












In-Network AR/CG Traffic Classification Entirely Deployed in the Programmable Data Plane: Unlocking RTP Features and L4S Integration

Alireza Shirmarz , Mateus N. Bragatto , Fábio Luciano Verdi ,
Suneet Kumar Singh , Christian Rothenberg , Gyanesh Patra , Gergely Pongrácz 
 Federal University of São Carlos (UFSCar), Sorocaba, SP, Brazil,
 NTNU Norwegian University of Science and Technology (NTNU), Norway,
 Universidade Estadual de Campinas (UNICAMP), Campinas, SP, Brazil,
 Ericsson Research, Budapest, Hungary
{ashirmarz, mateusbragatto, verdi}@ufscar.br — suneet.k.singh@ntnu.no
chesteve@dca.fee.unicamp.br — gergely.pongracz@ericsson.com

Abstract—This paper presents an in-network machine learning (ML) approach for classifying Augmented Reality (AR) and Cloud Gaming (CG) traffic using programmable hardware. Random Forest (RF) models are deployed in a P4¹ data plane capable of processing Real-time Transport Protocol (RTP) traffic features like Frame Size (FS) and Inter-Frame Interval (IFI) for efficient classification. The classifier marks AR and CG traffic with Explicit Congestion Notification (ECN) codepoints to integrate with the Low Latency, Low Loss, Scalable Throughput (L4S) features of the programmable switch. The RF model prioritizes AR/CG traffic using Differentiated Services Code-Point (DSCP) assignments and modular ECN marking. The classification performance is evaluated using accuracy, precision, recall, and F1-score, while time overhead is assessed based on nodal processing time incurred during deployment by replaying AR/CG traffic. The P4 implementations for V1Model and Tofino Native Architecture (TNA)² are all publicly available.

Index Terms—AR/CG traffic classification, Machine Learning, P4, Deployment, L4S

I. INTRODUCTION

The rapid growth of delay-sensitive application traffic, driven by advancements in real-time communication and interactive applications, poses significant challenges for network management. Applications such as AR and CG are exceptionally critical due to their stringent requirements for ultra-low latency and negligible packet loss [6], [14]. These competitive and real-time interactive services demand far greater sensitivity than other delay-sensitive traffic types [19].

AR and CG traffic, characterized by their real-time and competitive interactive requirements, necessitates stringent prioritization to mitigate the adverse effects of end-to-end latency. For AR, latency or packet loss can severely degrade the user experience, causing issues like motion sickness [10], [19], while for CG, even minor latency can drastically affect gameplay quality and competitiveness, reducing user satisfaction and Quality of

Experience (QoE) [5], [6], [19]. Network operators effectively manage queuing delays, making their involvement essential for ensuring these applications meet their stringent performance requirements. To address the stringent latency and bandwidth requirements of AR and CG traffic, network operators must implement prioritization strategies on the network operators bottleneck that ensure performance reliability and foster the AR/CG broader adoption across the Internet.

In order to provide operators with the ability to deliver ultra-low latency and reliable performance for delay-sensitive applications, recent research [5], [9], [14], [22] has advocated for the combination of the L4S architecture [2] with an automatic classifier for AR and CG traffic.

L4S achieves this through components like Explicit Congestion Notification (ECN) [15] and dual-queue Active Queue Management (AQM) [16], which facilitate scalable congestion control and compatibility with legacy systems, supporting incremental deployment. The L4S architecture uses the ECT(1) codepoint in the ECN field of the IP header to identify packets eligible for specialized treatment [2], [15]. Since the host typically handles ECT(1) marking in L4S, there are open challenges related to the risk of ECT(1) mismarking by malicious entities or improper configuration, which can undermine service guarantees and compromise performance for high-priority traffic [9], [12], [15]. This paper presents a pioneering implementation of a P4 data plane *RF-based AR/CG traffic classifier*, extending our prior work [19] by integrating its previously demonstrated accuracy and robustness into a P4 pipeline. AR and CG traffic are marked with *ECT(1)* to be directed to the L4S queue for higher QoS. DSCP marking further differentiates AR and CG, enabling tailored network prioritization and resource allocation by the network operators. While *L4S queuing performance* falls outside the scope of this study, our work focuses on evaluating the *deployed model classification accuracy* and *time overhead* dependent on traffic classification, ECT(1) marking, and DSCP assignment. Our testbed evaluation demonstrates deployment feasibility,

¹P4: Programming Protocol-independent Packet Processors

²The P4 v1model is a fixed reference pipeline used in simulators and P4Pi targets for standard packet processing. TNA is a flexible, programmable architecture used in Tofino ASICs for customized packet processing.

using V1Model on the P4Pi³ and TNA on the Tofino2 for implementation and performance assessment.

Moreover, our proposed approach incorporates RTP-based features to differentiate AR/CG traffic from other streaming traffic, improving the classification accuracy between AR and CG. This work is the first to deploy RTP FS and IFI within a P4 data plane for classification purposes. RTP frame features enhance differentiation between video streaming applications like AR and CG by making RTP data accessible in the data plane. This enables network devices to monitor RTP-based video traffic and classify packets for differentiated service levels. The fully In-network deployed automatic classification, ECN marking, and DSCP assignment accuracy and time overhead are evaluated in this paper.

Altogether, the key specific contributions of this paper can be presented as follows:

- We developed a retrained and pruned Random Forest (RF) model for classifying AR/CG traffic within the pipeline of P4 programmable switches (Tofino, P4Pi). The resulting P4 implementation supports pioneering functionalities related to Real-time Transport Protocol (RTP) features, including Frame Size (FS) and Inter-Frame Interval (IFI), directly within the data plane.
- We augmented the pipeline implementation after AR/CG classification to mark the identified traffic with Explicit Congestion Notification (ECN) codepoint ECT(1), turning AR/CG traffic eligible for L4S treatment. This marking module operates independently of the host and is compatible with the L4S architecture. Furthermore, the system assigns Differentiated Services Code Point (DSCP) values, specifically EF(46) for AR traffic and AF41(34) for CG traffic, allowing for distinct traffic prioritization.⁴
- We experimentally evaluated the classification performance and time overhead of the deployed AR/CG classification and ECT(1) marking modules on a Tofino2 testbed. Beyond a proof of concept hardware implementation, the obtained results serve to unlock the potential of in-network AR/CG traffic classification and marking integrated into L4S-enabled networks.
- All P4 code related to both the V1Model and TNA architectures is publicly available to foster research reproducibility.⁵

The paper is organized as follows. Section II reviews related works in ML model deployment on data plane for CG and AR traffic. Section III details the proposed architecture and the algorithm for deploying the AR/CG RF classification model and ECT(1) marker on the data plane using P4, V1Model, and TNA. Section IV evaluates the RF model for classification, along with the ECT(1) marking and DSCP assignment components, to assess their feasibility on programmable switches. Section V addresses the discussion, including challenges, key

insights, and limitations of the proposed approach, alongside evaluation constraints. Finally, Section VI concludes the paper.

II. RELATED WORK

The advent of programming languages such as P4 [1] and Network Programming Language (NPL), coupled with the integration of ML techniques for traffic classification [13], [23], has significantly advanced the capabilities of programmable data planes. ML applications in programmable networks span various use cases, including identifying heavy-hitter flows, mitigating Denial-of-Service (DoS) attacks, and anomaly [23].

Several studies have focused on CG traffic classification. Graff et al. [4] analyzed ML-based approaches to distinguish CG from Non-CG traffic using features such as packet size, inter-arrival time (IAT), and total packet data. However, this work lacked hardware deployment, limiting its practical applicability. Ky et al. [7] introduced a hybrid architecture combining Decision Tree (DT) and unsupervised ML for CG traffic classification. In this approach, six traffic features were extracted on a P4-programmable Tofino switch, excluding features such as standard deviation due to hardware constraints. Classification was performed on an NFV node using an Unsupervised Anomaly Detection (USAD) model, while the Tofino switch handled feature extraction and communication with a Linux-based controller.

MATADOR [21] implemented a DT model on the TNA architecture for CG traffic classification. The system used features like packet count, packet size, and inter-packet gap (IPG) for binary classification of CG versus Non-CG traffic. This deployment demonstrated the feasibility of using programmable hardware for traffic classification but did not address more granular or multi-class scenarios (such as AR traffic).

AR traffic has received limited attention in hardware implementations. While some studies [8], [11], [17], [20] have explored AR-specific traffic patterns, gaps remain in feature extraction and hardware deployment. The unique real-time requirements of AR traffic call for more sophisticated and resource-efficient solutions, especially in programmable switches.

Building upon our previous work [19], we address these limitations by developing and deploying an enhanced RF model on a programmable Tofino switch. Unlike prior efforts [4], [7], [21], which primarily focused on binary classification of CG/Non-CG traffic, our method introduces a multi-class scheme that distinguishes ‘AR’, ‘CG’, and ‘Other’ (non-AR/CG) traffic categories. To overcome the limitations of packet-based features, we incorporate RTP-specific features, significantly improving classification accuracy and robustness. Furthermore, this work pioneers deploying RTP frame features directly within the data plane while enabling DSCP marking to ensure seamless compatibility with L4S architectures.

Our system optimizes and prunes the RF model for deployment entirely in the dataplane, balancing hardware constraints with high classification accuracy for AR/CG traffic. We implement three key functions within P4 pipelines using V1Model and TNA architectures: (1) feature extraction

³<https://eng.ox.ac.uk/computing/projects/programmable-hardware/p4pi>.

⁴<https://www.iana.org/assignments/dscp-registry/dscp-registry.xhtml>.

⁵<https://github.com/dcomp-leris/ARCG-DP-Deployment.git>

and computation, (2) RF-based AR/CG traffic classification, and (3) marking AR/CG packets with ECT(1) and assigning unique DSCP codepoints. This modular deployment operates independently of the host, seamlessly integrating with the L4S queuing system on P4Pi and Tofino2. The automated ML-based AR/CG classifier was evaluated on Tofino2 by replaying AR/CG and non-AR/CG traffic, measuring classification accuracy and the associated time overhead.

III. AR/CG CLASSIFICATION MODEL DEPLOYED ENTIRELY IN THE DATA PLANE

As illustrated in Fig. 1, packets originated from various applications (e.g., web browsing, VoD, video conferencing, streaming, RDP, AR, and CG) enter network devices, each demanding specific levels of service, such as low latency, minimal loss, and scalable throughput. These requirements can be effectively addressed using the L4S mechanism. L4S prioritizes traffic marked ECT(1), enabling low latency, minimal loss, and scalable throughput.

We propose an *automatic AR/CG flow classification* system that *marks packets' ECN* with *ECT(1)* independently of the host, redirecting them to the L4S queue for preferential treatment (Fig.1). This approach can enhance AR/CG performance using L4S without host dependency but introduces processing time overhead within the programmable network device that we consider in this work for assessment. Each traffic type ideally requires priority-based treatment, necessitating more complex scheduling strategies. This may also require heterogeneous solutions to address the resource limitations of Tofino switches. While our work integrates seamlessly with L4S—providing output as input for L4S—the design of L4S scheduling and the evaluation of AR/CG QoS and QoE have been left for future work.

To implement the model entirely in the data plane, we utilize the Math Unit, Hash, MAUs, and Registers available in programmable switches (P4Pi and Tofino2). These components enable the feature extraction which is a key task for AR/CG traffic classification. In this work, different from the previous ones that use only packet-level features, we use the following features: PS, IPI, FS and IFI. Based on the RF *flow* classifier's output, *packets* are marked with ECT(1) and appropriate DSCP codepoints corresponding to their flow type. These markings enable integration with the L4S queuing mechanism, as illustrated in Fig. 1, ensuring modularity and compatibility with the L4S system before packets enter the queuing stage.

A. Methodology

In this work, we optimize and retrain the DT and RF models introduced in our previous research for AR/CG traffic classification, utilizing the same training dataset [19]. The refined model is implemented using P4 pipelines, which include the ingress pipeline, match/action (M/A) processing, and the egress pipeline. *Feature extraction* and the *optimized RF model*, chosen for its higher accuracy, are fully deployed within the data plane. The deployment was initially conducted

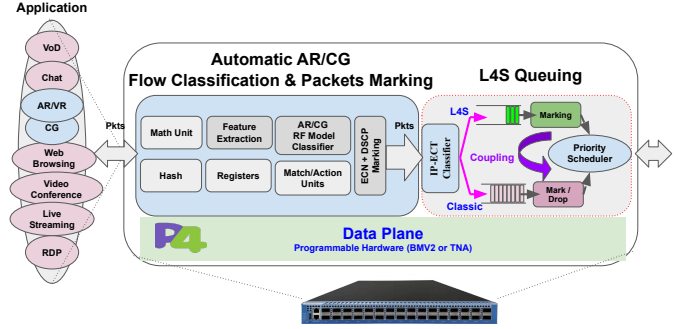


Fig. 1: AR/CG Traffic Classification & L4S Queuing.

on the *VIModel* and subsequently extended to the *TNA* for Tofino switches which both are publically available⁵.

1) *Development Strategies*: The *feature extraction* and *AR/CG RF model classifier* are fully developed in the data plane to compute flow features and classify flows into AR, CG, or other (non-AR/CG) types. The *ECN and DSCP marking* component ensures ECT(1) and DSCP codepoints are assigned to AR/CG packets at line rate (Fig. 1). While this enables in-line classification and marking, it introduces *processing overhead* and challenges in maintaining *classification accuracy*, requiring a balance between *accuracy* and *processing efficiency*. To address this, we employ three key strategies: (a) *Packet Cloning*, where packets are duplicated during ingress, M/A, and egress, enabling independent classification and feature extraction while slightly increasing resource usage but maintaining smooth latency for original packet forwarding; (b) *Flow-based RF Model for Classification*, which classifies traffic at the flow level instead of per packet, improving accuracy and leveraging RTP features to enhance classification performance in video streaming applications; and (c) *Shared Registers*, enabling synchronization between cloned and original packets for flow classification and accurate ECT(1) marking of original traffic in the data plane.

2) *Algorithm*: The algorithm for automatic flow classification and packet marking is detailed in Algorithm 1.

Algorithm 1 is implemented in the P4 pipelines. Incoming packets are processed to extract the $Flow_{ID}$ from the *destination IP*, *destination port*, and *transport protocol*, which is then hashed using CRC_8 to index a shared register for flow classification and packet marking. Packets are cloned using egress-to-ingress⁶ recirculation for feature computation (PS, IPI, FS, IFI) and flow classification, with results stored in the register at the $Flow_{ID}$ index. Original packets retrieve the classification result from the register for marking with ECT(1) and DSCP codepoints (46 for AR, 34 for CG), while unclassified (0) or Other (3) packets remain unchanged. For evaluation, DSCP codepoint 50 is assigned to "Other" packets

⁶We use egress mirroring in the Tofino2 to clone the packets in the egress deparser to recirculate the post-deparsed packets to the ingress for packet processing and we drop the cloned packets in the ingress.

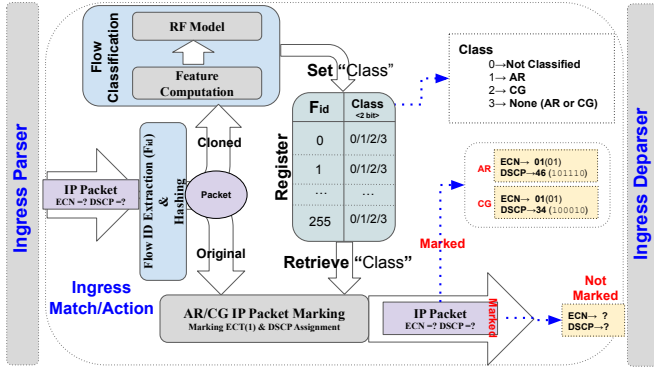


Fig. 2: Ingress Pipeline for Flow Classification & Packet Marking.

to distinguish classified from unclassified traffic.

ECN marking with ECT(1) is performed in the ingress, while L4S queue allocation occurs in the ingress after the proposed AR/CG automatic ECT(1) marking component. To enable flow-based classification, packets are cloned at the egress and recirculated to the ingress, where features (PS, IPI, FS, IFI) are extracted and sent back along with $FlowID$. As shown in Fig. 2, the ingress parser processes both original and cloned packets, with the cloned packets corresponding to previous packets (introducing a lag) and filtered by ingress port in Tofino2.

A shared register, consisting of 256 segments with 2-bit fields, synchronizes flow-based feature extraction/classification with packet-based marking. Classification results are stored in a $\langle Key, Value \rangle$ structure, where $FlowID$ serves as the key and flow class (0: unclassified, 1: AR, 2: CG, 3: Other) as the value. Original packets reference this register for ECN marking and DSCP assignment (46 for AR, 34 for CG, 50 for Other). Unclassified flows do not disrupt marking/DSCP assignment, as they remain unchanged. To assess the impact of cloning, unclassified packets are distinguished from classified ones. In the ingress, cloned packets aggregate and store features using

Algorithm 1 Automatic AR/CG Flow Classification & Packets Marking Algorithm

```

1: Input: Packet  $P$ 
2: Output: Marked packet  $P$  with ECT and DSCP
3: // *** Flow Identification & Traffic Mirroring *** //
4: Extract  $FlowID$  &  $Hash$  ( $CRC_8$ )
5: Clone Packets
6: // *** Packets Processing in Parallel Pipelines *** //
7: if Packet is cloned then
8:   Compute flow Packet/RTP features (e.g., IPI, IFI)
9:   Classify flow using Match-Action Table ( $MAT$ )
10:  Set flow class in register (AR: 1, CG: 2, Other: 3)
11: else
12:   Retrieve Packet Class using  $flowID$ 
13:   if Flow class == AR then
14:     Set  $P.ECN = 1$ ; Set  $P.DSCP = 46$ 
15:   else if Flow class == CG then
16:     Set  $P.ECN = 1$ ; Set  $P.DSCP = 34$ 
17:   else
18:     No marking (retain original ECN & DSCP)
19:   end if
20: end if

```

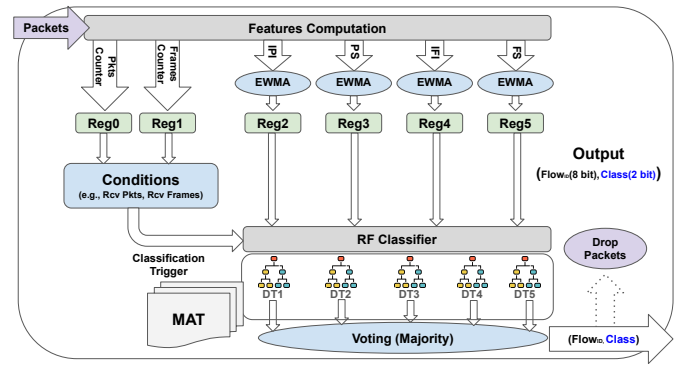


Fig. 3: Feature Computation & Flow Classification.

Exponentially Weighted Moving Average (EWMA), triggering the RF model once conditions are met to classify the flow and update the shared register. As shown in Fig. 3, features are extracted in the egress to preserve fidelity, while classification and feature computation (EWMA, storage, condition checks) occur in the ingress to avoid cloning-induced delays.

B. DT/RF Model Pre-processing for P4 Pipeline Deployment

We optimized the DT and RF models using the datasets from [19], tuning max_depth , $min_samples_split$, and $n_estimators$ through an iterative grid search (range 1–10) to balance performance and resource efficiency. The optimal parameters— $max_depth = 3$, $min_samples_split = 2$, and $n_estimators = 5$ —were selected to achieve the highest accuracy with minimal resource usage, with each RF tree having at most 15 nodes. The pruned models were trained, validated on 10% of the dataset, and saved as a .pkl file for further processing. As shown in Table I, RF was chosen for deployment due to its superior accuracy and efficiency.

DT/RF Rules Extraction. The serialized model file (.pkl) is used to extract hierarchical if-then rules based on feature values. Each feature threshold determines whether to continue to another condition or reach a final class. Using Python, we developed a script to process the RF model (or a single-tree DT) and generate if-clauses, thresholds, and their sequence, forming a rule set for each tree.

Threshold Tuning. Thresholds for each feature (PS, IPI, FS, IFI) are adjusted to meet hardware constraints, ensuring integer values for PS and FS and converting IPI/IFI to nanoseconds ($\times 10^9$) for hardware timestamps consideration. Training thresholds are rounded for precision. To optimize P4 deployment, rules are simplified and merged, minimizing if-

TABLE I: Pruned DT/RF Model Evaluation

Model	Class	Acc (%)	Prec (%)	Rec (%)	F1_s (%)
DT	AR	86	88	98	92
	CG		67	97	79
	Other		90	78	83
RF	AR	94	93	98	95
	CG		80	97	88
	Other		93	88	90

then conditions. The final rule set is implemented in a match-action table in P4.

Classification Outcome Encoding. Two bits are allocated for the RF classifier output, supporting four values: 0 (unclassified), 1 (AR), 2 (CG), and 3 (Other). This encoding is used for metadata read/write and shared register operations, as shown in Fig. 2.

Flow Classification Outcome. To aggregate the final classification outcome, the majority vote among the five DT outputs determines the RF flow classification, as shown in Fig. 3.

C. Feature Extraction, Computation & RF Classification

Packet-based and RTP frame-based features are extracted in the egress from original packets to preserve timing fidelity, while feature aggregation for RF classification occurs in the ingress using cloned packets and their headers. Extracted features are recirculated to the ingress pipeline using an egress mirror header, which carries PS, IPI, FS, and IFI in addition to packets common headers. This header is parsed in the ingress for feature computation and preparation for the RF flow classifier. Aggregated features (Registers 2–5) are computed using EWMA as RF inputs. Since classification requires a minimum number of packets or frames, Register0 (packet count) and Register1 (RTP frame count) track data availability (Fig. 3). A trade-off exists between accuracy (more packets/frames improve classification) and time overhead (increased processing time), making it configurable. For accuracy assessment, we use the minimum required frames and packets to balance performance and latency. The classification output is encoded in 10 bits (*FlowID* (8 bits) and *Class* (2 bits)). The five DTs of RF are implemented using MATs in the ingress for both V1Model and TNA.

Packet-Based Features. On packet arrival, the total IP packet length (*PS*) is extracted in the egress and recirculated to the ingress via the mirror header. The (*IPI*) is extracted as $IPI = T(Pkt_i) - T(Pkt_{i-1})$, where $T(Pkt_i)$ and $T(Pkt_{i-1})$ are the current and last packet timestamps.

RTP Frame-Based Features. The RTP protocol⁷ is a standard for delivering audio and video over IP networks, enabling real-time streaming through timestamping, sequence numbering, and payload type identification [18]. RTP packets are encapsulated in the UDP payload with a 12-byte header, which remains unencrypted even in Secure Real-time Transport Protocol⁸ (SRTP). We define the RTP header in the UDP packet payload first 12 bytes in P4. In video streaming, encoded frames span multiple UDP packets, with the final packet in a frame marked by the Marker bit (set to 1), while other UDP packets remain 0. Unlike IP and transport layers, which explicitly define protocols (e.g., UDP (17), TCP (6)), RTP lacks a deterministic identifier and must be extracted from the UDP payload. We utilize the Marker bit to detect RTP frame boundaries for video streaming across different encoding.

To extract *FS* and *IFI*, we use the *Marker*, *sequence number*, and *timestamp* fields from the RTP header. The *Marker* bit

indicates the last packet of an RTP frame. The difference in *sequence numbers* between two consecutive packets with the *Marker* set determines the number of packets (frm_{pkts}) in a frame.

The difference between timestamps of consecutive packets with the *Marker* set provides the *IFI*.

The number of packets in frame i , denoted as $frm_{pkts}(i)$, is calculated as the difference between the sequence numbers of the current frame $frm_{seq}(i)$ and the previous frame $frm_{seq}(i-1)$, as shown in Eq. 1.

$$frm_{pkts}(i) = frm_{seq}(i) - frm_{seq}(i-1), \quad i > 0 \quad (1)$$

The frame size (*FS*) is calculated by multiplying $frm_{pkts}(i)$ with $PS_{frm_i}^m$, the size of the last UDP packet in $frame_i$ where the *Marker* is **True**, as shown in Eq. 2.

$$FS_i = frm_{pkts}(i) \times PS_{frm_i}^m, \quad i > 0 \quad (2)$$

IFI is calculated as the difference between the *RTP* timestamps of successive frames, $T(frm_i)$ and $T(frm_{i-1})$, as shown in Eq. 3.

$$IFI_i = T(frm_i) - T(frm_{i-1}), \quad i > 0 \quad (3)$$

Dynamic Time Window. Packet-based features are typically extracted within a fixed time window, but RTP's variable frame size complicates this in flow-based classification. IFI requires at least three RTP frames, while packet-based features depend on the number of packets. To balance accuracy and latency, we introduce a trigger mechanism that activates RF classification when either 3 RTP frames or 20 packets are collected. The Marker bit identifies RTP frames, which average 5.6 packets per frame⁹. If no frames appear within 20 packets, classification is triggered. These parameters are adjustable based on accuracy and time overhead requirements.

EWMA Computation. We use an EWMA with $\alpha = 0.95$ (Eq. 4). In V1Model, this is implemented using addition and bitwise shifts to replace multiplication and division with logical operations. In contrast, TNA leverages the Low Pass Filter (LPF) extern in 'Sample Mode' to approximate the EWMA. While fixing $\alpha = 0.95$ in 'Sample Mode' is challenging, it is essential to ensure filtering accuracy on Tofino. In Eq. 4, S_t represents the value stored in the register, while S_{t-1} denotes the current value of the feature.

$$S_t = \alpha \times S_t + (1 - \alpha) \times S_{t-1}, \quad 0 < \alpha \leq 1 \quad (4)$$

We set $\alpha = 0.95$ to ensure the EWMA aligns with threshold averages, providing consistent and accurate flow classification results.

RF Model Classifier. The RF classifier consists of five DTs, with the final classification determined by majority voting. These DTs process the EWMA of features (*PS*, *IPI*, *FS*, *IFI*) extracted from cloned packets and trigger classification when sufficient *packets* or *RTP frames* are collected (Fig. 3). DTs

⁷<https://datatracker.ietf.org/doc/html/rfc3550>

⁸<https://datatracker.ietf.org/doc/html/rfc3711>

⁹This average number is based on the RTP frames we examined the available PCAPs.

apply rules from Subsection III-B to classify flows, and a counter determines the majority class. The *result* is stored in the shared register using $FlowID$ as the index.

D. P4 Code Resources V1Model vs TNA

We first implemented the P4 code in the V1Model on P4Pi and then migrated it to TNA for Tofino2, ensuring compatibility with Tofino2's syntax, architecture, and hardware constraints. Key adaptations include: (a) replacing the custom EWMA from V1Model with the LPF extern in TNA; (b) calculating FS as the sum of UDP packet lengths between two packets with the RTP marker set to **True**, due to the lack of multiplication support in Tofino2; and (c) Tofino2's bottleneck is the Traffic Manager (TM) with a throughput of 6 BPPS (4×1.5 GHz) supporting 12.9 Tbps. To mitigate the impact of mirrored traffic on bandwidth, cloned packets are dropped in the ingress before reaching TM. (d) Given Tofino2's limited per-stage resources, processing is structured across its 20 stages: header parsing and flow ID extraction (2 stages), feature extraction (4 stages), EWMA computation (1 stage), RF classification with 5 decision trees (5 stages), ECT(1)/DSCP marking (4 stages), queue management (3 stages), and deparsing (1 stage).

IV. EXPERIMENTAL EVALUATION

We now turn the attention to the *classification performance* and *time overhead* of deploying the RF-based AR/CG classifier with ECT(1)/DSCP assignment in actual P4 programmable data planes. To assess accuracy at both *flow* and *packet* granularity, we initially developed the P4 code in P4Pi and migrated the P4 code to Tofino2. The evaluation focuses on two key aspects: performance impact due to hardware constraints and simplifications and the additional processing time introduced by the deployed classifier and marking components.

A. Testbed

Fig. 4 illustrates our testbed consisting of two servers connected to a Tofino2 programmable switch. Server 1 (Intel® Xeon® D-1518, 64 GB RAM, Ubuntu 18.04) uses *tcpreplay* to inject PCAP traces (AR, CG, and other applications) at speeds up to 10 Gbps, while Server 2 (Intel® Core™ i7-7700K, 32 GB RAM, Ubuntu 24.10) captures incoming traffic using *tshark* and processes INT data with the implemented Python scripts.

B. Experimental Design & Procedure

To evaluate the P4 pipeline model, we define two terms: *deployed* and *undeployed*. The *deployed model* integrates feature extraction, classification with the pruned RF model, ECT(1) marking, and DSCP assignment directly in the P4 data plane, running on hardware (Tofino2) (Fig. 4). The *undeployed model*, in contrast, uses the pruned RF model stored as a .pkl file. Features are manually extracted and provided to the model using Python for prediction, serving as a *baseline* to compare classification performance with the deployed model.

Each packet's *nodal processing time* is computed as $T_{(egress,i)} - T_{(ingress,i)}$, which refers to the timestamp of

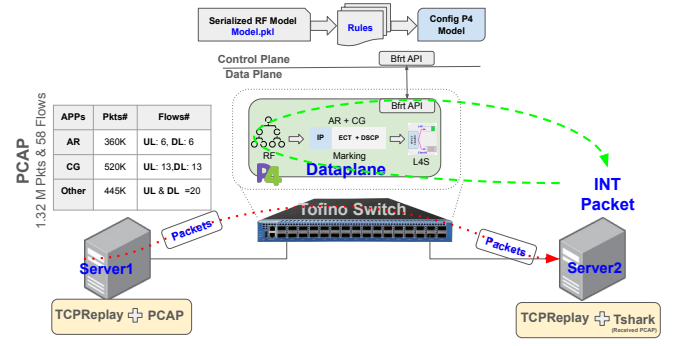


Fig. 4: Testbed for experimental evaluation. The red dotted line (red) shows the traffic path by TCPReplay flows. The dashed line (green) represents the path of INT packets.

TABLE II: Network Traffic Flows and Packets

Class	Direction	PCAPs	Flows	Packets
AR [19]	UL	Stream1-6	6	180k
	DL	Stream7	6	180K
CG [4], [19]	UL	ex10 - ex23	13	90K
	DL	ex10 - ex23	13	430K
Other [3], [4]	UL/DL	WB,RDP, VoD,VC,LS	20	445K

egress and ingress pipeline in P4 related to packet i . This per-packet value is accumulated in a register, and the cumulative packet count is likewise recorded. We consider the time overhead of the *deployed classification* against a baseline P4 program that performs simple forwarding to measure the nodal processing time as the base line time.

1) *Experiment Workflow*: To conduct the experiment, we deploy the P4 implementation on a programmable switch. First, the P4 code is compiled using *p4c* for V1Model and *bf-p4c* for TNA, then deployed the binary file on the programmable switch (Fig. 4). The P4 code includes components for RF model-based classification, ECT(1) marking, DSCP assignment, and INT packet processing to collect telemetry data. We prepared PCAP files containing traffic for AR, CG, and other applications using previously collected datasets. AR traffic was sourced from [19], while CG traffic was obtained from cloud-gaming traces [4] [19]. Traffic for other (non-AR/CG) applications, such as web browsing (WB), Remote Desktop Protocol (RDP), Video on Demand (VoD) (e.g., YouTube), video conferencing (VC), and live streaming (LS), was sourced from CIC datasets [3], [4]. For the experiment, AR and CG flows were organized into uplink (UL) and downlink (DL) traffic, and the PCAP files were sliced into continuous segments of consecutive packets. The number of packets and flows (distinct destination IPs and ports) per direction used in the experiments is detailed in Table II. To ensure generalization, the experiment was repeated three times, with each iteration using different flows and packets for AR and CG in both UL & DL directions. However, for the "other" traffic class, the same PCAP file was used for both directions.

TCPReplay is used to replay the PCAPs listed in Table II,

with received packets captured on Server 2 using Tshark for analysis. The replay rate matches the original capture rate, and each direction (UL & DL) takes approximately 13 minutes (780 seconds) to complete. Each experiment includes 58 flows and 1.325 million packets, extracted from referenced datasets. To ensure reliable and comprehensive results, the experiment is repeated three times with different PCAP slices, each containing 1.325 million packets divided into UL & DL traffic (Table II).

2) *Experiment Workflow*: The PCAP files captured on Server 2 are analyzed to evaluate classification performance, while INT is used to collect the programmable switch's *nodal processing time* and *round-trip time (RTT)*. INT packets are sent and received by Server 2 at a rate of 1 packet per second (PPS) and recirculate through the switch's loopback port, collecting accumulative *nodal processing time* and the total number of processed packets to compute the average nodal processing time in the server 2 with these two values for each INT packet. Each INT packet carries four fields: (a) $T_{(sent,i)}$, the timestamp recorded by Server 2 when sending the packet; (b) $\Delta t = T_{(egress,i)} - T_{(ingress,i)}$, the cumulative nodal processing time; N , (c) N , the total number of processed packets; and (d) $T_{(received,i)}$, the timestamp recorded by Server 2 upon receiving the packet. As the INT packet passes through the switch, it acquires Δt and N from the data plane before looping back to Server 2. The RTT for INT packet the index i is calculated as $RTT_i = T_{(received,i)} - T_{(sent,i)}$, and the average nodal processing time, derived from each INT packet, represents the real-time performance of the deployed model (e.g., Tofino2 or P4Pi).

C. Classification Performance

1) *Metrics*: We evaluate classification performance using accuracy, precision, recall, and F1-score. True Positive (TP) represents correctly classified packets of a given class, True Negative (TN) represents packets correctly identified as not belonging to the class, False Positive (FP) represents packets incorrectly classified as belonging to the class, and False Negative (FN) represents packets of the class incorrectly classified as another one. These definitions are extended to evaluate performance across all three classes (AR, CG, and Other).

2) *Analysis Methodology*: The PCAP file collected on Server 2 using Tshark is compared with the source PCAP to evaluate TP and FP for AR, CG, and other traffic. AR packets are marked with ECT(1) and DSCP=46, CG packets with ECT(1) and DSCP=34, other packets with DSCP=50, and unclassified packets default to DSCP=0. UL & DL traffic are analyzed separately to assess classification performance across directions. In the UL, AR traffic includes 180K packets (6 flows), CG traffic has 90K packets (13 flows), and other traffic accounts for 445K packets (20 flows). In the DL, AR traffic remains 180K packets, CG traffic increases to 430K packets, and other traffic stays at 445K packets. Correct DSCP assignment is expected for all classes. The TP for ECT(1) includes AR and CG packets, totaling 270K in the UL and

TABLE III: ECT(1) Marking Analysis: TP, FP, and Not-Classified Packet Percentages and Counts.

	Perc. (%)		# Pkts		Total pkts ($\times 3$)	
	UL	DL	UL	DL	UL	DL
TP (AR+CG)	89	92	720K	1683K	270K	610K
FP (Others)	15	9.2	200K	122K	445K	445K
Not-Class.	9	4.7	193K	86K	715K	1055K

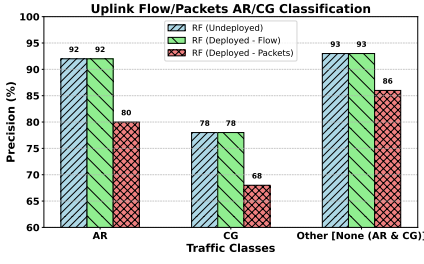
610K in the DL, across 19 flows. The FP is determined by "other" class packets incorrectly marked with ECT(1). Classification performance metrics for AR, CG, and other traffic are calculated for each replay slice, and the average over three slices is reported.

3) *AR/CG Classification Performance*: To demonstrate classification performance, precision, recall, and F1-score metrics are reported for UL & DL traffic in Fig. 5. The average results for the undeployed (baseline) and deployed RF models are compared at both flow and packet levels, highlighting the deployment's impact on packet processing. Subfigures illustrate precision (Subfigures 5a and 5d), recall (Subfigures 5b and 5e), and F1-score (Subfigures 5c and 5f) for UL & DL traffic, providing a clear comparison of performance and overhead across key metrics.

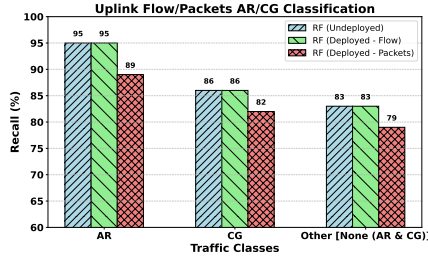
4) *ECT(1) Marking Performance*: The ECT(1) marking performance is summarized in Table III, where true positives (TP) represent correctly marked AR/CG packets, while false positives (FP) correspond to other traffic incorrectly marked with ECT(1). Unclassified AR/CG packets are also reported for comparison. The table presents three key metrics: (a) Percentage (%) of packets marked correctly, incorrectly, or left unclassified. (b) Number of packets categorized by UL and DL, and (c) total packet count over three experiment runs. These results demonstrate the effectiveness of the deployed automatic ECT(1) marker, highlighting its accuracy and potential impact on L4S queueing.

5) *Deployed vs Undeployed Classification Performance*: Using DSCP codepoints for AR (46), CG (34), and others (50), the deployed and undeployed models achieve equivalent flow-level classification accuracy, as shown in Fig. 5. UL traffic has higher precision than DL, with a 7% precision drop for AR and CG and a 9% drop for others in DL. Recall decreases slightly by 2% across all classes from UL to DL, while F1-scores drop by 4% for AR, 5% for CG, and 6% for others. Despite these minor losses, classification remains highly accurate. However, the longer duration of AR and CG flows amplifies the impact of misclassification, which can be mitigated by more frequent classification during long flows to enhance reliability.

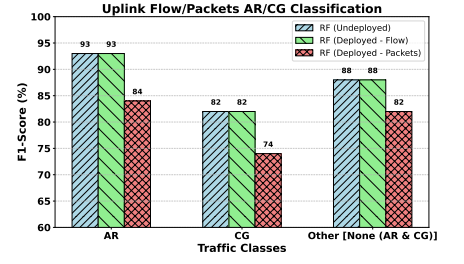
6) *Packet Cloning*: We employ egress mirroring to clone packets and process them in the ingress pipe, processing flow feature extraction, classification, and ECT(1)/DSCP marking independently. While Tofino2's 12.9 Tbps capacity minimizes the throughput loss effect, cloned packets are dropped after feature extraction/classification in the ingress pipeline to prevent TM bottlenecks. Future work will improve cloning with *selective packet mirroring* instead of duplicating all packets



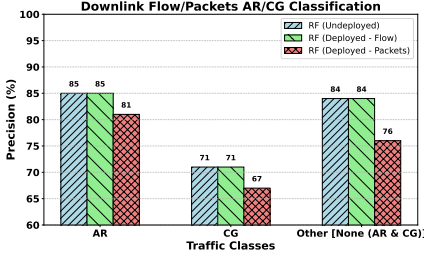
(a) Uplink Traffic Classification Precision



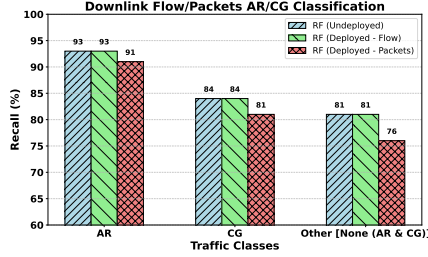
(b) Uplink Traffic Classification Recall



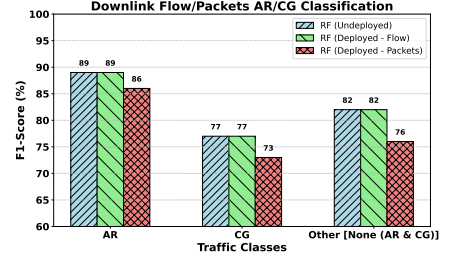
(c) Uplink Traffic Classification F1-Score



(d) Downlink Traffic Classification Precision



(e) Downlink Traffic Classification Recall



(f) Downlink Traffic Classification F1-score

Fig. 5: Uplink/Downlink Traffic Classification Performance, Precision, Recall, and F1-Score for Deployed vs Undeployed RF Model. Classification at flow and packet granularity for the deployed model.

TABLE IV: Classification/Marking Overhead vs Without Classification Data plane

	Avg	Std	P90	P95	P99
Without Class.	1.98^{-7}	1.44^{-7}	3.44^{-7}	3.47^{-7}	3.49^{-7}
Class. + Mark.	2.19^{-6}	3.79^{-6}	3.49^{-5}	3.72^{-5}	5.05^{-5}

and leverage *recirculation* across Tofino's four pipelines in TNA instead of egress mirroring to enhance parallelism.

D. Time Overhead

1) *Metric*: We define time overhead based on *nodal processing time*, which represents the classification time within the switch pipeline. This is compared to the average Round Trip Time (RTT) measured using INT. Eq. 5 shows the fraction of the RTT consumed by classification. For N processed packets, $(egress, i)$ and $(ingress, i)$ are the egress and ingress timestamps of packet i . The RTT represents the round-trip time from Server 2, through the switch, and back (Fig. 4).

$$Time\ overhead = \frac{\frac{1}{N} \sum_{i=1}^N (T_{(egress, i)} - T_{(ingress, i)})}{RTT} \quad (5)$$

2) *Analysis Methodology*: The *INT* packet *RTT* and *average nodal processing time* are computed for each INT packet across 780 samples per slice. The three slices are merged, resulting in 2,340 samples (780×3), to calculate the time overhead (Eq. 5). The merged data is used to compute the average, standard deviation, and 90th, 95th, and 99th percentiles, providing insights into the time overhead across the three experiments as shown in Table IV.

V. DISCUSSION

Flow Classification vs Packet Marking/DSCP Assignment.

Flow classification introduces a delay of approximately 20 packets per flow for feature computation, impacting shorter flows more significantly. This delay leaves some initial packets unclassified, affecting QoS and QoE for AR and CG flows, a challenge for future research. The overhead reflects the gap between flow-based classification and packet-based DSCP assignment. For AR (180K packets in UL & DL), 162.2K UL and 163.8K DL packets were correctly classified (TP). For CG (90K UL and 430K DL packets), 73.8K UL and 348.3K DL packets were TP. Overall, 585.55K of 715K UL packets and 850.3K of 1.055M DL packets were correctly classified, leaving 204.7K misclassified. Misclassification is due to RF model accuracy and thresholds, while unclassified packets result from classification delays, identifying areas for future improvement.

ECT(1) Marking Accuracy. As shown in Table III, ECT(1) marking achieves high accuracy for AR/CG packets, with TP rates of 89% in UL and 92% in DL, ensuring prioritization for the L4S queuing mechanism. However, FP marking, where non-AR/CG traffic is mistakenly marked and prioritized, is higher in UL (15%) than DL (9.2%), adding unnecessary overhead to the L4S mechanism. Additionally, due to feature computation delays, some packets remain unclassified and are not marked with ECT(1), resulting in the loss of L4S prioritization. These unclassified packets account for 9% in UL and 4.7% in DL, potentially degrading QoS & QoE, which will be explored in future work. This minimal loss highlights the impact of traffic mirroring, flow-based classification, and packet-based marking caused by the data plane development.

Time overhead. Table IV shows that nodal processing contributes minimally to RTT. The fraction of nodal processing time to RTT increased by 6.8×10^{-4} due to classification, ECT(1) marking, and DSCP assignment fully deployed in the data plane, while absolute nodal processing rose by 21.42 μ s compared to simple forwarding. Although the overhead is negligible, future refinements can reduce even more the processing time while maintaining efficient classification and marking.



VI. CONCLUDING REMARKS

In this work, we presented an RF model fully integrated into P4 pipelines for programmable switches, including Tofino2. The model autonomously classifies network traffic into AR, CG, and other (non-AR/CG), assigning DSCP codepoints—EF(46) for AR and AF41(34) for CG—while marking ECT(1) for L4S queuing independent of host configurations. The implementation is modular and embedded in the ingress pipeline for ECT(1) marking. To enhance classification accuracy for RTP-based video streaming, we proposed data plane feature extraction (FS and IFI), taking the first step toward RTP protocol integration in data planes.

Our hardware deployment matches the classification accuracy of the software model, confirming no adverse impact. Overhead from feature extraction, classification, DSCP assignment, and ECT(1) marking remains minimal, affirming deployment viability. High true positive rates for ECT(1) marking (89% (UL) and 92% (DL)) demonstrate reliable L4S prioritization, with marking accuracy loss average under 9.2%. However, 9% (UL) and 4.7% (DL) of AR/CG/Other packets remain unclassified, potentially affecting QoS/QoE. Additionally, 15% (UL) and 9.2% (DL) of ECT(1) marked packets belong to the 'Other' class, increasing L4S queuing.

Future work includes parallel classification and marking across four pipelines in Tofino, selective packet cloning to reduce time overhead, and AR/CG QoS/QoE evaluation alongside L4S integration for enhanced deployment efficiency. Additionally, scheduling L4S queuing based on application requirements is a promising direction for future research.

ACKNOWLEDGMENT

This work was supported by Ericsson Telecomunicações Ltda., and by the Sao Paulo Research Foundation (FAPESP),  grant 2021/00199-8, CPE SMARTNESS .

REFERENCES

- [1] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, et al. P4: Programming protocol-independent packet processors. *ACM SIGCOMM Computer Communication Review*, 44(3):87–95, 2014.
- [2] Bob Briscoe, Koen De Schepper, Marcelo Bagnulo, and Greg White. Low Latency, Low Loss, and Scalable Throughput (L4S) Internet Service: Architecture. RFC 9330, January 2023.
- [3] G. G. Gil, A. H. Lashkari, M. Mamun, and A. A. Ghorbani. Characterization of encrypted and vpn traffic using time-related features. In *Proceedings of the 2nd international conference on information systems security and privacy (ICISSP 2016)*, pages 407–414. SciTePress Setúbal, Portugal, 2016.
- [4] P. Graff, X. Marchal, T. Cholez, B. Mathieu, and O. Festor. Efficient identification of cloud gaming traffic at the edge. In *NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium*, pages 1–10. IEEE, 2023.
- [5] P. Graff, X. Marchal, T. Cholez, B. Mathieu, S. Tuffin, and O. Festor. Improving cloud gaming traffic qos: a comparison between class-based queuing policy and l4s. In *Network Traffic Measurement and Analysis Conference (TMA 2024)*, page 10. IEEE, 2024.
- [6] G. Kougioumtzidis, A. Vlahov, V. K. Poulkov, P. I. Lazaridis, and Z. D. Zaharis. Qoe prediction for gaming video streaming in o-ran using convolutional neural networks. *IEEE Open Journal of the Communications Society*, 5:1167–1181, 2024.
- [7] J. R. Ky, P. Graff, B. Mathieu, and T. Cholez. A hybrid p4/nfv architecture for cloud gaming traffic detection with unsupervised ml. In *2023 IEEE Symposium on Computers and Communications (ISCC)*, pages 733–738. IEEE, 2023.
- [8] M. Lecci, M. Drago, A. Zanella, and M. Zorzi. An open framework for analyzing and modeling xr network traffic. *IEEE Access*, 9:129782–129795, 2021.
- [9] M. Letourneau, K. B. N'Djore, G. Doyen, B. Mathieu, R. Cogranne, and H. N. Nguyen. Assessing the threats targeting low latency traffic: the case of l4s. In *17th International Conference on Network and Service Management (CNSM)*, pages 544–550. IEEE, 2021.
- [10] M. Mayfield. Army hopeful troubled headset program is finally looking up, April 2024. Accessed: 2024-12-28.
- [11] D. G. Morín, D. Medda, A. Iossifides, P. Chatzimisios, A. G. Armada, A. Villegas, and P. Pérez. An extended reality offloading ip traffic dataset and models. *IEEE Transactions on Mobile Computing*, 2023.
- [12] S. Nádas, G. Gombos, F. Fejes, and S. Laki. A congestion control independent l4s scheduler. In *Proceedings of the Applied Networking Research Workshop*, pages 45–51, 2020.
- [13] M. A. Ridwan, N. A. M. Radzi, F. Abdullah, and Y. E. Jalil. Applications of machine learning in networking: a survey of current issues and future challenges. *IEEE Access*, 9:52523–52556, 2021.
- [14] B. Sarpkaya, F. Fund, and S. Panwar. To adopt or not to adopt l4s-compatible congestion control? understanding performance in a partial l4s deployment. *arXiv e-prints*, pages arXiv-2411, 2024.
- [15] Koen De Schepper and Bob Briscoe. The Explicit Congestion Notification (ECN) Protocol for Low Latency, Low Loss, and Scalable Throughput (L4S). RFC 9331, January 2023.
- [16] Koen De Schepper, Bob Briscoe, and Greg White. Dual-Queue Coupled Active Queue Management (AQM) for Low Latency, Low Loss, and Scalable Throughput (L4S). RFC 9332, January 2023.
- [17] P. Schulz, A. Traßl, N. Schwarzenberg, and G. Fettweis. Analysis and modeling of downlink traffic in cloud-rendering architectures for augmented reality. In *4th 5G World Forum (5GWF)*, pages 188–193. IEEE, 2021.
- [18] Henning Schulzrinne, Stephen L. Casner, Ron Frederick, and Van Jacobson. RTP: A Transport Protocol for Real-Time Applications. RFC 3550, July 2003.
- [19] A. Shirmarz, F. L. Verdi, S. K. Singh, and C. E. Rothenberg. From pixels to packets: Traffic classification of augmented reality and cloud gaming. In *IEEE 10th International Conference on Network Softwarization (NetSoft)*, pages 195–203, 2024.
- [20] A. Shirmarz, F. L. Verdi, S. K. Singh, and C. E. Rothenberg. Posmac: Powering up in-network ar/cg traffic classification with online learning. In *IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 2024.
- [21] S. K. Singh, C. E. Rothenberg, A. Shirmarz, F. L. Verdi, I. Haque, G. Patra, and G. Pongrácz. Matador: ML-based cloud gaming traffic detection entirely in programmable hardware. In *IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, pages 1–5, 2024.
- [22] J. Son, Y. Sanchez, C. Hampe, D. Schnieders, T. Schierl, and C. Hellge. L4s congestion control algorithm for interactive low latency applications over 5g. In *2023 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1002–1007. IEEE, 2023.
- [23] C. Zheng, X. Hong, D. Ding, S. Vargaftik, Y. Ben-Itzhak, and N. Zilberman. In-network machine learning using programmable network devices: A survey. *IEEE Communications Surveys & Tutorials*, 2023.