Multi-task learning for identifying multi-activity situations and application type from network traffic

Ahcene Boumhand, Kamal Singh, Yassine Hadjadj-Aoul, Matthieu Liewig, César Viho

Multi-task learning for identifying multi-activity situations and application type from network traffic

Ahcene Boumhand[$], Kamal Singh[*], Yassine Hadjadj-Aoul[$], Matthieu Liewig[+], and César Viho[$]
[+]*Orange Labs, Rennes, France*
firstname.lastname@orange.com
[*]*Univ Jean Monnet, IOGS, CNRS, UMR 5516, LaHC, F - 42023 Saint-Etienne, France*
firstname.lastname@univ-st-etienne.fr
[$]*Univ Rennes, Inria, CNRS, IRISA, UMR 6074, F-35000 Rennes, France*
firstname.lastname@univ-rennes.fr

***Abstract*—Optimizing networks to meet user needs has been a long-standing goal for the various key players in the network sector. To this end, a large number of studies have addressed the case of classifying network traffic into a set of activities (e.g., streaming) and applications (e.g., Spotify). Nonetheless, the fast-paced growth of the digital market has favored the advent of new consuming habits such as the simultaneous performance of multiple activities. This concept is referred to as multi-activity situations or media multi-tasking. Conceiving solutions that can cope with these emerging consuming patterns may enable network operators and service providers to better adapt their network management solutions and commercial plans. In this paper, we propose a novel approach that can deal with a challenging scenario comprising both single-activity and multi-activity situations. The proposed approach pre-processes a network trace over a time-window and then determines to which situation type it belongs. Furthermore, it identifies the type of the activities being performed and the applications being used (e.g., chat on Facebook & streaming on Spotify). Our experiments highlighted that our solution is able to achieve a satisfactory level of performance despite the complexity of the scenario that we target. Indeed, our obtained results are comparable to state-of-the-art techniques addressing less challenging scenarios that involve only single-activity situations.**

***Index Terms*—Network traffic classification, activity recognition, multi-activity situations, deep learning, multi-task learning.**

I. Introduction

Recognizing the activities associated with network flows is crucial for optimizing key network management processes such as quality of service (QoS) and resource planning. In this regard, numerous studies from the network traffic classification field [1] have covered the diverse elements that intervene in users' interactions with digital services. These elements include the activities being performed (e.g., streaming) and the applications being used (e.g., YouTube). Nevertheless, the fast-paced evolution of the digital market has induced a significant shift in users' consuming habits. In fact, the proliferation of devices and applications has paved the way to the emergence of the concept of media multi-tasking or multi-activity situations. This concept refers to the simultaneous performance of two or more activities, such as mailing and streaming, among others.

The concept of multi-activity situations has already attracted several recent studies from the field of social sciences [2] [3]. These studies have shown the widespread occurrence of multi-activity situations among various communities of the population (e.g., teenagers, teleworkers) and their potential impact on health and productivity. Additionally, the concept of multi-activity situations can nurture the interest of operators and service providers in gaining insights into the modern consumption patterns and requirements of their clients. This will enable these stakeholders to better adapt their network management solutions and commercial plans. However, in spite of its potential significance, none of the previous studies on network traffic classification have approached the concept of multi-activity situations. These studies have considered only the case when a single-activity (e.g., chatting, or mailing, or streaming, etc.) is performed at a time. To the best of our knowledge, our previous work in [4] is the first study to treat the concept of multi-activity situations as a network traffic classification task. In our aforementioned study, we proposed a new method that generates multi-activity network traces starting from single-activity network traces together with a classifier that pre-processes a multi-activity network trace and then assigns it into a predicted multi-activity class.

In our present work, we extend the scope of our solution to cope with a more challenging and complete scenario that involves both single-activity and multi-activity network traces at the same time. This solution consists of the following main contributions:

- We propose a framework that can pre-process a given time-window of a network trace and then predict whether it corresponds to a single-activity situation or a multi-activity situation (i.e., situation-type identification).
- Additionally, we endow our proposed framework with two supplementary tasks that aim to recognize the type of the activities being performed (e.g., chat, chat & streaming) and the type of the applications within which the activities are being performed (e.g., chat on Facebook, chat on Facebook & streaming on Spotify).
- We deliver the three distinct outputs (i.e., situation-type, activity-type, application-type) of our framework in one pass leveraging a sole deep learning model thanks to the use of the multi-task learning paradigm.

The remainder of this paper is organized as follows. Section II reviews a set of selected works that are related to network traffic classification. In Section III, we present the main components of our proposed framework. Section IV describes the experimental parameters and resources that we leveraged to create an instance of our proposed framework. Section V depicts the performance of our solution together with a comparison with some related works. In Section VI, some of the potential improvements and technical choices regarding our methodology are discussed. Finally, Section VII concludes the paper while giving the main directions for future work.

II. Related work

To associate a given network flow to its corresponding activity type or application type classes, researchers and practitioners usually apply a plethora of techniques that belong to the network traffic classification field. These techniques range from classical approaches such as deep packet inspection (DPI) to modern deep learning (DL) algorithms that reach state-of-the-art results in this field.

For instance, to differentiate between a set of activity types (e.g., mailing, VoIP), authors in [5] used a combination of K-means and random forests (RF) over a set of statistical features (e.g., mean time between packets' arrival). Whereas in [6], authors proposed an attention-aided LSTM as well as a hierarchical attention network (HAN) that receive a sequence of packets' payloads. A similar approach was proposed in [7] by leveraging a two-dimensional CNN and a LSTM.

In a similar manner, to distinguish between a given set of applications (e.g., Skype, YouTube), authors in [8] based their method on Markov chain and RF models to analyze both sequential behavior and statistical characteristics (e.g., packet size distribution) of a network flow. In [9], multi-head attention was leveraged over a sequence of packets' payloads and metadata (e.g., packet position, packet size). In [10], the authors employed a multi-modal representation to cover the different types of information that can be extracted from a bi-directional flow. Their first proposed modality (i.e., payload modality) consisted of a predefined amount of payload extracted from the beginning of the corresponding bi-directional flow. The second modality (i.e., protocol fields modality) is a time series that coveys metadata of the initial packets that form the given bi-directional flow (e.g., TCP window size). Subsequently, the first modality was handled leveraging a one-dimensional CNN while the second modality was treated using bi-GRU.

A common limitation of the previously cited studies is that they treat only the case when a single activity is performed at a time. These techniques are therefore not capable of detecting multi-activity situations. Furthermore, single activities do not cover all real-world scenarios where a user may perform two or several activities simultaneously. These are some of the reasons that led us to investigate the detection of multi-activity situations.

Multi-task learning (MTL) is a machine learning (ML) approach that enables the training and the performance of multiple tasks simultaneously with a sole ML model on a single input. This approach aims to exploit the commonalities between the gathered tasks and enhance their overall performance. Recently, MTL has gained a significant amount of attention in the network traffic classification field following its notable performance in other research fields.

For instance, in [11], MTL was used to train a model that can identify the activity type of a flow along with its duration and bandwidth. In this case, the model relied on a one-dimensional CNN that receives a sequence of packets' metadata (e.g., packet size, packet direction). In [12], MTL was used for both traffic volume prediction and activity type identification where the best results were achieved when a sequence-to-sequence auto-encoder (AE) was leveraged for common features representation. Similarly, the work in [13] applied MTL for application type, activity type, and encryption type identification relying on a combination of bi-directional GRU and one-dimensional CNN over packets' payloads and metadata (e.g., TCP window size, packet direction).

In our study, we adopt the MTL paradigm to gather three complementary tasks that aim to thoroughly describe an activity situation based on a time-window of a given network trace. Specifically, the first task determines whether the corresponding time-window represents a single-activity situation or a multi-activity situation. The second task determines the types of the activities being performed. Whereas the third task determines the types of the applications that host the conducted activities.

III. SMAR: **S**ingle and **M**ulti **A**ctivity **R**ecognition framework

Hereafter, we enumerate first a set of assumptions and specifications that guide some conceptual choices in our proposed solution (Section III-A). Then, we describe thoroughly each of the main building blocks that comprise our conceived framework (Section III-B to Section III-D). The overall framework is illustrated in Figure 1.

A. Hypotheses and requirements

To alleviate the complexity of the targeted problem, we defined a list of assumptions for our solution, as follows:

- When generating multi-activity network traces, we assume that multi-activity situations are constituted only of two simultaneous activities (i.e., dual-activity situations). This aligns with the assumptions adopted in studies from the social sciences field that have addressed the concept of multi-activity situations [2] [3].
- For our multi-activity network traces, the recorded activities are presumed to be conducted by a unique user on a single or multiple devices.

Additionally, we have met some self-imposed competing requirements that dictate us to find a balance between reaction time and providing the classifier with enough information to perform its predictions accurately. These requirements are listed below:

- The designed solution has to deliver inferences that are as accurate as possible.

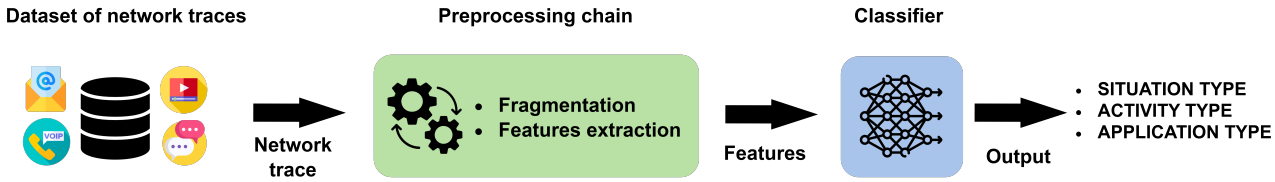

Fig. 1. Illustration of our SMAR framework. (Icons used in the image are from freepik.com and flaticon.com)

- The solution has to infer within $10s$, involving a time-span of the input time-window and the time required for the pre-processing and prediction operations.

B. Dataset of single-activity and multi-activity network traces

The overall dataset can be separated into two subsets, a set of single-activity network traces and a set of multi-activity network traces. Each set can be described as an ensemble of (N) network traces (i.e., raw PCAP files) $D = (T_1, T_2, T_3, \ldots, T_N)$ where each network trace is an ordered sequence of packets. We suppose that each network trace is associated with three labels that refer to the situation type, the activity type, and the application type of the activity that was conducted when the corresponding trace was captured.

C. Preprocessing chain

This component is composed of two steps. First, each trace is cut into equal slices. Then, representative features are extracted from each slice to be provided as input to the classifier. These steps are described below:

1) Fragmentation: In this step, incoming traces from the source dataset undergo a fragmentation operation as a primary step for preparing an adequate input for the classifier. Each network trace from the source dataset is split into time windows of (W) seconds. This time window is a hyper-parameter that has to be chosen wisely in a manner that enables us to satisfy some self-imposed constraints (see section III-A).

2) Features extraction: To enhance our representation of the fragment to classify, we extract distinct types of features that aim to describe the network behavior of the corresponding fragment according to varied aspects. This set of features involves features that are based on fragments' packets sizes and inter-arrival times as well as features that attempt to summarize the global characteristics of the fragment and its flow related structure (e.g., number of distinct IP addresses within the corresponding fragment). In fact, packets sizes and inter-arrival times are widely used in the network traffic classification field and have shown their usefulness in differentiating between traffic types in numerous works like in [14], [11]. Moreover, the global characteristics of a given fragment and its flow related structure should contribute in differentiating between single-activity and multi-activity fragments. The set of features that we leverage in our framework are grouped into three subsets that we exhibit below:

- Packets inter-arrival times related features: In this subset, fragments that are issued from the previous step are represented as time series of (L) time steps where each time-step is a sub-time-window of size (W/L) seconds. This representation will allow us to have a fine-grained description of each fragment since the corresponding subset of features will not be computed over the entire window but over each sub-time-window. The following features are computed over the packets that are contained within each time-step of the time series: mean, variance, skewness, and kurtosis of packets inter-arrival times.
- Packets sizes related features: In this subset, the distribution related to the sizes of packets that are contained within the corresponding fragment is represented as a histogram. This histogram is constituted of $(B = P/S)$ bins, where (P) is the maximum length of a packet in bytes while (S) is the length of a bin in bytes. Consequently, the (i^{th}) bin of the histogram contains the frequency of the packets whose sizes fall within the interval $[i \times S, (i+1) \times S]$, where (i) is an integer that belongs to the interval $[0, B-1]$.
- Global characteristics related features: This subset includes four features that are computed over the packets that are contained within the whole fragment. This involves: the number of packets contained within the fragment, the size of the fragment that can be defined as the sum of the sizes of its packets, the number of distinct IP addresses that are leveraged by the packets that reside within the corresponding fragment, the number of distinct port numbers that are leveraged by the packets that reside within the corresponding fragment.

D. Single-activity and multi-activity network traces classifier

The conceived classifier is endowed with three distinct outputs that aim to describe the classified fragment in a fine-grained manner. These outputs are described in what follows:

- Situation-type: This output indicates whether the corresponding fragment belongs to a single-activity situation or a multi-activity situation.
- Activity-type: This output indicates the type of the single-activity or multi-activity that is being performed (e.g., chat, chat & streaming).
- Application-type: This output determines the applications within which the single-activity or multi-activity is being performed (e.g., Facebook, Facebook & Spotify).

IV. Experimental Setup

In this section, we depict the resources, tools, and hyper-parameters that we selected to carry out our methodology that has been described in Section III.

A. Dataset of single-activity and multi-activity network traces

As a source for our single-activity network traces samples, we have leveraged the ISCXVPN2016 dataset [14]. This dataset is publicly available and has been used in several previous works to train and test their network traffic classification models. The ISCXVPN2016 dataset is constituted of a set of network traces that are labeled according to the particular activity that was performed when the trace was captured along with the application within which the activity was performed. It is important to note that the subset of single-activity network traces is further processed for generating the subset of multi-activity network traces. This has restrained our selection on part of the single-activity network traces that are available in the ISCXVPN2016 dataset due to the storage and computational complexity of the multi-activity traces generation process (described in [4]). Thus, the following list of activities and their corresponding applications were selected to constitute our subset of single-activity network traces: Chat (Facebook, Google Hangouts), Mailing (Thunderbird), Streaming (Spotify, YouTube), VoIP (Skype, VoIPBuster).

The resulting subset is then split into three sets: the training set (60%), the validation set (20%), and the test set (20%).

In order to generate our subset of multi-activity network traces, we applied the multi-activity traces generation process that is described in [4] on each set (i.e., the training set, the validation set, and the test set) of the subset of single-activity network traces separately. This yielded into a subset of multi-activity network traces that is constituted of the classes that we cite in what follows:

- Chat & Mailing (Facebook & Thunderbird, Google Hangouts & Thunderbird)
- Chat & Streaming (Facebook & Spotify, Facebook & YouTube, Google Hangouts & Spotify, Google Hangouts & YouTube)
- Chat & VoIP (Facebook & Skype, Facebook & VoIPBuster, Google Hangouts & Skype, Google Hangouts & VoIPBuster)
- Mailing & Streaming (Thunderbird & Spotify, Thunderbird & YouTube)
- Mailing & VoIP (Thunderbird & Skype, Thunderbird & VoIPBuster)
- Streaming & VoIP (Spotify & Skype, Spotify & VoIPBuster, YouTube & Skype, YouTube & VoIPBuster).

Consequently, the overall dataset can be separated into two classes according to the situation-type label, 10 classes according to the activity-type label, and 25 classes according to the application-type label.

B. Configuration parameters

For the pre-processing chain, we set the configuration parameter W to 10 seconds in the fragmentation step (see section III-A and section VI). Moreover, to compute packets inter-arrival times related features in the features extraction step, we set the parameter L to 40. This means that for this subset of features, the classifier receives as input a time-window of 10 seconds that is represented as a time series (i.e., segmented into sub-time-windows) of 40 time-steps. Furthermore, each time-step is constituted of the set of features that are described in Section III-C2 and that are computed over the packets that are contained within the $10/40$ seconds related to this time-step.

Similarly, in order to extract features related to packet sizes, we set the parameters P and S to 1500 and 1, respectively. It is important to note that the selected value of P corresponds to the maximum transmission unit (MTU) while the value of S was picked to provide a detailed representation of the distribution related to the sizes of packets that reside within the corresponding fragment.

C. Classifier

1) Multi-label based output: Since our classifier deals with the specific case of multi-activity instances, it has to be able to assign multiple non-mutually exclusive labels to such instances (e.g., the label chat and the label streaming). This type of classification is called as multi-label classification.

Multi-label classification is a classification task that labels each sample with one or multiple labels from a set of possible target labels. Formally, when classifying a sample, a binary output is assigned to each class where positive classes are denoted with 1 and negative classes with 0.

For our classifier, the outputs that indicate the activity type and application type can be modeled leveraging a multi-label output. This means that the activity-type output is represented by harnessing a one-dimensional binary array of size 4 whose elements are dedicated to the following classes: Chat, Mailing, Streaming, and VoIP. Similarly, the application-type output can be represented using a one-dimensional binary array of size 7 whose elements are attributed to the classes: Facebook, Hangouts, Thunderbird, Spotify, YouTube, Skype, and VoIPBuster.

2) Multi-task learning based architecture: The three outputs of our conceived classifier are predicted using three separate tasks. These tasks are trained jointly leveraging the MTL paradigm. One way to apply MTL on a neural network (NN) based model is to utilize the hard parameter sharing technique. This technique is widely used and consists of sharing the hidden layers (i.e., shared representation) among all tasks while keeping additional task-specific layers that are tuned independently. The overall architecture of our MTL model that leverages the hard parameter sharing technique is depicted in Figure 2 and is described in what follows.

For the shared representation layers, to handle the time series that carries packets inter-arrival times related features, we utilized 5 blocks of a transformer's encoder where we kept the default architecture while we set the multi-head attention hyper-parameter number of heads to 4 and the dimension of key to 4. Transformers are known for their ability to capture short-range and long-range dependencies in sequences and time series. Besides, Transformers often showcase superior performance than attention-aided recurrent neural networks. The transformer's encoder was then followed by a layer of one-dimensional global average pooling, a dense layer with 16 units, and a dropout layer with a rate that was set to 0.1.

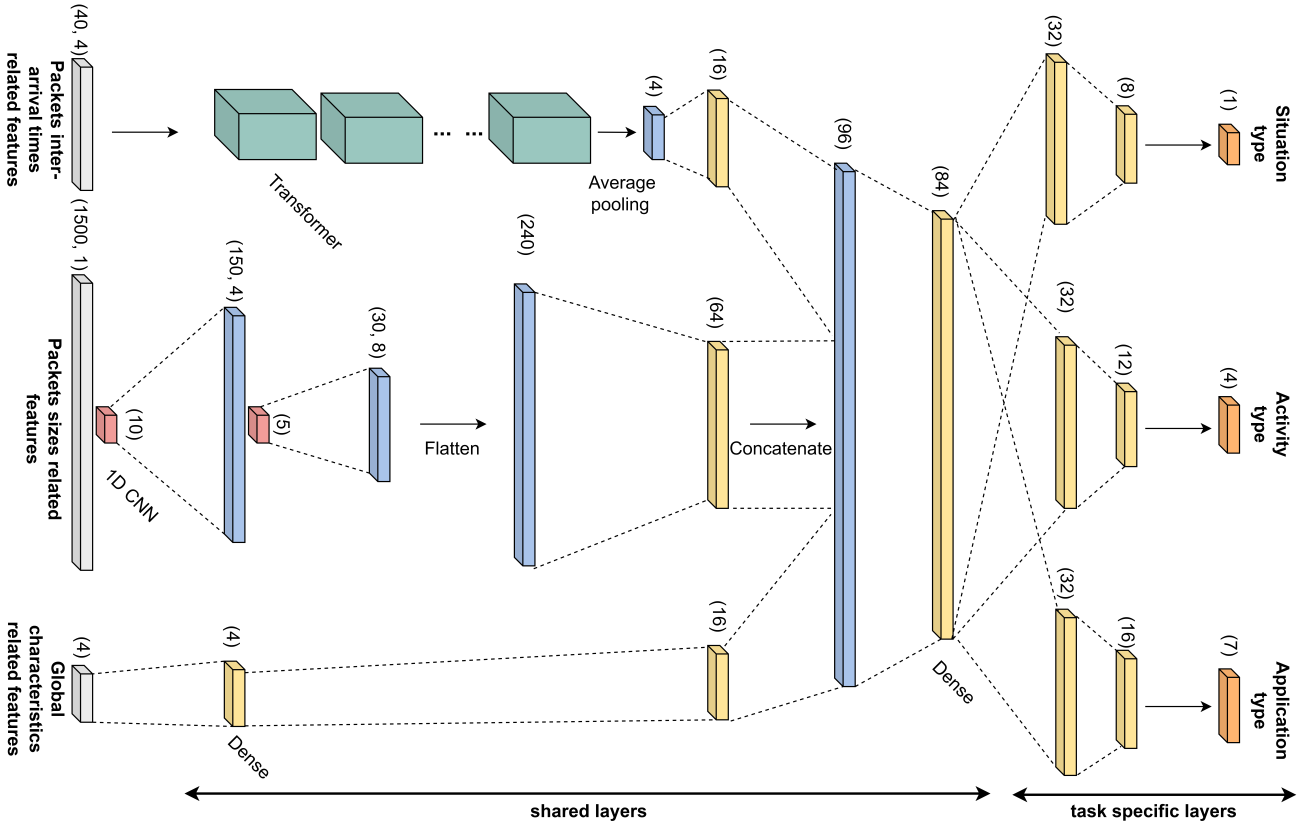

Fig. 2. Architecture of our multi-task learning model

In a similar manner, to treat the histogram that conveys packets sizes related features, we leveraged a succession of two one-dimensional CNN layers. For the first CNN layer, the number of filters was set to 4, while the kernel size was set to 10, and the stride was fixed to 10. For the second CNN layer, the number of filters was set to 8, while the kernel size and stride parameters were set both to 5. One-dimensional CNN are endowed with the ability to capture shift-invariant patterns and spatial dependencies over the targeted input dimension. Besides, using a stack of CNN layers generally creates a hierarchical feature representation in hidden layers, hence allowing the model to learn more intricate and structured patterns. The output of the one-dimensional CNN layers was then flattened and followed by a dense layer with 64 units, and a dropout layer with a rate that was set to 0.1.

Lastly, the set of global characteristics related features are received by a sequence of two dense layers of 4 and 16 units, respectively. These dense layers are then followed by a dropout layer with a rate that was set to 0.1. Dense layers are suited for tabular features such as the set of global characteristics related features. In fact, tabular features are generally mutually independent and don't expose spatial or temporal dependencies.

Posterior to that, the three separate dense layers that carry the representations that are extracted from the three sets of features are concatenated then are fed to a sole dense layer of 84 units. Thus, this last layer aggregates the acquired representations to provide a shared representation to the distinct task-specific layers.

For the task-specific layers, we attributed first a dense layer of 32 units to each of the three tasks. Then, for the situation-type identification task, we utilized a dense layer of 8 units followed by an output layer of 1 unit. For the activity-type identification task, we leveraged a dense layer of 12 units and an output layer of 4 units. Lastly, for the application-type identification task, we harnessed a succession of a dense layer of 16 units and an output layer of 7 units. In addition, it is worth mentioning that we leveraged the ReLU function as an activation function for all dense layers except for the output layers that utilize the Sigmoid function.

In order to prepare the data for the DL models, we used normalization to scale the features to the range $[-1, 1]$. Besides, we leveraged the up-sampling and down-sampling techniques to mitigate the impact of the imbalanced dataset. Finally, to train our MTL model, we attributed an equivalent level of priority to each of the three tasks comprising our model when computing the overall loss function. Additionally, we leveraged the Adam optimization algorithm as an optimizer and used a batch size of 128. Moreover, we used the early stopping technique to avoid over-fitting.

The MTL model was implemented using TensorFlow 2, while the experiments were run on a PC that is equipped with

an Intel $i7$ CPU, 32 GB of RAM, and two GPUs (NVIDIA Quadro RTX 3000, and NVIDIA GeForce RTX 2060).

V. Results

A. Per-task performance assessment

To assess the performance of our classification model, we summarize in Table I the results that are achieved by each of its three constituent tasks. These results are exhibited using the metric weighted F1 score (w-F1) for the three tasks. Additionally, we harness the metrics accuracy score (Acc-sr) and hamming loss (H-loss) as supplementary metrics for the activity-type and application-type identification tasks. These two metrics are suitable to models that utilize a multi-label output. The accuracy score metric computes the ratio of samples where the set of predicted labels strictly matches the true set of labels to the total number of samples. The hamming loss reports the fraction of labels that are incorrectly predicted.

As can be observed from Table I, the reported values for the metric weighted F1 score highlight that the three tasks comprising our model are able to perform their classification tasks with a high degree of confidence. Indeed, the recorded value for w-F1 is above 0.90 for all tasks and even surpasses 0.95 for the situation-type and activity-type identification tasks. Furthermore, the robustness of the activity-type identification task ($2nd$ column) is evident in its accuracy score, which reaches 0.88. This value indicates that the corresponding task is able to predict correctly the whole set of labels for the vast majority of the samples. Furthermore, the recorded value for the hamming loss metric (i.e., 0.04) shows that in most cases where the corresponding task fails to predict the complete set of labels for a given sample, it succeeds in predicting it partially. Similarly, for the application-type identification task ($3rd$ column), the recorded value for the accuracy score (i.e., 0.75) shows that this task is able to predict correctly all the labels for a considerable proportion of the samples. Whereas, the reported hamming loss value (i.e., 0.05) indicates that the corresponding task misclassifies only a small proportion of the labels over the whole set of samples.

B. Comparison with related works

To compare the performance of our solution with related studies from the network traffic classification field, we selected a set of baselines that employed the ISCXVPN2016 dataset to classify network traffic into a set of predefined activities and applications. In this regard, we summarize in Table II the properties related to each baseline. The $2nd$ column indicates the way with which the corresponding baseline segments a network trace to prepare the input for the classification model (i.e., classification unit). The $3rd$ column describes the ML model that is used to perform the classification. The $4th$ column exhibits the number of trainable parameters (Model parameters) that comprise the classification model. The $5th$ and $6th$ columns showcase the output classes and the obtained results (F1 score) for the activity-type and application-type identification tasks. Lastly, the $7th$ column indicates whether the corresponding study is able to deal with multi-activity situations, whereas the $8th$ column indicates whether the classification model employs the MTL paradigm.

As can be observed from the $7th$ column of Table II, the cited baselines consider only the case of single-activity situations. To the best of our knowledge, our current work and its predecessor [4] are the sole studies that target the specific case of multi-activity situations.

Additionally, it can be noticed from the columns showcasing the results regarding both the activity-type and application-type identification tasks (i.e., $5th$ and $6th$ columns of Table II, respectively) that even though our framework addresses a challenging case (i.e., multi-activity situations), it succeeds at attaining comparable results to those reported in [15] and [16] and even surpasses significantly those recorded in [13]. Nonetheless, it is worth mentioning that while the solutions in [15] and [16] exhibit slightly better results than ours, these works rely partially or completely on leveraging packets' payloads as input for their classification models. This type of features implies cumbersome computations during the training and inference phases. Besides, packets' payloads may convey some information (e.g., TLS handshake fields) that bias the performance of the trained model and impact its generalization ability. On the other hand, our model deals with multi-activity traces that comprise bi-directional flows belonging to distinct activities as well as noise flows. This entails a composite network traffic behavior that differs significantly from single-activity traces network traffic behavior. Furthermore, the studies that are cited in [15] and [16] utilize separate models for the activity-type and application-type identification tasks whereas our solution gathers the two tasks in a single model harnessing the multi-task learning paradigm. Lastly, for the application-type identification task, our model considers a larger number of output classes (i.e., 25) compared to the other three baselines.

Finally, regarding the complexity of our classification model (i.e., column 4 of Table II), the reported values show that our model leverages the lowest number of trainable parameters among the three other baselines. This property is desirable for DL models as it implies a reduced training and inference time and more adaptability to constrained environments.

VI. Discussion

In this section, we discuss some limitations regarding our conceived approach and their potential improvements that can be addressed in our future studies.

To create the subset of multi-activity network traces, we applied our proposed multi-activity traces generation process [4] on a set of single-activity traces from the ISCXVPN2016 dataset. Thus, the resulting multi-activity traces can be described as synthetic. Nevertheless, for the sake of assessing the performance of our framework in real environments, it can be of relevant interest to collect multi-activity network traces that are issued from real multi-activity situations.

In our experimental setup, we opted for a value of 10 seconds for the configuration parameter W. This value was chosen heuristically to align with the specifications that we mentioned in Section III-A. Nonetheless, this configuration

TABLE I
PER-TASK PERFORMANCE OF OUR MULTI-TASK LEARNING MODEL

Task	Situation-type identification	Activity-type identification			Application-type identification		
	w-F1	w-F1	Acc-sr	H-loss	w-F1	Acc-sr	H-loss
Results	0.95	0.96	0.88	0.04	0.90	0.75	0.05

TABLE II
COMPARISON WITH SOME RELATED WORKS THAT ARE DEDICATED TO ACTIVITY-TYPE AND APPLICATION-TYPE IDENTIFICATION ON ISCXVPN2016

Ref	Classification unit	Classification model	Model parameters	Activity type		Application type		Multi-activity	MTL
				Classes	F1 score	Classes	F1 score		
[13]	Bidirectional flow	Bi-GRU, 1D-CNN	0.97×10^6	6	0.79	15	0.66	No	**Yes**
[15]	Packet	1D-CNN, S-AE	3.00×10^6	6	0.93	17	**0.98**	No	No
[16]	Bidirectional flow	1D-CNN, ResNet, Bi-GRU	1.29×10^6	7	**0.98**	17	0.96	No	No
SMAR	Time window	Transformer, 1D-CNN	$\mathbf{0.04} \times 10^6$	**10**	0.96	**25**	0.90	**Supported**	**Yes**

parameter could have been selected more elaborately by varying the window size within a range of predefined values and then selecting the one that offers the optimal results.

Lastly, in our current solution, we consider only multi-activity situations that are constituted of two concurrent activities. This conceptual choice was adopted to alleviate the computational burden induced by the multi-activity traces generation process. Nevertheless, it is relevant to state that even though our network traces dataset involves only dual-activity traces, our classifier is adapted to handle multi-activity traces of higher order owing to its use of multi-label outputs.

VII. CONCLUSION

In this paper, we presented a methodology called SMAR that can cope with a challenging scenario that comprises both single-activity and multi-activity situations. The proposed methodology pre-processes a network trace over a time-window and then determines to which situation type it belongs. Furthermore, it identifies the type of the activities being performed and the applications being used. The exhibited methodology may endow researchers with insights about conceiving network traffic classification solutions that are adapted to the emerging changes in users' consumption patterns.

To assess the performance of our solution, we conducted a set of experiments revealing that our classification model is able to attain a satisfactory level of performance for its three constituent tasks. Additionally, comparing our work against a set of notable studies from the state of the art revealed that our instantiated framework is able to achieve comparable results to those studies even if they addressed a less challenging scenario that involves only single-activity situations.

For our future research paths, it can be of relevant interest to assess the performance of our solution on datasets that are collected from distinct environments. This may help us inspect the sensibility of our classification model to the variation of network configuration parameters. Lastly, it may be of clear interest to investigate the interpretability of our model's predictions leveraging techniques of explainable AI field.

REFERENCES

[1] O. Salman, I. H. Elhajj, A. Kayssi, and A. Chehab, "A review on machine learning–based approaches for internet traffic classification," *Annals of Telecommunications*, vol. 75, pp. 673–710, 2020.

[2] E. Ophir, C. Nass, and A. D. Wagner, "Cognitive control in media multitaskers," *Proceedings of the National Academy of Sciences*, vol. 106, no. 37, pp. 15583–15587, 2009.

[3] E. Beuckels, G. Ye, L. Hudders, and V. Cauberghe, "Media multitasking: A bibliometric approach and literature review," *Frontiers in psychology*, vol. 12, 2021.

[4] A. Boumhand, K. Singh, Y. Hadjadj-Aoul, M. Liewig, and C. Viho, "Network traffic classification for detecting multi-activity situations," in *2023 IEEE Symposium on Computers and Communications (ISCC)*, pp. 681–687, IEEE, 2023.

[5] V. Labayen, E. Magaña, D. Morató, and M. Izal, "Online classification of user activities using machine learning on network traffic," *Computer Networks*, vol. 181, 2020.

[6] H. Yao, C. Liu, P. Zhang, S. Wu, C. Jiang, and S. Yu, "Identification of encrypted traffic through attention mechanism based long short term memory," *IEEE Transactions on Big Data*, vol. 8, no. 1, pp. 241–252, 2019.

[7] Z. Zou, J. Ge, H. Zheng, Y. Wu, C. Han, and Z. Yao, "Encrypted traffic classification with a convolutional long short-term memory neural network," in *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pp. 329–334, IEEE, 2018.

[8] C. Xiang, Q. Chen, M. Xue, and H. Zhu, "Appclassifier: automated app inference on encrypted traffic via meta data analysis," in *2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–7, 2018.

[9] J. Cheng, R. He, E. Yuepeng, Y. Wu, J. You, and T. Li, "Real-time encrypted traffic classification via lightweight neural networks," in *GLOBECOM IEEE Global Communications Conference*, pp. 1–6, 2020.

[10] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapè, "Mimetic: Mobile encrypted traffic classification using multimodal deep learning," *Computer networks*, vol. 165, 2019.

[11] S. Rezaei and X. Liu, "Multitask learning for network traffic classification," in *2020 29th International Conference on Computer Communications and Networks (ICCCN)*, pp. 1–9, IEEE, 2020.

[12] A. Rago, G. Piro, G. Boggia, and P. Dini, "Multi-task learning at the mobile edge: An effective way to combine traffic classification and prediction," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 9, pp. 10362–10374, 2020.

[13] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Distiller: Encrypted traffic classification via multimodal multitask deep learning," *Journal of Network and Computer Applications*, vol. 183, 2021.

[14] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related features," in *Proceedings of the 2nd international conference on information systems security and privacy (ICISSP)*, pp. 407–414, 2016.

[15] M. Lotfollahi, M. Jafari Siavoshani, R. Shirali Hossein Zade, and M. Saberian, "Deep packet: A novel approach for encrypted traffic classification using deep learning," *Soft Computing*, vol. 24, no. 3, pp. 1999–2012, 2020.

[16] C. Dong, C. Zhang, Z. Lu, B. Liu, and B. Jiang, "Cetanalytics: Comprehensive effective traffic information analytics for encrypted traffic classification," *Computer Networks*, vol. 176, p. 107258, 2020.