Federated Feature Engineering for Semi-Supervised Classification of QUIC Flows

Saira Bano*†, Achilles Machumilane*†, Lorenzo Valerio‡, Pietro Cassarà†, Alberto Gotta†
*Department of Information Engineering, University of Pisa, Pisa, Italy
‡CNR, Institute of Informatics and Telematics, Pisa, Italy
†CNR, Institute of Information Science and Technologies, Pisa, Italy

Abstract—**Automatic traffic classification is becoming more and more critical in traffic engineering due to the current trend of encrypting transport information (e.g., behind HTTP encrypted tunnels), which prevents intermediate nodes from accessing end-to-end packet headers. However, such information is crucial for traffic shaping, network slicing, and Quality of Service management and for preventing network intrusion or anomalies detection. This paper proposes a federated learning architecture for supporting traffic engineering based on automated traffic clustering and classification. However, this is done in an unsupervised and cooperative manner, such that federated nodes participate in increasing the global knowledge of the network thus enhancing the accuracy in either the prevention of anomalies and intrusions or the set up of new traffic sources and the relative services.**

I. Introduction

Traffic characterization depends on the information observable from the probed client-server data streams. The availability of such information depends on the employed communication protocols and the modality used to archive the communications [1]. Also, data processing techniques play a role in information extraction. When traffic is composed of non-encrypted data flows, it is usually possible to perform a complete analysis. In such a scenario, the packet content can be inspected by accessing the packet header and the payload. In this case, applications, protocols and other information related to the generation of data flows are identifiable, meaning that we can separate the flows concerning the couples of client-server exchanging them.

Access to transport protocol information allows header compression/suppression when capacity is scarce, adapt the protocol behaviour to the characteristics of the network bottlenecks (e.g. HTTP acceleration and split-connections), and implement multi-class per-hop behaviour in the absence of another IP signalling. A solution is urgently needed to avoid losing these benefits when encrypting transport headers [2] or other HTTP tunnels are prevalent. Traditional methods of traffic analysis was based on Deep Packet Inspection (DPI) [3]. The term DPI refers to a set of analysis tools to extract information from the headers and payload and classify the flows. However, with the increasing number of new applications, which no longer have fixed port numbers that can be queried but adopt random port strategies, the accuracy of DPI are gradually declining. Nevertheless, since the advent of the encrypted transport headers, as in QUIC [4], the range of applicability of DPI has been progressively fading out, paving the way for new from blind data extraction. Among other classification methods, Machine Learning (ML) is gaining popularity as a substitute to DPI due to its performance and ease of implementation.

Machine Learning (ML) can be broadly described as the study of algorithms able to identify patterns in data traces without human intervention. The two main features of ML are the ability to improve model estimation through experience (for which a substantial amount of relevant data need to be fed to the algorithm) and the ability to describe a vast number of models, including numerical and categorical data.
We divide the set of ML methods into three families:

Supervised learning refers to a category of algorithms that determine an empirical input-output mapping based on a series of input-output samples. Supervised learning requires a labelled training dataset consisting of pairs of input vectors (called features) and their associated output (called labels), generated by the process under consideration.

Unsupervised learning mainly deals with unlabelled data. An unsupervised learning algorithm aims to discover patterns providing a compact representation of the data space. The most common family of unsupervised learning is clustering. Clustering methods include algorithms that divide the dataset into groups based on a similarity criterion derived from a chosen metric.

Semi-supervised learning. As the name suggests, this class is an intermediate step between supervised and unsupervised learning, which combines a small amount of labelled data with many unlabelled data during the training.

The purpose of Semi-supervised methods is to reduce the cost associated with the acquisition of labelled data by substituting it with unlabelled data (relatively less expensive). The goal is to acquire knowledge on the missing labels by evaluating the similarity of unlabelled data to labelled one. To do so, often semi-supervised methods define tasks of data inspection or optimized manual labelling of the unlabelled data to provide adequate information. Even if the semi-supervised techniques can reduce the computational costs of the learning, the effort for data labelling for the starting training phase can still be heavy, so the semi-supervised techniques can be combined with learning protocols and data processing methods to improve the learning performance in terms of computational and communication loads.

In order to lighten both computational and communication costs due to the semi-supervised techniques, it is possible to use methods of dimensional reduction and federated learning protocols. In the first case, a feature selection can be adopted to reduce computational and communication costs compared to the elaboration of the native dataset; the feature selection can also reduce the overfitting problem. In the second case, the federation allocates part of the computational operations on the end devices, thus reducing the computational load for a central server and the number of control messages exchanged.

In this paper, we provide the following contributions:

- a regression algorithm based on Neural Network suitable for working on constrained devices in a semi-supervised fashions;
- a federated version of the regression algorithm;
- a federated dimensionality reduction procedure based on an information-based feature selection;
- a comparison with some state-of-art feature selection procedures;
- a numerical analysis of the performance obtained by using both centralized and federated versions of the regression algorithm, also considering the feature selection procedure.

The rest of the paper is organized as follows. In Section II, the related works are presented. Section III presents the general methodology and a description of the scenario. The performance evaluation is shown in Section IV. Conclusions in Section V.

II. Related Work

Internet traffic classification using supervised, unsupervised, and semi-supervised learning has been widely investigated in literature [1] and several solutions applied to computer vision and time series analysis have been studied in application to traffic monitoring and analysis. What differs in applying such methods to the traffic analysis is how to extract the features x_i, i.e., the variables that allow outputting the classification y. We can divide the contributions into three classes: *Statistics-based methods*, *Correlation-based methods*, and *Fingerprint-based methods*.

1) Statistics-based methods: they are based on the representation of traffic flows through statistical flow level features, which aim at capturing the characterizing patterns of a traffic flow. These features are, for instance, statistical metrics related to the number of packets transmitted, the number of bytes transmitted, the packet inter arrival time, and the packet size of a flow. In [5], 23 metrics (maximum length (bytes), minimum, mean, median, standard deviation and cumulative length, then inter arrival time (ms), etc.) are applied over five objects (total packets, forward and backward packets, handshake and data transfer packets). In [6], the statistical features are extracted from a vector of packet lengths, taking into account three packet series: incoming, outgoing and bidirectional flows packets. A pool of 9 classifiers is combined to form the MCS architecture. In [7], features are: protocol, source port number, destination port number (that are referred to as classic features), maximum, minimum, mean and variance of the packet size (referred to as statistical features).

2) Correlation-based methods: they are characterized by including knowledge on the correlation among flows, to help perform the classification task. These methods are often based on the so-called Bag-of-Flows (BoF), which are entities that group correlated flows. Correlation among flows is evaluated in literature according to a set of heuristics. A typical example is the 3-tuple identifier, which groups the flows over a link based on the destination IP address, the destination port, and the transport protocol. In [8] a self-learning classifier called SLIC (Self Learning Intelligent Classifier), capable of performing continuous model updates, is proposed. The SLIC scheme consists of two main components: a classifier based on k-Nearest Neighbours (k-NN) and a decision-maker. The classification method is based on the pre-processing of network traffic into BoF, based on the 3-tuple heuristic. Two semi-supervised methods are proposed in [9], [10], to classify the traffic with respect to the communication protocols employed. The role of the previously seen BoF is carried by the equivalence set constraints in [9], [10] that are built using the 3-tuple heuristic and indicate that a set of flows are likely to share the same application layer protocol and hence have to be classified equally.

3) Fingerprint-based methods: they are based on extracting a fingerprint, i.e., a vector or function, possibly a probability density function, capable of summarizing the main characteristics that are common to a specific traffic class. Fingerprinting requires a significant amount of data per class to be identified. In [11] traffic fingerprinting is used to classify IP flows produced by network applications exchanging data through TCP connections such as HTTP, SMTP, SSH, etc.., by looking at client-server or server-client flows. After observing several flows from the same protocol, the gathered statistical information is used to build protocol fingerprints which are used to classify an unknown flow. A vector of Probability Density Functions (PDF) is estimated from a training set of flows generated by the same known protocol, to build the protocol fingerprints. In [12] a method to represent communication patterns in a compact way is proposed. Communications are abstractly considered like sets of messages exchanged, represented as vectors in a multidimensional space. Each set of messages is treated as a set of observations of a random variable with an unknown probability distribution. The joint distribution of features that describe a flow is represented as a single vector of fixed dimension, that is, the fingerprint.

In the last years, the proliferation of encrypted traffic is leading to an increment of *flow-based methods* that rely on the analysis of statistical or time-series features using ML. These include Naive Bayes (NB), Support Vector Machine (SVM), Random Forest (RF), and K-Nearest Neighbours (KNN) [2], [13]–[15]. Reference [16] states that the most burdensome task in building an ML model for traffic classification is data labelling, which requires human intervention, whereas capture of large traces is readily available. Their approach is to use a semi-supervised method, where only a subset of flows needs to be labelled to enable accurate predictions. This approach was fine-tuned in [17] where a deep convolutional generative adversarial network (DGCAN) was used to classify encrypted connections. Their approach provided an accuracy of 89% when only 10% of the dataset was labelled.

In [18], the classifier is able to detect several classes of services with encrypted traffic with good accuracy. While the initial analysis considers 1400 features, it is shown that through a method of features reduction, the significant set can be reduced to only three features.

III. Problem Definition and Methodology

The problem we have faced in this paper concerns estimating the percentage of traffic due to either a given protocol or a class of application within a data stream containing many types of data flows. Since the output of our traffic analysis is a variable with real value and no in a categorical set, i.e. an integer set. Hence, we considered the regression techniques in [19] applied to a scenario of Edge Computing, where the end devices are resource-constrained.

The reference scenario implies that the information to train the regression models is distributed over a set of end devices, which utilize this information to perform a local training of the regression model on the local dataset, i.e., on the traffic captured by each one. Then, a centralized Edge Server performs the merging of the knowledge derived by each end device. To this aim, we have developed a regression algorithm based on a Neural Network (NN) with a single hidden layer, using the Adam optimizer, and its federated version. We also combine the regression algorithm with a feature selection technique that can be used still in a federated manner. Then, we have analyzed the accuracy of the federated regression model compared to the centralized one and how the feature selection affects the computational and communication loads during model training, and its accuracy. In the following sections we provide more details about developed procedures.

A. Regression-based Algorithm

The regression algorithm developed in this paper is based on a Neural Network (NN) with a single hidden layer, using the ADAM optimizer [20]. The ADAM optimization algorithm is a more efficient version of the well-known Stochastic Gradient Descent procedure to update network weights iteratively through training data.

ADAM algorithm is suitable for addressing non-convex optimization problems even when running on constrained devices. The main advantages in using this algorithm lie in: it is straightforward to implement, it is computationally efficient, it can work online, it requires little memory requirements, it is suitable for non-stationary objective functions, even when their gradient is very noisy or sparse. Differently from the Stochastic Gradient Descent, the ADAM algorithm computes individual adaptive learning rates for different parameters from the estimates of first and second moments of the gradients. The algorithm maintains a learning rate for each network weight, which is separately adapted as learning unfolds. Precisely, instead of adapting the parameter learning rates based on the average first moment, Adam also uses the average of the second moments of the gradients. The average used by the algorithm is the exponential moving average of both the gradient and the squared gradient, with the parameters that control the decay rate of the average adapted step by step.
Note that, ADAM algorithm combines the advantages of both algorithms, the Adaptive Gradient Algorithm (AdaGrad) and the Root Mean Square Propagation (RMSProp). The former allows to maintain a per-parameter learning rate improving the performance on problems with sparse gradients, and the last one still maintains per-parameter learning rates that in this algorithm are adapted using the average of current magnitudes of the gradients for the weight.
The developed federated procedure to estimate the percentage of traffic due to either a given protocol or a class of application within a data stream containing many types of data flows, through the resolution of a regression problem, is discussed in the following.

$\mathbf{Step_0}$ The Edge Server performs the supervised learning of the regression model (hard-labelling), using a pre-elaborated dataset, then this model (global model) is propagated toward the end devices;
$\mathbf{Step_1}$ Each end device acquires local data and performs data labelling (soft-labelling), exploiting the global model;
$\mathbf{Step_2}$ Each end device uses its soft labels to perform the feature selection, then the scheme of feature selection is transmitted toward the Edge Server;
$\mathbf{Step_3}$ The Edge Server federates the feature selection schemes and propagates back to the end devices the federated feature selection scheme;
$\mathbf{Step_4}$ Each end device trains its local regression models using the restricted set of features evaluated through the federated scheme. The restricted trained regression model is transmitted toward the Edge Server;
$\mathbf{Step_5}$ The Edge Server averages the received restricted regression models and merges this averaged model with the global regression model available until that time, the Edge Server propagate the new global regression model toward the end devices.

Hence, a new round of local learning of the regression model and a new local feature selection can be achieved starting from the $\mathbf{Step_1}$. The following section explains the procedure to achieve the feature selection and federation of the selection schemes available at the end devices.

B. Feature Selection Algorithm

The Feature Selection (FS) algorithm used in this paper has been derived in both its centralised and federated versions from [21], and it can be formally stated as follows:

Definition (FS Problem)**.** *Given the input data matrix* $\mathbf{X}$ *composed by* n *samples of* m *features (*$\mathbf{X} \in \mathbb{R}^{n\times m}$*), and* *the target attributes' (or labels) vector $\mathbf{y} \in \mathbb{R}^n$, the feature selection problem is to find a k-dimensional subset $\mathbf{K} \subseteq \mathbf{X}$ with $k \leq m$, by which we can characterize $\mathbf{y}$.*

The algorithm used in this work is based on the measure of Mutual Information (MI) that measures the amount of information obtained about the class attribute through the set of selected features. The MI is related to the entropy $\mathbf{H}(\cdot)$ that measures the uncertain of a random variable, as shown in the following equation [22], [23].

$$\mathbf{I}(\mathbf{U};\mathbf{y}) = \mathbf{H}(\mathbf{y}) - \mathbf{H}(\mathbf{y}|\mathbf{U}), \tag{1}$$

In the previous equation $\mathbf{U} = \{\mathbf{x}_1 \cdots \mathbf{x}_k \mid k \leq m\} \subseteq \mathbf{X}$ is the subset of selected features, and $\mathbf{H}(\mathbf{y}|\mathbf{U})$ is the conditional entropy that measures the amount of information needed to describe $\mathbf{y}$, conditioned by the information carried by $\mathbf{U}$. In brief, the $\mathbf{I}(\mathbf{U};\mathbf{y})$ represents the dependence between $\mathbf{U}$ and $\mathbf{y}$, i.e., the greatest is the value $\mathbf{I}$, the greatest is the information carried by $\mathbf{U}$ on $\mathbf{y}$. The features selected in $\mathbf{U}$ called Essential Attributes (EAs), provide the maximum value for the equation (1).

In [22], [23] authors prove that the solution for the faced feature selection problem can be found solving the optimization problem in the equation (2),

$$\arg\max_{\mathbf{U}} \mathbf{I}(\mathbf{U};\mathbf{y}) \tag{2}$$
$$\mathbf{U} = \{\mathbf{x}_1 \cdots \mathbf{x}_k \mid k \leq m\} \subseteq \mathbf{X}$$

assuming the features independent among them, authors also provide the incremental version of the previous algorithm, shown in the equation (3).

$$\arg\max_{\mathbf{x}_j \in \mathbf{X} \setminus \mathbf{U}} \mathbf{I}(\mathbf{x}_j;\mathbf{y}|\mathbf{U}), \tag{3}$$
$$\mathbf{U} = \{\mathbf{x}_1 \cdots \mathbf{x}_{k-1} \mid k \leq m\} \subseteq \mathbf{X}.$$

With the Cross-Entropy-based [24] feature selection algorithm used in this paper, we provide the solution for the algorithm in equation (2), making negligible the assumption of the independence among the features. This means that, instead of selecting one EA at a time, we can select a set of EAs jointly. The solution provided by the Cross-Entropy-based algorithm is the distribution probability $\mathbf{p} = [p_1, \cdots p_m]$ of selecting features in $\mathbf{U}$ to get the maximum for the problem in the equation (2).

The Federated Feature Selection (FFS) procedure adopted in this paper exploits the Bayesian's theorem to merge the local distribution probability $\mathbf{p}$ into the global one. Formally, we assume that each node acquires a number of *i.i.d.* records n^l to address the optimization problem (2), and that the nodes share the same set of features $\mathbf{X}$. The global probability $\mathbf{p}^{\mathbf{G}}$ used for the FS can be written as follows:

$$\mathbf{p}^G = \sum_l \mathbf{p}^l q^l, \tag{4}$$

where $\mathbf{p}^{(l)}$ is the probability distribution of selecting the features at the node l, and $q^{(l)}$ is the weight of this distribution probability. As shown in equation (5),the weight is proportionally to the size of its local dataset compared to the whole amount of data present in the system. In this way, we can contrast situations where local datasets are heterogeneous w.r.t. the size.

$$q^l = \frac{n^l}{\sum_l n^l} \tag{5}$$

In the next section we provide the perform analysis of the procedure discussed in this section.

IV. Numerical Analysis

The dataset used to analyze the performance of the developed semi-supervised procedure is the QUIC Dataset [25]. The dataset contains flow packets based on QUIC and non-QUIC traffic, the latter generated from five Google services: Google Drive, Google Docs, Google Music, Google Search and Youtube. Authors have used several configuration systems for generating and acquiring data based on operating systems such as Windows 7, 8, 10, Ubuntu 16.4. Authors also developed several scripts using Selenium WebDriver and AutoIt tools to mimic human behaviour when capturing data. This approach allowed them to capture a dataset with more than $20 \cdot 10^6$ records without significant human effort. The data are recorded in many directories, one for each service, containing many files deriving from different days of acquisition. The records in the file contain three fields: Unix timestamp, the acquisition time of the record measured from the first one acquired and the packet length.

We developed Python-based scripts to merge all the files synchronizing them using the Unix timestamps. Hence, we obtained a single file containing all the acquired records for QUIC and non-QUIC traffic for the five classes of service. We also developed Python-based scripts to perform the feature extraction, which provides the following features: number of QUIC packets within a sampling window, 25-th, 50-th, 75-th and 90-th percentiles of inter-arrival time of packets, 25-th, 50-th, 75-th and 90-th percentile of packet sizes. The sampling window chosen to evaluate the statistics stated above is set to 1 second. For each record of extracted features, we have evaluated two kinds of class labels: the first one called *Class Label QUIC* is the percentage of QUIC based traffic within the considered sampling window, the second one called *Class Label Service* is the percentage of traffic due to a class service within the considered sampling window. We have partitioned the dataset of extracted features into two sets: the first containing the 20% of the samples used to perform the supervised training of the regression model as stated in the $\mathbf{Step_0}$ of the procedure. The second contains the 80% of the samples that have been distributed over a set of 10 end devices to perform the local unsupervised training of the regression model as stated in the procedure from $\mathbf{Step_1}$ to $\mathbf{Step_5}$.

We have analyzed the performance of the proposed Cross-Entropy-based (CE) feature selection algorithm over the features extracted from the QUIC Dataset. We compared CE's performance in detecting the quota of QUIC traffic over non

QUIC traffic in a sample of traffic dump to those obtained using other information-based feature selection algorithms such as mRMR [26], CMIM [27], and DSR [28], which provide the solution for the optimization problem in equation (3). We have also compared the Cross-Entropy-based algorithm with the Analysis of Variance technique. This technique, known as ANOVA [29], determines the variance of all the features and divide them into systematic and random factors, where random factors have no impact on the learning as the variance of these features are zero. The higher the F-score, the higher the variance between the means of the two populations. We assume that features with zero variance add no information by considering the relation between the target variable and feature vectors. In our case, ANOVA is employed to compare the means of the features and determine a set of features that efficiently contribute to the traffic classification.

Table I shows the results of the feature selection scheme obtained with the methods discussed above. Most feature

		Features Selection Scheme				
Feature		CE	ANOVA	CMIM	DISR	mRMR
1)	$\overline{N}$	0	0	0	1	1
2)	ΔT_{25-th}	0	1	0	0	1
3)	ΔT_{50-th}	1	0	1	1	0
4)	ΔT_{75-th}	0	1	0	0	0
5)	ΔT_{90-th}	0	0	0	0	0
6)	Ln_{25-th}	1	0	1	0	0
7)	Ln_{50-th}	1	1	1	1	1
8)	Ln_{75-th}	1	1	1	1	1
9)	Ln_{90-th}	1	1	1	1	1

TABLE I: Extracted Features Set

selection methods provide a ranking of the analyzed features. Instead, our method provides automatically the minimal set of features, which is 5 in this case, guaranteeing the best estimation for the regression model. For this reason, to obtain the minimum set of features by the other methods, we select the first five features of ranking provided by them. Another advantage provided by the Cross-Entropy-based algorithm is that we can implement a federated version of it. Instead of the others methods, the federated version does not exist to the best of our knowledge. Table II shows the comparison among the performance obtained with the regression model for identifying the percentage of QUIC traffic in the received flow adopting the subsets of features selected with the various methods discussed previously. The Cross-Entropy and CMIM algorithms allow a performance degradation in the regression model's performance of just under 1% *w.r.t.* no feature selection; instead, we obtain a degradation close to 5% with the ANOVA, and close to 10% with both DSR and mRMR, compared to the performance obtained with the whole set of features. Note that while achieving the same performance with Cross-Entropy-based and CMIM algorithms, the former allows automatic calculation of the minimum number of features and can also be implemented in a federated version. In Table (III), we provide the performance analysis in terms of control traffic (mesured in MBs), when both FS and the regression model are federated. Precisely we compare the performance of the

Method		RMSE	Selected Feat. #
No Feature selection	–	0.0121	9 (All)
Centralised FS	CE	0.0122	5
	ANOVA	0.0127	top 5
	CMIM	0.0123	top 5
	DSR	0.0133	top 5
	mRMR	0.0133	top 5

TABLE II: Supervised Feature selection. Performance of regression model trained using the features selected by each method.

procedure used to train the regression model, when this is achieved without using model federation and feature selection or Centralized Regression (CR), using the federation of the models or Federated Regression (FR), and using the federation of the models trained over the selected set of features or Restricted Federated Regression (RFR). The values in the column

Procedure	Avrg. Traffic (Fed./F.S.)	Conv. Rounds (Fed./F.S)	RMSE
CR	5.8	-	0.0122
FR	0.44/-	8/-	0.0152
RFR	0.315/0.108	10/12	0.0219

TABLE III: Performance comparison among the regression procedures: centralized, federated and restricted federated.

Avrg. Traffic show the volume of control traffic, generated by all the nodes during the learning procedure to train the regression model. The Centralized Regression generates the highest load because in each learning round all the nodes use a regression model estimate using a Neural Network with input layer based on all the available data records. The Federated Regression involve lighter regression models than the previous one because the input layer of the Neural Network is based on a smaller dataset, that is, the dataset available at the node. In this case, all the control traffic is due to the federated procedure. Finally, the Restricted Federated Regression procedure generates the minimum control traffic, since it uses the local dataset containing only the records of the selected features. Note that in this last case, 0.108 MB of the 0.315 MB control traffic are due to the federated feature selection procedure that involves data exchanging between the Edge Server and end devices.

The values in the column *Conv. Rounds* show the number of communication rounds required by the federated procedure to merge the local models into the global one. We assume that the convergence for the global model is reached when the difference between the weights of two consecutive learning rounds is in average lesser than 1%. For the Restricted Federated Regression procedure the average number of communication rounds required to converge is also shown during a learning round.

Finally, the values in the column *RMSE* show the values of the RMSE evaluated between the ground truth and the trained regression model trained using the procedures discussed until now. Obviously the minimum RMSE is obtained when using the Centralized Regression because we use whole dataset to fed the Neural Network. Reducing the amount of information used as input in the Neural Network the RMSE increase. Note that, the RMSE's degradation is less fast than the growth of the control traffic, for example the RMSE of the Restricted Federated Regression is $\sim 1.8\times$ the RMSE of the Centralized Regression, but the control traffic of the Centralized Regression is more than $50\times$ that of the Restricted Federated Regression control traffic.

V. Conclusion

In this paper, we have developed a procedure to capture the quote of QUIC traffic within a data flow using a regression model trained with a semi-supervised technique. We have also tested the effect of the federation of the regression models trained locally by a set of nodes. Finally, we have jointly adopted a novel feature selection technique with the federation to reduce communication and computational cost. Based on the Cross-Entropy method, the feature selection algorithm maximizes the mutual information of the selected features and class attributes, i.e., the quote of QUIC traffic. The proposed Cross-Entropy algorithm was utilized and compared in a centralized manner and a federated distributed one. Selected features provide the same performance of the whole set with a negligible RMSE error while outperforming the other baselines of the 5-10% in the centralized/supervised approach. In addition, Cross-Entropy was also tested in a federated manner and with a semi-supervised approach with local soft labelling. Such a distributed approach has significantly reduced the traffic overhead required to train the regression model, but at the expense of a higher RMSE, higher than the centralized baseline, but preserving the number of communication rounds needed to federate the models.

Acknowledgment

This work has been partially supported by ESA SatNEx V project contract n. 4000130962/20/NL/NL/FE, and by TEACHING H2020 projects (GA #871385).

References

[1] F. Pacheco, E. Exposito, M. Gineste, C. Baudoin, and J. Aguilar, "Towards the deployment of machine learning solutions in network traffic classification: A systematic survey," *IEEE Communications Surveys Tutorials*, vol. 21, no. 2, pp. 1988–2014, 2019.

[2] A. Dainotti, A. Pescape, and K. C. Claffy, "Issues and future directions in traffic classification," *IEEE Network*, vol. 26, no. 1, pp. 35–40, 2012.

[3] T. Bujlow, V. Carela-Español, and P. Barlet-Ros, "Independent comparison of popular dpi tools for traffic classification," *Computer Networks*, vol. 76, pp. 75–89, 2015.

[4] J. Iyengar and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport," RFC 9000, May 2021.

[5] Y. Zhang, S. Zhao, J. Zhang, X. Ma, and F. Huang, "Stnn: A novel tls/ssl encrypted traffic classification system based on stereo transform neural network," in *2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 2019, pp. 907–910.

[6] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Multi-classification approaches for classifying mobile app traffic," *Journal of Network and Computer Applications*, vol. 103, pp. 131–145, 2018.

[7] D. Tong, Y. R. Qu, and V. K. Prasanna, "Accelerating decision tree based traffic classification on fpga and multicore platforms," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 11, pp. 3046–3059, 2017.

[8] D. M. Divakaran, L. Su, Y. S. Liau, and V. L. Thing, "Slic: Self-learning intelligent classifier for network traffic," *Computer Networks*, vol. 91, pp. 283–297, 2015.

[9] Y. Wang, Y. Xiang, J. Zhang, W. Zhou, G. Wei, and L. T. Yang, "Internet traffic classification using constrained clustering," *IEEE transactions on parallel and distributed systems*, vol. 25, no. 11, pp. 2932–2943, 2013.

[10] Y. Wang, Y. Xiang, J. Zhang, and S. Yu, "A novel semi-supervised approach for network traffic clustering," in *2011 5th International Conference on Network and System Security*. IEEE, 2011, pp. 169–175.

[11] M. Crotti, M. Dusi, F. Gringoli, and L. Salgarelli, "Traffic classification through simple statistical fingerprinting," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 1, pp. 5–16, 2007.

[12] J. Kohout and T. Pevnỳ, "Network traffic fingerprinting based on approximated kernel two-sample test," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 3, pp. 788–801, 2017.

[13] F. Pacheco, E. Exposito, and M. Gineste, "A framework to classify heterogeneous internet traffic with machine learning and deep learning techniques for satellite communications," *Computer Networks*, vol. 173, pp. 107–213, 2020.

[14] G.-L. Sun, Y. Xue, Y. Dong, D. Wang, and C. Li, "An Novel Hybrid Method for Effectively Classifying Encrypted Traffic," in *In proc. of GLOBECOM 2010*, 2010, pp. 1–5.

[15] P. Velan, M. Čermák, P. Čeleda, and M. Drašar, "A survey of methods for encrypted traffic classification and analysis," *Netw.*, vol. 25, no. 5, p. 355–374, Sep. 2015.

[16] S. Rezaei and X. Liu, "How to Achieve High Classification Accuracy with Just a Few Labels: A Semi-supervised Approach Using Sampled Packets," in *in proc. of ICDM 2019*, 2019.

[17] A. S. Iliyasu and H. Deng, "Semi-supervised encrypted traffic classification with deep convolutional generative adversarial networks," *IEEE Access*, vol. 8, pp. 118–126, 2020.

[18] V. Tong, H. A. Tran, S. Souihi, and A. Mellouk, "A novel quic traffic classifier based on convolutional neural networks," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.

[19] M. Fernández-Delgado, M. Sirsat, E. Cernadas, S. Alawadi, S. Barro, and M. Febrero-Bande, "An extensive experimental survey of regression methods," *Neural Networks*, vol. 111, pp. 11–34, 2019.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[21] P. Cassarà, A. Gotta, and L. Valerio, "Federated feature selection for cyber-physical systems of systems," *arXiv preprint arXiv:2109.11323*, 2021.

[22] R. McEliece, *In The Theory of Information and Coding: A Mathematical Framework for Communication*, ser. Encyclopedia of Mathematics and Its Applications. MA, USA: Addison-Wesley Publishing Company: Reading, 1977, vol. Vol. 3.

[23] J. Cover, T.M. Thomas, *Elements of Information Theory*. New York, NY, USA: John Wiley and Sons, Inc, 1991.

[24] D. Rubinstein, R.Y. Kroese, *The Cross-Entropy Method: a Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, Machine Learning*. Springer, 2004.

[25] R. Shahbaz and L. Xin, "Dataset QUIC," http://https://drive.google.com/drive/folders/1Pvev0hJ82usPh6dWDlz7Lv8L6h3JpWhE, 2018, [Online; accessed 30-Oct-2018].

[26] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.

[27] P. E. Meyer, C. Schretter, and G. Bontempi, "Information-theoretic feature selection in microarray data using variable complementarity," *IEEE Journal of Selected Topics in Signal Processing*, vol. 2, no. 3, pp. 261–274, 2008.

[28] F. Fleuret, "Fast binary feature selection with conditional mutual information." *Journal of Machine learning research*, vol. 5, no. 9, 2004.

[29] R. M. Heiberger and E. Neuwirth, "One-way anova," in *R through excel*. Springer, 2009, pp. 165–191.