Federated Representation Learning for Encrypted Application Type Classification in beyond 5G RAN

Sid Ali Hamideche, Marie Line Alberi Morel, Kamal Singh, César Viho

Federated Representation Learning for Encrypted Application Type Classification in beyond 5G RAN

Sid Ali Hamideche
Univ Rennes, IRISA
Rennes, France
sid-ali.hamideche@irisa.fr

Marie Line Alberi Morel
Nokia Paris Saclay
Massy, France
marie_line.alberi-morel@nokia.com

Kamal Singh
Laboratoire Hubert Curien
Saint-Etienne, France
kamal.singh@univ-st-etienne.fr

César Viho
Univ Rennes, IRISA
Rennes, France
cesar.viho@irisa.fr

***Abstract*—Mobile application classification is essential for advanced network management and application-based QoS policy enforcement in future, AI-enhanced, beyond 5G and 6G mobile networks. This article proposes to use AI methods to categorize applications as functional types (e.g., Video, Audio, Browsing) despite encryption and limited labeled data. We tackle these challenges through unsupervised representation learning, which maximizes the use of abundant unlabeled data in mobile networks. Due to the distributed nature of beyond 5G and 6G networks, we use this method in federated learning scenarios and compare it to the centralized ones. Our findings highlight that unsupervised learning improves model performance, especially with scarce labeled data. Additionally, federated learning provides effective results as compared to centralized methods.**

***Index Terms*—Application classification, Beyond 5G, 6G, Deep learning, Representation learning, Federated learning**

I. Introduction

The transition to advanced 5G networks, has brought major changes in how we live, work, and connect. With more devices and users seeking high-speed and low-latency connectivity, future generations need advanced networks beyond 5G and 6G. Users also expect new services including multimedia offering like high resolution video streaming and low latency gaming. To address these challenges, AI integration in mobile networks has become essential. AI can provide optimal management and scaling based on users' needs and habits, such as the type of applications being used. Concrete scenarios that exploit user application knowledge to improve network performance include resource allocation and migration, traffic shaping, compression, prioritization, and congestion control, QoS control and enforcement among others. Consequently, detecting mobile applications is important. Application classification methods include: port-based, payload-based (e.g, Deep Packet Inspection), statistical-based, and deep learning methods. However modern networks introduce challenges like encryption and dynamic port changes that limit the performance of the first two methods. Furthermore the statistical methods require manual feature extraction which is time-consuming and requires human expertise. In response, recently, deep machine learning algorithms are increasingly used to identify applications through network data analysis. In this work, we focus on deep learning methods for application classification.

Supervised deep learning models have gained significant attention in recent years due to their ability to automatically learn complex patterns from large datasets. This makes them well-suited for traffic classification tasks, where it is challenging to manually extract relevant features, especially with encrypted traffic. However, the major constraints associated with supervised deep learning models is the manual data labeling. It is expensive and complex, which can limit the scalability and efficiency of these models in real-world applications. As a result, techniques that leverage unlabeled data have become more prominent. One such approach is unsupervised learning which does not require labeled data at all. However, these methods often under-perform when compared directly with supervised learning models. Another type of technique that has gained attention is semi-supervised learning which combines the benefits of both supervised and unsupervised learning. They can leverage both labeled and unlabeled data, making them particularly useful in situations where a limited amount of labeled data is available. Recent research in semi-supervised learning involves unsupervised representation learning as a pre-training step which can use large amounts of unlabeled data which is easy to acquire. This approach is specifically designed to learn patterns and extract high-level, generalizable features that can be used for a variety of tasks. These learned representations are then combined with a fine-tuning step using labeled data for the specific task at hand, referred to as the downstream task. These techniques have demonstrated success in various fields, including natural language processing [1], [2] and computer vision [3] while only few works have explored those in the context of application classification [4], [5].

Another challenge facing data-driven systems is data centralization, which requires significant bandwidth to transfer data to a central server where it is processed and models trained. To overcome this issue, federated learning has emerged as a solution. Unlike centralized methods, it involves training the model on the edge using local data rather than sending data to a central server. Only models, which are significantly smaller than data, are sent the server, where they are aggregated to represent the learned knowledge from all distributed entities. This approach can capitalize on the inherently distributed nature of beyond 5G and 6G networks Future wireless networks are built using modern architectures such as O-RAN (Open Radio Access Network) that are massively distributed systems and designed to include AI in mind with components such as RIC (Radio Intelligent Controller).

Building on this, in our work, we also investigate similar supervised deep learning approaches as those studied in [6]–[9] to distinguish between various types of mobile application services. This idea aligns with the QoS Class Identifier (QCI)

categorization used by 3GPP (Third Generation Partnership Project). We use these approaches as baselines to differentiate popular services between buffered video or audio (download), non-real-time video, live conversational video calls and voice calls, and browsing activities. To go further, our contribution is to introduce semi-supervised and federated learning techniques to improve the performances in the face of limited labels and distributed networks. In addition, our work stands out from most others in its focus on mobile network traffic, rather than PC-generated traffic, thus presenting a more complex scenario.

Key contributions are:

1) **Semi-supervised learning for application type application (SSL):** To explore unlabeled data, we employ semi-supervised learning which combines unsupervised representation learning pre-training with supervised fine-tuning for application type classification. This methodology has shown success in various other fields; however, its application to network traffic [4], [5], particularly mobile network traffic, has been limited. In our work, we utilize Denoising Autoencoder (DAE), for pre-training to improves performances compared to supervised learning approaches [6]–[8], and maintain the performances when there is a limited amount of data.
2) **Federated semi-supervised Learning (FL-SSL):** We adopt federated learning to match the highly distributed nature of next-generation mobile networks, and to avoid drawbacks brought by data centralization, such as additional bandwidth and latency. In our work, we implement federated learning for the unsupervised representation learning step, which does not require labeled data and can be trained locally without human intervention. In contrast to most work on Federated Learning for application classification, our solution is intended to be implemented on the RAN side not on the user device.

This paper is organized as follows. Sec. II presents related work. Sec. III discusses the data. Sec. IV outlines the application classification method. Sec. V presents the performances. Lastly, Sec. VI provides conclusions and perspectives.

II. Related Work

Traffic classification encompasses several tasks, including intrusion detection, protocol detection, and application identification. Application classification can be further divided into other sub-tasks which can involve recognizing the name of an application (e.g., Google, Facebook, or Spotify). Alternatively, it may refer to type determination, also known as application categorization, which consists of identifying the function or purpose of the application (e.g., video, voice, instant messaging). These tasks can be accomplished using similar methods, either traditional or machine learning-based. However, recent years have seen a shift away from traditional methods, such as Deep Packet Inspection (DPI), port-based, and payload-based analysis [10]–[13]. This transition is primarily driven by the prevalence of encryption like SSL/TLS in modern networks, making DPI techniques obsolete.

Machine learning approaches, particularly deep learning methods, have gained significant prominence in recent times [6], [14]–[18]. Deep learning models can automatically learn high-level representations from data, reducing the need for manual feature engineering. These models demonstrate scalability to large datasets and high-dimensional spaces. Additionally, they have shown the ability to identify application-specific patterns and signatures in raw payload content. However, encryption limits what patterns and signatures these models can learn. As a result, some research has focused on using time series of packet statistics instead [9], [19]. Recently, multimodal models that combine both approaches have been proposed [20], offering a promising solution. Another area of study is representation learning, which aims to address the challenges of data labeling. One approach is ET-BERT [4], a version of BERT (NLP) adapted for network traffic pre-training and classification. Similarly, YaTC [5] employs transformer-based Masked Autoencoders (MAE) for network traffic pre-training. Multitask learning is also employed in [21], [22] as a means of reducing the amount of labelled data required. The popularity of federated learning has grown considerably, particularly leveraging the user device for training application classification models. The main motivation is to address privacy and security concerns associated with centralized learning [23]–[25].

Our work uses real-world data from various mobile applications, presenting a challenging labeling task due to the noisy and complex nature of the traffic. In contrast to most application classification studies, our approach does not isolate specific types of data or control for only foreground apps being labeled. Instead, we collect and analyze real-world phone traffic, unlike laboratory-based studies that rely on computer tools to capture traffic in a controlled environment. We also explore representation learning and federated learning for application classification in mobile networks, an area that has not been extensively investigated to our knowledge. This research direction aligns with the decentralized nature of new generation mobile networks, where data is generated and processed at the edge rather than in centralized locations.

III. Data

A. Data processing

The data collected includes pcap files of network traffic for smartphones and corresponding label files captured on the devices. They indicate the time intervals and the type of application being used during that duration (e.g., Video (Download); start: ”Date” 19:31:06; end: “Date” 20:12:48.. This signifies that the user was consuming “Video (Download)” data during this time frame. However, some labeling noise occurs because not all packets within the labeled interval correspond to the application being used. In mobile phones, numerous applications run in the background.

Thus, to minimize the amount of data, particularly in case of data-intensive applications such as video), we compute statistics of packets per second instead of utilizing all the packets. This approach results in a single measurement point

per second. We begin by identifying the flow having the maximal data rate (in bytes) within a one-second period. A flow is uniquely identified by a 5-tuple consisting of [IP source, port source, IP destination, port destination, protocol]. We then proceed to extract features from this identified flow:

- Raw payload content (encrypted) of the largest packet.
- Total number of packets transmitted per second for both uplink and downlink directions.
- Total amount of data (Bytes) transferred per second for each direction: downlink and uplink.
- Mean interarrival (seconds) between successive packets.

Table I presents the pre-processed dataset statistics, showing the total number of data points and data collection campaign duration (the collection wasn't 24h/24h, hence the number of data points is smaller compared to the duration).

B. Application type versus labels

We focus on application classification for identifying service groups. Determining whether an application belongs to Video or Audio is more important than identifying particular names, as one name can be used for multiple service groups. For instance, Facebook can be used for video streaming, audio calls, and text messaging. Its identification is less interesting for a network that wants, for example, to allocate more resources for a video session than an audio session due to its high bandwidth consumption.

For our work, we have selected a comprehensive set of labels that effectively capture a wide range of user activities and are aligned with $3GPP$ service classes. These labels are: **Video (Download)**, **Audio (Download)**, **Browsing**, **Voice Call (live)**, and **Video Call (live)**. Accurately distinguishing between these categories is of paramount importance. For instance, the ability to differentiate between 'Video (Download)' and 'Video Call' is essential. These categories have vastly different requirements and priorities from a network perspective. A call application, known for its sensitivity to latency, requires different network management compared to other Video-type applications. Buffering strategies can be employed in the latter case, and latency is not a major concern. The network might prioritize Voice Calls over other types of traffic to ensure quality and continuity of real-time communication. Figure 1 shows the usage frequency of different application categories, revealing popular types and usage trends in our dataset. It also underscores classes imbalance.

C. Data input relation to application type

Application classification tasks is widely performed using IP packets. IP traffic can be analyzed using various methods, including manual computation of relevant features or direct use of raw payload as input. In our work, we have chosen to employ both methods. Consequently, we utilize multimodal learning, combining two types of inputs (modalities): TCP or UDP payload content and a time series of statistical data.

- **TCP or UDP Payload content (encrypted):** the content of payload. [17], [18] have shown that deep learning models can discern patterns and identify applications, even with encrypted network traffic.
- **Time Series of Packet Statistics:** A sequence of data points of length L seconds of the same flow. This input captures temporal signatures and application patterns. This is also less affected by encryption.

To examine the connection between application types and network traffic statistics, we conduct an in-depth visual analysis. Fig. 2b shows cumulative distribution functions for downlink and uplink data, revealing that "Video-based" applications produce larger amounts of data, while "Call-based" applications produce almost identical amounts of data in both directions. Fig. 2a displays variations in total data sizes (bytes/second) for targeted application classes over a 30-second time frame. This figure shows that real-time applications produce a constant amount of data throughout the time frame, as content is generated continuously rather than buffered. However, browsing applications are more complex and diverse. They can generate small text-based data or large media, resulting in traffic patterns similar to video or audio.

TABLE I: Data collection description

Total number of data points : 113951	**Period :** ~ 119 $days$
Labeled: 40666 and Unlabeled: 73285	Not continuously

IV. Application type classification

A. Classification in O-RAN

Application type classification in RAN is a key problem. Placing the classifier within the RAN directly benefits the management function, avoiding intricate cross-layer interactions that would occur if implemented in user equipment or the core network. This approach prevents additional delays detrimental to real-time low latency applications (uRLLC), as well as enhancing resource allocation, L2 packet scheduling, and traffic shaping for other applications (eMBB) with specific Quality of Service (QoS) requirements.

In an O-RAN network, traffic data can be accessed at the Protocol Data Unit (PDCP), which is located either at the Central Unit (CU) or Distributed Unit (DU), depending on the split option being used. The training itself can be implemented in either a near-Real-Time or non-real-time RIC. Furthermore,

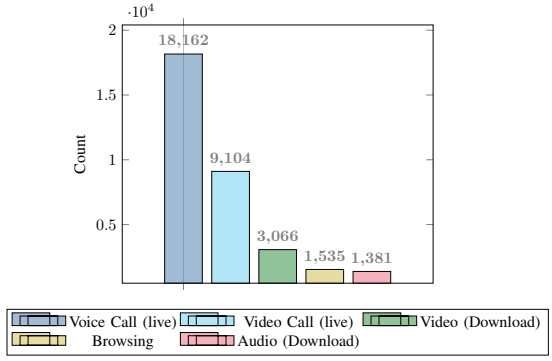
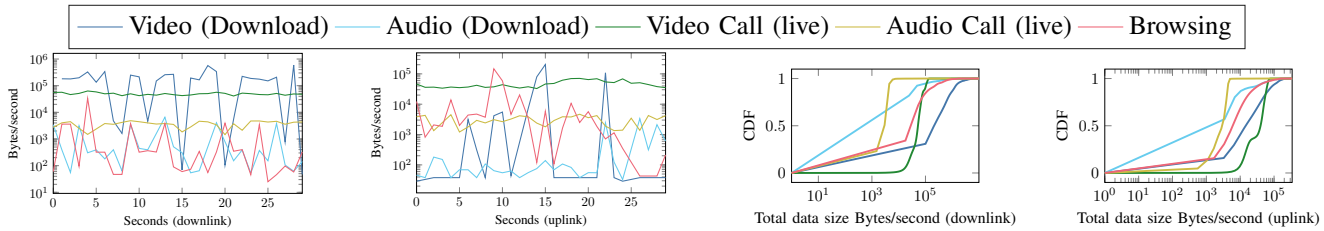


Fig. 1: Data distribution: Number of data points for each class.



(a) Evolution of data size (Bytes/second) by in 30-seconds time window (b) Cumulative distribution of total data size (Bytes/second)

Fig. 2: Data size for different application categorizations : (a) time evolution and (b) cumulative probability distribution

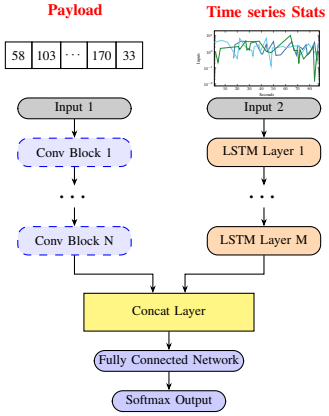


Fig. 3: Multimodal architecture for application classification.

the distributed architecture of O-RAN can also be leveraged for optimal application classification, as will be discussed later.

B. Multimodal deep learning for classification

To effectively utilize the two aforementioned input types, we adopted a multimodal architecture. This architecture combines Convolutional Neural Network (CNN) layers for extracting features from raw payload content and Long Short-Term Memory (LSTM) networks for extracting features from time-series packets statistics. The extracted features from each input are concatenated. Then, these features are passed through fully connected layers, followed by a softmax layer representing the output mobile user application class. Fig. 3 illustrates the proposed architecture of the two-branch model. Each branch is trained traditionally like the monomodal training.

C. Unsupervised representation learning

To leverage unlabeled data, we employ representation learning to pre-train the first layers of the models. Specifically, we train the feature extraction layers from each input of the multimodal model. For every input, we train a denoising autoencoder (DAE) with an encoder that shares similar parameters as the feature extraction layers (2 DAEs since we have two modalities). This technique is useful for learning patterns and meaningful representations without label [26], [27]. Once trained, the encoder part of each input is used to initialize the feature extraction layers, followed by supervised application classification, a process called transfer learning. Here, we have a source task (DAE) and a downstream task: application classification. This approach is also a type of semi-supervised learning as it uses both unlabeled data for pre-training and labeled data for fine-tuning.

D. Federated representation learning

As mobile networks are massively distributed and machine learning models require large amounts of data for training, classical approaches suggest gathering data in a centralized location as seen in Fig. 4a. However, this approach has several drawbacks. Data must be transmitted from their locations to a central server and stored there, for example, from CU/DU to RIC in O-RAN. This requires a significant amount of network bandwidth, especially for eMBB applications that already demand large bandwidth, and necessitates a powerful server to process all the incoming data. In addition to other concerns that large bandwidth can bring, such as cost and carbon footprint, having user data stored in a single centralized server poses privacy risks. Consequently, federated learning has emerged as a modern solution in mobile network research.

This technique allows multiple models to train locally without data transmission, transferring only learned knowledge like weights or deep learning model parameters to the server. The server aggregates these parameters and returns updated model. In our study, we use this method for unsupervised representation learning to be able to exploit large unlabeled data in distributed entities. For supervised training, we minimize labeled data usage and assume that relatively small amount of it is present on the server. We categorize data based on unique cell IDs to mimic real-life scenarios where each cell (representing CU/DU in O-RAN) has distinct data and entities.

As illustrated in Figure 4b, we showcase an implementation of federated representation learning for application classification in O-RAN networks. We have multiple edge units (CU/DUs) acting as distributed clients where local models are trained, and a near-RT RIC as the server to aggregate the models for federated learning. Fine-tuning for application type classification is also carried out at the RIC. Figure 4a illustrates a centralized approach where data is instead sent to the near-RT RIC for unsupervised pre-training.

V. Experiments

This section describes the configuration setup and then presents the experiment results. We show and discuss the performance of new proposed approaches applied to the classification of applications compared with that of the reference methods. We aim to examine the extent to which unsupervised representation learning can prevent significant performance decreases when there are a limited amount of labeled data.

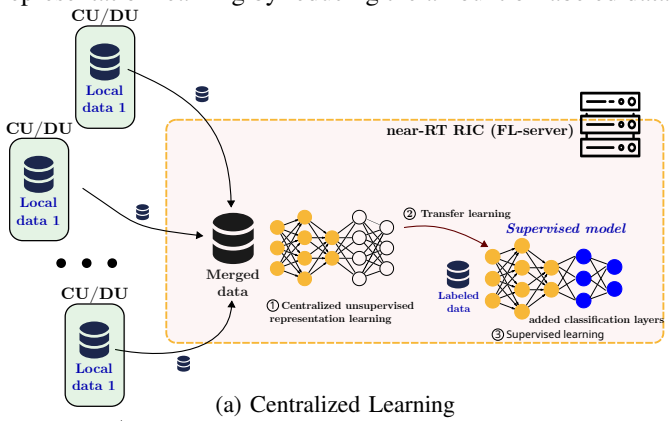
TABLE II: Methods tested for application classification.

Learning Mode	Approach Central	Pre-training	Fine-tuning
Central	SL / Payload [6], [7]	No	CNN
Central	SL / Time series [9]	No	LSTM
Central	SL / Multimodal	No	Multimodal
Central	C-SSL	DAE	Multimodal
Federated	FL-SSL	Federated DAE	Multimodal

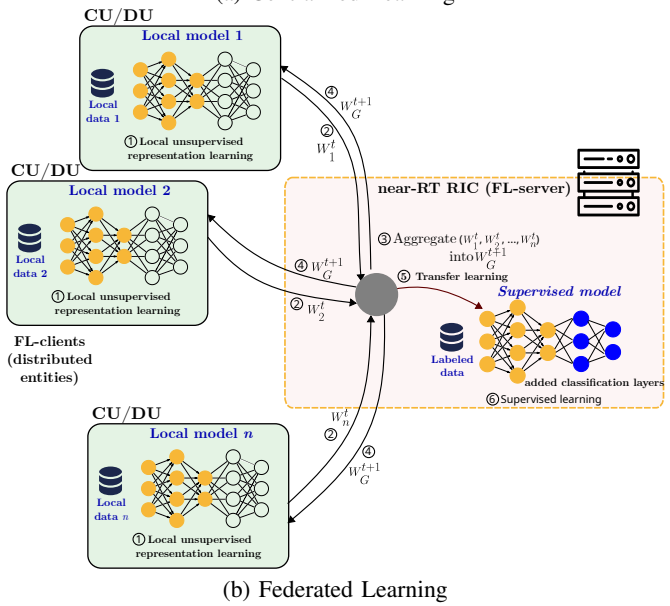
A. Configuration setup

Table II summarizes all the approaches used in the experiments for application classification (baseline and new) using Central Learning (**CL**) as well as Federated Learning (**FL**) mode. The baseline models classify the application using CNN [6], [7] or LSTM [9] model respectively without pre-training. They are referred to as: **SL** (Supervised Learning). All the other models comprise a multimodal supervised model with or without a pre-training step. They are referred to as, respectively: **C-SSL** (Centralized Semi-supervised Learning) and **FL-SSL** (Federated Semi-supervised learning).

To assess performance, we use Accuracy and F1-Score metrics. We also investigate the effect of introducing unsupervised representation learning by reducing the amount of labeled data from 100% (all labeled data) to less than 20%. As shown in Tab. I, the total number of labeled data is 40,666 points and unlabeled data is 73,285 points. It is always 100% when used. We separately analyzed the performance metrics for each class and visualized performance over individual classes using the confusion matrix. Lastly, we evaluate the bandwidth savings for federated learning vs. centralized learning.



(a) Centralized Learning



(b) Federated Learning

Fig. 4: Proposed application classification approach for O-RAN in centralized and federated learning

B. Overall Performances

Tab. III presents the performance of all tested models with 100% and 20% labeled data. The results indicate that models pre-trained using unsupervised representation learning, regardless of centralized or federated learning configurations, outperform models without pre-training, although a slight decrease in performance was observed for federated learning techniques in comparison to centralized learning. In addition, the results also shows that multimodal approaches that explore both raw payload and time-series of packets statistics outperform the ones that uses only one of those and not both.

Fig. 5 illustrates F1 score versus labeled data size. The impact of unsupervised representation learning as a pre-training step is evident in all cases, as the gap in performances grows with the decrease in the amount of labeled data. With 100% of labeled data, unsupervised pre-training improves performances. And with 50% and 25%, pre-trained models outperform non pre-trained models using 100% labeled data. Moreover, even when using less than 20% of initial labeled data (as shown in Tab. III), pre-training helps the model maintain good performances above 70%, not far from performances of using 100% data without pre-taining, whereas models that do not exploit unlabeled data collapse with F1-scores below 50%. This shows that unlabeled data adds value and reduces labeled data need for application classification through unsupervised representation learning. The models pre-trained using federated learning perform equivalently to models pre-trained using centralized learning with only a slight decrease in performance which is significant given FL's advantages.

C. Application wise evaluation

Here, we provide an individual evaluation for each class. Fig. 6 displays confusion matrices for both and non pre-trained and pre-trained models. With ratios of labeled data usage 100% and 20%. The impact of pre-training on improving accuracy in detecting applications correctly, particularly when using only 20% labeled data, can be clearly observed from these matrices. In the absence of pre-training for 20% labeled data, the model struggled to distinguish between "Video (Download)" and "Video Call" (Figure 6d) as opposed to pre-trained models (Figures 6e and 6f). The browsing class,

TABLE III: Accuracy and F1-Score comparison

	100% labeled data		20% labeled data	
	Accuracy	F1-score	Accuracy	F1-score
SL / Payload [6], [18]	58.49	52.05	34.85	23.15
SL / Time series [9]	74.15	71.32	47.58	38.78
SL / Multimodal	77.42	76.10	49.61	40.82
C-SSL	82.19	82.04	75.55	72.91
FL-SSL	79.76	79.52	73.17	72.37

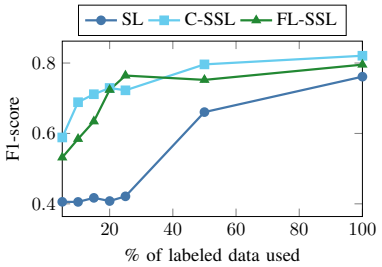


Fig. 5: F1-Score performances highlighting the added value of pre-training as the number of labeled data decrease

which is typically the most challenging due to its diversity (it may include applications from other classes), was also better handled overall by the pre-trained model. This superior performance can be attributed to the fact that supervised models with insufficient labeled data and no exploration of unlabeled data were unable to learn specific patterns defining different classes (Video and Video Call which are both video-based). However, models that effectively leveraged unlabeled data learned patterns that may not have been available in limited labeled data.

D. Centralized vs federated: bandwidth overhead

One of the primary aims of utilizing FL in mobile networks is to save bandwidth. The amount of bandwidth saved by not sending traffic data to a central server depends on the number of weight exchanges required in FL. In a CL setup, clients would need to send data of size ~ 185 GB per user (the total size of pcap files generated by one user in our study) and ~ 1.9 TB for 10 users as shown in Table IV. In contrast in FL, models is trained for 100 epochs and weights is exchanged every 5 epochs. Consequently, each local client sends data to the server 20 times and the server has to send same amount (aggregated weights to each client). So, for 3 local clients, the total number of weight exchanges amounts to $(20+20)\times 3 = 120$. Furthermore, each weight exchange involves transmitting a model with a size of ~ 1.5 MB. Therefore, the total data exchanged between clients and server during the entire FL training process is roughly equal to 1.5 MB $\times 120 = 180$ MB. This is significantly smaller than the minimal data size transmitted for CL. Assuming that the data processing and filtering is done on local clients rather than on central server, the local clients would still send ~ 1.3 GB of data per user or ~ 13 GB for 10 users. Thus, this is still larger than the size of data exchanged for FL. The observation is that more users lead to more data being collected. Consequently, the efficiency thanks to FL becomes increasingly higher. Despite the limited amount of data we used in our experiment, we still observed FL's advantages. In real-world applications, networks generate significantly larger quantities of data, resulting in even more important bandwidth save using FL rather than CL. Tab. V shows the total amount of weights transmitted for FL for different number of clients (assuming one model per client) versus the data size generated in CL for 1 user. More clients mean more data is collected for the training phase but also more diverse situations are observed. This would lead to increased classification performance. The table indicates that even in larger networks where more than 3 DUs/CUs train their own models, the bandwidth savings using FL remain significantly superior as compared to transmitting data to a central server (i.e., 5.4 GB for training 100 CUs/DUs versus 185 GB per user in CL).

TABLE IV: Size of info exchanged during centralized learning training w.r.t number of users in network

Number of users	1	5	10
Data size exchanged (centralized)	185 GB	925 GB	1.9 TB

VI. Conclusion

We showed the added value of unsupervised representation learning for mobile networks to identify five popular types of Application. We have demonstrated that Denoising Autoencoders (DAE) can learn meaningful representations that are effectively transferred to this downstream tasks. Our results reveal improved performances, particularly with limited amounts of labeled data. Even using only 20% of the labeled data that non pre-trained models require, pre-trained models maintain an F1-score above 70%. In contrast, no pre-trained models collapse with an F1-score under 50%. Our research also illustrates that pre-training can be performed using federated learning, avoiding sending data to a central server while maintaining similar results to centralized learning. However, this approach may be challenging to deploy in real-world mobile networks, as it requires sufficient processing power to train the model, at each distributed node. The use of a multimodal architecture combining CNN and LSTM may increase computational complexity, which could be a challenge for resource-constrained devices at the network edge. In future works, it would be worthwhile to experiment with this approach in real-world networks where these problems can be addressed. Additionally, other potential issues that may arise during real-world implementation, such as introduced additional latency, will need to be investigated. We also plan to investigate more representation learning techniques and alternative source tasks beyond Denoising Autoencoders. Furthermore, we intend to explore multitask learning by incorporating auxiliary tasks to enhance application classification performances.

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

TABLE V: Total size of models to be transmitted w.r.t to number of clients (models) in federated learning compared to total data of 1 user transmitted in centralized learning

Number of clients/models	CL (traffic data)	FL (models weights)
3	~ 185 GB	~ 180 MB
10		~ 540 MB
100		~ 5.4 GB

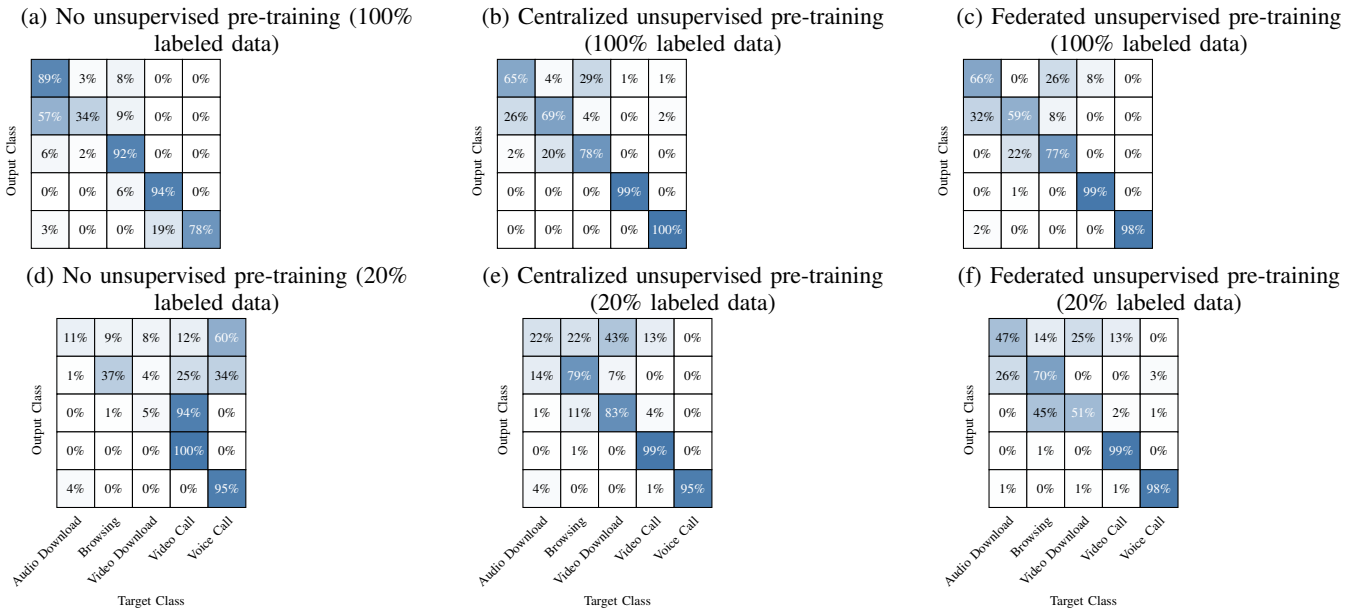


Fig. 6: Confusion matrix of the models using 100% and 20% ratio sizes of labeled data

[2] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever *et al.*, "Improving language understanding by generative pre-training," 2018.

[3] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*. PMLR, 2020, pp. 1597–1607.

[4] X. Lin, G. Xiong, G. Gou, Z. Li, J. Shi, and J. Yu, "ET-BERT: A Contextualized Datagram Representation with Pre-training Transformers for Encrypted Traffic Classification," in *Proceedings of the ACM Web Conference 2022*, Apr. 2022, pp. 633–642.

[5] "Yet Another Traffic Classifier: A Masked Autoencoder Based Traffic Transformer with Multi-Level Flow Representation | Proceedings of the AAAI Conference on Artificial Intelligence," https://ojs.aaai.org/index.php/AAAI/article/view/25674, 2023.

[6] Z. Wang, "The Applications of Deep Learning on Traffic Identification," p. 10, 2015.

[7] M. Lotfollahi, R. S. H. Zade, M. J. Siavoshani, and M. Saberian, "Deep Packet: A Novel Approach For Encrypted Traffic Classification Using Deep Learning," *arXiv:1709.02656 [cs]*, Jul. 2018.

[8] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescape, "Mobile Encrypted Traffic Classification Using Deep Learning," in *2018 Network Traffic Measurement and Analysis Conference (TMA)*. Vienna: IEEE, Jun. 2018, pp. 1–8.

[9] S. Rezaei and X. Liu, "Deep learning for encrypted traffic classification: An overview," *IEEE communications magazine*, vol. 57, no. 5, 2019.

[10] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "Blinc: multilevel traffic classification in the dark," in *Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, 2005, pp. 229–240.

[11] A. Callado, C. Kamienski, G. Szabó, B. P. Gero, J. Kelner, S. Fernandes, and D. Sadok, "A survey on internet traffic identification," *IEEE communications surveys & tutorials*, vol. 11, no. 3, pp. 37–52, 2009.

[12] A. W. Moore and D. Zuev, "Internet traffic classification using bayesian analysis techniques," in *Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, 2005, pp. 50–60.

[13] M. Iliofotou, P. Pappu, M. Faloutsos, M. Mitzenmacher, S. Singh, and G. Varghese, "Network monitoring using traffic dispersion graphs (tdgs)," in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, 2007, pp. 315–320.

[14] T. Auld, A. W. Moore, and S. F. Gull, "Bayesian neural networks for internet traffic classification," *IEEE Transactions on neural networks*, vol. 18, no. 1, pp. 223–239, 2007.

[15] W. Li and A. W. Moore, "A machine learning approach for efficient traffic classification," in *2007 15th International symposium on modeling, analysis, and simulation of computer and telecommunication systems*. IEEE, 2007, pp. 310–317.

[16] M. Soysal and E. G. Schmidt, "Machine learning algorithms for accurate flow-based network traffic classification: Evaluation and comparison," *Performance Evaluation*, vol. 67, no. 6, pp. 451–467, 2010.

[17] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," in *2017 IEEE international conference on intelligence and security informatics (ISI)*. IEEE, 2017, pp. 43–48.

[18] M. Lotfollahi, M. J. Siavoshani, R. S. H. Zade, and M. Saberian, "Deep packet: a novel approach for encrypted traffic classification using deep learning," *Soft Computing*, vol. 24, pp. 1999 – 2012, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:35187639

[19] M. Wang, Y. Cui, X. Wang, S. Xiao, and J. Jiang, "Machine learning for networking: Workflow, advances and opportunities," *Ieee Network*, vol. 32, no. 2, pp. 92–99, 2017.

[20] 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.

[21] K. Wang, J. Gao, and X. Lei, "Mtc: A multi-task model for encrypted network traffic classification based on transformer and 1d-cnn." *Intelligent Automation & Soft Computing*, vol. 37, no. 1, 2023.

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

[23] H. Mun and Y. Lee, "Internet traffic classification with federated learning," *Electronics*, vol. 10, no. 1, p. 27, 2020.

[24] E. Bakopoulou, B. Tillman, and A. Markopoulou, "Fedpacket: A federated learning approach to mobile packet classification," *IEEE Transactions on Mobile Computing*, vol. 21, no. 10, pp. 3609–3628, 2021.

[25] M. Abbasi, A. Taherkordi, and A. Shahraki, "Flitc: A novel federated learning-based method for iot traffic classification," in *2022 IEEE International Conference on Smart Computing (SMARTCOMP)*. IEEE, 2022, pp. 206–212.

[26] D. Erhan, A. Courville, Y. Bengio, and P. Vincent, "Why does unsupervised pre-training help deep learning?" in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 201–208.

[27] S. A. Hamideche, M. L. Alberi Morel, K. Singh, and C. Viho, "Federated Representation Learning for Indoor-Outdoor Detection in Beyond 5G Networks," in *2023 IEEE Globecom Workshops (GC Wkshps)*. Kuala Lumpur, Malaysia: IEEE, Dec. 2023, pp. 860–865.