MEGATRON: Machine Learning in 5G with Analysis of Traffic in Open Radio Access Networks

Mauro Belgiovine, Jerry Gu, Joshua Groen, Miquel Sirera, Utku Demir, Kaushik Chowdhury
Institute for the Wireless Internet of Things, Northeastern University, Boston, MA, USA
{belgiovine.m, gu.je, groen.j, sirera.m, u.demir, k.chowdhury}@northeastern.edu

***Abstract*—With the advent of 5G and next-generation cellular networks and the increasing complexity of assigning users traffic types for efficient resource allocation, Open Radio Access Networks (O-RAN) offer intelligent virtualized frameworks for optimizing network operations related to supporting diverse types of traffic. In this paper, we utilize the native support for machine learning in O-RAN to develop a transformer-based 5G traffic classification system that identifies, with high accuracy, conditions when broadband, machine-to-machine type communication, and ultra-reliable low-latency communication are present. By utilizing distinct temporal slices of O-RAN-defined key performance indicators generated from traffic captures as inputs (as opposed to directly accessing user-plane data) and filtering for non-critical control traffic, we ensure user confidentiality while maintaining a high degree of classification performance. Our transformer model is able to achieve an average offline accuracy of $99\%+$ for the longest traffic slice length, with the online deployment achieving an average of $\sim 90\%$ accuracy across all slice lengths.**

***Index Terms*—O-RAN, 5G, Transformers, Traffic Classification, Network Slicing**

I. Introduction

5G and next-generation cellular systems are evolving to meet the demands of high data rates, low latency, and large-scale communication. To accommodate these goals with efficient network resource allocation, 5G introduces three traffic classes: i) enhanced Mobile BroadBand (eMBB), ii) massive Machine Type Communications (mMTC), and iii) Ultra Reliable Low Latency Communications (URLLC). Such classification prevents the over-allocation of network resources for users, which opens the door for improved coexistence with efficient network resource allocation. The challenge here, however, is how to perform the first step of the traffic classification. Typically, mobile users explicitly convey their requested service in order to negotiate their assignment to a specific traffic slice [1], although such an approach raises privacy and confidentiality concerns. Additionally, the presence of *control* (ctrl) traffic, i.e., traffic where no application data is being sent while the frame resources are still used to maintain the state of user connection, can overlay other traffic types. This mixture of control and data traffic further complicates the problem of traffic classification.

• **Traffic classification challenges:** Recent performance measurements [2], [3] in deployed 5G networks indicate that the main contributor to latency spikes and decreased throughput is within the LTE Enhanced Packet Core (EPC) or 5G Core Network (CN), and not the Radio Access Network (RAN). These studies also suggest that end-to-end throughput and latency often do not match desired 5G performance benchmarks due to improper buffer size and poor TCP congestion control in the CN. Diverse requirements of the various network slices in 5G-NR necessitate the meeting of two conditions: i) specifically designing the entire network path for slices, and ii) properly classifying traffic demands into correct network slices [4]. Network slicing has several benefits related to latency reduction and throughput improvements, which indirectly leads to a reduction in power consumption and support for network scaling. However, given the concerns of violating user privacy, we must also preserve the confidentiality of users' network activity while performing traffic classification as a precursor to network slicing [5]. While significant efforts have been made to design dynamic network slicing algorithms (see Sec. II-B), these works focus on functions in the RAN portion of the network.

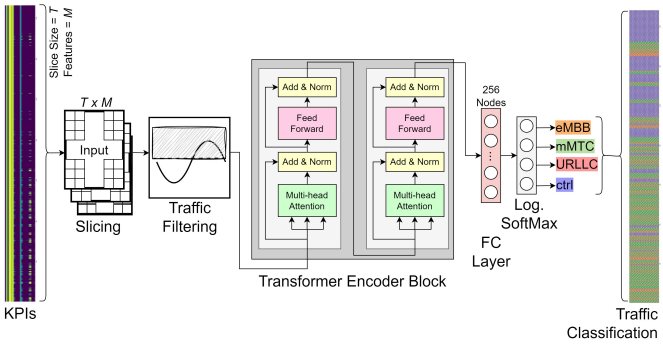


Fig. 1: Transformer-based 5G traffic classification system used in TRACTOR.

• **Transformer-based solution:** To address the challenges of identifying what traffic types are currently passing through the network, we propose utilizing machine learning (ML) algorithms–specifically transformer models [6]. Our ML model is packaged in an eXtended application (xApp) that runs within the near real-time (RT) radio intelligence controller, with the requirement that retrieving the key performance indicators (KPIs) requested from the gNB and transmitting the decision of the model back to gNB cumulatively takes less than 1s of time. Considering that cellular network users are inherently dynamic both in terms of movement and traffic usage, the xApp that performs such traffic classification and then assigns appropriate and optimal network resources, i.e., *slicing*, must keep up with the variations in traffic traces of these users. This raises interesting questions on whether to perform traffic classification in the xApp with fewer sets of KPIs (T) or to incur additional time overhead of increasing T

979-8-3503-7099-7/24/$31.00 ©2024 IEEE

for better classification outcomes.

Our contributions are as follows:

- We propose a transformer-based model that classifies different traffic patterns in compliance with 5G O-RAN standard-defined traffic classes and ML operations.
- We utilize a unique traffic filtering scheme that detects the presence of non-critical ctrl traffic in eMBB, mMTC, and URLLC traffic. We preprocess our training data and validate ctrl traffic detection at run time such that any ctrl traffic that exists within captures of other types of traffic is more likely to be correctly detected, ultimately improving our model performance.
- We implement the transformer model for traffic classification on an individual user basis, preserving privacy through observations made from KPIs. Similar to [7], we also implement our model over a variety of input temporal scales for traffic data collected both offline and online through an xApp deployed on Colosseum [8], the largest wireless RF network emulator with real radios in the loop. We analyze the effect of traffic pattern variations on classification to further mimic dynamic network usage.

II. Related Work

A. O-RAN Background

In [9], the authors provide a detailed presentation and insight into the O-RAN paradigm, architecture, interfaces, security, algorithms to deploy, and future research areas. As an early noteworthy implementation, [10] creates a prototype testbed for O-RAN for next-generation operations using USRPs. Their implementation is based on the software radio system RAN (srsRAN), focusing on the E2 interface that connects the RAN intelligent controller (RIC) to the other the RAN nodes by developing two xApps for the near-RT RIC that are for KPI collection and RAN slicing. The work in [11] introduces an open and virtualized prototyping platform for modern cellular systems, called SCOPE, which is a ready-to-use portable container that embodies various wireless deployments.

B. Traffic Classification

Traditional IP traffic classification relies on packet inspection, as seen in [12] and [13]. However, these methods fail when packets are encrypted at the network or data-link layer. To overcome this, statistical traffic properties were used to identify applications, which naturally led to the use of ML in traffic classification, including encrypted traffic. Generally, encrypted traffic classification techniques either use traffic flows defined by a 5-tuple (source IP, source port, destination IP, destination port, and transport-level protocol) [14] or the entire encrypted packet [15]–[17] as the input. Both of these options present a high risk to privacy and security, especially when paired with other information unique to a cellular environment, such as physical location. Furthermore, in an O-RAN system, the near-RT RIC does not have access to the entire packet, encrypted or otherwise.

The work in [18] selects the best-performing scheduling policy and resource block assignment in each network slice using deep reinforcement learning fed with data generated in the Colosseum emulator. Both UE assignment and the rewards in the DRL agents are based on knowing the traffic slice *a priori*. The authors in [19] aim to predict incoming traffic at the LTE base station (eNB) using supervised ML and test their model using simulated bursty LTE traffic data. The work in [20] predicts traffic type (among IM, web browsing, and video data) in an upcoming 5-minute period using the previous 3 hours of traffic data in a framework that consists of α-stable models, dictionary learning, and alternating direction method (ADM) using 7 million users' OTA 2G-4G application layer data. The authors in [21] perform network slicing and scheduling on an xApp that is based on srsRAN using policy-driven heuristic methods.

In [22], the authors present SenseORAN to detect radar pulses within the CBRS band using O-RAN networks. Their design paradigm of reusing existing cellular infrastructure with O-RAN-compliant sensing and communication slices aims to eliminate the need for dedicated spectrum sensors along the coastline as well as severe restrictions on the transmit power for modern LTE operators. The gNB uses a YOLOv3 module as an xApp that is trained to detect radar signals present within spectrograms generated from I/Q samples collected during the regular uplink cellular operation.

III. System Architecture

A. O-RAN Architecture

We implement a software-defined RAN using a framework based on SCOPE [11] for both the gNB and UE. Our near-RT RIC, a part of the ColO-RAN framework [23], provides vital functionalities such as support for the E2 interface for data collection and control communication with RAN nodes, alongside a sample xApp for collecting fundamental KPIs from the gNB. We add several key elements to this framework by creating a custom traffic generator and developing an xApp named **TR**affic **A**nalysis and **C**lassification **T**ool for **O-R**AN, or TRACTOR [7]. Our present study aims to enhance the functionality of this xApp and our contributions will be integrated into the public release of TRACTOR's code base.

B. ML Components

1) Dataset and Preprocessing: We perform our experiments using the same dataset collected in [7], which is based on a set of real 5G cellular traffic data obtained using PCAPDroid [24] and 5G handsets. Briefly, the dataset contains wireless packet captures of different applications, which we categorize under different types of wireless traffic, namely, eMBB, mMTC, and URLLC traffic. We replay all traffic on Colosseum via TRACTOR, emulating packets exchanged between the real UE and gNB and collecting the resultant stream of KPIs, which are used as the input to our model. To characterize features of ctrl traffic, we also separately collect a set of KPIs that represent instances of communication, where the UE is idle while still being associated with the gNB.

To form our input, we stack slices of T consecutive KPIs, which are sampled by the gNB every 250 ms. The values of T are given in Table I, and each slice contains $M = 17$ KPIs, resulting in a $T \times M$ 2D input, as shown in Fig. 1. The different T values provide insight into how traffic indicators evolve over a $T \times 250$ ms timespan; thus, T needs to be carefully chosen

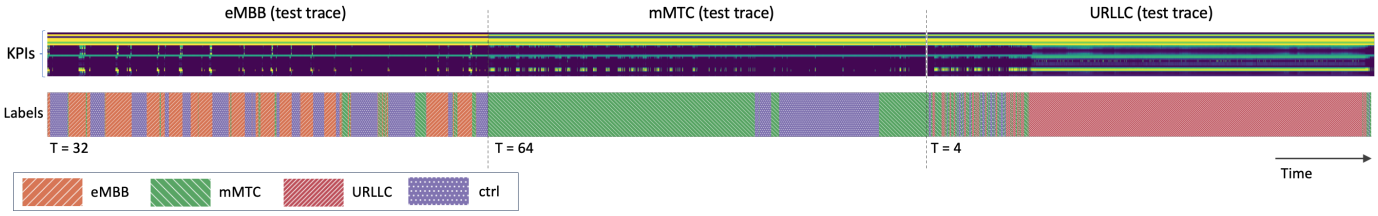


Fig. 2: A visualization of input KPIs for different traffic patterns (top) and the relative classifier output (bottom).

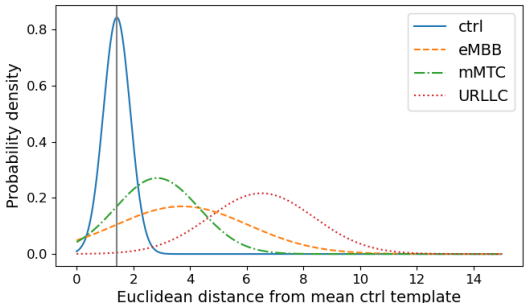


Fig. 3: Visualization of PDF for Euclidean difference for each traffic class ($T = 16$) when compared to mean ctrl template $\mathcal{T}$. The solid vertical line is the Euclidean distance value used as the threshold for ctrl relabeling: if the norm of the difference between $\mathcal{T}$ and an input sample falls below this threshold, the sample is automatically relabeled as ctrl traffic.

based on the timing requirements of the traffic classification task. A visualization of the input KPI stream is shown in Fig. 2. We use a sliding window approach with overlap of $T-1$ KPI samples to form slices of our inputs and make a new prediction every 250 ms once the first $T-1$ samples are collected. Input slices are preprocessed in the dataset via *min-max* normalization over each KPI feature in order to keep every input feature within a [0,1] range. We then randomize the order of slices and partition 80% of input samples to train the model and retain the other 20% for offline validation purposes. After training and validation datasets are defined, we obtain 111.6K training samples and 27.9K validation samples, covering the main 3 traffic types and ctrl classes.

For online testing purposes, we generate a separate set of traffic traces, one for each traffic type, and collect a new set KPIs that are used to test our models trained offline. While replaying each traffic type through Colosseum may result in different input patterns due to the varying nature of the network conditions, our intuition is that the classifier will be able to generalize well in the presence of unseen sets of KPIs, given the similar category of traces provided during training.

2) Data Filtering: During traffic generation, the gNB and UE do not actively communicate at all times; ctrl messages are regularly exchanged to maintain UE association. Additionally, depending on the UE's activities, ctrl traffic might be found in any of the other three traffic categories, so we consider ctrl as a fourth "class" of traffic that can be used to identify idle portions of communication. Due to this coexistence, it is inevitable that samples from a non-ctrl traffic type will be misclassified due to the presence of ctrl traffic; thus, we design a filtering and relabeling heuristic, which we call *Idle Traffic Removal* (ITR) heuristic, that we use to relabel idle portions of non-ctrl traffic samples and validate ctrl detections online via threshold comparison. This threshold, as shown in Fig. 3, is dynamically calculated during dataset generation by deriving the $\mathcal{N}(\mu^*, \sigma)$ probability density function (PDF) of Euclidean distance between all known ctrl samples and the $T \times M$ mean features of ctrl samples in our training dataset, based on the assumption that KPI values in ctrl traffic remain quasi-constant along T due to its idle nature. Once such a *mean ctrl template* and the PDF of Euclidean distance for ctrl samples is computed, we compute the same distance metric for each non-ctrl traffic sample: if the norm of difference is below the mean of PDF, we relabel the sample as ctrl, as it is statistically similar to the average ctrl sample configuration. We formulate the ITR heuristic as follows:

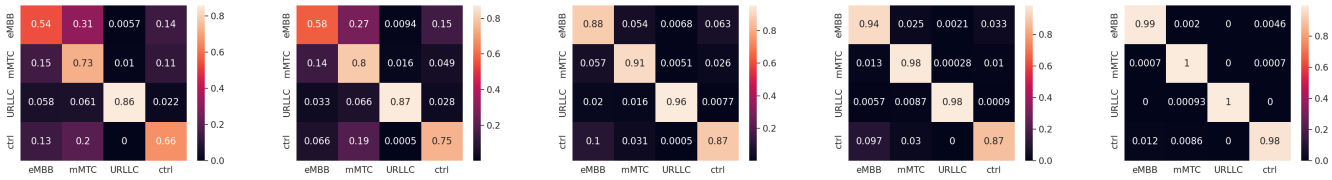
$$\text{ITR}(\langle \mathbf{X}, y \rangle, \mathcal{T}) = \begin{cases} \langle \mathbf{X}, \text{ctrl} \rangle, & \text{if } \|\mathbf{X} - \mathcal{T}\|_2 < \mu^* \\ \langle \mathbf{X}, y \rangle, & \text{otherwise} \end{cases}$$

where $\langle \mathbf{X}, y \rangle$ is a sample input and label pair, $\mathcal{T}$ is mean ctrl template used as reference and μ^* is the threshold, corresponding to the mean of normal PDF derived above.

T	Num. params	Inference time (ms)
4	164,704	0.7575 ± 0.0162
8	182,112	0.7640 ± 0.0419
16	216,928	0.7716 ± 0.0142
32	286,560	0.7264 ± 0.0293
64	425,824	0.7537 ± 0.0440

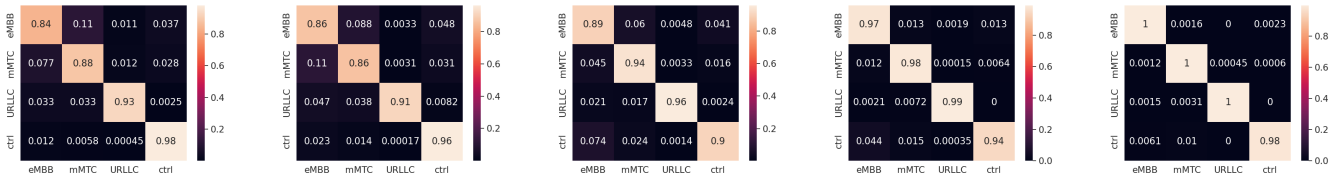
TABLE I: Parameters and inference time for each slice size.

3) Transformer: Since transformer models are designed to process sequential data, we treat the selected T consecutive time samples of the KPIs as a sequence. This means that we consider the KPIs sampled at a particular time as a *token* or element of the sequence and we have a contextualized representation of each token from the sampled KPIs at a given time step. After generating the dataset, we pass each input of T tokens through transformer encoder layers [6], as shown in Fig. 1. These representations are then flattened and fed to a Fully Connected (FC) layer that consists of 256 neurons followed by ReLU activation. Finally, a LogSoftmax output layer calculates the log probabilities of an input slice belonging to each traffic type. We train our model for 350 epochs using the Adam optimizer with learning rate 10^{-2}, reduced by an order of magnitude when a plateau in loss is encountered and an early stopping criterion to halt training when the loss does not improve for a sufficient number of epochs. The parameters and inference times for various sizes of T are shown in Table I, with all computations performed on an NVIDIA A100 GPU. Note that, as the average inference time is less than 1 ms



(a) (b) (c) (d) (e)

Fig. 4: Offline transformer confusion matrices for input sizes $T = \{4, 8, 16, 32, 64\}$ (left to right) without traffic filtering.



(a) (b) (c) (d) (e)

Fig. 5: Offline transformer confusion matrices for input sizes $T = \{4, 8, 16, 32, 64\}$ (left to right) with traffic filtering.

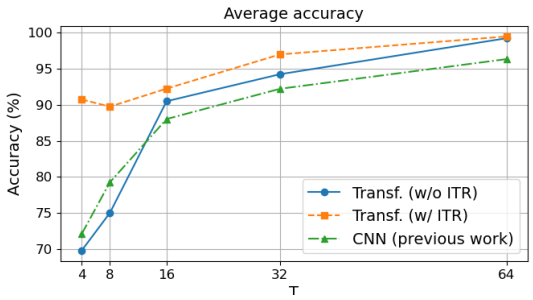


Fig. 6: Average offline accuracy of classifiers for different $T = \{4, 8, 16, 32, 64\}$. Performance is compared among transformers with and without the ITR heuristic and CNN model presented in [7].

on GPU and 1.5 ms on CPU, no latency is introduced by the inference operation when sampling KPIs every 250 ms; therefore, our approach is feasible for real-time deployment and suitable for xApp operations (i.e., event reaction time between 10 ms and 1 s).

IV. Results

We evaluate how the transformer-based classifier performs both offline and online when trained on both unfiltered and filtered traffic samples, considering configurations of slice length $T = \{4, 8, 16, 32, 64\}$.

A. Offline Results

We display the offline evaluation on validation data of proposed transformer model without traffic filtering in Fig. 4 and with traffic filtering in Fig. 5 for all T values considered. The proposed relabeling approach improves average classification accuracy up to 20% as shown in Fig. 6. Here, for comparison, we also include the results obtained via the convolutional neural network (CNN) from our previous work [7], which uses the same dataset for training and validation. The effectiveness of traffic filtering in our offline evaluation is clear, with an average improvement of $\sim 8\%$ over the transformer trained without traffic filtering and the CNN presented in the previous work when considering all T settings.

It is important to note that, while we focus on relabeling only idle portions of traffic at training time, we obtain a boost

Traffic Type	T = 4	T = 8	T = 16	T = 32	T = 64
Without Filtering					
eMBB	**66.34**	48.85	**65.25**	67.13	**95.28**
mMTC	**94.49**	**94.83**	75.10	**99.62**	90.79
URLLC	79.16	76.43	79.9	82.37	78.32
With Filtering					
eMBB	37.82	**58.73**	48.15	**81.17**	85.85
mMTC	83.75	85.80	**82.20**	85.85	**98.36**
URLLC	**87.73**	**82.32**	**83.80**	**85.76**	**78.50**

TABLE II: Online classification accuracy (%) with and without traffic filtering for $T = \{4, 8, 16, 32, 64\}$ using the same test traces in both sets of results; results in bold indicate best performance.

in classification accuracy across all traffic types, validating the need for more precise labeling of input traces to enhance the accuracy of the proposed classifier.

Regardless of filtering, it is clear that a longer T length significantly improves classification accuracy, from an average accuracy of $\sim 80\%$ for $T = 4$ to $99\%+$ for $T = 64$. However, we note that out of the non-ctrl traffic classes, URLLC is the least impacted by T size. This may be due to the temporal features of eMBB and mMTC traffic, which we observe to be "bursty" in nature and, thus, may be harder to distinguish from each other in comparison to URLLC, which typically involves long instances of active traffic exchange (e.g., long video chats and voice calls).

B. Online Results

The online evaluation results are shown in Table II, with separate results for traffic without and with filtering. We measure the average accuracy as follows:

$$acc = \frac{T_p}{N_s - N_{ctrl}}$$

where T_p is the number of true positives matching the traffic class considered to be groundtruth while replaying a given test trace, N_s is the total number of inputs to the classifier generated from the sequential KPIs and N_{ctrl} is the number of samples classified as ctrl by the classifier and whose label is validated via the ITR heuristic described in Sec. III-B2.

We observe a similar trend to our earlier work [7] in which there is no universal slice size that gives the best performance

for all classes, which highlights the challenges of deploying traffic classification methods that potentially consider different time scales to identify each traffic type. Nonetheless, these results are largely consistent with the offline results in that a longer T length generally provides higher classification accuracy. We also note that in most cases, with the exception of $T = 4$, traffic filtering does provide more stable performance and higher classification accuracy for a majority of traffic captures (two out of three non-ctrl classes for each classifier), and it always provides higher detection accuracy for URLLC traffic. The drop in performance in some traffic instances suggests that the proposed filtering method might also be sensitive to parameter T, possibly over-filtering samples that might resemble ctrl patterns (especially for lower values of T) and highlighting the need for more fine-grained threshold levels, possibly computed on a per-class basis. Finally, using the ITR heuristic, we observe that the models trained with proposed ctrl filtering approach show an average validation rate of 99.7% for all traffic types in test data, while the models trained without filtering only reaches 86.4%, further confirming that idle traffic detection is statistically more accurate with the proposed relabeling method.

V. Conclusions

Fulfilling the varied and competing promises of 5G and next-generation cellular networks will require an automated approach to traffic classification and slice assignment. With a full-stack srsRAN-based O-RAN traffic generation that natively supports incorporating ML models, we demonstrate the feasibility of using a transformer architecture with data filtering for traffic classification while simultaneously maintaining privacy and security via the use of KPIs instead of user traffic. Our proposed model reaches up to $99\%+$ of accuracy on offline data when longer temporal inputs are provided while reaching an average of $\sim 90\%$ accuracy across all slice lengths when tested on new traces collected online. Overall, our work demonstrates the potential for next-generation networks to deliver along the orthogonal performance axes of traffic classification and fine-grained user needs in a secure manner.

Acknowledgement

The authors gratefully acknowledge the funding support from the U.S. National Science Foundation (NSF) grants CNS 2112471 and CNS 1925601 along with the Roux Institute and Harold Alfond Foundation.

References

[1] V. K. Choyi, A. Abdel-Hamid, Y. Shah, S. Ferdi, and A. Brusilovsky, "Network slice selection, assignment and routing within 5g networks," in *2016 IEEE Conference on Standards for Communications and Networking (CSCN)*, 2016, pp. 1–7.

[2] A. Narayanan, X. Zhang, R. Zhu, A. Hassan, S. Jin, X. Zhu, X. Zhang, D. Rybkin, Z. Yang, Z. M. Mao *et al.*, "A variegated look at 5G in the wild: performance, power, and QoE implications," in *ACM SIGMCOMM*, 2021.

[3] D. Xu, A. Zhou, X. Zhang, G. Wang, X. Liu, C. An, Y. Shi, L. Liu, and H. Ma, "Understanding operational 5G: A first measurement study on its coverage, performance and energy consumption," in *ACM SIGCOMM*, 2020.

[4] Cisco, "QoS: Classification Configuration Guide, Cisco IOS XE 17," Cisco, Tech. Rep., April 2020.

[5] J. Groen, B. Kim, and K. Chowdhury, "The cost of securing o-ran," in *ICC 2023 - IEEE International Conference on Communications*, 2023.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[7] J. Groen, M. Belgiovine, U. Demir, B. Kim, and K. Chowdhury, "Tractor: Traffic analysis and classification tool for open ran," 2023. [Online]. Available: https://arxiv.org/abs/2312.07896

[8] L. Bonati, P. Johari, M. Polese, S. D'Oro, S. Mohanti, M. Tehrani-Moayyed, D. Villa, S. Shrivastava, C. Tassie, K. Yoder, A. Bagga, P. Patel, V. Petkov, M. Seltser, F. Restuccia, A. Gosain, K. R. Chowdhury, S. Basagni, and T. Melodia, "Colosseum: Large-scale wireless experimentation through hardware-in-the-loop network emulation," in *2021 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2021, pp. 105–113.

[9] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "Understanding O-RAN: Architecture, Interfaces, Algorithms, Security, and Research Challenges," *IEEE Communications Surveys & Tutorials*, 2023.

[10] P. S. Upadhyaya, A. S. Abdalla, V. Marojevic, J. H. Reed, and V. K. Shah, "Prototyping next-generation o-ran research testbeds with sdrs," 2022. [Online]. Available: https://arxiv.org/abs/2205.13178

[11] L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "Scope: An open and softwarized prototyping platform for nextg systems," in *ACM MobiSys*, 2021.

[12] T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *IEEE communications surveys & tutorials*, vol. 10, no. 4, pp. 56–76, 2008.

[13] Y. Li, B. Liang, and A. Tizghadam, "Robust Online Learning against Malicious Manipulation and Feedback Delay With Application to Network Flow Classification," *IEEE Journal on Selected Areas in Communications*, 2021.

[14] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Mobile encrypted traffic classification using deep learning," in *2018 Network traffic measurement and analysis conference (TMA)*. IEEE, 2018, pp. 1–8.

[15] M. Lotfollahi, M. Jafari Siavoshani, R. Shirali Hossein Zade, and M. Saberian, "Deep packet: A novel approach for encrypted traffic classification using deep learning," *Soft Computing*, 2020.

[16] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related," in *ICISSP*, 2016.

[17] S. Rezaei and X. Liu, "Deep learning for encrypted traffic classification: An overview," *IEEE Communications Magazine*, 2019.

[18] L. Bonati, S. D'Oro, M. Polese, S. Basagni, and T. Melodia, "Intelligence and learning in O-RAN for data-driven NextG cellular networks," *IEEE Communications Magazine*, 2021.

[19] T. N. Weerasinghe, I. A. Balapuwaduge, and F. Y. Li, "Supervised learning based arrival prediction and dynamic preamble allocation for bursty traffic," in *IEEE INFOCOM Workshops*, 2019.

[20] R. Li, Z. Zhao, J. Zheng, C. Mei, Y. Cai, and H. Zhang, "The learning and prediction of application-level traffic data in cellular networks," *IEEE Transactions on Wireless Communications*, 2017.

[21] D. Johnson, D. Maas, and J. Van Der Merwe, "NexRAN: Closed-loop RAN slicing in POWDER-A top-to-bottom open-source open-RAN use case," in *Proceedings of the 15th ACM Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization*, 2022.

[22] G. Reus-Muns, P. S. Upadhyaya, U. Demir, N. Stephenson, N. Soltani, V. K. Shah, and K. R. Chowdhury, "Senseoran: O-ran based radar detection in the cbrs band," *IEEE Journal on Selected Areas in Communications*, pp. 1–1, 2023.

[23] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "Colo-ran: Developing machine learning-based xapps for open ran closed-loop control on programmable experimental platforms," 2021. [Online]. Available: https://arxiv.org/abs/2112.09559

[24] E. Faranda, "PCAPdroid." [Online]. Available: https://emanuele-f.github.io/PCAPdroid/