Estimating Video on Demand QoE from Network QoS through ICMP Probes

Estimating Video on Demand QoE from Network QoS through ICMP Probes

Gilson Miranda Jr.*†, Daniel Fernandes Macedo†, Johann M. Marquez-Barja*
*University of Antwerp - imec, IDLab, Faculty of Applied Engineering - Antwerp, Belgium
†Universidade Federal de Minas Gerais - Computer Science Department - Minas Gerais, Brazil
E-mail: {gilson.miranda, johann.marquez-barja}@uantwerpen.be, damacedo@dcc.ufmg.br

***Abstract*—With the increasing traffic of Video on Demand (VoD), network providers are seeking to deliver high Quality of Experience (QoE) for their users. Many methods have been proposed to assess VoD-related QoE. Some of them rely on client instrumentation and reporting QoE information to network elements, such as Server and Network Assisted DASH, others are based on statistical methods that make QoE inferences using monitored network conditions, such as throughput and delays. In this article, we present a practical method to estimate QoE for VoD using the widely supported Internet Control Message Protocol (ICMP) probes. Measured network conditions are used as input to a Machine Learning (ML) model that estimates QoE in terms of Mean Opinion Score (MOS), based on the ITU-T P.1203 Recommendation. The estimation encompasses video quality switches and playback stalls. We estimate MOS with an average Root Mean Square Error (RMSE) of 1.05 for a catalog of 25 different videos, training a model with sessions of the shortest video, and evaluating the generalization to the full catalog. We performed experiments using a virtualized setup as well as in a Wide Area Network.**

***Index Terms*—Quality of Service, Quality of Experience, DASH video, Machine Learning**

I. Introduction

Video content represents a significant amount of current IP traffic. According to Cisco, 82 % of network traffic will be composed by video by 2022 [1], being mostly comprised by Video on Demand (VoD) services such as Netflix, YouTube, among others. With the prominence of VoD services and the pressure that they pose to network resources, operators are expected to keep up with the increasing demand for performance to satisfy user expectations. An Accenture survey shows that 60% of users are dissatisfied with their connectivity and network experience [2]. Network operators should assess user satisfaction in order to avoid customer churn. Both industry and academia have drawn their attention to Quality of Experience (QoE), which indicates the users' degree of satisfaction or annoyance when consuming services or applications [3]. Operators must then employ QoE measurements to improve their resource provisioning and troubleshooting policies [4].

Most VoD services are now based on HTTP Adaptive Streaming (HAS) [5], [6]. In HAS the QoE is degraded by events such as playback stalls, initial buffering delay and video quality variations [7]. Different methods have been proposed to gather this information on user devices and estimate user QoE [8], [9]. Although such information is easily computed on the streaming client application, operators must reach agreements with each video service provider to obtain it.

Due to the lack of interactions with the client's playback software, operators must resort to indirect QoE measurements. This is the preferred method for operators because it maintains their client's privacy and avoids the introduction of specialized equipment [9]. Limitations of the state of the art are the reliance on technology-dependent information (e.g. quality of signal [10]), limiting the application of the solution on network with other technologies, and the use of monitoring software that does not cover the last mile of the connection [11].

In this work, we propose and evaluate a method for QoE assessment based on network-level Quality of Service (QoS). To measure end-to-end QoS between the VoD server and a VoD client, we employ active Internet Control Message Protocol (ICMP) probing. The network QoS measurements are fed into a regression tree ensemble-based Machine Learning (ML) model. The model estimates Mean Opinion Score (MOS) according to the ITU-T P.1203 Recommendation, encompassing playback stalls, video quality switches and user equipment characteristics [8]. We focus on deployments where small scale Content Delivery Networks (CDNs) are deployed within Internet Service Providers (ISPs) domains, known as CDN-ISP or Mobile Edge Computing (MEC) [12], [13]. For such deployments, more flexibility in regards of probing frequency and link monitoring is expected. However, deployments traversing multiple domains can also be monitored due to the general support of ICMP probing, albeit with more restrictions and lower accuracy. Our method has the following benefits: (i) it is widely supported by recent and legacy network equipment due to the use of ICMP probes; (ii) it is privacy-preserving, since active probing replaces Deep Packet Inspection (DPI) techniques; (iii) it works with encrypted video traffic; and (iv) it allows the operator to take preventive actions even before the video flow starts.

We performed experiments using a controlled environment to generate a dataset to train and evaluate the inference model. We first provide a set of analyses about the relation between network QoS conditions and the QoE they generate, giving insights on the expected performance of the proposed method. The method provided MOS inferences with Root Mean Square Error (RMSE) of 1.05, for a model trained using samples of all available videos in the catalog, as well as for a model trained with samples of a single video. Furthermore, the model trained with samples of a single video of the catalog was able to generalize to the rest of the catalog.

This article is an improvement of our previous work [14],

adding the following aspects:

- In previous work [14], we evaluated the inference model using samples from a single video. We now compare a similar model trained with data from a single video with another one created with samples of all videos.
- The inference model now uses more network input metrics such as percentiles and standard deviation of measurements. The training dataset is now labeled using Mode 3 of ITU.T P.1203. Mode 3 is more comprehensive, considering more video characteristics, better reflecting user's perceived QoE.
- This article shows the feasibility of the method on more realistic network conditions. We added experiments over wireless links, and crossing different domains and geographically distributed testbeds.
- We evaluate the effects of lower probing rates, considering cases where ICMP probing rate is restricted on cross-domain measurements. We present the results evaluating sampling rates from 0.25 up to 42 samples per second.
- We provide a comprehensive analysis of the individual impact of uplink and downlink network QoS conditions on the resulting MOS, investigating the general differences in network conditions between sessions that presented high MOS and low MOS.

This article is organized as follows: Section II discusses related work. Section III describes the proposed method, while the setup of experiments are detailed in Section IV. The results are presented and discussed in Section V. Section VII presents the conclusion and future work. We use several acronyms throughout the article, and a list of them can be found at the end of the document.

II. Related Work

Focusing on cellular networks, Costa et al. [11] use network measurements to estimate Application QoS (AQoS) conditions of video streaming, and from AQoS predict the user's QoE. They first use Decision Tree (DT) to map delay, jitter, throughput and packet loss into AQoS metrics of *startup time*, *stall count* and *total stall time*. Network measurements are performed using NetMetric [15], requiring probes as close as possible to the points of interest. The experiments do not evaluate adaptive video streaming, being restricted to one video with 1080p resolution and one with 720p resolution.

Relying on DPI for granular traffic analysis, the method presented by Huysegems et al. [16] reconstructs a video session by analyzing packets that pass through an intermediate node. Information such as number of segments, video duration and quality levels are extracted from the manifest file, while other information such as segment sizes and requested video quality level are obtained through traffic inspection. The session is then reconstructed to obtain the QoE-related parameters, e.g. rebuffering events and bit rate variation. In view of the increasing use of encrypted traffic [17], traffic inspection may be infeasible. The method proposed by Ge and Wang [18] overcomes this limitation using an HTTP proxy, however it also requires flow inspection to identify video streams. The QoE-related metrics estimated by the algorithm in [18] are initial playback delay, number of rebuffering events and their duration. For evaluation, the authors used one video encoded in one representation, thus the performance of the system is not clear when quality adaptation algorithms are active.

Addressing challenges imposed by encrypted traffic, Khokhar et al. [17] present a method to estimate MOS based on network-level measurements and ITU-T P.1203 [8]. The authors created a dataset by consuming YouTube videos while emulating network impairments using Traffic Control (TC)[1]. Throughput, packet inter-arrival times, chunk sizes and other information were also included, totaling 48 features. Multiple models were trained to estimate different QoE-related playback characteristics, e.g. occurrence of playback stalls, whether video playback started or not, occurrence of quality switches and MOS. Despite using information about network QoS (e.g. delay, jitter, bandwidth and loss rate), the work does not address how to monitor this information in real networks.

Also focusing on encrypted YouTube traffic, the work by Seufert et al. [19] presents ViCrypt: a stream-based ML approach to predict stalling of video streaming in real time. The video session is analyzed in one-second slots, but a combination of all slots can be used to evaluate the complete session. An ML model based on Random Forests (RF) is used to predict whether a slot contains a stalling or not. For this the model takes 208 input features, including uplink and downlink TCP/UDP packet counts, upload versus download ratio, among others. Prediction of stall occurrence achieves an accuracy of 94.67% using the complete set of features, however, train and evaluation were performed using a highly unbalanced dataset, in which over 70% of video sessions did not present stalls. The work from Wassermann et al. [20] also employs ViCrypt on a similar context as by Seufert et al. [19], but extends it to also predict video resolution and bitrate. The dataset consisted of videos in 6 different resolutions, with 55% of data with vertical resolution of 480p. To estimate resolution the authors used a classification method, achieving 66% accuracy with a k-Nearest Neighbors (kNN) method. A RF method was used for bitrate estimation, returning predictions with 233 kbps of mean absolute error.

Table I summarizes the works analyzed in this section. The first column references the work. Column *QoS Metrics* indicate which network-level QoS information are used as input for the QoE estimation, while column *QoE Metrics* describes the QoE formulation. The *Monitoring Type* column describes how the QoS metrics are obtained in the proposal: through active or passive monitoring, specialized monitoring software, and if DPI is employed. The column *Video Catalog* describes the video catalog used in each work. The *Generalization* column indicates whether the model was validated with videos outside of the training set. As shown on the table, as far as we are aware, the work presented in this paper is the first one to explicitly evaluate model inference generalization. We focus on works that utilize Network-level QoS (NQoS) to estimate QoE, as we consider that NQoS can be directly influenced by management actions of an ISP. Other approaches that focus on mapping AQoS (e.g. initial buffering time, duration of

[1] http://man7.org/linux/man-pages/man8/tc.8.html

resolution drops) to QoE such as the work by Bampis et al. [21], were not included.

In this work we encode videos in 10 quality levels, thus the Dynamic Adaptive Streaming over HTTP (DASH) adaptation algorithm can adjust playback quality during a session as it would perform on a real VoD service. Our MOS estimation approach comprises playback stalls, stall duration, playback quality, and quality switches through ITU-T P.1203 Recommendation. Methods that estimate MOS based only on stalls may fail to identify low QoE when playback is smooth but at the lowest or oscillating video quality, also reducing user experience [7]. Using active probing, our method is suitable to monitor VoD services that transmit encrypted or unencrypted traffic. Moreover, network monitoring is based on ICMP probing that is widely supported by new and legacy network equipment. This method is technology agnostic, and eliminates the need for specialized monitoring tools.

III. Proposed Solution

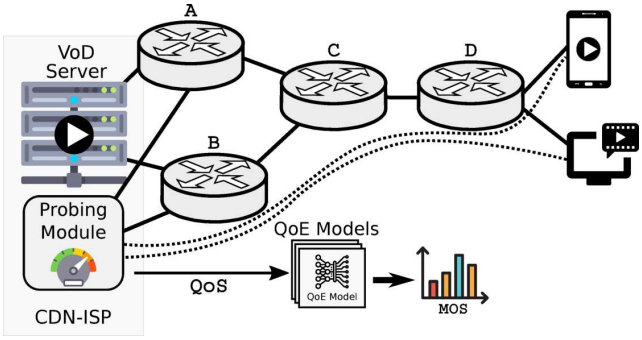
We propose a solution that is tailored to ISPs that manage their networks using QoE-based Service-Level Agreements (SLAs). It monitors network-level QoS through active ICMP probing and employs an ML model to estimate the QoE level of video flows. It overcomes one of the main challenges in ISP traffic monitoring: how to obtain meaningful information from both encrypted and unencrypted traffic while preserving the user's privacy. Our solution can be integrated into self-management software in order to implement QoE-aware management, providing the necessary feedback for management control loops. For example:

- **Tuning Wi-Fi parameters:** Moura et al. [24] use the estimated MOS of user flows to reconfigure the channel and the transmit power of Wi-Fi access points;
- **Improved routing:** Costa et al. [25] use a QoE inference model for traffic routing.
- **Scheduling in datacenters:** Carvalho et at. [26] rely on the inferred QoE to schedule containers on Kubernetes.

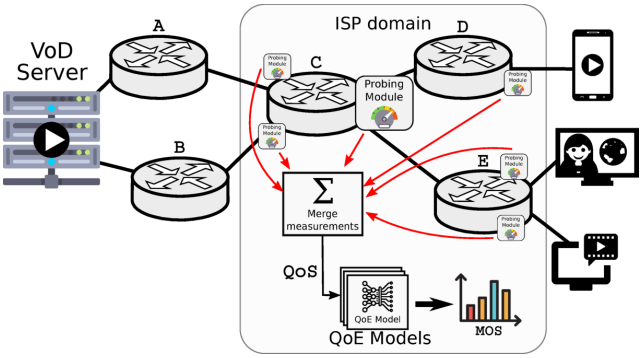
Before describing the implementation details, we present the envisaged use cases.

A. Use cases

There are two use cases for our monitoring tool shown in Figure 1. The first is in the context of CDN-ISP partnership or MEC [12], [13], depicted in Figure 1a. This use case considers a partnership between ISPs and Over The Top (OTT) services to deploy a small scale VoD CDN within the ISPs domains. The monitoring software is installed in the best vantage point (near to the server), and the probes are free from third-party traffic shaping. This is the ideal use case for the tool, and also simplifies the identification of the startup of a new video session, as well as the classification of the traffic as VoD. Traffic classification is out of the scope of this article, however, the methods presented by Dias et al. [27], and Lotfollahi et al. [28] can be combined with our solution to detect and classify VoD traffic. The first method classifies individual flows that belong to a VoD session using a modified Naïve Bayes algorithm, taking features of IP headers of a flow as input. The second method uses Deep Learning to automatically extract features and classify traffic types and applications, by processing packets as vectorized byte streams. Both methods can perform real-time classification of a network flow and determine that a VoD streaming session is in progress with over 90 % accuracy.



(a) Use case example in a CDN-ISP context



(b) Use case example restricted to ISP domain

Fig. 1. Use cases of the proposed solution

Figure 1b shows a more complex deployment, where the VoD server is outside of the domain of the ISP. Because the tool is installed in the middle of the end-to-end path, multiple instances of the Probing Module (PM) are required in order to monitor all links involved. Information gathered for a common link can be used to perform estimates for multiple clients, as is the case between routers C and E. Measurements from the multiple instances can then be combined following the process described in Section 8 of ITU-T Y.1541 Recommendation [29], and the QoE for each client can be estimated. This second scenario is the most challenging for the tool, as routers outside of the ISP may block ICMP traffic or perform rate limitations (which may reduce the precision of the QoE estimation).

The use of our method may be limited in contexts where clients are behind firewalls (restricting ICMP probing) or Network Address Translation (NAT). However, the network provider can still perform measurements until before the link applying such restrictions, giving useful information about whether QoE-impairing conditions occur in its own domain or in the last-mile (e.g. user's personal Wi-Fi network). Moreover, with the wide adoption of IPv6, the reachability issues posed by NAT tend to be diminished.

Our solution is composed of two modules: The PM measures QoS conditions between the VoD server and a VoD client and adapts the probing frequency to varying network

TABLE I
SUMMARY ON RELATED WORK

Reference	QoS Metrics	QoE Metrics	Monitoring Type	Video Catalog	Generalization
[11]	Delay, Jitter, Throughput, Loss	A mathematical function derived from observations in [5], [22], [23]	Active measurements using NetMetric [15]	One video clip encoded in two fixed quality levels (720p and 1080p)	✗
[16]	TCP/HTTP session reconstruction based on DPI	Video quality, quality switches, and playback stalls	Passive (using DPI)	Not specified	Not specified
[18]	DPI and download time of video segments	Initial playout delay, buffer level, playback stalls, quality switches	Passive monitoring of TCP/HTTP session	One video encoded in one quality level	✗
[17]	Up to 48 features	MOS derived from ITU-T P.1203 Recommendation [8]	Not specified	A catalog of over 1 million videos from YouTube	✗
[19]	69 packet-level statistics of the video flow	Initial delay, playback stalls, and stall duration	Passive monitoring	A catalog of videos from YouTube	Not specified
[20]	69 packet-level statistics of the video flow	Video resolution and bitrate	Passive monitoring	A catalog of videos from YouTube	Not specified
This work	RTT, Jitter, PLR	MOS derived from ITU-T P.1203 Recommendation [8]	Active ICMP probing	25 videos encoded in 10 quality levels	✓

conditions. The second component is an MOS inference model created using supervised ML. It receives the QoS measurements and returns an MOS inference based on ITU-T P.1203 Recommendation.

B. Probing Module (PM)

We implement the PM based on the *fping*[2] tool for ICMP probing. It measures Round-Trip Time (RTT), jitter (i.e. delay variation in a period) and Packet Loss Rate (PLR) between VoD server and client. ICMP probing is a network probing method widely supported by network devices, allowing the PM to be adopted even on networks with legacy equipment. Moreover, end-to-end measurements can be achieved without requiring installation of new software on clients or servers. The PM aims to obtain N probing samples during a window of T seconds. To achieve this, it runs multiple probing threads, each one performing independent requests and reporting to a central thread. Each thread makes a probing attempt and waits for the reply before reporting the observed RTT or a timeout, which is treated as a packet loss, to the central thread. The interval between probe attempts is adjusted on-the-fly according to the observed RTT to avoid generating more traffic than needed while still collecting N samples during T seconds.

For probing frequency adaptation we developed the Algorithm 1. Input parameters are the monitoring time window (in milliseconds), minimum amount of samples to be collected, and the IP of the host to be probed. Line 1 creates a probing thread and adds it to a list of probes. In the main loop (line 2) the first operation is to gather and consolidate data from probing threads (lines 3 to 5). Then, the algorithm calculates the required statistics for the inference model: mean RTT, median RTT, standard deviation of RTT, 10th and 90th quantiles of RTT; mean jitter, median jitter, standard deviation of jitter, 10th and 90th quantiles of jitter; PLR (line 7). Results are published to other services like the QoE model in line 8. Probing frequency adaptation starts in line 9, calculating if the number of existing probes provide the required amount of samples. If the minimum sampling frequency cannot be achieved, new probing threads are created (line 11).

[2]https://fping.org/

Algorithm 1 Algorithm for ICMP probing

Input: timeWindow_ms, minSamples, destination

```
1:  probes.add(spawnProbe(destination))
2:  while True do
3:      for all probes do
4:          probeData = getProbeData()
5:          allData.concat(probeData)
6:      if allData > 0 then
7:          rtt, jitter, loss, statistics = getQoS(allData)
8:          publishQoS(rtt, jitter, loss, statistics)
9:          samplesPossible = timeWindow / rtt * probes.len()
10:         if samplesPossible < minSamples then
11:             probes.add(spawnProbe(destination))
12:         if allData.length < 0.9 * minSamples or
                allData.length > 1.2 * minSamples then
13:             interval = max(1, ((timeWindow * probes.len()) /
                (minSamples * 1.1)) - rtt)
14:             for all probes do
15:                 probe.setInterval(interval)
16:         removeOldData(allData, timeWindow)
17:     if (samplesPossible > 1.5 * minSamples) and
            (probes.len() > 1) then
18:         probes.remove()
```

Line 12 checks if the sampling frequency for the threads is adequate. Looser limits are defined to avoid constant corrections caused by delay variation. The interval between ICMP requests is calculated in line 13. A constant value of 1.1 is multiplied to $minSamples$ to account for other delay sources during program execution (e.g. inter-process communication), and was defined empirically. The calculated interval is applied to all probes from line 14 to 15. Line 16 discards data older than the time window. If network conditions change and the number of probes is higher than the necessary, lines 17 and 18 remove probing threads. Each fping process waits a timeout of 2 seconds before considering a packet loss. This value must

be adjusted in case of higher network delays.

C. QoE Model

Formulation: The QoE Model is an ML model that takes QoS data as input and returns the inferred QoE in terms of MOS. Methods based on Decision Trees (DT) have shown better results (predictions with lower RMSE) when mapping QoS to QoE than other methods [30], which motivated the use of a DT-based approach. Through the last decades, ensemble techniques such as *boosting* have been developed to combine multiple DTs and create more accurate models. Boosting is an ensemble technique, that combines several *weak* (or low-accuracy) learners to build a *strong* learner capable of performing accurate predictions. A *weak* learner performs slightly better than random guessing, while a *strong* learner is said to be close to the perfect performance [31]. During the training process new trees are added to predict the residuals (or errors) of the existing trees, and then combined to provide the final predictions. Due to the superior performance achieved by gradient boosting methods over many different ML problems [32], we opted for this type of algorithm and the eXtreme Gradient Boosting (XGBoost) framework [33].

The model is formally defined in (1), and is a function of eleven variables: mean RTT, median RTT, standard deviation of RTT, 10th and 90th quantiles of RTT; mean Jitter, median Jitter, standard deviation of Jitter, 10th and 90th quantile of Jitter; and Packet Loss Ratio. All these elements are represented in 1 as QoS. The output of the model is a MOS value between 1 and 5. The MOS values used in this work are estimates based on ITU-T P.1203 Recommendation and obtained using the software[3] provided by Raake et al. [34] and Robitza et al. [35].

$$f(QoS) \mapsto MOS_{ITU} \in \mathbb{R} \mid 1 \leq MOS_{ITU} \leq 5 \quad (1)$$

PLR is calculated by $1 - (P_{rep}/P_{req}) * 100$, where P_{rep} is the number of probe replies received and P_{req} are the number of probe requests sent during the time window. It is worth noting that the estimator does not perform throughput measurements, however, metrics such as RTT and PLR have been used in previous work to obtain effective throughput and estimate link capacity. For example, Padhye et al. [36] and Chen et al. [37] present a model to compute the throughput of a TCP transfer as function of PLR and RTT, Chan et al. [38] use RTT and packet dispersion to measure asymmetric link capacity. As will be shown in the results section, the model accuracy achieved is similar for various ranges of bandwidth setups.

Training set: We created a labeled dataset to train the model using supervised learning as follows. Each sample is composed by the eleven QoS elements mentioned above and a label, i.e. the MOS for the video session at the time the input features were recorded. To create this dataset, we setup a video server with a catalog of videos encoded with a similar process as used by VoD services, and instrumented a client to record playback information such as resolution and rebuffering events. The MOS for each data point was then estimated based on the ITU-T P.1203 Recommendation [8]. This client instrumentation process is required only for model training, during system operation the MOS is estimated without requiring client feedback. Further details about the videos, their encoding format, and the complete data collection process are provided in Section IV.

[3]https://github.com/itu-p1203/itu-p1203

D. Assumptions, limitations and overhead

We now describe some assumptions of the proposal. Then, we proceed to requirements and limitations related to the PM, as well as an analysis of the generated overhead.

Assumption #1 - uniform video encodings: We aim to create a QoE inference model suited for all videos hosted by the same VoD server. Therefore, it is important to have a standardized number of resolutions for the videos. To contextualize this, Figure 2 illustrates the typical operation of a DASH-based VoD service. An HTTP server stores multiple *representations* of a given video, each representation is a version of the video that can be encoded in a different format in terms of resolution, codec and bitrate. If each video is encoded following a different standard in terms of resolution, codec, bitrate, and other parameters, the relation between file sizes of video segments and a specific video quality can have little to no correlation. Therefore, our solution requires all videos and their representations to be prepared following a consistent standard (same number of representations and same configuration for each representation). The *bitrate ladder* we used for all videos is described on Table III.

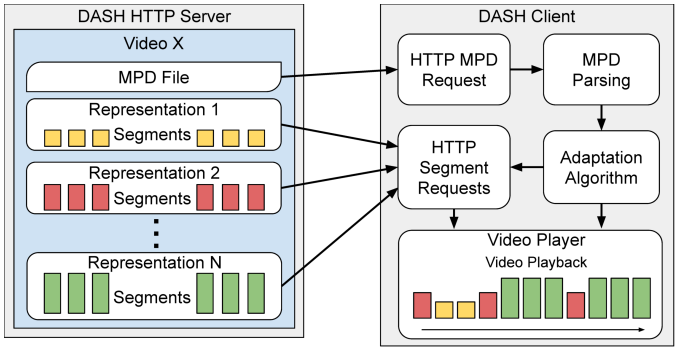


Fig. 2. DASH server and client operation.

Assumption #2 - A high probing frequency is needed to accurately measure packet losses: A preliminary analysis of data collected in our testbed showed that MOS is strongly affected by PLR, as shown in Figure 3 (data collection methodology is detailed in Section IV). A PLR below 1 % can make the MOS value drop from 5 to approximately 3, a significant impact on user's QoE. If we aim to detect PLR with a granularity of 0.1 %, then we need historical data of at least 1,000 probing attempts. Hence, the PM is configured to store results of the last 1,000 attempts. The timeliness of probing data is also relevant. The PM holds data only of a recent window of time, which is configured as 30 seconds in this work. This value was determined based on the buffer

length of the VoD client: as our client was configured to buffer up to 12 seconds of video, we used a window that is twice as large as the buffer in a way that the playback conditions reflect a recent network state, not a previous situation of when the buffer was filled. Based on these requirements, a probing rate of about 33 packets per second is necessary.

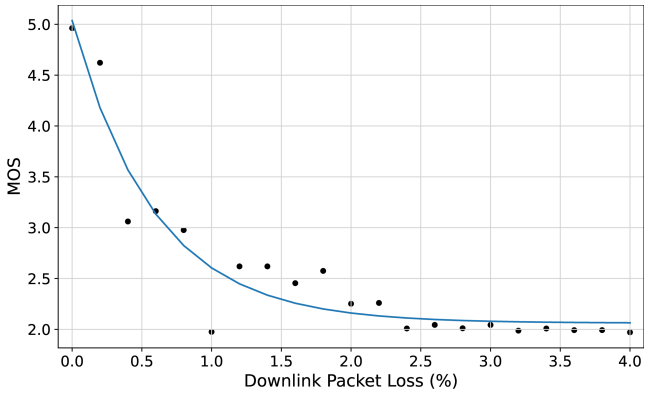


Fig. 3. MOS achieved according to PLR

Assumption #3 - Audio track is equal for every representation: The audio track of a video usually is not an issue in QoE for two reasons: first, a common audio track is usually shared among all video quality levels, so it practically adds a constant overhead to all video representations. Second, in terms of file size, the audio track is much smaller than the video.

Limitation #1 - Model considers symmetrical links: Our method cannot differentiate uplink from downlink conditions, as it is based on ICMP request-reply. ICMP brings already mentioned benefits, such as operation without additional software at the edges, however it limits the precision of the PLR measurements. Section V-A will show how the precision of VoD QoE is impacted by this limitation.

Limitation #2 - ICMP probing restricted on some networks: ICMP rate-limiting policies will reduce the precision of the proposal. Because of that, section V-E evaluates the precision using lower probing rates.

Overhead: Probing overhead depends on: i) ICMP data size; ii) protocol overhead; iii) *timeWindow* setting; iv) *minSamples* setting. Factors (i), (iii) and (iv) are configured in the algorithm. Factor (ii) depends on overhead generated by underlying networks. The per second probing overhead on a given direction (O_{dir}, where direction is downlink or uplink), in bits per second, is given by Equation 2. The first term gives the probing frequency required to obtain $minSamples$. To calculate the overhead, $timeWindow$ is used in seconds. The result is multiplied by the size of the probes (S_{icmp}) plus protocol overhead (P_{ov}). This is an approximation, since different technologies in the path may change the size of P_{ov}.

$$O_{dir} = \frac{minSamples}{timeWindow_{seconds}} \times (S_{icmp} + P_{ov}) \qquad (2)$$

IV. Experiment Setup

Figure 4 shows the setup used to perform the experiments, composed by three Docker containers. A *Server* runs NGINX HTTP server [39]. The DASH client is based on the reference player provided by DASH Industry Forum[4], modified to collect playback metrics and executed using Firefox Web Browser. Playback metrics are stored locally to avoid additional traffic on the client's network. Network impairments are generated using TC. *Server* and *QoS Monitor* are deployed on the same network point, so similar impairments are applied to both containers.

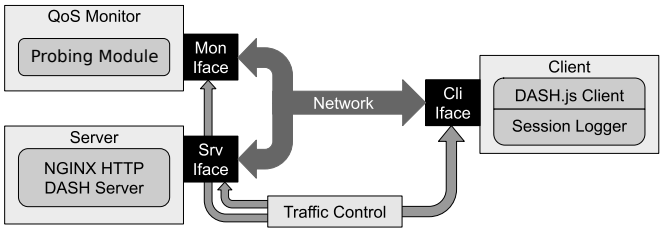


Fig. 4. Experimental environment

TABLE II
Sample videos

Video	Duration	Type
[1]Sony Another World (another)	00:03:11	Nature
[1]Sony Another World 2 (another2)	00:03:06	Nature
[1]Samsung: Around The World (aworld)	00:05:39	Documentary
[1]Panasonic Football Barcelona (barcelona)	00:03:14	Sports
[2]Big Buck Bunny (bbb)	00:10:35	Animation
[1]Sony The Fountains Of Bellagio (bellagio)	00:03:43	Arts
[1]LG La Boheme (boheme)	00:04:29	Music Video
[1]Samsung Power of Curve (curve)	00:03:15	Promotional
[1]Samsung The Quiet Czech (czech)	00:03:24	Documentary
[1]Samsung Phantom Flex (flex)	00:03:07	Promotional
[1]LG Garden (garden)	00:03:05	Promotional
[1]LG Cymatic Jazz (jazz)	00:04:58	Concert
[1]Jimix Put Your Hands Up (jimix)	00:03:56	Music Video
[1]Samsung Landscape (landscape)	00:03:10	Nature
[1]Panasonic Lumix (lumix)	00:03:07	Documentary
[4]Sintel (sintel)	00:14:48	Animation
[1]LG Slam Dunk (slam)	00:02:56	Sports
[1]Sony Surfing (surfing)	00:02:59	Sports
[1]Samsung Lovely Swiss (swiss)	00:03:41	Documentary
[3]Tears of Steel (tearsofsteel)	00:12:14	Short film
[1]Samsung Travel With My Pet (travel)	00:02:35	Documentary
[1]TravelXP HDR/HLG (travelxp)	00:05:00	Documentary
[1]Samsung & RedBull See the Unexpected (unexpected)	00:03:18	Sports
[1]Life Untouched (untouched)	00:03:18	Nature
[1]Samsung 7 Wonders Of The World (wonders)	00:03:51	Documentary

[1]http://4kmedia.org [2]https://peach.blender.org/
[3]https://mango.blender.org/ [4]https://durian.blender.org/

The server offers 25 videos listed in Table II. The table shows video names on the source website, a short name in parenthesis for reference in this work, video duration, and an indication of content type. All videos were encoded in 10 representations (i.e. 10 versions of the videos with specific resolutions and bitrates), as described on Table III. A Media Presentation Description (MPD) file describes representations,

[4]https://dashif.org/

location of files and directives for quality adaptation (e.g. *bandwidth* field). For each representation the MPD bandwidth field was fixed with the values shown in the fourth column. The videos were split into *segments* of four seconds. The DASH client adjusts playback quality by selecting segments according to network conditions, then segments are assembled and played sequentially. All videos were encoded using the *ffmpeg*[5] tool, H.264 codec (x264 implementation[6]) with no audio track, as our focus is on video quality.

TABLE III
VIDEO REPRESENTATIONS

Representation	Resolution	Bitrate	MPD Bandwidth
1	320x180	200 kbps	256,000 bps
2	320x180	400 kbps	512,000 bps
3	480x270	600 kbps	760,000 bps
4	640x360	800 kbps	1,020,000 bps
5	640x360	1,000 kbps	1,260,000 bps
6	768x432	1,500 kbps	1,900,000 bps
7	1024x576	2,500 kbps	3,160,000 bps
8	1280x720	4,000 kbps	4,960,000 bps
9	1920x1080	8,000 kbps	10,000,000 bps
10	3840x2160	12,000 kbps	15,000,000 bps

We used TC to limit network bandwith and insert delay, jitter and PLR, setting different conditions in upstream and downstream for each session. We drawn bandwidth and delays values from a uniform distribution, the former between 0 and 500 Mbps, and the latter from 0 to 800 ms. For jitter we drawn values based on the delay set for the session, being a random uniform value between 0 and $0.5 \times session\_delay$. Due to the sensitivity of playback quality to PLR, we opted for a Gamma distribution (with shape $k = 0.3$, and scale $\theta = 1$) instead of a uniform distribution. The Gamma distribution was derived from the traffic characterization found on *Measurement Lab (M-Lab)*[7]. Although we used a wide bandwidth range, the throughput between containers is highly affected by delay, jitter and PLR. Therefore, even settings with over 300 Mbps of bandwidth could present low MOS due to other impairments. Jitter and PLR values applied with TC are upper bounds for a random uniform sampling, therefore, those values can oscillate during a single video session.

For evaluation purposes and dataset creation, the video streaming client was instrumented to collect the following metrics in one second intervals: i) *Current video representation being played*, as described in Table III; ii) *Playback rate:* indicates whether the video playback is stalled or not; iii) *Timestamp:* marks the time the metric was collected, used to determine stall characteristics and to synchronize client logs and ICMP measurements. The collected metrics were used to calculate our ground truth MOS using ITU-T P.1203. This standard defines four "*modes of operation*" (from 0 to 3) with increasing levels of inspection of media playback and input complexity. We used the operation mode 3, which offers the most accurate estimates of MOS. This way, the MOS inferences returned by our method will better reflect the user experience. MOS calculation used the software[8] provided by Raake et al. [34] and Robitza et al. [35], setting *device type* as PC, *display resolution* as 3840x2160, and *viewing distance* as 150 cm. The player was configured with a buffer of 12 seconds. This way the network oscillations are more quickly reflected on playback quality. Other configurations of the player were kept as default.

[5]http://ffmpeg.org/
[6]https://www.videolan.org/developers/x264.html
[7]https://www.measurementlab.net/

V. RESULTS

A. Data Analysis

We executed a total of 183,876 sessions, being 63,879 of the "travel" video and the remaining with the other videos (an average of 5,000 sessions each). Each second of a video session became a data point in our dataset, resulting in over 44 million samples, with more samples of the "travel" video due to its shorter duration.

Table IV shows the Spearman correlation between the QoS metrics and the resulting MOS. The first eight rows of the table show the correlation between values set with TC and the resulting MOS. The last seven rows show the correlations between the main metrics collected by the PM and the resulting MOS. The bandwidth values showed a weak correlation with MOS. This was caused by the wide bandwidth range used for tests. Sessions with lower bandwidth values presented a stronger correlation with MOS, so we performed a more in-depth analysis for all variables.

TABLE IV
QOS AND MOS SPEARMAN CORRELATION

QoS Condition	Correlation to MOS	p-value
Downlink Bandwidth	0.0230	$p < 0.001$
Uplink Bandwidth	0.0003	0.0884
Downlink Delay	-0.2732	$p < 0.001$
Uplink Delay	-0.1650	$p < 0.001$
Downlink Jitter	-0.2697	$p < 0.001$
Uplink Jitter	-0.1246	$p < 0.001$
Downlink PLR	-0.7175	$p < 0.001$
Uplink PLR	-0.0067	$p < 0.001$
RTT (mean)	-0.2956	$p < 0.001$
RTT (median)	-0.3065	$p < 0.001$
RTT (std. dev.)	-0.1993	$p < 0.001$
Jitter (mean)	-0.2437	$p < 0.001$
Jitter (median)	-0.2745	$p < 0.001$
Jitter (std. dev)	-0.1960	$p < 0.001$
PLR	-0.4346	$p < 0.001$

Figure 5 shows the absolute correlation values for each metric. The figure shows that the effect of downlink bandwidth decreases from 0.66 (on sessions with bandwidth below 5 Mbps) to 0.27 (on sessions with bandwidth up to 25 Mbps). On the other hand, the correlation with downlink PLR quickly scales from 0.10 (at 0.1 %) to 0.61 (at 2 %). We can also observe that while the correlation between MOS and bandwidth drops, it becomes slightly stronger with delay and jitter. Correlations with uplink bandwidth and PLR are low for the entire evaluated range. PLR showed dissonant values for uplink and downlink. Downlink PLR showed strong correlation with MOS while uplink loss shows a weak correlation. It

[8]https://github.com/itu-p1203/itu-p1203

is important to highlight this difference, since our PM does not differentiate downlink from uplink loss. The implications of this limitation will be shown later in this section. It should be noted that a configured bandwidth of 80 Mbps, for example, does not translate into 80 Mbps of throughput. The effective throughput is also subject to conditions like delays and PLR.

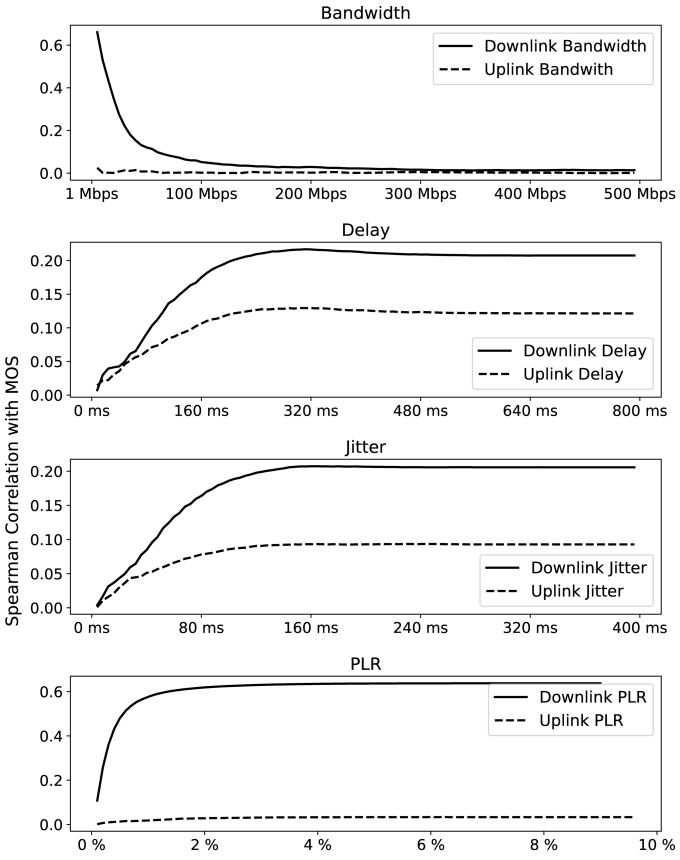


Fig. 5. Correlation according to network impairment limits

Figure 6 shows the distribution of values found in two distinct classes of sessions. We compare the distribution of QoS values for sessions with high QoE (MOS $>$ 4) and sessions with low QoE (MOS $<$ 2), in order to highlight the differences between sessions with high overall QoE and sessions with low overall QoE. Figure 6a shows that sessions with low QoE had a slightly higher amount of samples with downlink bandwidth between 0 and 10 Mbps. On the other hand, for delay and jitter (Figures 6b and 6c), we see that sessions of high quality had usually lower values, specially in downlink. Figure 6d highlights the different effect of packet losses in uplink and downlink: while high quality sessions had downlink PLR closer to zero, the uplink PLR of high quality and low quality shows the same distribution curve. These observations obtained through the experimental setup can be leveraged to define queuing policies regarding packet dropping, buffer length allocation and other control operations to optimize QoE for VoD.

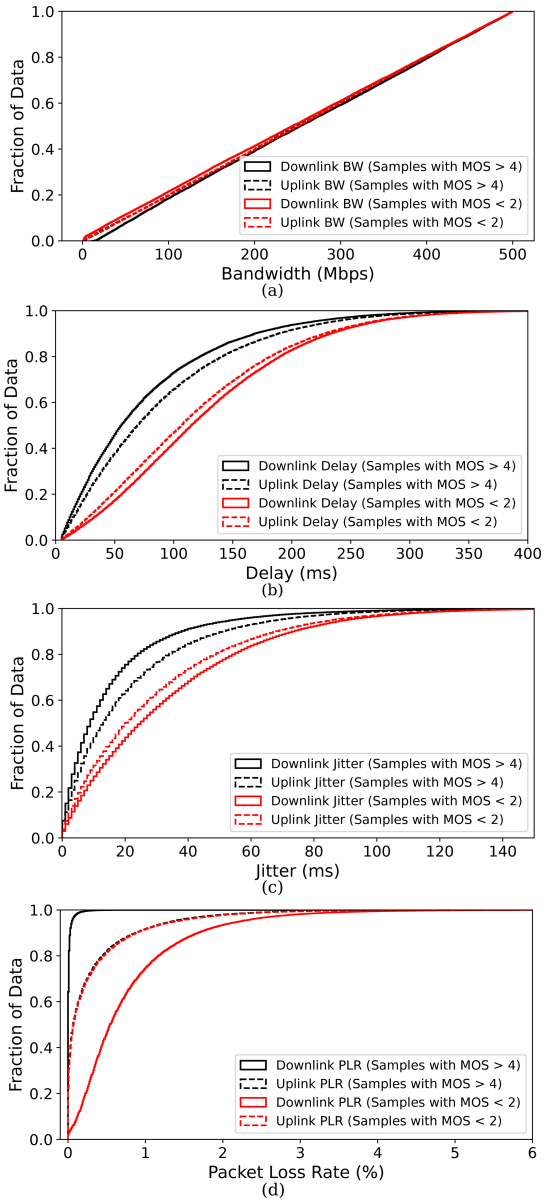


Fig. 6. Distribution of QoS values on sessions with high overall quality (MOS $>$ 4) and sessions with low overall quality (MOS $<$ 2).

B. Model Training

We evaluate two approaches for model training. The first uses data from all 25 videos to train a model (hereby referred as *Full* model). The second uses only data of the "travel" video for training, and generalization is evaluated with the remaining videos. This second model is referred as *Reduced*. The "travel" video was selected due to its shorter duration, which allows us to collect more video sessions on a given period of time. With this approach we were also able to execute more sessions with different QoS conditions. If this model shows satisfactory generalization capacity, the building time of the inference model can be significantly accelerated. For the *Full* model, a third of the sessions of each video were used for hyperparameter tuning, training and Cross Validation (CV), leaving the other two thirds for generalization analysis. For the *Reduced* model we used data from 55,893 sessions of the "travel" video for hyperparameter tuning, training and CV. 7,986 sessions of "travel" and all sessions of the other videos were used to evaluate the generalization of the *Reduced*

model.

We used random search to define the models' hyperparameters[9] [41]. Combinations of randomly selected hyperparameter values are evaluated and the set that yields the best model is selected. Random search has shown many advantages over grid search (another widely used technique for hyperparameter tuning), usually generating better models and requiring less computational time [41].

Table V describes the evaluated and selected values after 200 trials, for each model. The first three parameters introduce randomization to the training and improve the generalization capacity. Column Sampling by Tree (*colsample_bytree*) defines the percentage of randomly selected features to be used during creation of each tree. Column Sampling by Level (*colsample_bylevel*) has a similar effect, but for each level of depth in a tree. Row Sampling (*subsampling*) determines the percentage of training data to be sampled and used for training at each iteration. The Learning Rate (*learning_rate*) determines model updates and training speed. *Alpha* is a regularization term that impacts training performance and model accuracy. The Maximum Depth of each decision tree is determined by *max_depth*, and the maximum number of trees in all cases were 1000. We also used early stopping to interrupt the training after 20 iterations without accuracy improvement.

TABLE V
XGBOOST HYPERPARAMETERS EVALUATED AND SELECTED WITH RANDOM SEARCH

Hyperparameter	Evaluated Values	Selected Values	
		Full	Reduced
colsample_bytree	uniform(0.1, 1)	0.76	0.88
colsample_bylevel	uniform(0.1, 1)	0.86	0.62
subsample	uniform(0.1, 1)	0.11	0.43
learning_rate	loguniform(0.005, 0.5)	0.07	0.10
alpha	uniform(1, 5)	2	2
max_depth	uniform(1, 5)	4	4

Table VI shows the overall results obtained with each model. The first row shows the RMSE of 3-Fold CV and the standard deviation during this phase. The lower RMSE obtained by the *Reduced* model in 3-Fold CV is due to using data from the same video for train and validation, which offers less variation of video content. The same effect is observed when the final model is evaluated. On the other hand, the *Full* model generalized better than the *Reduced* model, which is expected since it was trained using samples of all videos.

TABLE VI
OVERALL INFERENCE ACCURACY OF TRAINED MODELS

Evaluation	MOS RMSE	
	Full	Reduced
3-Fold CV (Std. Dev.)	1.0462 (0.0050)	1.0364 (0.0051)
Final Model	1.0211	1.0065
Generalization	1.0376	1.0419

[9]Hyperparameters are configurations that cannot be inferred from data during training and impact on model accuracy and generalization [40]. More information about the hyperparameters can be found on https://sites.google.com/view/lauraepp/parameters

C. Generalization Analysis

For the generalization analysis we evaluate how much accuracy is lost when we train the model using data from sessions of a single video. Table VII shows the RMSE obtained with the *Full* and *Reduced* models for sessions of each individual video. The data used for this test was not part of training or hyperparameter search phases. The first column indicates the video, the second column shows the inference RMSE using the *Full* model, the third column gives the RMSE with the *Reduced* model, and the last column indicates the percent difference between RMSE values. Negative values in the fourth column indicates that the *Reduced* model generalized better for a given video than the *Full* model. This occurred for 11 videos, while for the other 14 videos the *Full* model generalized better. The *Full* model obtained a slightly lower average RMSE. Nevertheless, the results show that the accuracy loss by using the *Reduced* model is negligible, with the advantage that this model can be built faster.

TABLE VII
INFERENCE RMSE ACCORDING TO VIDEO

Video	Full	Reduced	% Difference
another	1.0468	1.0411	-0.5460
another2	1.0122	1.0197	0.7382
aworld	1.0493	1.0458	-0.3341
barcelona	1.1185	1.1184	-0.0089
bbb	1.0167	1.0110	-0.5622
bellagio	1.0659	1.0581	-0.7344
boheme	1.0664	1.0676	0.1124
curve	1.0572	1.0609	0.3493
czech	1.0364	1.0341	-0.2221
flex	1.0492	1.0673	1.7103
garden	1.0780	1.0820	0.3703
jazz	1.0665	1.0759	0.8775
jimix	1.0491	1.0536	0.4280
landscape	1.0710	1.0785	0.6978
lumix	1.1126	1.1108	-0.1619
sintel	1.0015	1.0016	0.0099
slam	1.0845	1.0758	-0.8054
surfing	1.0250	1.0232	-0.1757
swiss	1.0370	1.0442	0.6919
tearsofsteel	0.9841	0.9880	0.3955
travel	1.0352	1.0298	-0.5230
travelxp	1.0734	1.0793	0.5481
unexpected	1.0479	1.0614	1.2800
untouched	1.0479	1.0486	0.0667
wonders	1.0635	1.0625	-0.0940
Average	**1.0518**	**1.0535**	**0.1649**

D. MOS Inference

Figure 7 shows the error according to MOS range using the *Reduced* model. We observed higher inference errors in cases where the MOS is high (between 4 and 5). On the other hand, when MOS is below 4, the distribution of error values is similar for all classes of MOS, with errors below 1 in approximately 80 % of samples. This result can be taken as a pessimistic behavior of the method, inferring a low QoE when the client is actually receiving a high QoE, making the method more precise when the client is not receiving optimal QoE. This is a consequence of the limitation of the PM to distinguish downlink from uplink PLR.

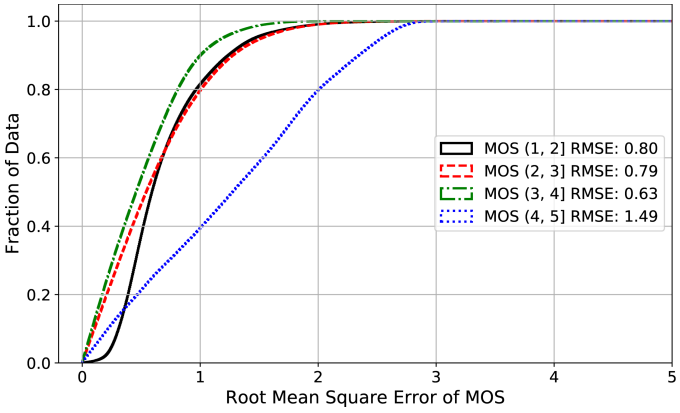


Fig. 7. MOS inference error by MOS class

We also analyzed the RMSE obtained for sessions with different configurations of bandwidth, since this metric is not measured by the PM. We split the data into 9 different ranges of downlink bandwidth and analyzed the distribution of errors for each class. The results shown in Figure 8 indicate that inference errors per class were consistent with the average RMSE shown on Table VII. Even though the PM does not monitor bandwidth, the relation between the other QoS metrics and user MOS can still be captured by the inference model.

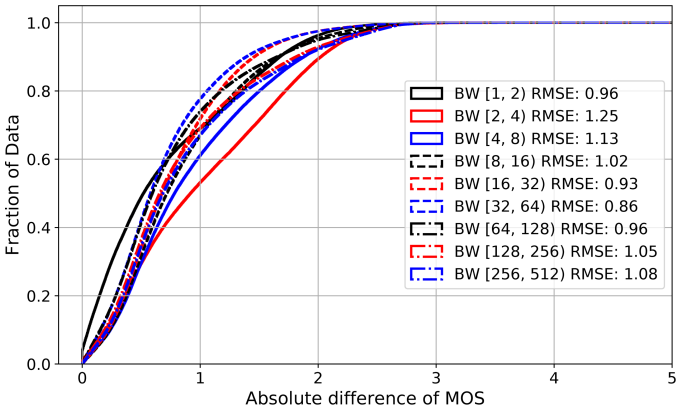


Fig. 8. MOS inference error by bandwidth range

E. Probing Module Performance

Network devices can be configured to limit ICMP probing rate and affect the precision of inferences. We performed a set of experiments to evaluate the accuracy loss when a lower probing rate is selected either to reduce probing overhead, or due to limitation by endpoints. In Figure 9 we show the absolute error obtained for each "*minSamples*" configuration of Algorithm 1. Results indicate similar errors for configurations of 8, 16 and 32 samples. From 64 to 1,024 samples the median error dropped from 1 to 0.6. These results indicate that it is possible to obtain estimates even with probing restrictions, albeit with slightly higher error. Based on Equation 2 we estimated a probing overhead of 27 Kbps during monitoring. The traffic generated to constantly monitor a video session for one hour would generate approximately 11.5 MBytes of additional traffic, using *minSamples* configuration of 1,000. This value can be even lower if the PM is configured to use fewer samples or monitor for shorter intervals. This overhead can take up to 1.4% of traffic if the video is served at lowest quality. If we consider the video served at highest quality the value is negligible.

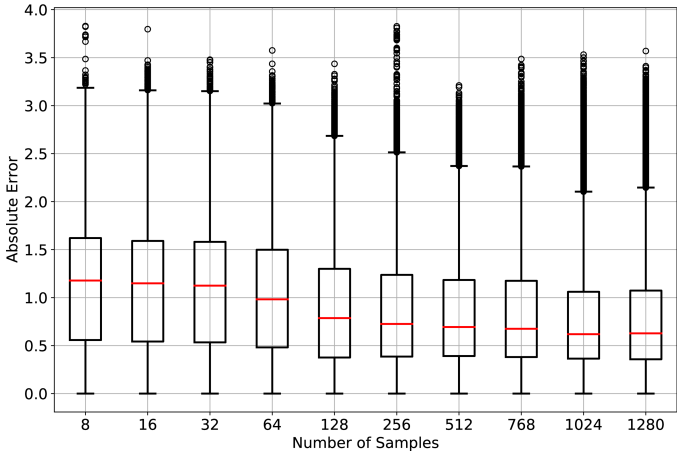


Fig. 9. Inference error according to number of collected samples

VI. Testbed Results

We performed experiments on a production network using four videos with different content, namely "*another*", "*barcelona*", "*jimix*", and "*travel*". Our experiments were performed using two distinct testbeds in Belgium, shown in Figure 10. The first is Virtual Wall[10] in the city of Ghent, and the second is CityLab[11] [42] in the city of Antwerp, both separated by approximately 53Km crossing administrative domains of Ghent University and University of Antwerp. We deployed the VoD server at the Virtual Wall testbed. Using CityLab we set up 10 different pairs of Wi-Fi Access Point (AP) and Clients, detailed on Table VIII. With the exception of *Setup 1*, used as baseline, we selected pairs that the testbed reported less than ideal connectivity conditions (e.g. asymmetric link reliability, or high noise). On the table we also show the number of other APs that our nodes detected operating on the same channel. The APs were connected to the server and allowed the clients to consume the videos. The monitor container, also deployed at Virtual Wall, performed ICMP probing and MOS inferences. More information about node specifications and locations can be found on the testbed website.

The connection between VirtualWall and CityLab nodes (i.e., between VoD server and the AP) was stable, achieving throughput of over 500Mbps with RTT of 3.01 ms, jitter lower than 0.5 ms, and 0 % of PLR. Therefore, network impairments during tests were caused by interference and connectivity issues on the wireless segment. We performed 10 repetitions for each video in each setup. Figure 11 shows the distribution of MOS of the sessions performed in the testbed, indicating

[10]https://doc.ilabt.imec.be/ilabt/virtualwall/
[11]https://doc.lab.cityofthings.eu/wiki/Main_Page

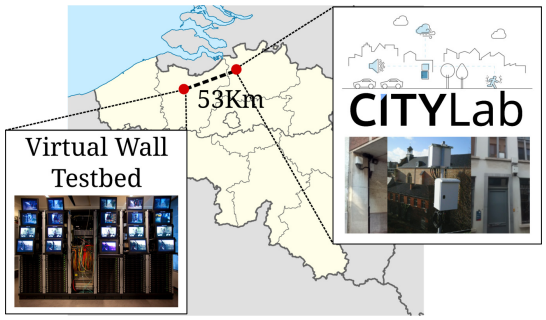


Fig. 10. Experimental testbed in Belgium

TABLE VIII
SETUP OF TESTBED NODE PAIRS AND OTHER APS DETECTED CONFIGURED ON THE SAME CHANNEL

Setup	Node Pair		Other APs	
	AP	Client	AP	Client
1	6	72	6	6
2	71	6	9	6
3	24	28	1	5
4	14	18	10	4
5	34	35	7	27
6	14	15	4	1
7	4	5	4	4
8	3	5	5	4
9	8	12	1	0
10	4	36	8	18

30 % of samples with the highest level of MOS. However, we still have significant amount of samples around MOS values of two, three and four.

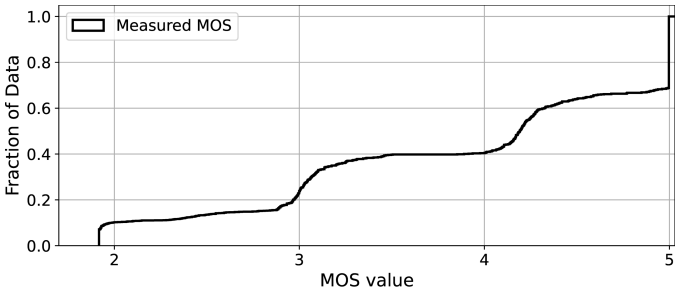


Fig. 11. Distribution of MOS values during sessions in the testbed

Table IX shows the inference RMSE obtained for each video and setup of the testbed, using the *Reduced Model*. The last column of the table shows the RMSE achieved using all videos on the same setup. Similarly, the last row shows the RMSE for each video across all setups. The cell on the bottom right shows the overall RMSE achieved in the testbed experiments. We observed that on setups 2, 4, 5, and 10 the RMSEs were above the expected. Comparing with Table VIII, these setups had more networks configured on the same channel near to our nodes, which may cause higher link fluctuations and link asymmetry. Such link asymmetry may reduce the accuracy of the models, as mentioned on *Limitation #1*. Nevertheless, the results indicate that the method is capable of providing MOS estimates within the expected error margin, especially considering the challenges posed by the wireless connections.

TABLE IX
INFERENCE RMSE ON TESTBED

	RMSE by Video and Setup				
	another	barcelona	jimix	travel	All videos
Setup 1	0.30	0.26	0.24	0.28	**0.27**
Setup 2	1.68	1.17	1.44	1.06	**1.38**
Setup 3	1.19	1.13	1.14	1.26	**1.17**
Setup 4	1.89	1.45	1.93	1.85	**1.78**
Setup 5	1.76	1.62	1.83	1.58	**1.71**
Setup 6	1.08	1.28	1.27	0.77	**1.15**
Setup 7	1.02	0.88	1.01	1.18	**1.02**
Setup 8	0.97	0.76	0.75	0.81	**0.82**
Setup 9	1.08	0.77	1.13	1.13	**1.04**
Setup 10	1.79	1.64	1.46	1.40	**1.58**
All Setups	**1.37**	**1.18**	**1.32**	**1.21**	**1.28**

Figure 12 shows the distribution of inference errors of both models on the testbed experiments. For the testbed results we observe that the *Full Model* is slightly more accurate. This is expected since this model was trained using more data and from more different videos. These results are also in line with the results of Table VII, that shows the *Full Model* with marginally better performance. The results obtained with the wireless testbed indicate that despite not using data collected on wireless networks, the method can still provide relevant feedback across different technologies.

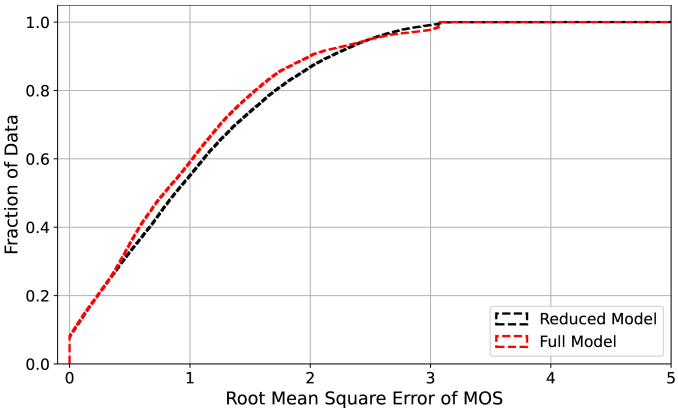


Fig. 12. Distribution of RMSE on testbed using both Full and Reduced models

VII. CONCLUSION AND FUTURE WORK

In this work we propose a practical method for QoE inference for DASH VoD. Different from other methods in the literature, ours does not require instrumentation of client devices, modification of existing network elements, deployment of monitoring tools in multiple network points, or deep inspection of video traffic. Instead, a Probing Module performs active ICMP probing using a freely available ping tool. As ICMP is widely supported by network devices, the PM can be used even with legacy equipment. Network measurements are fed to a ML model that takes QoS values as input and estimates MOS based on the ITU-T P.1203 Recommendation.

We evaluated two methods of creating such model and concluded that a model trained using samples obtained with a single video can perform MOS inferences with similar

accuracy if the videos available on a VoD server are prepared following a consistent bitrate ladder. The monitoring overhead can take up to 1.4 % of traffic if the video is served at lowest quality, however, at the highest quality levels the probing overhead is negligible. Results in a production networks show the applicability of the solution, despite ICMP rate limiting policies. In future work we plan to perform more experiments in real deployments and with clients that employ different quality adaptation techniques. We will also investigate the combination of QoS measurements provided by different monitoring tools, for example, using In-band Network Telemetry (INT) to measure QoS within the provider's network, and the ICMP measurements to cover the last mile of links. We aim to apply this method as a feedback signal for an automated network control loop. We will evaluate methods to identify asymmetric links and their applicability to enhance the PM. This way we can improve QoE estimation accuracy over asymmetric packet loss conditions.

Acknowledgment

This work was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001, CNPq (funding agency from the Brazilian federal government), FAPEMIG (Minas Gerais State Funding Agency), and São Paulo Research Foundation (FAPESP) with Brazilian Internet Steering Committee (CGI.br), grants 2018/23097-3 and 2020/05182-3.

The work has also been supported by the Horizon 2020 Fed4FIRE+ project, Grant Agreement No. 723638, and by the FLEXNET project: "Flexible IoT Networks for Value Creators" (Celtic 2016/3), in the Eureka Celtic-Next Cluster.

Glossary

AP	Access Point
AQoS	Application QoS
CDN	Content Delivery Network
CV	Cross Validation
DASH	Dynamic Adaptive Streaming over HTTP
DPI	Deep Packet Inspection
DT	Decision Tree
HAS	HTTP Adaptive Streaming
ICMP	Internet Control Message Protocol
INT	In-band Network Telemetry
ISP	Internet Service Provider
MEC	Mobile Edge Computing
ML	Machine Learning
MOS	Mean Opinion Score
MPD	Media Presentation Description
NAT	Network Address Translation
NQoS	Network-level QoS
OTT	Over The Top
PLR	Packet Loss Rate
PM	Probing Module
QoE	Quality of Experience
QoS	Quality of Service
RF	Random Forests
RMSE	Root Mean Square Error
RTT	Round-Trip Time
SLA	Service-Level Agreement
TC	Traffic Control
VoD	Video on Demand
XGBoost	eXtreme Gradient Boosting

References

[1] Cisco, "White paper: Cisco Visual Networking Index: Forecast and Trends, 2017–2022," *Cisco*, 2018, Accessed on: Oct. 16, 2020. [Online]. Available: https://www.cisco.com/c/dam/m/en_us/network-intelligence/service-provider/digital-transformation/knowledge-network-webinars/pdfs/1213-business-services-ckn.pdf

[2] Accenture, "Majority of Smartphone Users Surveyed Unhappy with Service and Ready to Switch Mobile Providers, Accenture Screenager Report Finds," 2016, Accessed on: Oct. 16, 2020. [Online]. Available: https://newsroom.accenture.com/news/majority-of-smartphone-users-surveyed-unhappy-with-service-and-ready-to-switch-mobile-providers-accenture-screenager-report-finds.htm

[3] I. Rec, "P. 10: Vocabulary for performance and quality of service, amendment 2: New definitions for inclusion in recommendation itu-t p. 10/g. 100," *Int. Telecomm. Union, Geneva*, 2008. [Online]. Available: https://www.itu.int/rec/T-REC-P.10

[4] T. Mangla, E. Zegura, M. Ammar, E. Halepovic, K. W. Hwang, R. Jana, and M. Platania, "VideoNOC: Assessing video QoE for network operators using passive measurements," *ACM Multimedia Systems Conference, (MMSys)*, pp. 101–112, 2018. [Online]. Available: https://doi.org/10.1145/3204949.3204956

[5] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hoßfeld, and P. Tran-Gia, "A Survey on Quality of Experience of HTTP Adaptive Streaming," *IEEE Communications Surveys Tutorials*, vol. 17, no. 1, pp. 469–492, 2015. [Online]. Available: https://doi.org/10.1109/COMST.2014.2360940

[6] Sandvine, "Global Internet Phenomena Report: Latin America and North America," Tech. Rep., 2016, Accessed on: Oct. 16, 2020. [Online]. Available: https://www.sandvine.com/hubfs/downloads/archive/2016-global-internet-phenomena-report-latin-america-and-north-america.pdf

[7] P. Casas, M. Seufert, F. Wamser, B. Gardlo, A. Sackl, and R. Schatz, "Next to You: Monitoring Quality of Experience in Cellular Networks from the End-Devices," *IEEE Transactions on Network and Service Management*, vol. 13, no. 2, pp. 181–196, 2016. [Online]. Available: https://doi.org/10.1109/TNSM.2016.2537645

[8] ITU-T Telecommunication Standardization Sector, "ITU-T Rec P.1203: Parametric bitstream-based quality assessment of progressive download and adaptive audiovisual streaming services over reliable transport," 2017. [Online]. Available: https://www.itu.int/rec/T-REC-P.1203

[9] N. Barman and M. G. Martini, "QoE Modeling for HTTP Adaptive Video Streaming-A Survey and Open Challenges," *IEEE Access*, vol. 7, pp. 30 831–30 859, 2019. [Online]. Available: https://doi.org/10.1109/ACCESS.2019.2901778

[10] K. Jia, Y. Guo, Y. Chen, and Y. Zhao, "Measuring and predicting quality of experience of DASH-based video streaming over LTE," *International Symposium on Wireless Personal Multimedia Communications, WPMC*, pp. 102–107, 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7954462

[11] R. I. T. Da Costa Filho, W. Lautenschlager, N. Kagami, V. Roesler, and L. P. Gaspary, "Network fortune cookie: Using network measurements to predict video streaming performance and QoE," *2016 IEEE Global Communications Conference, GLOBECOM 2016 - Proceedings*, pp. 2–7, 2016. [Online]. Available: https://doi.org/10.1109/GLOCOM.2016.7842022

[12] B. Frank, I. Poese, Y. Lin, G. Smaragdakis, A. Feldmann, B. Maggs, J. Rake, S. Uhlig, and R. Weber, "Pushing CDN-ISP collaboration to the limit," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 3, pp. 34–44, 2013. [Online]. Available: https://doi.org/10.1145/2500098.2500103

[13] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, "Collaborative mobile edge computing in 5G networks: New paradigms, scenarios, and challenges," *IEEE Communications Magazine*, vol. 55, no. 4, pp. 54–61, 2017. [Online]. Available: https://doi.org/10.1109/MCOM.2017.1600863

[14] G. Miranda, D. F. Macedo, and J. M. Marquez-Barja, "A QoE Inference Method for DASH Video Using ICMP Probing," in *2020 16th International Conference on Network and Service Management (CNSM)*, 2020, pp. 1–5. [Online]. Available: https://doi.org/10.23919/CNSM50824.2020.9269120

[15] G. L. Dos Santos, V. T. Guimarães, J. G. Silveira, A. T. Vieira, J. A. De Oliveira Neto, R. I. T. Da Costa Filho, and R. Balbinot, "UAMA: A unified architecture for active measurements in IP networks End-to-end objetive quality indicators," *10th IFIP/IEEE International Symposium on Integrated Network Management 2007, IM '07*, pp. 246–253, 2007. [Online]. Available: https://doi.org/10.1109/INM.2007.374789

[16] R. Huysegems, B. De Vleeschauwer, K. De Schepper, C. Hawinkel, T. Wu, K. Laevens, and W. Van Leekwijck, "Session reconstruction for HTTP adaptive streaming: Laying the foundation for network-based QoE monitoring," in *2012 IEEE 20th International Workshop on Quality of Service*, 2012, pp. 1–9. [Online]. Available: https://doi.org/10.1109/IWQoS.2012.6245987

[17] M. J. Khokhar, T. Ehlinger, and C. Barakat, "From network traffic measurements to QoE for internet video," *IFIP Networking Conference*, pp. 1–9, 2019. [Online]. Available: https://doi.org/10.23919/IFIPNetworking.2019.8816854

[18] C. Ge and N. Wang, "Real-time QoE estimation of DASH-based mobile video applications through edge computing," *INFOCOM 2018 - IEEE Conference on Computer Communications Workshops*, pp. 766–771, 2018. [Online]. Available: https://doi.org/10.1109/INFCOMW.2018.8406935

[19] M. Seufert, P. Casas, N. Wehner, L. Gang, and K. Li, "Stream-based Machine Learning for Real-time QoE Analysis of Encrypted Video Streaming Traffic," *Proceedings of the 2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops, ICIN 2019*, pp. 76–81, 2019. [Online]. Available: https://doi.org/10.1109/ICIN.2019.8685901

[20] S. Wassermann, M. Seufert, P. Casas, L. Gang, and K. Li, "Let me decrypt your beauty: Real-time prediction of video resolution and bitrate for encrypted video streaming," in *TMA 2019 - Proceedings of the 3rd Network Traffic Measurement and Analysis Conference*, 2019, pp. 199–200. [Online]. Available: https://doi.org/10.23919/TMA.2019.8784589

[21] C. G. Bampis, Z. Li, A. K. Moorthy, I. Katsavounidis, A. Aaron, and A. C. Bovik, "Study of temporal effects on subjective video quality of experience," *IEEE Transactions on Image Processing*, vol. 26, no. 11, pp. 5217–5231, 2017. [Online]. Available: https://doi.org/10.1109/TIP.2017.2729891

[22] T. Hossfeld, S. Egger, R. Schatz, M. Fiedler, K. Masuch, and C. Lorentzen, "Initial delay vs. interruptions: Between the devil and the deep blue sea," in *2012 Fourth International Workshop on Quality of Multimedia Experience*, 2012, pp. 1–6. [Online]. Available: https://doi.org/10.1109/QoMEX.2012.6263849

[23] P. Casas, R. Schatz, and T. Hoßfeld, "Monitoring youtube qoe: Is your mobile network delivering the right experience to your customers?" in *2013 IEEE Wireless Communications and Networking Conference (WCNC)*, 2013, pp. 1609–1614. [Online]. Available: https://doi.org/10.1109/WCNC.2013.6554804

[24] H. D. Moura, D. F. Macedo, and M. A. Vieira, "Wireless control using reinforcement learning for practical web qoe," *Computer Communications*, vol. 154, pp. 331–346, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366419314860

[25] R. Iraja Tavares Da Costa Filho, W. Lautenschlager, N. Kagami, M. Caggiani Luizelli, V. Roesler, and L. Paschoal Gaspary, "Scalable QoE-aware Path Selection in SDN-based Mobile Networks," *Proceedings - IEEE INFOCOM*, pp. 989–997, 2018. [Online]. Available: https://doi.org/10.1109/INFOCOM.2018.8486427

[26] M. Carvalho and D. F. Macedo, "QoE-Aware Container Scheduler for Co-located Cloud Environments," in *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2021, pp. 286–294. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9463920

[27] K. L. Dias, M. A. Pongelupe, W. M. Caminhas, and L. de Errico, "An innovative approach for real-time network traffic classification," *Computer Networks*, vol. 158, pp. 143–157, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128618305541

[28] 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*, vol. 24, no. 3, pp. 1999–2012, Feb 2020. [Online]. Available: https://doi.org/10.1007/s00500-019-04030-2

[29] ITU Telecommunication Standardization Sector, "ITU-T Rec Y.1541: Network performance objectives for IP-based services," 2011. [Online]. Available: https://www.itu.int/rec/T-REC-Y.1541

[30] S. Aroussi and A. Mellouk, "Statistical evaluation for quality of experience prediction based on quality of service parameters," *International Conference on Telecommunications (ICT)*, pp. 1–5, 2016. [Online]. Available: https://doi.org/10.1109/ICT.2016.7500364

[31] Z.-H. Zhou, *Ensemble methods: foundations and algorithms*. Chapman and Hall/CRC, 2019.

[32] D. Nielsen, "Tree Boosting with XGBoost - Why does XGBoost Win Every Machine Learning Competition?" Master's thesis, NTNU, 2016. [Online]. Available: https://ntnuopen.ntnu.no/ntnu-xmlui/bitstream/handle/11250/2433761/16128_FULLTEXT.pdf

[33] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2016. [Online]. Available: https://doi.org/10.1145/2939672.2939785

[34] A. Raake, M.-N. Garcia, W. Robitza, P. List, S. Göring, and B. Feiten, "A bitstream-based, scalable video-quality model for HTTP adaptive streaming: ITU-T P.1203.1," in *International Conference on Quality of Multimedia Experience (QoMEX)*, May 2017. [Online]. Available: https://doi.org/10.1109/QoMEX.2017.7965631

[35] W. Robitza, S. Göring, A. Raake, D. Lindegren, G. Heikkilä, J. Gustafsson, P. List, B. Feiten, U. Wüstenhagen, M.-N. Garcia, K. Yamagishi, and S. Broom, "HTTP Adaptive Streaming QoE Estimation with ITU-T Rec. P.1203 – Open Databases and Software," in *ACM Multimedia Systems Conference*, 2018. [Online]. Available: https://doi.org/10.1145/3204949.3208124

[36] J. Padhye, V. Firoiu, D. F. Towsley, and J. F. Kurose, "Modeling TCP Reno performance: a simple model and its empirical validation," *IEEE/ACM Transactions on Networking*, vol. 8, no. 2, pp. 133–145, 2000. [Online]. Available: https://doi.org/10.1109/90.842137

[37] Zesheng Chen, Tian Bu, M. Ammar, and D. Towsley, "Comments on "Modeling TCP reno performance: a simple model and its empirical Validation"," *IEEE/ACM Transactions on Networking*, vol. 14, no. 2, pp. 451–453, 2006. [Online]. Available: https://doi.org/10.1109/TNET.2006.872541

[38] E. W. Chan, A. Chen, X. Luo, R. K. Mok, W. Li, and R. K. Chang, "TRIO: Measuring asymmetric capacity with three minimum round-trip times," *Proceedings of the 7th Conference on Emerging Networking EXperiments and Technologies, CoNEXT'11*, 2011. [Online]. Available: https://doi.org/10.1145/2079296.2079311

[39] W. Reese, "Nginx: The high-performance web server and reverse proxy," *Linux J.*, no. 173, Sep. 2008. [Online]. Available: https://doi.org/10.5555/1412202.1412204

[40] M. Kuhn and K. Johnson, *Applied predictive modeling*. Springer, 2013, vol. 26. [Online]. Available: https://doi.org/10.1007/978-1-4614-6849-3

[41] J. Bergstra and Y. Bengio, "Random Search for Hyper-Parameter Optimization," *J. Mach. Learn. Res.*, vol. 13, no. null, p. 281–305, Feb. 2012. [Online]. Available: https://doi.org/10.5555/2188385.2188395

[42] J. Struye, B. Braem, S. Latré, and J. Marquez-Barja, "The CityLab testbed — Large-scale multi-technology wireless experimentation in a city environment: Neural network-based interference prediction in a smart city," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2018, pp. 529–534. [Online]. Available: https://doi.org/10.1109/INFCOMW.2018.8407018

Gilson Miranda Jr. holds B.Sc. and M.Sc. degrees in Computer Science from Federal University of Lavras (UFLA), Brazil. In 2017 started his PhD in Computer Science at the Federal University of Minas Gerais (UFMG), Brazil. Now is pursuing a Joint PhD in Applied Engineering at the University of Antwerp, Belgium, where he is carrying his research with IDLab. His main research interests are programmable networks, wireless networks, and machine learning applied to network management.

Daniel F. Macedo is Professor Adjunto (equivalent to the Associate Professor level in the US) in the Computer Science Department (DCC) in Federal University of Minas Gerais (UFMG), Brazil. He holds the CNPq research productivity scholarship level 2, a scholarship granted to the most performing researchers in Brazilian academia. He was a post-doc researcher in UFMG, Brazil. He holds a PhD in computer science from Université Pierre et Marie Curie-Paris VI. He also holds a M.Sc and a B.Sc. in Computer Science from UFMG. His main research interests are wireless networks, ad hoc and sensor network, network management, Delay-tolerant networking, software-defined networking and autonomic computing.

Johann Marquez-Barja is a Professor at University of Antwerp, as well as a Professor in IMEC, Belgium. He is leading the Wireless Cluster at IDLab/imec Antwerp. He was and is involved in several European research projects with leading roles. He is a member of ACM, and a Senior member of the IEEE Communications Society, IEEE Vehicular Technology Society, and IEEE Education Society where he participates in the board of the Standards Committee. His main research interests are: 5G advanced architectures including edge computing; flexible and programmable future end-to-end networks; IoT communications and applications. He is also interested in vehicular communications, mobility, and smart cities deployments. Prof. Marquez-Barja is co-leading the Citylab Smart City testbed, part of the City of Things programme, and the SmartHighway testbed, both located in Antwerp, Belgium. He has given several keynotes and invited talks in different major events, as well as received 30 awards in his career so far, and co-authored more than 180 published works including publications, editorials, and books. He is also serving as Editor and Guest editor for different International Journals, as well as participating in several Technical Programme and Organizing Committees for several worldwide conferences/congresses.