A Federated Continual Learning Framework for Sustainable Network Anomaly Detection in O-RAN

Chafika Benzaïd*, Fahim Muhtasim Hossain*, Tarik Taleb*†, Pedro Merino Gómez‡, and Michael Dieudonne§
*University of Oulu, Oulu, Finland, †Ruhr University Bochum, Bochum, Germany,
‡Universidad de Malaga, Malaga, Spain, §Keysight Technologies, Belgium
Emails: chafika.benzaid@oulu.fi, fahimtonmoy108@gmail.com, tarik.taleb@oulu.fi, pmerino@uma.es, michael_dieudonne@keysight.com

***Abstract*—The distributed and disaggregated nature of 5G and beyond (B5G) networks has spurred interest in federated learning (FL) for empowering privacy-preserving collaborative network anomaly detection at the edge. However, FL is prone to catastrophic forgetting (CF), where prior knowledge is forgotten while sequentially learning new attack patterns from a stream of data. Few studies addressed CF issue in network anomaly detection using Continual Learning (CL), but focusing on centralized models rather than FL and overlooking integration in B5G. To fill this gap, we propose TenaxDoS, a novel framework that combines FL with a replay memory-based CL strategy to foster sustainable and cooperative network anomaly detection in an Open Radio Access Network (O-RAN) environment in B5G networks. The experimental results on a dataset from a real 5G test network show TenaxDoS's superior overall performance, stability and effective mitigation of CF, yielding a remembering of past knowledge of above 98.8%.**

***Index Terms*—B5G, DDoS, Federated Continual Learning, Network Anomaly Detection, O-RAN.**

I. Introduction

The emerging Open Radio Access Network (O-RAN) architecture [1] is a notable effort to achieve 5G and beyond (B5G) networks promises for ultra low latency, high data speeds, and massive connectivity, owing to its its openness, disaggregation and embedded intelligence features. Those features are set to drive the wide-spread adoption of private 5G networks, tailored to the specific performance and security needs of delivered services [2]. However, the promises of B5G are offset by an increased attack surface, mostly due to the network's dispersed and disaggregated architecture.

Network anomaly detection systems are crucial for protecting B5G networks against emerging cyberthreats, ensuring they deliver their promises. Identifying Distributed Denial of Service (DDoS) attacks is especially vital. DDoS attacks have the capacity to quickly overload the network or exhaust the server's computational and memory resources, leading to substantial service disruptions and possible data loss [3]. The ever-evolving sophistication of DDoS techniques challenge the effectiveness of traditional rule-based detection systems. This has spurred the adoption of machine learning (ML) and deep learning (DL) approaches to improve generalization and detection accuracy, thanks to their capacity in spotting subtle network changes caused by a malicious behavior.

The highly distributed and disaggregated nature of B5G networks has stimulated the recent increasing interest in adopting distributed learning, particularly federated learning (FL), for empowering collaborative network anomaly detection at the edge [4]. Using FL, the distributed network nodes can develop a shared anomaly prediction model cooperatively while keeping all their training data locally. This will result in greatly improving the effectiveness and timeliness of anomaly detection, thanks to local knowledge sharing among network nodes. Moreover, limiting the information exchange to the model parameters helps in fostering data privacy preservation, a key requisite to support compliance with data sharing restrictions of private 5G networks.

Despite of their advantages, ML, DL, and FL-based anomaly detection systems are still facing challenges. Catastrophic Forgetting (CF) [5], a situation where the learning model tends to forget previously acquired patterns when exposed to fresh data or attack patterns, is a critical problem. Overlooking this issue will significantly undermine the attack detection effectiveness, consequently reducing the reliability of ML/DL/FL-powered security defense solutions and calling into question the trust placed in their decisions. Therefore, addressing CF issue is paramount for maintaining resilience in identifying the continually evolving and constantly emerging DDoS attacks in B5G networks.

Continual Learning (CL) [6] is poised as a viable remedy to lessen the CF issue. CL enables ML/DL models to accommodate new knowledge while preserving previously learned experiences, which fosters the capability of incrementally learning from a stream of data. A number of approaches have been developed in CL to combat CF, with *replay buffers* being among the most promising. Replay buffers [7] preserve a balance between old and new information by storing a portion of past data for subsequent "replay" during training. Although CL has demonstrated success in a number of fields, including computer vision, its use in network security is still unexplored. To the best of our knowledge, very few contributions (e.g., [8], [9]) have explored for network anomaly detection. However, most of them have addressed CF in centralized ML/DL-based models rather than FL. Moreover, none of existing works have explored integrating CL strategies in B5G networks.

To fill this gap, we propose a novel framework, coined TenaxDoS, that exploits and combines the potential of CL and FL to empower sustainable and cooperative privacy-preserving network anomaly detection in B5G networks. Leveraging the disaggregated and distributed nature of O-RAN architecture coupled with the in-built intelligence via intelligent

controllers, we demonstrate the integration of TenaxDoS in an O-RAN environment. Such integration not only promotes timely and continuous detection of network anomalies at the edge, but also fosters multi-operator collaboration in a privacy-preserving way. This makes TenaxDoS a promising solution to stimulate collaboration among private 5G networks. To the best of our knowledge, TenaxDoS is the first solution using CL and FL for network anomaly detection in O-RAN.

The remainder of this paper is structured as follows. Section II presents an overview of related work in the literature. Section III provides background information on Federated Continual Learning (FCL) and O-RAN architecture. Section IV introduces the proposed TenaxDoS framework, detailing its architecture and the design of the FCL-based DDoS anomaly detection model. Section V describes the experimental setup and presents the performance evaluation results. Finally, Section VI concludes the paper and suggests future research directions.

II. Related work and limitations

Extensive research has focused on tackling DDoS attacks in next-generation networks, leveraging the potential of ML/DL through either supervised approaches [3], [10] or unsupervised anomaly detection approaches [11], [12]. To support data privacy preservation, there is a recent shift towards empowering collaborative network anomaly detection leveraging FL [13]. For instance, Hireche *et al.* [4] leveraged the capabilities of programmable data plane and FL to empower fully distributed self-detection and mitigation of DDoS attacks, fostering the realization of secure self-driving networks. In [14], a peer-to-peer MLP-based FL approach is developed for detecting network anomalies in O-RAN.

Until very recently, the problem of CF has been overlooked in most existing ML/DL/FL-based network anomaly detection solutions, assuming that all training data are available at the same time. To the best of our knowledge, very few contributions have aimed at addressing CF in the context of network anomaly detection by adopting CL paradigm. However, most of them (e.g., [8]) have tackled CF concern in centralized ML/DL-based network anomaly detection system instead of FL. As far as we know, [9] is the only work considering the integration of CL in FL-based network anomaly detection. Nevertheless, the authors did not provide details on the created tasks and did not assess the effectiveness of the proposed solution in mitigating CF. Moreover, none of the aforementioned studies have investigated the integration of the proposed CL strategies in B5G networks. To fill this gap, we propose a novel FCL-based network anomaly detection approach that can be integrated in O-RAN environment.

III. Background

A. Federated Continual Learning

Combining CL and FL, FCL enables collaborative learning by continuously training on a stream of distributed private data, fostering privacy-preserving knowledge sharing while mitigating CF [15]. In CL, three scenarios are distinguished depending on whether task identity is known at test time and, if not, whether it must be determined [16]: (i) Task-incremental learning; (ii) Domain-incremental learning; and (iii) Class-incremental learning. In our study, we face a domain-incremental learning scenario, as the output classes (i.e., legitimate or malicious) remain constant despite task changes. The CL methods used for alleviating CF fall into three categories [7], namely: (i) Regularization-based methods, which preserve learned knowledge by restricting model parameter updates; (ii) Parameter isolation based methods, which prevent CF by allocating separate parameters per task; and (iii) Rehearsal-based methods, which tackle CF by storing past samples into a fixed-size or generative memory for subsequent replay when learning new tasks. In this study, we adopt a rehearsal-based strategy, owing to its proven superiority in terms of simplicity, performance and resource efficiency [7].

B. O-RAN Architecture

In O-RAN architecture [1], the RAN components are virtualized and disaggregated, using open interfaces and intelligent controllers for interconnection and optimization. These features provides greater flexibility, openness, and enhanced visibility and security. As illustrated in Fig. 1, O-RAN architecture includes the Radio Unit (O-RU), Distributed Unit (O-DU), Central Unit control plane (O-CU-CP) and Central Unit user plane (O-CU-UP) on the radio side, and the Service Management and Orchestration (SMO) framework on the management side. The network intelligence is enabled by the RAN Intelligent Controller (RIC), incorporating AI/ML into its decision-making. The RIC comprises a near-real time controller (Near-RT RIC) handling delay-sensitive control functions and a non-real time controller (Non-RT RIC) for control operations with more relaxed latency needs. The control functions at Near-RT RIC and Non-RT RIC are handled by specialized applications called xApps and rApps, respectively. The Near-RT RIC connects with O-DU, O-CU-UP and O-CU-CP (known as E2 nodes) via E2 interface.

IV. TenaxDoS – A FCL-based DDoS Detection Framework

In this section, we present TenaxDoS, a novel framework that leverages and combines the potential of CL and FL to empower sustainable and cooperative network attack detection in an O-RAN environment in B5G networks.

A. Framework Overview

Fig. 1 illustrates the overall architecture of TenaxDoS framework integrated in the O-RAN architecture. The framework capitalizes on RAN function disaggregation and the native support of intelligence in O-RAN to enable swift and fully autonomous detection and mitigation of DDoS attacks at the edge. Attack detection is powered by a DL model trained following an FCL approach. This facilitates knowledge sharing for increased attack detection accuracy while avoiding the sharing of local data and effectively addressing the catastrophic forgetting concern. The TenaxDoS framework encom-

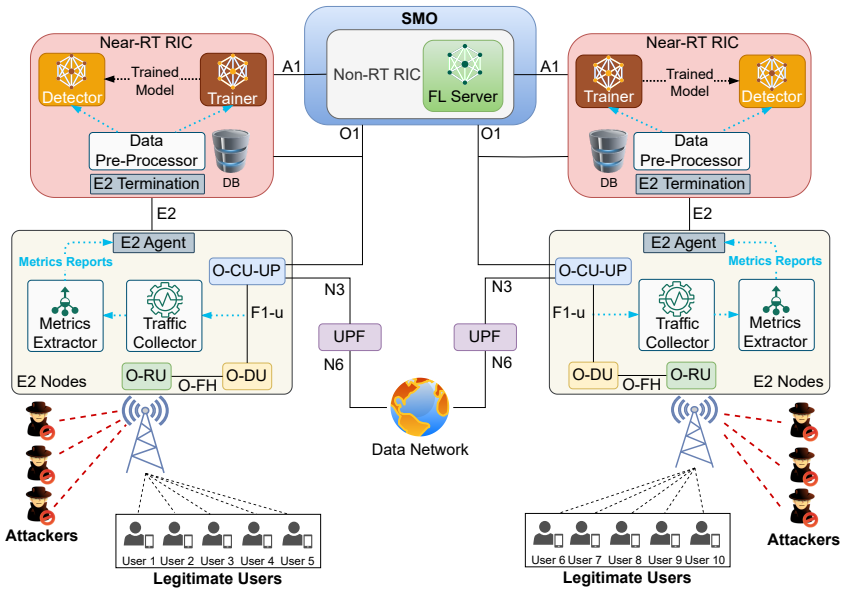


Fig. 1: The overall architecture of TenaxDoS framework.

passes six core components that provide traffic monitoring, data processing, attack detection, and FCL training services.

The Traffic Collector continuously captures the user plane traffic over the F1-u interface. This is feasible in the context of private networks with full access to the GPRS Tunneling Protocol (GTP) tunnel [17]. The collected traffic is periodically communicated to the Metrics Extractor to extract the network flow's features pertinent to identify malicious patterns caused by DDoS attacks. The extracted features are streamed to the near-RT RIC in the form of metrics reports via the E2 interface. The metrics reports are conveyed to the Data Pre-Processor xApp for transforming the raw data into appropriate format to fit for the DL model during training and inference phases. This includes data cleansing, encoding, and standardization, as elaborated in Section IV-C. During the inference phase, the pre-processed network flow's features are passed to the Detector xApp to decide on the legitimacy or maliciousness of the received traffic. The Detector incorporates a DL-powered DDoS detection model trained following an FCL approach. If a malicious traffic pattern is detected, the Detector issues a security policy for enforcement on E2 Nodes to mitigate the the DDoS attack. The security policy can, for example, consist in blocking the F1-u GTP tunnel of the user equipment's Protocol Data Unit (PDU) session from which the malicious traffic is coming.

The model training is carried out by the FL Server rApp and Model Trainer xApps deployed at the Non-RT RIC and the connected Near-RT RICs, respectively. Each Model Trainer xApp trains the local DL-powered DDoS detection model using the local metrics reports collected from the attached base stations. To prevent catastrophic forgetting, the Model Trainer adopts a CL approach for training the local model. The local model updates (i.e., parameters) are sent to the Non-RT RIC for aggregation. The FL Server rApp acts as the central server in charge of training the global model by aggregating the local models updates. The parameters of the global and local models are exchanged between FL Server and Model Trainers over O1 interface. Details on the model structure and the training process will be elaborated further in the subsequent sections IV-D and IV-E.

B. DDoS Attack Types

We assume that the attacker is able to carry out three common forms of DoS/DDoS attacks, namely: volume-based, protocol-based, and application layer attacks [10]. In volume-based (D)DoS attacks, the attacker aims at overloading the network's available bandwidth by generating a large volume of traffic. Examples of volume-based (D)DoS attacks include ICMP flood and UDP flood. Protocol-based (D)DoS attacks exploit the network protocol weaknesses to drain the network resources. SYN flood, SYN scan, and TCP connect scan attacks, which manipulate the three-way handshake connection establishment procedure of TCP protocol, are common examples of protocol-based (D)DoS attacks. Finally, application-layer (D)DoS attacks abuse application-layer protocols and services (e.g., HTTP and DNS) to exhaust the server's resources. It is worth mentioning that application-layer (D)DoS attacks are often more challenging to spot, owing to their capability to imitate legitimate behavior with low network bandwidth usage [3]. HTTP flood and slow-rate DoS are well-known examples of application-layer DDoS attacks.

C. Data Preparation

1) Task-based Representation: The local training dataset $\mathcal{D}$ consists of a series of T DDoS attack detection tasks that arrive sequentially over time. Each task t corresponds to a dataset $\mathcal{D}_t$ containing both legitimate and malicious network flows The attack classes are distinct across various tasks; i.e., each DDoS attack class is exclusively present in exactly one task.

2) Data Preprocessing: The data pre-processing module transforms the network flow data of a given task into a

format that fits for the DL model. Firstly, the data are cleaned by removing all missing and infinity values. The categorical features are then converted to numerical ones using one-hot encoding method, yielding a binary feature for each unique categorical value. After applying one-hot encoding, each network flow will be characterized by 96 features in addition to a label identifying the flow's class (i.e., legitimate or malicious). To remove scaling differences, the values of features are standardized by scaling each feature to have zero mean and unit variance.

D. Local Model Structure

Motivated by its capacity in uncovering complex non-linear patterns, a DL model is adopted for detecting network anomalies caused by DDoS attacks. The detection model is built using Multi-Layer Perceptron (MLP) algorithm. It consists of an input layer of 96 neurons associated to the pre-processed network flow's features, 3 hidden layers with 64 neurons each, and 1 output layer containing only one neuron for binary classification. The model's output is the network flow's class; i.e., legitimate or malicious). The Rectified Linear Unit (ReLU) and Sigmoid are used as activation functions in the hidden layers and the output layer, respectively.

E. Federated Continual Learning Training Process

The training task of the DDoS anomaly detection model is carried out cooperatively by a set of K Near-RT RICs $\{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_K\}$ via the Trainer xApps. Each Near-RT RIC $\mathcal{C}_k$ trains the local DL model on a stream of data $\mathcal{D}_k$, which consists of T DDoS attack detection tasks that arrive sequentially over time. At each time step $t \in \{1, 2, \ldots, T\}$, the stream $\mathcal{D}_k$ reveals n_k network flow samples $\{(x_i, y_i)\}_{i=1}^{n_k} \sim \mathcal{D}_k^t$, where x_i and y_i are, respectively, the network flow's features vector and the associated label. Let $\mathcal{D}$ denotes the global input space which satisfies $\mathcal{D} = \cup_{k=1}^{K} \cup_{t=1}^{T} \mathcal{D}_k^t$ and $|\mathcal{D}| = \sum_{k=1}^{K} \sum_{t=1}^{T} n_k^t = N_{\mathcal{D}}$. The goal of FCL is to minimize the following objective function:

$$\underset{\omega}{argmin}\Big\{\mathcal{L}(\omega) \triangleq \frac{1}{N_{\mathcal{D}}} \sum_{k=1}^{K} \sum_{t=1}^{T} \sum_{(x_i, y_i) \in \mathcal{D}_k^t} l(f(x_i, \omega); y_i)\Big\} \tag{1}$$

where $\mathcal{L}(.)$ denotes the global loss function and $l(.;.)$ is the local loss function on one data sample. $f(x_i, \omega)$ is the predicted label for network flow sample x_i using the global model weight vector ω.

The overall FCL training process is provided in Algorithm1. The process is divided into three phases: the initialization, the local training, and the aggregation.

1) Initialization: The FCL training process starts with the FL Server hosted at Non-RT RIC initiating and sending the global model parameters and the associated hyper-parameters (e.g., local training epochs, learning rate, loss function) to every Trainer at Near-RT RICs involved in the training process.

2) Local Training: For each data stream $\mathcal{D}_k^t$, at each communication round r, each Trainer $\mathcal{C}_k$ first updates its local model parameters ω_k using the global model parameters received from the FL Server. The local model parameters are then optimized over the local training dataset to minimize the loss function. Considering a binary classification problem, the binary-cross entropy is used as a loss function, which is formulated on one network flow sample (x_i, y_i) as:

$$l(\hat{y}_i, y_i) = -\left[y_i * log(\hat{y}_i) + (1 - y_i) * log(1 - \hat{y}_i)\right] \tag{2}$$

where $\hat{y}_i = f(x_i, \omega_k)$ is the predicted label of the ground truth target label y_i.

To tackle CF, the local training follows a CL approach. Motivated by its proven superiority in CL, we adopt a rehearsal-based strategy with reservoir sampling. Specifically, each Trainer maintains a fixed-size buffer memory $\mathcal{B}_k$, where it stores representative network flow samples from past tasks for subsequent replay when learning new tasks. The reservoir sampling technique decides whether to keep or reject a new sample based on a given probability when new samples come in and the buffer fills up [7]. This probability is expressed as $\frac{|\mathcal{B}_k|}{\sum_{i=1}^{t} n_k^i}$, where $|\mathcal{B}_k|$ is the size of the buffer memory $\mathcal{B}_k$ and $\sum_{i=1}^{t} n_k^i$ is the total number of traffic samples observed by Trainer $\mathcal{C}_k$ up to task t. Thus, the local model is trained on the union of the current task samples $\mathcal{D}_k^t$ and the buffered samples $\mathcal{B}_k$.

Once the training is finished, the Trainer transmits the optimized local parameters for the current round to the FL Server for aggregation.

3) Federated Aggregation: The FL Server aggregates the received local model parameters $\{\omega_k^t\}_{k=1}^{K}$ into the global parameters ω_g, allowing to capture the global knowledge on

Algorithm 1 Federated Continual Learning Training.

Input:
$\mathcal{C}$: Set of FL Trainers; $\mathcal{D}_k^t$: Dataset stream of Trainer $\mathcal{C}_k$;
T: Number of tasks; R: Number of communication rounds;
E: Number of local epochs; $\mathcal{B}_k$: Replay buffer of Trainer $\mathcal{C}_k$;
ω_g: Global model parameters; $\mathcal{M}_k$: Local model of Trainer $\mathcal{C}_k$;
ω_k: Local model parameters of Trainer $\mathcal{C}_k$;
$\mathcal{M}_k$: Local model of Trainer $\mathcal{C}_k$

for Trainer $\mathcal{C}_k \in \mathcal{C}$ **do**
  $\mathcal{B}_k \leftarrow \{\}$;
end for
FL Server initializes the global model $\mathcal{M}$ and its parameters ω_g;
FL Server distributes $\mathcal{M}$ and ω_g to all Trainers $\mathcal{C}_k \in \mathcal{C}$;
for $t = 1$ to T **do**
  for $r = 1$ to R **do**
    for Trainer $\mathcal{C}_k \in \mathcal{C}$ in parallel **do**
      Receive the model parameters sent by the FL Server;
      $\omega_k \leftarrow \omega_g$;
      ▷ Train local model to minimize Eq.(2) for solving the local CL problem
      $\omega_k \leftarrow \mathcal{M}_k.Train(\omega_k, \mathcal{D}_k^t \cup \mathcal{B}_k, E)$;
      Send ω_k to FL Server;
      if $r == R$ **then**
        $\mathcal{B}_k \leftarrow \mathcal{B}_k.Update(\mathcal{D}_k^t)$;
      end if
    end for
    ▷ FL Server aggregates the local weights using Eq.(3)
    $\omega_g \leftarrow Aggregate(\{\omega_k\}_{k=1}^{K})$;
    FL Server distributes ω_g to all Trainers $\mathcal{C}_k \in \mathcal{C}$;
  end for
end for

DDoS anomaly detection. In this study, Federated Averaging (FedAvg) [18] is used for model aggregation, as follows:

$$\omega_g = \frac{\sum_{k=1}^{K} n_k^t \cdot \omega_k}{\sum_{k=1}^{K} n_k^t} \quad (3)$$

The aggregated parameters are then disseminated by the FL Server to all Trainers for the next training round.

V. Performance Evaluation

A. Experimental Settings

TABLE I: Nine tasks extracted from 5G-NIDD for CL setting.

Task	DDoS Class	DDoS Type	Task	DDoS Class	DDoS Type	Task	DDoS Class	DDoS Type
1	Goldeneye	A	4	SYNSCAN	P	7	Torshammer	A
2	ICMPFlood	V	5	Slowloris	A	8	UDPFlood	V
3	SYNFlood	P	6	TCPConnect	P	9	UDPScan	P

Note: **A** – Application-layer DDoS, **V** – Volume-based DDoS, **P** – Protocol-based DDoS

1) Dataset Description: The experiments are based on 5G-NIDD dataset [10]. The dataset contains a combination of malicious and legitimate traffic generated by actual mobile devices attached to the 5G test network 5GTN (https://5gtn.fi/) through two base stations. The malicious traffic is produced by several DDoS attacks as summarized in Table I.

For CL setting, the dataset is split into 9 tasks corresponding to the 9 attack sessions conducted to generate 5G-NIDD. We took advantage of already available data separated in CSV files per BS and attack session to create the 9 tasks. The pre-processing phase results into a total of 308991 (250019) network flow samples distributed across the 9 tasks of BS1 (BS2), where each flow is characterized by 96 features in addition to the flow's label.

2) Environmental Setup: Fig.2 illustrates the testbed setup. We used three (03) virtual machines (VMs) on an OpenStack cloud, each of them running Ubuntu 22.04.5 LTS operating system. Two VMs, with a configuration of 16 vCPU and 58GB RAM, act as the Trainers associated to the two base stations. The third VM, with a configuration of 4 vCPU and 7.8GB RAM serves as the FL Server.

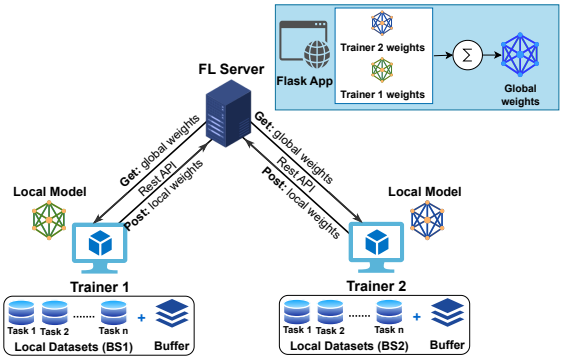


Fig. 2: Experimental Setup.

The Trainers and FL Server are implemented using Python 3.8. The interaction between the Trainers and the FL server is performed through REST-full interfaces implemented using the lightweight web server framework Flask (https://flask.palletsprojects.com/en/2.3.x/). The designed MLP model is built using Pytorch 2.0.1. The reservoir sampling-based CL strategy is implemented using Avalanche [19], an end-to-end CL library based on Pytorch. For each of the 9 tasks in 5G-NIDD dataset, the local MLP model is trained for 3 rounds, with 10 epochs in each round. The local training is carried out with a batch size of 128, Adam as an optimizer, a learning out of 0.01, and a replay buffer size of 5000 network flow samples.

3) Performance Metrics: Let $pm \in \mathbb{R}^{T \times T}$ be the train-test performance metric (i.e., accuracy, recall, precision and F1-score) matrix, where $pm_{i,j}$ is the performance metric of the model on the test set of task t_j after training the model on task t_i. The effectiveness of TenaxDoS in cooperatively detecting DDoS anomalies while tackling catastrophic forgetting is assessed by measuring the following metrics [20]:

- **Average Performance Metric** is measured after the model has been trained continually till task $\{t\}_{t=1}^{T}$ as follows:

$$PM_t = \frac{\sum_{i=1}^{t} \sum_{j=1}^{i} pm_{i,j}}{\frac{t(t+1)}{2}} \quad (4)$$

  where PM refers to average accuracy (ACC), precision (P), F1-score (F1) or recall (R).
- **Backward Transfer** measures the degree of forgetting older tasks after training on a new task. It is defined as:

$$BWT = \frac{\sum_{i=2}^{T} \sum_{j=1}^{i-1} (acc_{i,j} - acc_{j,j})}{\frac{T(T-1)}{2}} \quad (5)$$

- **Forward Transfer** quantifies the model's ability to perform zero-shot learning; i.e., the ability to generalize to future tasks. It is defined as:

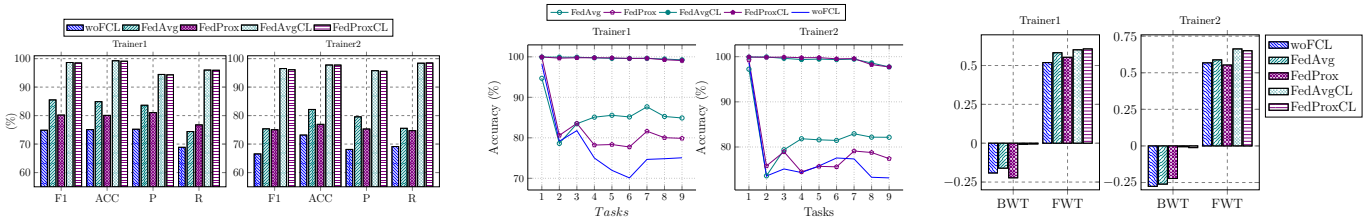
$$FWT = \frac{\sum_{i=1}^{T} \sum_{j=i+1}^{T} acc_{i,j}}{\frac{T(T-1)}{2}} \quad (6)$$

The performance metrics are measured for each Trainer by testing on the unseen data of the other Trainer's tasks. The reported results are the average over five runs. To better demonstrate the advantage of combining FL and CL, the FCL performance results are compared to those without using FCL and those when only FL is used. In addition to FedAvg, the performances are also evaluated using FedProx [21].

B. Performance Results

1) Overall Performances: Fig. 3(a) shows the global per-Trainer average performances after training on all tasks (i.e., $t = T$ in Eq.(4)). We observe that conventional FL training (i.e., FedAvg and FedProx) improves the performance compared to training without FCL (i.e., woFCL), thanks to the inter-Trainers knowledge transfer. However, it fails to handle the shifts in the data distribution across tasks over time. On the other hand, FCL-based training (i.e., FedAvgCL and FedProxCL) significantly boosts performances, surpassing FL training by at least 14.4% in accuracy, 10.8% in precision, 13.1% in F1, and 19.2% in recall. This improvement is ascribed to the replay memory's capacity to retain knowledge from previous tasks.

2) Catastrophic Forgetting: Fig. 3(b) summarizes the evolution of the average accuracy (ACC_t) as new tasks are



(a) Overall Performance

(b) Average Accuracy

(c) Catastrophic foregetting

Fig. 3: Performance results of TenaxDoS.

learned. It is observed that FCL-based training prevents interference from new tasks with the knowledge learned from previous task, resulting in high stability and fostering sustainable AI-powered DDoS anomaly detection. This statement is further corroborated by the achieved BWT. As shown in Fig. 3(c), FCL-based training effectively mitigates CF, exhibiting a negligible BWT that allows to achieve a remembering of past knowledge of above 98.8%. This is owed to the presence of a replay memory, which prevents the sudden loss of previous task knowledge. The results in Fig. 3(c) show also that less forgetting leads to enhanced forward transfer.

VI. Conclusion and Future Research Direction

In this paper, we proposed TenaxDoS, a novel framework that leverages the potential of FCL to enable sustainable and cooperative network anomaly detection in an O-RAN environment. Leveraging FCL and integration in O-RAN, TenaxDoS not only promotes timely and continuous detection of network anomalies at the edge but also fosters multi-operator collaboration in a privacy-preserving way. The experimental results demonstrated the superiority of TenaxDoS in accurately detecting DDoS attacks against 5G services while effectively alleviating CF problem. In the future, we intend to extend TenaxDoS to support fully decentralized asynchronous FCL, eliminating central aggregation and enabling handling more realistic scenarios where DDoS attacks arrive to base stations with different orderings and in asynchronous time frames.

Acknowledgement

This research work is partially supported by the Research Council of Finland (former Academy of Finland) under 6G Flagship program (Grant No. 346208), and the European Union's Horizon Europe research and innovation programme HORIZON-JU-SNS-2022 under the 6G-SANDBOX project (Grant No. 101096328) and the RIGOUROUS project (Grant No. 101095933). The paper reflects only the authors' views. The Commission is not responsible for any use that may be made of the information it contains.

References

[1] O-RAN Alliance, "O-RAN Architecture Description 9.0," in *Technical Specification*, June 2023.

[2] A. Bellin *et al.*, "Autonomous Private Mobile Networks: State of the Art and Future Challenges," *IEEE Commun. Standards Magazine*, vol. 7, pp. 24–31, June 2023.

[3] C. Benzaïd, M. Boukhalfa, and T. Taleb, "Robust Self-Protection against Application-layer (D) DoS Attacks in SDN Environment," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*. IEEE, 2020, pp. 1–6.

[4] O. Hireche, C. Benzaïd, and T. Taleb, "Deep Data Plane Programming and AI for Zero-Trust Self-driven Networking in beyond 5G," *Computer Networks*, vol. 203, p. 108668, Feb. 2022.

[5] R. M. French, "Catastrophic Forgetting in Connectionist Networks," *Trends in Cognitive Sciences*, vol. 3, no. 4, pp. 128–135, 1999.

[6] D. Rolnick *et al.*, "Experience Replay for Continual Learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[7] M. D. Lange *et al.*, "A Continual Learning Survey: Defying Forgetting in Classification Tasks," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 44, pp. 3366–3385, Jul. 2021.

[8] S. Prasath *et al.*, "Analysis of Continual Learning Models for Intrusion Detection System," *IEEE Access*, vol. 10, Nov. 2022.

[9] Z. Zhang *et al.*, "Communication-efficient Federated Continual Learning for Distributed Learning System with Non-IID data," *Sci. China Inf. Sci.*, vol. 66, p. 122102:1–122102:20, Dec. 2022.

[10] S. Samarakoon *et al.*, "5G-NIDD: A Comprehensive Network Intrusion Detection Dataset Generated over 5G Wireless Network," *ArXiv*, vol. abs/2212.01298, 2022.

[11] C. Benzaïd, T. Taleb, and J. Song, "AI-Based Autonomic and Scalable Security Management Architecture for Secure Network Slicing in B5G," *IEEE Network*, vol. 36, no. 6, pp. 165–174, 2022.

[12] C. Benzaïd, T. Taleb *et al.*, "A Deep Transfer Learning-powered EDoS Detection Mechanism for 5G and Beyond Network Slicing," in *Proc. IEEE Global Commun. Conf. (To appear)*, Dec. 2023.

[13] S. Agrawal *et al.*, "Federated Learning for Intrusion Detection System: Concepts, Challenges and Future Directions," *Comput. Commun.*, vol. 195, pp. 346–361, Nov. 2022.

[14] D. Attanayaka *et al.*, "Peer-to-Peer Federated Learning based Anomaly Detection for Open Radio Access Networks," in *Proc. IEEE Int. Conf. Commun. (ICC 2023)*, June 2023.

[15] J. Yoon *et al.*, "Federated Continual Learning with Weighted Inter-client Transfer," in *Proc. 38th Int. Conf. Mach. Learning*, vol. PMLR 139, Jul. 2021, pp. 12 073–12 086.

[16] G. M. van de Ven and A. S. Tolias, "Three scenarios for continual learning," *ArXiv*, vol. abs/1904.07734, 2019.

[17] C. A. García-Pérez and P. Merino, "Experimental Evaluation of Fog Computing Techniques to Reduce Latency in LTE Networks," *Trans. Emerg. Telecommun. Technol.*, vol. 29, Apr. 2018.

[18] H. Brendan McMahan *et al.*, "Communication-efficient Learning of Deep Networks from Decentralized Data," *ArXiv*, vol. 1602.05629, 2016.

[19] V. Lomonaco *et al.*, "Avalanche: an End-to-End Library for Continual Learning," in *Proc. 2021 IEEE/CVF Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW)*, 2021, pp. 3595–3605.

[20] N. Díaz-Rodríguez *et al.*, "Don't forget, there is more than forgetting: new metrics for Continual Learning," in *Workshop on Continual Learning, NeurIPS 2018*, Dec. 2018.

[21] T. Li *et al.*, "Federated Optimization in Heterogeneous Networks," in *Proc. of Mach. Learning and Syst.*, vol. 2, March 2020, pp. 429–450.