to which degree the traffic generated by these apps and their different use cases (i.e. activities: audio-call, video-call, and chat) can be forecast at packet level. The experimental analysis investigates the performance of the considered deep learning architectures, in terms of both traffic-prediction accuracy and complexity, and the related trade-off. Equally important, our work is a first attempt at interpreting the results obtained by these predictors via eXplainable Artificial Intelligence (XAI).**

***Index Terms*—communication apps; collaboration apps; COVID-19; deep learning; encrypted traffic; multitask approaches; traffic prediction; XAI.**

I. Introduction

During the COVID-19 pandemic, Internet traffic, especially from residential users, has witnessed a significant growth (+15–20% in terms of volume) [1], as in lockdowns and social distancing periods people engaged in remote work and education, and online commerce and entertainment activities as well. This shift has had a measurable impact on network performance [2]. Such sudden variations do not play well with planning and management, and pushed for automated and adaptive management of network resources [3]. A responsive system should be able to react instantaneously to observed network traffic, but an even better system would be *proactive*, anticipating the incoming future traffic, to have the time to effectively process the best reaction, and enact it. This need in the last decade motivated a number of solutions for predicting network traffic, usually in the form of *aggregates* (e.g., total volume, average rate) over a wide span of time intervals.

This work is partially supported by the grant "O. Carlini" funded by GARR, the Italian Research Program "PON *Ricerca e Innovazione* 2014–2020 (PON R&I) – Asse IV *REACT-EU* – Azione IV.4", the "Centro Nazionale HPC, Big Data e Quantum Computing – ICSC" and the "RESTART" Project funded by MUR, and the "ADDITIONAL" Project funded by Vietsch Foundation.

By leveraging the latest advancements in Machine Learning (ML)—i.e. Deep Neural Networks (DNNs)—we consider an especially hard prediction problem at its finest granularity: *packet-level network-traffic prediction*. With a limited memory of previously observed packets, we propose a model able to predict the direction (upstream/downstream) of the next packet, its inter-arrival time, and its (transport-layer) payload length. This allows for fine-grained and prompt traffic management (i.e. at biflow level and short-time scale) toward the design of next-generation networks. We evaluate our proposal on traffic generated by *communication-and-collaboration mobile applications* (*CC apps*, in short), a type of network traffic that is of high practical interest, having contributed to the reshaping of Internet traffic worldwide during and following the COVID-19 pandemic.[1] Moreover, these apps could significantly benefit from a dynamic resource management as each of them can offer a mix of functionalities (viz. activities) including highly-interactive and relatively high-bandwidth (audio- and video-call) and little-interactive and low-bandwidth (chat) ones.

Herein, we specifically analyze the performance of different *multitask Deep Learning (DL) models*, related to different *apps* and different *activities* and trained with different *granularity levels*. We train and evaluate such DL models on traffic from CC apps that have seen the highest increase in usage during the COVID-19 pandemic (namely, `Skype`, `Teams`, `Webex`, and `Zoom`) [4], with their performance being interpreted at per-app, per-activity, and per-packet granularity.

Indeed, to contrast one of the main drawbacks of models based on Artificial Intelligence (particularly DNNs), namely their black-box nature, we apply *eXplainable Artificial Intelligence (XAI)* techniques to shed light on observed performance, and we link it to the properties of the traffic. The derived insights, on the one hand, provide additional confidence in the results of the proposed solution, on the other hand, point to possible improvements. To the best of our knowledge, this is the first work proposing an XAI-evaluated DL approach to predict network traffic at per-packet granularity.

The rest of the paper is organized as follows: Sec. II surveys literature predicting network traffic or analyzing explainability of DL models, focusing on network traffic analysis; Sec. III

[1] Sandvine, "The Global Internet Phenomena Report", 2022.

provides the methodology on multitask DL-based traffic prediction and the adopted XAI tool; Sec. IV describes the dataset collection, the apps/activities selection rationale, and the evaluation metrics; the experimental evaluation is provided in Sec. V; Sec. VI ends with conclusions and future perspectives.

II. Related Work

Predicting network traffic has attracted the interest of the scientific community, which has approached distinct practical network problems by casting them into a variety of forecasting tasks. Here, we discuss these works in terms of (a) prediction granularity, (b) application context, and (c) methodology. The section ends with the positioning of our contribution.

A remarkable amount of works aims at *predicting* the evolution of *traffic aggregates* (e.g., packet rates, volumes) over time, at different time-resolutions ranging from $\leq$ 1s, to few seconds, minutes, or even hours and days [5, 6, 7, 8]. As predicting *aggregated* traffic predates fine-grain predictions, DL approaches have been first applied to this arguably simpler problem. An early work [6] compares the performance of simple neural network approaches to deep ones. A more complex approach (a meta-learning adversarial scheme) is proposed in [9] for short-term prediction of aggregated traffic volume, to inform cellular downlink scheduling and sleep scheduling in user mobile devices. In [10] several deep sequences models are implemented, including Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU), to predict real traffic with and without outliers, investigating the significance of outlier detection in real-world traffic prediction.

Other research aims to predict the evolution of traffic features at *finer grain*. However, fine-grained prediction approaches primarily involve video traffic and target the forecast of video frame attributes [11]. Conversely, a *smaller body of work tackles packet-level analysis and addresses prediction at fine granularity while focusing solely on network-layer factors* [12, 13, 14]. Hidden Markov Models (HMMs) are adopted in [12] to model Internet traffic sources (i.e. SMTP, HTTP, online gaming, and messaging) based on inter-packet time and packet size. More recent works have focused on the prediction of network traffic generated by *mobile apps*. Specifically, HMMs and Markov Chains are investigated in [13] to characterize and predict network traffic generated by mobile apps. Montieri et al. [14] investigate and specialize a set of architectures (i.e. Convolutional, Recurrent, and Composite Neural Networks) to predict mobile-app traffic at packet-level granularity, also assessing the impact of multi-modality or varying memory size.

Positioning w.r.t. traffic prediction literature: the latter paper [14] represents the most related work as it addresses the same prediction problem (i.e. network-traffic prediction at packet level) leveraging DL approaches. Prompted by the popularity gained by *CC apps* with the COVID-19 pandemic, in this work we focus on their traffic, investigating the suitability of DL solutions to predict their fine-grained characteristics. Also, in this work, we leverage a *more detailed ground truth* that includes the specific activity performed by the users (beyond the app generating the traffic), thus investigating its impact on the capability of the DL model to predict traffic characteristics. Finally, we investigate the *suitability of considering a single model* to predict CC-app traffic against per-app specific models.

Concerning the **interpretability and reliability of models used for traffic analysis**, recently a bunch of researches have applied XAI techniques aiming at improving performance, robustness, reliability, and feasibility of AI models [15]. The works [16, 17] use LIME to provide *post-hoc explanations* for models *predicting QoE* in YouTube video streaming and targeting *video bit-rate adaptation*, respectively. Conversely, in [18] post-hoc explanations via the *Layerwise Relevance Propagation* approach (exploiting the layered structure of the neural network to capitalize recursive computation) are capitalized in a DNN-based *anomaly detection* framework whose goal is improving users' trust in the detected anomalies.

Recent research efforts have also applied various XAI approaches to interpret, refine, and improve solutions tackling *network-traffic classification*. A lightweight and *explainable-by-design* Convolutional Neural Network (CNN) is devised in [19] via comparison of parts of input data with per-class prototypes. Post-hoc analysis built upon *Shapley values* is employed by [20], presenting a *sample-dependent* (viz. local-interpretation) method based on Deep SHAP [21] to assess the importance contribution of classifiers' inputs on the outcomes obtained through an 1D-CNN for mobile-app traffic classification. In a recent work [22], we investigate interpretability and reliability to improve the behavior of state-of-the-art multimodal DL traffic classifiers by applying *global-interpretation* XAI techniques based on Deep SHAP, and methods to both assess and improve the reliability of classifiers. Finally, specifically focusing on *network-traffic prediction*, in our previous works [13, 14] we have applied *Markovian Distillation* for interpreting prediction results provided by ML/DL solutions.

Positioning w.r.t. traffic analysis via XAI: in this work, we tackle the black-box nature of traffic prediction solutions, *providing interpretation of opaque DL models* via XAI tools. Specifically, we inspect them in depth by providing *interpretability results based on* Deep SHAP. The above solution overcomes the interpretability limitations incurred by Markovian solutions due to limited memory [13, 14]. Equally important, to the best of our knowledge, no other work leverages XAI to investigate the interpretability of DL solutions aiming at predicting network traffic, regardless of both the aggregation granularity and the specific DL solution implemented.

III. Methodology

A. Multitask Deep Learning Traffic Prediction

We aim to predict network traffic generated by CC apps via a DL approach at the *finest granularity*, i.e. at packet-level. Indeed, we choose the widely-used bidirectional flow (biflow) as the traffic object constituting the (multivariate) time series

associated to the prediction task. A biflow encompasses all the packets sharing the same 5-tuple (IP src, IP dst, port src, port dst, protocol) in both upstream and downstream directions [14].

Given a biflow up to its n^{th} packet, our goal is to predict P *traffic parameters* associated to the $(n+1)^{th}$ packet: the vector $\boldsymbol{x}^{n+1}$ is the desired *output* of DL architecture. Such predictions are based on observations of previous values of the same traffic parameters, gathered by the predictor in a *memory window* of size W: these observations $\boldsymbol{x}^{n}, \ldots, \boldsymbol{x}^{n-(W-1)}$ are the *input* of the DL architecture. We stress that we construct the input via an *incremental windowing* approach based on a sliding memory window with a unit stride: we exploit incrementally-sized sets of samples until reaching the maximum size W. When the prediction task has an accumulated memory $\leq W$, we apply a (left) zero-padding up to W samples. This allows the prediction to be performed on the initial part of the biflow and/or for biflows shorter than W. We take advantage of *multitask architectures* that jointly address multiple prediction tasks, one for each of the P parameters considered.[2] Accordingly, we are targeting the design of a *single* DL model in the form:

$$\hat{\boldsymbol{x}}^{n+1} = \mathcal{P}(\boldsymbol{x}^{n}, \boldsymbol{x}^{n-1}, \ldots, \boldsymbol{x}^{n-(W-1)}) \tag{1}$$

where $\hat{\boldsymbol{x}}^{n+1}$ denotes the prediction vector associated to $\boldsymbol{x}^{n+1}$.

Traffic Parameters. We aim at predicting $P = 3$ traffic parameters of the packets belonging to the same biflow: (i) the *direction* (DIR) as a binary value indicating if the packet is downstream or upstream; (ii) the *payload length* (PL), that is the size (in bytes) of the transport-layer payload; (iii) the *inter-arrival time* (IAT), namely the time between the arrival of two consecutive packets. Before extracting these parameters from raw packet sequences, we remove zero-payload packets and recalculate the IATs (with a minimum granularity of 1μs) by saturating to the 99^{th}-percentile value (197.90 ms) of their distributions to remove the effect of outliers. Finally, we use a min-max scaler to constrain the PL and IAT within $[0, 1]$, a common procedure when exploiting DL models [14].

DL Architectures. We employ *three* categories of DL models: convolutional, recurrent and the composition of both, whose hyperparameters are set based on state-of-the-art works [14, 23, 24]. All the architectures are terminated by one dense layer (with sigmoid activation) for each traffic parameter to predict. Regarding the former, we consider a *1D-CNN* made of a cascade of two convolutional layers (with 32 and 64 filters, respectively, and kernel size of 5) each followed by a max-pooling layer (with a pool size of 3), and one dense layer with 128 neurons. All these layers have a ReLU activation. On the other hand, the two recurrent architectures are a *GRU* and an *LSTM*, both unidirectional with 200 units and sigmoid activation. Finally, as a composite architecture, we employ an extended version of the *SeriesNet*, a DL model based on dilated causal convolutions. The overall architecture consists of 7 dilated causal convolutional layers (DCC in the following)—with 32 filters and dilation and size of 2—with SeLU activation. Each DCC has a residual connection between the input and output, while the last two having a dropout of 0.8. The sum of the parameterized skip connections of each DCC is passed through a ReLU activation and then used as the input of a 1×1 convolutional layer. Finally, the output of the latter is concatenated with the stacking of two LSTM layers (with 200 units each) and a dense layer (with 128 nodes).

Loss Specification and Training Details. We are concerned with the prediction of P *traffic parameters* of the next packet, collected in the vector $\boldsymbol{x}^{n+1}$. Accordingly, the DL architecture is trained to minimize a *weighted sum* of the losses of the P prediction tasks considered, namely:

$$\mathcal{L}\left(\boldsymbol{\theta}_{\text{shared}}, \{\boldsymbol{\theta}_p\}_{p=1}^{P}\right) \triangleq \sum_{p=1}^{P} \lambda_p \, \mathcal{L}_p\left(\boldsymbol{\theta}_{\text{shared}}, \boldsymbol{\theta}_p\right) \tag{2}$$

with the weight λ_p representing the preference level of the p^{th} task[3] in the multitask objective function, $\boldsymbol{\theta}_{\text{shared}}$ the parameters associated to the layers shared by the different tasks, while $\boldsymbol{\theta}_p$ collects the parameters associated *exclusively* to the p^{th} task. Given the multitask nature of these architectures, the loss function to optimize depends on the specific prediction task to address. In detail, we train the DL architectures to minimize the *binary cross-entropy* loss function for the prediction of binary DIR, whereas for the prediction of the non-binary PL and IAT, to minimize the *Mean Squared Error (MSE)* between the actual traffic parameter associated to the $(n+1)^{th}$ time instant of a given biflow and the corresponding prediction.[4] In this work, we adopt *two* training strategies that differ based on how traffic information is grouped, leading to models able to capture traffic characteristics not limited to the single app. Hence, we consider the following *granularity levels*:

- **per-app level (APP)**: a separate model is obtained for each application (i.e. one $\mathcal{P}(\cdot)$ matched to each app); the model is trained with traffic labeled with a specific application label (i.e. package information);
- **all apps (ALL)**: a single model is obtained for all apps (i.e. a single predictor $\mathcal{P}(\cdot)$ is used for all apps); the model is trained on whole traffic without taking into account the information about the app that generated it.

B. Interpretability of DIR Prediction via DEEP SHAP

Herein, we describe the methodology defined to investigate the *interpretability* of the outcomes of the probabilistic-level

[2] In our previous work [14], we experimentally showed that multitask architectures are appealing from both prediction performance and complexity (in terms of overall training time) viewpoints, outperforming single-task solutions—needing a specific DL architecture to predict each traffic parameter—in both aspects.

[3] Based on the optimization of the weights carried out in [14], herein we set $\lambda_1 = \lambda_2 = 0.45$ and $\lambda_3 = 0.10$, where $p = 1$, $p = 2$, and $p = 3$ are associated to the DIR, PL, and IAT prediction tasks, respectively.

[4] We perform the optimization by employing the *Adam* optimizer with a batch size of 32, a learning rate of 10^{-3}, and exponential decay rates for the estimates of the first-order and second-order moments equal to 0.9 and 0.999 (Keras default values), respectively. Overall, each DL architecture is trained for 100 epochs and *early-stopping* is used to prevent overfitting (patience of 4 epochs and a minimum delta of 10^{-4} measured on validation loss).

(binary) classification task (i.e. the prediction of DIR values) tackled via the considered DL architectures. Specifically, our interpretability analysis leverages DEEP SHAP [21].

The starting point for interpreting DL architectures is to consider a simpler explanation model $g(\cdot)$, which is designed to closely-approximate the original model $f(\cdot)$. Hereafter we focus on *local methods*, which explain the model $f(\boldsymbol{x})$ in the neighborhood of each particular instance $\boldsymbol{x}$—i.e. a *per-biflow* explanation in our case—using the so-called *simplified inputs* $\boldsymbol{x}'$ that map to the original ones via the mapping $\boldsymbol{x} = \boldsymbol{h}_{\boldsymbol{x}}(\boldsymbol{x}')$.

Herein, we consider the *Additive Feature Attribution (AFA)* functional form for the explanation model $g(\cdot)$:

$$g(\boldsymbol{z}^{'}) = \phi_0 + \sum_{m=1}^{M} \phi_m \, z_m^{'} \tag{3}$$

where $\boldsymbol{z}^{'} \in \{0,1\}^M$, M denotes the number of simplified inputs, and $\phi_m \in \mathbb{R}$. This class of explanation models associates an "effect" ϕ_m to each input. *Hence, the original model output $f(\boldsymbol{x})$ can be approximated by summing the effects of all input attributions.* A well-known approach to compute AFA solutions is via the so-called *Shapley values*, originating from cooperative game theory and specifying the contribution of player m to the payoff $v(\mathcal{P})$ achieved by the whole coalition $\mathcal{P}$. Unfortunately, since the exact computation of Shapley values grows exponentially with the input size M, we resort to the *SHapley Additive exPlanation (SHAP)*. This approximation computes the Shapley values in a computationally-efficient way and eliminates the need to re-train the models by approximating these values via the conditional expectation $f(\boldsymbol{h}_{\boldsymbol{x}}(\boldsymbol{z}^{'})) \approx \mathbb{E}\{f(\boldsymbol{z})|\boldsymbol{z}_{\mathcal{S}}\}$, where $\mathcal{S}$ denotes the set of non-zero indices within $\boldsymbol{z}^{'}$. Further, we use DEEP SHAP [21], an adaptation of the *DeepLIFT* algorithm for the evaluation of SHAP values. Specifically, DeepLIFT computes a compositional approximation of SHAP values: it uses the output expectation as the reference value, and it resorts to explicit Shapley equations for consistent linearization. The reference value is a user-defined parameter set to be an uninformative background value for the m^{th} input.

Per-sample explanation outcomes based on local methods are then aggregated to obtain *global explanations*. Since a soft-output can assume a range of different values, the absolute importance range of the m^{th} input may differ from sample to sample. Hence, our *global explanation* approach assumes the preliminary calculation of *range-normalized* SHAP values:

$$\widetilde{\phi}_m \triangleq \phi_m \,/\, \sum_{m=1}^{M} \phi_m \tag{4}$$

Considering $\widetilde{\phi}_m$ (as opposed to ϕ_m), allows focusing on the relative *importance* of each input and then draw out importance measures that do not depend on the peculiar architecture confidence (generally higher or lower) and consistently aggregate over the test samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$. Last, we solely focus on the aggregation of correctly-classified DIR samples [22]: this choice allows focusing on the correct behavior of a given DL-based traffic predictor, allowing to interpret its counter-intuitive (while right) decisions *a posteriori*.

IV. EXPERIMENTAL SETUP

In this section, we detail the experimental setup, providing information about the dataset and the evaluation metrics we refer to for the assessment of the prediction models.

Dataset Collection. In this work, we exploit the MIRAGE-COVID-CCMA-2022 public dataset[5] collected by students and researchers of the University of Napoli "Federico II" during Apr–Dec 2021 via the MIRAGE architecture [25] optimized to cope with the generation and capture of the traffic of CC apps. The experimenters used three mobile devices: a Google Nexus 6 and two Samsung Galaxy A5 with Android 10. In each capture session, the experimenters performed a specific activity using a certain CC app, aiming at generating traffic that reflects its common usage.[6] Each session resulted in a PCAP traffic trace and additional `netstat`[7] log-files reporting information on established network-connections. The latter were used to generate the ground-truth, namely to reliably label each biflow with (i) the *Android package-name* of the app and (ii) the specific *activity* performed by the user operating the device[8]. Overall, after the application of the incremental windowing approach—described in Sec. III—to construct the input of the DL architectures, the employed dataset encompasses 806k samples for `Skype`, 1.9M for `Teams`, 2.5M for `Webex`, and 1.2M for `Zoom`.

Apps' and Activities' Selection Rationale. The mobile apps used for business meetings, classes, and social interaction—in short, *CC apps*—have experienced a huge increment in utilization when "stay-at-home" orders were first issued worldwide and are still widely used. Accordingly, we have selected a subset of *four* apps from MIRAGE-COVID-CCMA-2022 based on popularity and utilization boost due to the COVID-19 pandemic [4]: `Skype`, `Teams`, `Webex`, and `Zoom`. As aforementioned, according to the app usage, the experimentation included the following live activities: *Chat (`Chat`)*—involves just two participants exchanging textual messages and/or multimedia content (e.g., images or GIFs); *Audio-call (`ACall`)*—involves just two participants transmitting only audio; *Video-call (`VCall`)*—involves many attendees which can transmit both video and audio (e.g., live events such as video calls between two or more attendees or webinars).

Evaluation Metrics. The performance evaluation of traffic prediction strategies is based on a solid *stratified five-fold cross-validation* setup: given an app-activity pair, 80% of the biflows constitute the training/validation set (80% is the actual

[5]To foster replicability and reproducibility we have publicly released the MIRAGE-COVID-CCMA-2022 dataset: https://traffic.comics.unina.it/mirage/mirage-covid-ccma-2022.html.

[6]Each traffic capture session spanned $15 \sim 80$ mins based on the activity and has been performed with the up-to-date version of the app. Also, to limit background traffic, network access has been disabled for all the apps but the one under test.

[7]https://linux.die.net/man/8/netstat.

[8]Activity labels were manually assigned based on the knowledge of the individual activity performed by the user during the specific capture.

training set, while 20% is assigned to the validation set)[9], and the remaining 20% the test set (used for evaluation purposes). Thus, for each evaluation metric, we provide its average value and standard deviation over the five folds. We use the *G-mean* to evaluate the prediction performance of the binary DIR traffic parameter:

$$\text{G-mean} \triangleq \sqrt{\rho_{dir}^{\text{dw}}\,\rho_{dir}^{\text{up}}} \tag{5}$$

with $\rho_{dir}^{\text{dw}} \triangleq \Pr(\hat{x}_{dir} = \text{DW} \mid x_{dir} = \text{DW})$ and $\rho_{dir}^{\text{up}} \triangleq \Pr(\hat{x}_{dir} = \text{UP} \mid x_{dir} = \text{UP})$. In this case, x_{dir} is associated to the sequence of actual DIRs and $\hat{x}_{dir}$ to those predicted, whereas DW and UP indicate downstream and upstream direction, respectively. The probability of correctly predicting the DIR of downstream/upstream packets (ρ_{dir}^{dw}/ρ_{dir}^{up}) is estimated as the ratio of correct predictions on a given direction by the overall number of samples associated to the same (true) direction. Conversely, to assess the prediction of PL and IAT, we adopt the *Root Mean Squared Error (RMSE)*:

$$\text{RMSE}_p \triangleq \sqrt{\frac{1}{\bar{N}}\sum_{j=1}^{\bar{N}_B}\sum_{n=1}^{\bar{N}_j-1}\left[\hat{x}_p^{n+1}(\bar{B}_j) - x_p^{n+1}(\bar{B}_j)\right]^2} \tag{6}$$

where $\bar{N}$ is the total number of predictions, $x_p^{n+1}(\bar{B}_j)$ the value of the p^{th} traffic parameter (with $p \in \{\text{PL}, \text{IAT}\}$) observed for packet $n+1$ from the j^{th} biflow $\bar{B}_j$ (of length $\bar{N}_j$), and $\hat{x}_p^{n+1}(\bar{B}_j)$ the corresponding value provided by the prediction model.

V. Experimental Evaluation

In the following, we first investigate traffic prediction performance for all apps and activities, attained by different multitask DL models, trained with different granularity-levels (Sec. V-A). Then, we explain the model behavior when predicting the next packet direction via DEEP SHAP (Sec. V-B).

A. Overview of Traffic Prediction Performance

Herein, we provide an overview of the performance of multitask DL models used for predicting DIR, PL, and IAT by adopting[10] a memory window size of $W = 10$. To this end, Figs. 1a, 1c, and 1e summarize the performance of all models—trained with different granularity levels (i.e. ALL and APP)—w.r.t. each of the apps considered. Specifically, focusing on training strategies (viz. granularity levels), we observe that models trained at the ALL level perform as well as or even outperform those at the APP level. Moreover, looking at the prediction of the specific traffic parameter, for DIR all the considered strategies attain a G-mean of $\approx 80\%$, with the only exception of Zoom showing $\approx 65\%$ G-mean. A similar outcome is also observed when looking at PL prediction: on Zoom the RMSE attained is $\approx +50$B w.r.t. the other apps. A slightly different—although more stable—performance picture is evident for IAT prediction, for which Skype exhibits the worst RMSE ($\approx$ 5ms higher than the other apps), also presenting the highest variability. This is mainly associated with the higher variability of IATs related to ACall and VCall activities (not shown for brevity). From a practical viewpoint, since the DIR error has a higher impact on the use of (packet-level) prediction results, in Figs. 1b, 1d, and 1f we focus on Zoom traffic prediction (i.e. the most problematic app on DIR prediction) dissected based on the specific activity performed by the user. Therein, the worst performance for DIR and PL prediction corresponds to the VCall activity, achieving the lowest G-mean ($\approx 62\%$) and the highest RMSE ($\approx$ 280B), respectively. Notably, IAT prediction shows an opposite trend, as VCall has an RMSE less than half of that of Chat and $2.5\times$ lower than that of ACall.

Finally, when looking at the impact of the specific DL architecture employed, it appears to have a negligible impact on prediction performance, both considering all the traffic generated by a given app and dissecting the traffic per activity.[11] Still, while performance is comparable, the complexity of the architectures considered—measured via the number of trainable parameters—varies significantly. In fact, comparing GRU, LSTM, and SeriesNet with CNN (i.e. the least complex with 27.5K parameters), there is a significant increase in complexity ranging from +96.1K (GRU) to +502.3K (SeriesNet).[12]

Take-Away Messages. The reported analysis shows that *a single model trained on the whole traffic of CC apps is enough*, as there seems to be no appreciable gain in training a specialized model for each app. This result has a significant practical impact, allowing to spare the training, deployment, and management of multiple models. Secondly, looking at the app-related performance, Zoom results the most problematic app to predict, especially for PL and DIR, while the others exhibit better and more stable performance. For the same features, VCall is the hardest to be predicted among Zoom-related activities. Finally, though the DL architectures considered provide similar performance, they differ greatly in computational complexity, with CNN being the least complex. Accordingly, in what follows we focus on the CNN trained on the whole traffic (namely, the ALL variant), constituting the best trade-off between complexity and prediction performance.

B. Interpreting DIR Prediction via DEEP SHAP

In this section, we analyze the relative importance of inputs—i.e. the three traffic parameters extracted from the last W observed packets of each biflow, namely transport-layer payload length (PL), direction (DIR), and inter-arrival time (IAT)—on the prediction of DIR, based on DEEP SHAP.

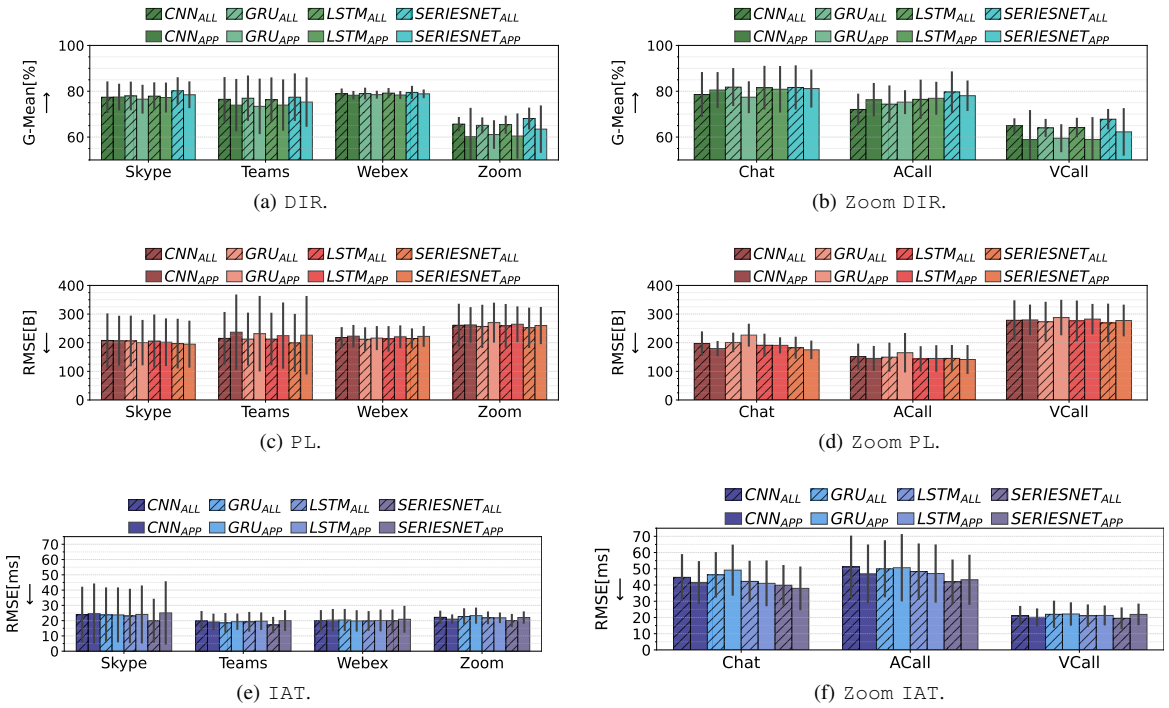
To this end, in Fig. 2, for each field, we report the median importance values across the last 10 packets (since $W = 10$)

[9]The validation set is exploited to properly capitalize the early-stopping technique employed during training to prevent overfitting.

[10]Preliminary results—not shown for brevity—showed that $W = 10$ constitutes the best trade-off between the complexity of the model (growing with W) and the effectiveness of the prediction.

[11]Similar considerations can be drawn also for Skype, Teams, and Webex whose per-activity performance figures are not shown for the sake of brevity.

[12]As discussed in Sec. III, for a fair comparison of the different architectures, we set identical tunable parameters concerning all controllable aspects of models' training procedure, namely: total number of epochs, optimizer, batch size, learning rate, and early-stopping.



(a) DIR. (b) Zoom DIR.

(c) PL. (d) Zoom PL.

(e) IAT. (f) Zoom IAT.

Fig. 1. Prediction performance of CNN, LSTM, GRU, and SeriesNet on DIR (a, b), PL (c, d), and IAT (e, f). Results refer to the prediction of traffic generated by *each app regardless of the specific activity* (a, c, e) and *Zoom when performing a specific activity* (b, d, f)—i.e. Chat, ACall, and VCall. Memory size is set to $W = 10$ and both APP and ALL granularity levels are considered. The arrow close to y-axis shows the desired trend.

of each biflow used to feed the model to obtain the prediction related to the following packet. In detail, by focusing on the prediction results provided by the ALL CNN model, the figure reports for each packet direction (i.e. upstream ↑ and downstream ↓) the breakdown on Skype and Zoom.[13] For a given direction, the importance of the observed packets varies according to the input type. Furthermore, for a given type of input, we note that the importance associated with the packets in the sequence varies according to the predicted direction.

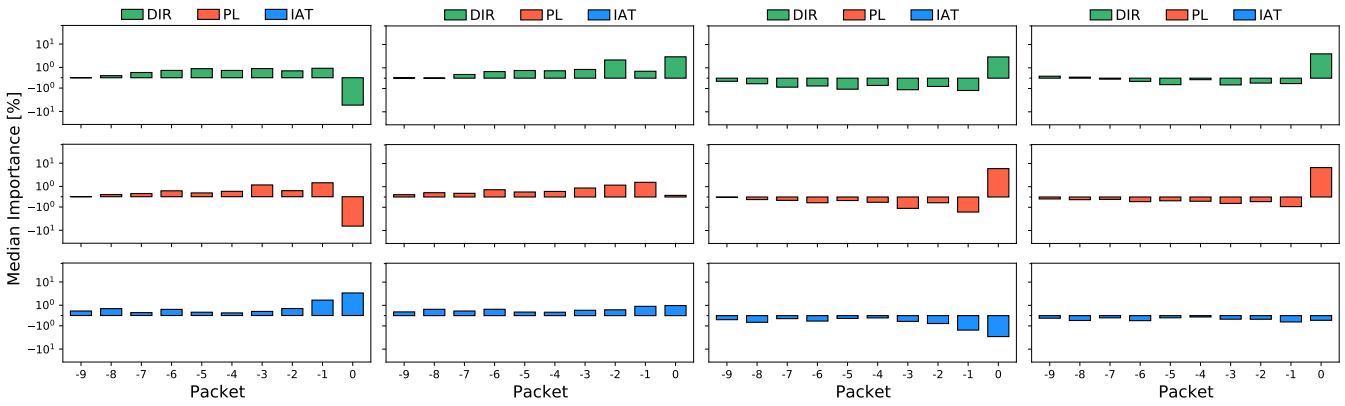
Specifically, when the model *correctly* predicts the *upstream* direction (↑, see Figs. 2a-2b), for both apps almost all the observed sequences of inputs contribute positively to the prediction, for which a higher importance is generally attributed to more recent packets. Nevertheless, looking at DIR and PL of the last observed packet we observe that these have a negligible impact or are even detrimental (negative score) to a correct prediction of the direction. This is especially true for Zoom, for which the last observed packet has a non-negligible negative importance—which also corresponds to the highest values in magnitude—indicating that this generally leads the model to predict the *downstream* direction. Conversely, moving to *downstream* prediction, we notice an inverse trend: DIR and PL of the last packet have a positive impact on the prediction. Also, their importance (in terms of magnitude) is significantly greater than that of the other packets. Based on this observation we can infer that DIR and PL of the last packet lead the model to correctly predict the *downstream* direction.

Take-Away Messages. The prediction of the direction of the next packet, based on a memory made of the last 10 packets, *is mostly influenced by the* $3 \sim 4$ *more recent packets observed*. Moreover, the direction and transport-layer payload length of the last observed packet can be either highly beneficial (downstream case) or detrimental (upstream case) to correctly predict the direction of the next packet.

[13]Related PL and IAT inputs, Webex and Teams showed similar behaviors to Zoom, while for DIR their behavior is similar to Skype.

VI. Conclusions and Future Directions

We tackled *packet-level traffic prediction* via Deep Learning architectures (a CNN, a GRU, an LSTM, and a SeriesNet) based on a publicly released dataset. The latter contains traffic traces of the four mobile apps whose usage has increased the most since the beginning of COVID-19 pandemic. We employed XAI approaches (i.e. DEEP SHAP) to contrast the black-box nature of these models and obtain actionable insights on the importance of specific subsets of input data. The analysis allowed to select a single model, a 1D-CNN trained on the whole traffic of all apps, representing the best trade-off between prediction performance and complexity. This solution has a substantial practical impact since it eliminates the need for multiple model training, deployment, and administration. The results show a variety of behaviors for the different apps, with Zoom being the hardest to predict, especially



(a) Zoom upstream. (b) Skype upstream. (c) Zoom downstream. (d) Skype downstream.

Fig. 2. Median importance (in log-scale) of each packet parameter (i.e., DIR, PL, and IAT) on the prediction of DIR for Zoom and Skype. The x-axis reports the packet index in chronological order, with 0 for the last observed packet, -1 the previous one, and so forth. Each app is separately analyzed according to the (true) direction of packets, as upstream ((a) and (b)) and downstream ((c) and (d)).

regarding the direction (65% G-mean) and the payload length (260B RMSE) of the next packet. Regarding the *activities* that can be performed by those apps, VCall results the hardest (resp. easiest) for predicting the direction and the payload length (resp. the timing). The fine-grained interpretation of the performance highlighted that using a limited memory of previously observed packets, model predictions are mainly influenced by the most recent packets. Also, in most cases, the last observed packet can be either very informative (for Skype and Zoom downstream) or confounding for the model (for Zoom upstream). **Future directions** will include: (i) interpretability analysis for payload length and inter-arrival time, (ii) reliability analysis of the model, (iii) use of XAI toward the improvement of multitask predictors, and (iv) lifelong learning to cope with concept drift due to app aging.

References

[1] A. Feldmann, O. Gasser, F. Lichtblau, E. Pujol, I. Poese, C. Dietzel, D. Wagner, M. Wichtlhuber, J. Tapiador, N. Vallina-Rodriguez, O. Hohlfeld, and G. Smaragdakis, "The Lockdown Effect: Implications of the COVID-19 Pandemic on Internet Traffic," in *ACM IMC'20*.

[2] M. Candela, V. Luconi, and A. Vecchio, "Impact of the COVID-19 pandemic on the Internet latency: A large-scale study," *Comp. Networks*, vol. 182, 2020.

[3] L. Velasco, R. Casellas, S. Llana, L. Gifre, R. Martínez, R. Vilalta, R. Muñoz, and M. Ruiz, "A control and management architecture supporting autonomic NFV services," *Photonic Network Communications*, vol. 37, no. 1, pp. 24–37, 2019.

[4] I. Guarino, G. Aceto, D. Ciuonzo, A. Montieri, V. Persico, and A. Pescapé, "Classification of communication and collaboration apps via advanced deep-learning approaches," in *IEEE CAMAD'21*.

[5] N. Ramakrishnan and T. Soni, "Network traffic prediction using recurrent neural networks," in *IEEE ICMLA'18*.

[6] T. P. Oliveira, J. S. Barbar, and A. S. Soares, "Computer network traffic prediction: a comparison between traditional and deep learning neural networks," *Int. J. of Big Data Intell.*, vol. 3, no. 1, pp. 28–37, 2016.

[7] C. Huang, C. Chiang, and Q. Li, "A study of deep learning networks on mobile traffic forecasting," in *IEEE PIMRC'17*.

[8] Y. Huo, Y. Yan, D. Du, Z. Wang, Y. Zhang, and Y. Yang, "Long-term span traffic prediction model based on STL decomposition and LSTM," in *IEEE APNOMS'19*.

[9] Q. He, A. Moayyedi, G. Dán, G. P. Koudouridis, and P. Tengkvist, "A Meta-Learning Scheme for Adaptive Short-Term Network Traffic Prediction," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 10, pp. 2271–2283, 2020.

[10] S. Saha, A. Haque, and G. Sidebottom, "Deep Sequence Modeling for Anomalous ISP Traffic Prediction," in *IEEE ICC'22*.

[11] A. Kalampogia and P. Koutsakis, "H.264 and H.265 Video Bandwidth Prediction," *IEEE Trans. Multimedia*, vol. 20, no. 1, pp. 171–182, 2018.

[12] A. Dainotti, A. Pescapé, P. Salvo Rossi, F. Palmieri, and G. Ventre, "Internet traffic modeling by means of Hidden Markov Models," *Comp. Networks*, vol. 52, no. 14, pp. 2645–2662, 2008.

[13] G. Aceto, G. Bovenzi, D. Ciuonzo, A. Montieri, V. Persico, and A. Pescapé, "Characterization and Prediction of Mobile-App Traffic Using Markov Modeling," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 1, pp. 907–925, 2021.

[14] A. Montieri, G. Bovenzi, G. Aceto, D. Ciuonzo, V. Persico, and A. Pescapè, "Packet-level prediction of mobile-app traffic using multitask Deep Learning," *Comp. Networks*, vol. 200, p. 108529, 2021.

[15] T. Zhang, H. Qiu, M. Mellia, Y. Li, H. Li, and K. Xu, "Interpreting AI for Networking: Where We Are and Where We Are Going," *IEEE Commun. Mag.*, vol. 60, no. 2, pp. 25–31, 2022.

[16] A. Morichetta, P. Casas, and M. Mellia, "EXPLAIN-IT: Towards Explainable AI for Unsupervised Network Traffic Analysis," in *ACM BigDAMA'19*.

[17] A. Dethise, M. Canini, and S. Kandula, "Cracking Open the Black Box: What Observations Can Tell Us About Reinforcement Learning Agents," in *ACM NetAI'19*.

[18] K. Amarasinghe, K. Kenney, and M. Manic, "Toward Explainable Deep Neural Network Based Anomaly Detection," in *IEEE HSI'18*.

[19] K. Fauvel, A. Finamore, L. Yang, F. Chen, and D. Rossi, "A Lightweight, Efficient and Explainable-by-Design Convolutional Neural Network for Internet Traffic Classification," *arXiv e-prints*, pp. arXiv–2202, 2022.

[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] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *NeurIPS'17*.

[22] A. Nascita, A. Montieri, G. Aceto, D. Ciuonzo, V. Persico, and A. Pescapé, "XAI meets mobile traffic classification: Understanding and improving multimodal deep learning architectures," *IEEE Trans. Netw. Serv. Manag.*, vol. 18, no. 4, pp. 4225–4246, 2021.

[23] 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 ISI'17*.

[24] Z. Shen, Y. Zhang, J. Lu, J. Xu, and G. Xiao, "SeriesNet: A Generative Time Series Forecasting Model," in *IEEE IJCNN'18*.

[25] G. Aceto, D. Ciuonzo, A. Montieri, V. Persico, and A. Pescapé, "MIRAGE: Mobile-app Traffic Capture and Ground-truth Creation," in *IEEE ICCCS'19*.