Machine Learning-based End-to-End QoE Monitoring Using Active Network Probing

Gilson Miranda Jr.*†, Esteban Municio*, Johann M. Marquez-Barja*, Daniel Fernandes Macedo†
*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, esteban.municio, johann.marquez-barja}@uantwerpen.be, damacedo@dcc.ufmg.br

***Abstract*—Video on Demand (VoD) is responsible for a significant amount of traffic on IP networks. To meet users' expectations, network operators need means to monitor and to identify when service quality is degraded in order to take actions to avoid customer churn. Many proposals in the literature correlate network Quality of Service (QoS) metrics with indicators of user Quality of Experience (QoE). However, most solutions cannot monitor end-to-end conditions without modification on video player applications or require deep packet inspection techniques, which may raise privacy issues. In previous work, we proposed a method to estimate QoE using active ICMP probing, which is widely supported by network devices and can be used for end-to-end network measurements. In this work, we improve our previous method by adding a secondary model that operates over the first step of QoE inferences. We also extend the evaluation of our approach by using two wireless and wired testbeds, reporting our results for different end-to-end setups subject to distinct connectivity conditions. Finally, we identify and discuss the advantages and limitations of our methods and assess their suitability in real-world production deployments.**

***Index Terms*—DASH Video, QoE, Machine Learning**

I. Introduction

Video on Demand (VoD) accounts for significant amounts of traffic over the Internet. Forecasts show that video traffic will comprise over 80% of all IP traffic by 2022 [1]. In order to keep users satisfied with their networking experience and avoid customer churn, content and network operators must be able to identify when user experiences are unsatisfactory. Because of the high relevance of VoD with the widespread consumption of services such as Netflix, YouTube, and Prime Video, network operators and researchers have been seeking ways to monitor user-perceived quality for such services [2]. However, network Quality of Service (QoS) metrics like bandwidth, delays, or Packet Loss Ratio (PLR) do not translate directly into the user-perceived quality of a VoD service [3]. Instead, the concept of Quality of Experience (QoE) is used to measure or estimate the user's subjective perception of a service.

QoE is defined by the International Telecommunication Union (ITU) as the degree of user satisfaction (delight or annoyance) with an application or service [4]. Previous work has shown that the relation between network QoS and QoE is not linear, and Machine Learning (ML) methods have been employed along the last decade to perform the mapping of objective network or application QoS metrics into objective or subjective QoE indicators [5]. Many proposals in the literature lack the ability to monitor the last-mile link, which in many cases is the network bottleneck, especially in wireless networks [6]. Therefore, QoE monitoring methods that neglect the last mile may be unable to detect degraded QoE.

User satisfaction is highly influenced by events such as video playback stalls, initial buffering delay, and oscillations in video quality [7]. Using client-side information to estimate user QoE is one of the approaches found in the literature [8], [9]. The problem with client-side QoE monitoring is that it requires changes in video players or the deployment of additional software on clients or servers. With network-only techniques, on the other hand, providers do not need to have explicit agreements with the content owners to improve the user QoE [5]. Further, many existing works require extensive knowledge about content characteristics or rely on technology-specific information (such as TCP headers, Wi-Fi link-layer counters) [10]–[12]. This makes it difficult to deploy such solutions on Internet Service Providers (ISPs) and more heterogeneous networks since they require access to the content for preprocessing or access to input metrics that are not present across different network technologies.

In previous work, we proposed a method to perform end-to-end QoE inference for VoD using active probing and an ML model that takes network QoS as input and estimates user QoE [13]. By using Internet Control Message Protocol (ICMP) probing, it also monitors the last-mile link, enabling network operators to identify issues between a Wi-Fi Access Point (AP) and a client. In the previous work, we used an emulated environment to collect data for model training and evaluation. The setup was useful for prototyping and parallel execution of experiments, allowing us to collect over 60,000 video sessions with distinct network conditions. This setup allowed us to identify the effect of different network QoS conditions on user QoE.

The contributions of this work are two-fold: first, we improved our previous model by evaluating more features that could offer better inferences and selected those that improved most the model accuracy. Secondly, this work validates and extends our previous emulated environment results, demonstrating the capacity of the method to estimate QoE using ICMP probing to measure network QoS conditions. Our evaluation comprises both wired and wireless network segments. The evaluation of our QoE inference method is carried out in a reproducible manner using two different real-world open

testbeds: CityLab[1] [14] as smart city/wireless testbed, and Virtual Wall[2] as a cloud testbed. For the wireless domain, we perform experiments using indoor and outdoor nodes, with different levels of noise and link conditions (e.g. suffering from Wi-Fi interference and signal oscillations existing in the city of Antwerp). For the wired domain, we deploy high-performance nodes to deliver video content from a catalog.

This work validates and extends our previous emulated environment results, demonstrating that the capacity of the method to estimate the quality level selected by the video client still stays within expected accuracy levels even in a realistic, large-scale scenario comprising both wired and wireless network segments. The new experiments highlighted the effects of link asymmetry on inference accuracy as observed on emulated setup. We added a secondary model that receives the inferred values and statistics about such inferences to further improve the accuracy of the QoE estimations. Finally, we discuss the advantages and limitations of our QoS inference model while ensuring the reproducibility of the experiments.

II. Related Work

Methods for QoE inference for video applications based on network measurements have been studied for many years. Such methods can provide insights for network operators about the experience being delivered to the users. Approaches based on Deep Packet Inspection (DPI) to reconstruct the video sessions and estimate stalls or quality switches are becoming infeasible with the wide adoption of encryption protocols [15]. To overcome these limitations, some proposals rely on the analysis of packet counters, the volume of UDP and TCP packets, and other indirect measurements. Such methods have been successfully used for estimating playback stalls, and video resolution for YouTube traffic [16], [17]. Although playback stalls represent one of the main causes for QoE degradation [18], video playback in low resolution or constant resolution changes can also severely hinder user experience [7].

The ITU-T P.1203 Recommendation describes models for quality assessment that comprise stalls, video resolution, resolution changes, and other information to provide better estimates of user experience with streamed media [8]. The work by Khokhar et al. [19] presents an ML-based method that takes network-level measurements and estimates QoE (in terms of Mean Opinion Score (MOS)) using ITU-T P.1203 models, for YouTube traffic. The input for the ML model comprises up to 48 features, including bandwidth, Round-Trip Time (RTT), jitter, packet inter-arrival times, along with features inferred from the traffic traces. Ul Mustafa et al. [20] also employ the ITU-T P.1203 to label a dataset, and train models that take packet-level statistics as input and classify QoE as *poor*, *average*, or *good*.

A common approach to creating training datasets is to use the Traffic Control (TC)[3] tool to insert network impairments between VoD server and client, and obtain a varied range of network-level QoS conditions [13], [19]–[21]. However, in many proposals the methods to perform the monitoring on real deployments are not specified [19], [20]. Active probing using specialized tools has been previously used to obtain network-level measurements, and then map such measurements into indicators of user experience [21]. A common issue with current monitoring tools is on how to cover the last mile of the end-to-end connection between server and client since such tools usually require agents to be installed on the device in order to perform measurements.

III. QoE Inference Using ICMP Probing

We proposed a method for QoE monitoring that uses the widely supported ICMP protocol, which allows us to perform end-to-end measurements through active probing [13]. Figure 1 gives an overview of our method. We consider a context where small-scale Content Delivery Networks (CDNs) are deployed within the domain of an ISP. The server offers a VoD service through Dynamic Adaptive Streaming over HTTP (DASH), which is the method used by the main VoD services nowadays. The ISP has no access to server logs but can deploy a Probing Module (PM) to perform active probing to the server and the client. The ISP can also configure routes in its domain so the majority of the probing flow follow the same route as the video flow.

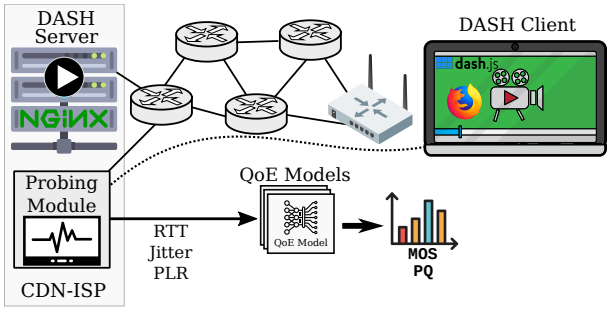

Fig. 1. Overview of the QoE inference method

The PM continuously measures the RTT, jitter, and PLR between server and client by using ICMP probing. This allows us to perform end-to-end measurements, as long as no restrictions are actively applied by any device. The PM runs concurrent threads independently probing the destination in adjustable intervals. A coordinator thread aggregates the results from the threads and adjusts the interval between probing messages based on the observed RTT values. This way the PM is able to obtain 1000 samples in the time window of 30 seconds, dynamically adjusting the probing frequency as network conditions change. Measurements older than 30 seconds are discarded, avoiding excessive resource usage.

Statistics about RTT, jitter, and PLR are given as input to ML models based on an ensemble method of regression trees, specifically eXtreme Gradient Boosting (XGBoost) [22]. We selected XGBoost as it has shown better performance than other methods across a variety of ML problems [23]. The

[1]https://doc.lab.cityofthings.eu/wiki/Main_Page
[2]https://doc.ilabt.imec.be/ilabt/virtualwall/
[3]https://man7.org/linux/man-pages/man8/tc.8.html

model maps the input QoS information into an MOS value between 1 and 5, based on ITU-T P.1203 Recommendation [8]. Since the model is based on supervised learning, it requires a labeled dataset for training. Therefore, we created such a dataset using an emulated setup, detailed in Section IV-B.

A. Improvements Over our Previous Model

In the previous work, our model received as inputs the mean RTT, mean jitter, and PLR over 1000 probing attempts in a window of 30 seconds. We evaluated other methods to aggregate such measurements and identified that *median* RTT, *median* jitter, as well as the 90th percentiles of RTT and jitter provided better results. Moreover, during our initial experimentation using the testbed, we identified that during sessions with sub-optimal QoE the inferred MOS also showed higher variation, and thus, higher standard deviation between inferred values along time. Therefore, incorporating such information into the model would be useful to improve its accuracy.

Figure 2 gives an overview of the updated inference method, composed of two models. Both models consume the same QoS statistics from the PM, i.e. median RTT, 90th percentile of RTT, median jitter, 90th percentile of jitter, and PLR. The *Primary Model* executes first, returning one MOS estimate each second, based on the most recent QoS statistics. The per-second MOS estimates from the Primary Model are accumulated by the *Postprocess* module, which generates six statistics: standard deviation of MOS values for the last 10, 20, and 30 seconds; and mean MOS values for the last 10, 20, and 30 seconds. After receiving the first output from the Primary Model the Postprocess module already starts generating the first statistics, allowing the Secondary model to run. However, only after the initial 30 seconds all the values can be calculated with the correct amount of data. Nonetheless, we did not observe significant errors caused by this initialization process in our experiments, and its duration is negligible in most cases.

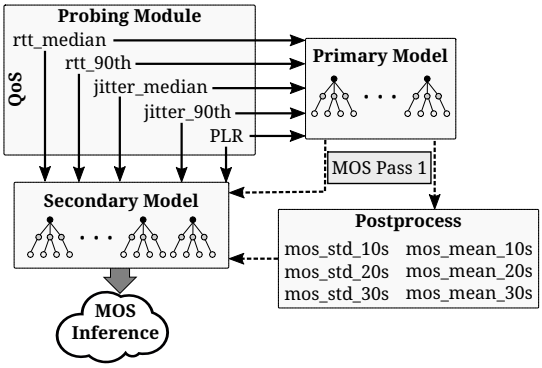

Fig. 2. Overview of the inference model.

The *Secondary Model* consumes the QoS statistics from the PM, the most recent MOS estimate provided by the Primary Model (MOS Pass 1), and concatenates with the statistics generated by the Postprocess module. The output of the Secondary Model is the final inference of MOS. The training of the Secondary Model requires augmenting the dataset with the MOS values estimated by the Primary Model. We detail the training process and how we split the dataset in Section IV-B. The performance achieved by both models in terms of inference accuracy is addressed in Section V.

IV. Experimental Setup

This section first presents the testbeds. Later on, it details the data collection using the emulated environment to create the training dataset and train the models. Finally, it describes the specific setups of the experiments.

A. Virtual Wall and CityLab Testbeds

There are several testbeds for network experimentation on different domains such as Cloud [24], IoT [25], Smart Highways [26] or Smart Cities [27]–[29]. The experiments in this work were carried using the Virtual Wall and the CityLab testbeds, (see Figure 3). Virtual Wall is a cloud deployment at Ghent University, while CityLab consists of dozens of nodes at different locations in the city of Antwerp.

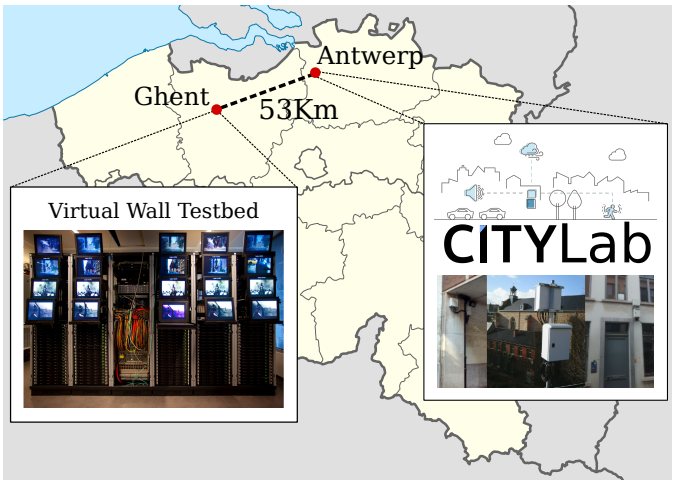

Fig. 3. Virtual Wall and CityLab testbeds in Belgium

Virtual Wall offers a wide range of node types, with over 10 different configurations and flavors. The nodes used in our experiments are the *pcgen2* nodes with 2 Quad-core Intel E5520 2.2GHz CPUs and 12GB of RAM. For more information about both testbeds, we recommend the imec-iLab.t documentation[4]. CityLab nodes are composed of PC Engines apu2c4[5] boards with an AMD GX-412TC 1GHz Quad-core CPU and 4GB DDR-1333 MHz of RAM. These nodes are equipped with multiple wireless cards of different technologies such as Wi-Fi, LoRa, and Dash 7.

B. Dataset Creation and Model Training

We created the dataset by using the emulated infrastructure shown in Figure 4. The three components were deployed as containers, and network impairments between DASH Server,

[4]https://doc.ilabt.imec.be/ilabt/index.html
[5]https://pcengines.ch/apu2c4.htm

PM, and DASH Client were inserted using the TC tool for Linux. The DASH server offered a catalog of 15 videos described in Table I, encoded in the quality levels described in Table II, using the H.264 codec, with no audio track and video segments of four seconds. The DASH client was based on the DASH Industry Forum reference player[6] version 4.0.0, modified to record playback statistics.

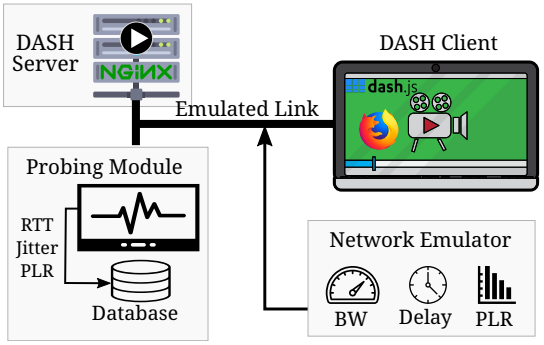

Fig. 4. Emulated setup for dataset creation, model training, and initial evaluation

TABLE I
SAMPLE VIDEOS

Video	Duration	Type
Another World (another)	00:03:11	Nature
Samsung: Around The World (aworld)	00:05:39	Documentary
Football Barcelona (barcelona)	00:03:14	Sports
Power of Curve (curve)	00:03:15	Promotional
Phantom Flex (flex)	00:03:07	Promotional
Garden (garden)	00:03:05	Promotional
Jimix Put Your Hands Up (jimix)	00:03:56	Music Video
Lumix (lumix)	00:03:07	Documentary
Slam Dunk (slam)	00:02:56	Sports
Surfing (surfing)	00:02:59	Sports
Lovely Swiss (swiss)	00:03:41	Documentary
Travel With My Pet (travel)	00:02:35	Documentary
TravelXP HDR/HLG (travelxp)	00:05:00	Documentary
Life Untouched (untouched)	00:03:18	Nature
7 Wonders Of The World (wonders)	00:03:51	Documentary

All videos obtained from http://4kmedia.org

With the emulated setup we executed the video sessions using Firefox web browser while recording network measurements and video playback quality metrics once per second using a custom script. At the end of this process, we had a dataset containing network QoS measurements and the video playback characteristics they generated. We labeled the dataset following the ITU-T P.1203 Recommendation, which provides an estimate of MOS for a video session encompassing multiple characteristics such as playback stalls, video quality switches, video resolution played, among others. For this, we relied on the software[7] provided by Robitza et al. [30] and Raake et al. [31]. For this work we used the *mode 3* MOS estimation on the P.1203 software, instead of the *mode 0* of our previous work, resulting in a more comprehensive estimation of the MOS of the video sessions. However, this may turn the function to be approximated by our inference model more complex, and therefore, the estimation error can be slightly higher than the observed in the previous work.

TABLE II
VIDEO REPRESENTATIONS (QUALITY LEVELS)

Representation	Resolution	Bitrate
0	320x180	200 kbps
1	320x180	400 kbps
2	480x270	600 kbps
3	640x360	800 kbps
4	640x360	1,000 kbps
5	768x432	1,500 kbps
6	1024x576	2,500 kbps
7	1280x720	4,000 kbps
8	1920x1080	8,000 kbps
9	3840x2160	12,000 kbps

The resulting dataset contained approximately 115,000 video sessions and over 23 million data points. We split the dataset into three parts, being 40 % for the training of the *Primary Model* (using the other 60 % for its evaluation), 40 % for the training of the *Secondary Model*, and the remaining 20 % to evaluate both models combined. We used the method of random search [32] to define the hyperparameters for each model. Those hyperparameters are configurations that cannot be inferred from the data during training and impact model accuracy and generalization capacity. The hyperparameters defined for each model, and details of the random search execution are presented in Section V-A.

C. Testbed Setup

We deployed the DASH Server container on a Virtual Wall node running the NGINX Server [33], and connected it to a CityLab node that operated as Wi-Fi AP using *hostapd*[8]. The server offered a subset of four videos from the emulated experiments: *another*, *barcelona*, *jimix*, and *travel*. We connected the DASH Server to the AP using a Generic Routing Encapsulation (GRE) tunnel over IPv6, which delivered excellent performance throughout the experiments. Our measurements showed a mean RTT of 3.01 ms, with 25 μs of jitter, and 0% PLR in all tests. The effective throughput measured using *iperf3*[9] was over 700 Mbps for UDP and TCP traffic.

We used the five setups described below for the experiments in CityLab. Node locations and further details can be found on CityLab documentation. For the outdoors deployments, we show the node pairs in Figure 5. We configured the AP on channel 6 (2,437 MHz) in all setups, using Atheros (ath10k) QCA9880 cards operating in IEEE 802.11g mode to evaluate the method with more challenging settings. Table III shows the link characteristics between the node pairs during each experiment. The values are estimates of the link quality

[6]https://github.com/Dash-Industry-Forum/dash.js
[7]https://github.com/itu-p1203/itu-p1203
[8]https://w1.fi/hostapd/
[9]https://github.com/esnet/iperf

between nodes provided by the CityLab testbed, and although they provide an indication of usual link quality, they may not reflect the exact condition during the experiments.

TABLE III
LINK QUALITY REPORTED BY CITYLAB

Setup	Node Pair	Avg. RSSI	Reliability
1	6 → 72	-52.55 dBm	99.00%
	72 → 6	-51.66 dBm	94.99%
2	71 → 6	-48.45 dBm	94.22%
	6 → 71	-51.40 dBm	97.61%
3	24 → 28	-73.85 dBm	98.40%
	28 → 24	-70.43 dBm	64.53%
4	14 → 18	-85.22 dBm	80.08%
	18 → 14	-80.48 dBm	71.54%
5	34 → 35	-57.54 dBm	98.20%
	35 → 34	-55.27 dBm	20.16%

We also analyzed the number of other networks in the range of each node and operating on the same channel. Traffic on those networks can cause interference and affect the quality of the video sessions in a similar way as

- **Setup 1:** nodes 6 (AP) and 72 (Client). Indoors deployment with nodes in close proximity, having 40 other APs in range, being 6 of them on the same channel. The same networks are detected by the AP and the client.
- **Setup 2:** nodes 71 (AP) and 6 (Client). Indoors deployment with nodes in close proximity. 37 other APs in range of the AP, 9 on the same channel. 41 other APs in range of the client, 6 on the same channel.
- **Setup 3:** nodes 24 (AP) and 28 (Client). Outdoors deployment with other 50 APs in range of the AP, 1 on the same channel. 20 other APs can be detected by the client, being 5 on the same channel.
- **Setup 4:** nodes 14 (AP) and 18 (Client). Outdoors deployment. 107 other APs in range of the AP, being 10 on the same channel. 39 other APs in range of the client, 4 on the same channel.
- **Setup 5:** nodes 34 (AP) and 35 (Client). Outdoors deployment. 37 APs in range of the AP, 7 on the same channel. 144 APs in range of the client, 27 on the same channel.

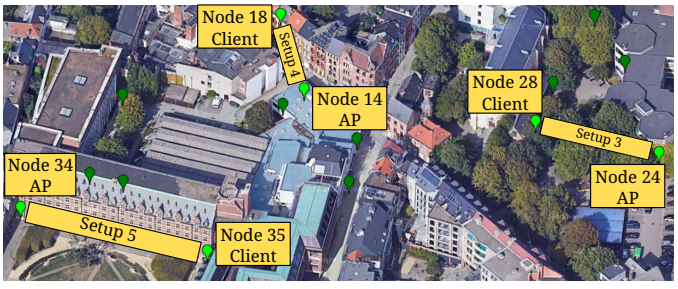

Fig. 5. Setups 3 to 5 using outdoors nodes of CityLab

V. RESULTS

This section first describes the results obtained with the inference model, highlighting the improvement achieved over the method described in our previous work. Then we present the results of the experiments using the testbed setups.

A. Model Training and Initial Evaluation

The first step of our model training process consists of hyperparameter tuning with 100 iterations of random search. At each iteration, a random set of hyperparameters was evaluated using 3-fold Cross-Validation (CV). The hyperparameter set that resulted in lower Root Mean Square Error (RMSE) on CV was selected for the final model training. Table IV describes the parameters that were evaluated and selected[10]. The maximum number of trees was 100 in all cases.

TABLE IV
HYPERPARAMETERS SELECTED USING RANDOM SEARCH

Hyperparameter	Evaluated	Selected	
		Primary	Secondary
colsample_bytree	uniform(0.1, 1)	0.94	0.90
colsample_bylevel	uniform(0.1, 1)	0.62	0.80
subsample	uniform(0.1, 1)	0.42	0.59
learning_rate	loguniform(0.005, 0.5)	0.16	0.22
alpha	uniform(1, 5)	2	3
max_depth	uniform(1, 5)	4	4

Table V shows the accuracy achieved by each model, for each video, and the accuracy over data of all videos combined. We evaluate model accuracy in terms of RMSE. We observe that the *Primary Model* used in this work already reduces the error marginally just by using a different set of input features. By adding the *Secondary Model* to the method we can further reduce the inference errors.

TABLE V
INFERENCE RMSE VALUES FOR EACH VIDEO

Video	Model in [13]	Primary Model	Secondary Model
another	1.08	1.07	1.06
aworld	1.09	1.08	1.03
barcelona	1.15	1.14	1.12
curve	1.11	1.09	1.09
flex	1.08	1.08	1.04
garden	1.11	1.10	1.09
jimix	1.10	1.09	1.08
lumix	1.16	1.14	1.12
slam	1.12	1.11	1.09
surfing	1.07	1.06	1.01
swiss	1.09	1.07	1.05
travel	1.08	1.07	1.04
travelxp	1.11	1.10	1.07
untouched	1.09	1.08	1.04
wonders	1.11	1.10	1.08
Combined	**1.09**	**1.08**	**1.05**

[10]Further detail about XGBoost hyperparameters can be found in https://sites.google.com/view/lauraepp/parameters

B. Testbed Results

For each setup described in Section IV-C we executed 20 repetitions for each video. Table VI presents the overall results obtained. The first column shows the mean MOS achieved across all sessions on each setup. The second column shows the mean of inferred values, and the third column shows the mean of RMSE values. The standard deviation for each metric is shown in parentheses. The results show that setups 1, 2, and 5 allowed video sessions with higher quality. Although the testbed information shows a highly asymmetric link (in terms of reliability) on setup 5, the MOS achieved during the sessions and the inference accuracy indicates that the link conditions were symmetric during our tests. On the other hand, setups 3 and 4 had worse network conditions.

TABLE VI
RESULTS OBTAINED ON EACH SETUP

Setup	Mean MOS	Mean Inferred	Mean RMSE
1	4.89 (0.26)	4.93 (0.10)	0.19 (0.22)
2	4.90 (0.27)	4.43 (0.56)	0.56 (0.55)
3	2.84 (0.41)	4.14 (0.67)	1.31 (0.61)
4	2.02 (0.23)	3.64 (0.57)	1.63 (0.57)
5	4.87 (0.29)	4.22 (0.45)	0.75 (0.46)

To further discuss the details of the inference method, Figures 6, 7, 8, and 9 show the measured and inferred values during some video sessions executed. The red lines show the inferred MOS, and the black lines show the measured MOS at the client's video player. Figure 6 shows a session of the "jimix" video on setup 1. The client constantly receives high MOS, and we can observe that the inferences are close to the measurements with slight oscillations. Figure 7 shows a session of the "travel" video on setup 2. In that case, the inferred values oscillate more, and in fact, for a period of 25 seconds at the beginning of the session there was a drop in MOS. This indicates that the network conditions could be improved in order to guarantee the highest possible QoE for the whole session.

Comparing these results with those in Table VI, we observe that the mean MOS on setup 1 was close to 5, similar to the inferred MOS values, resulting in an RMSE of 0.19. On setup 2, where slightly more oscillations were observed, the mean MOS was mostly high. However, the inferences presented higher error than on setup 1, with an RMSE of 0.56. From the values observed in Table VI, considering the means and standard deviations, show that the method offers measurements in line with the expected accuracy from Table V in most cases.

Sessions on setups 3 and 4 had MOS between 2 and 3. On setup 3 the sessions had more oscillations, as shown in the example of Figure 8 (a session of the "another" video on setup 3), and also on the standard deviation of mean inferred MOS values. We observe that during some periods on Figure 8 the inferred MOS values present higher errors. Nevertheless, for most of the session duration, the error is within the expected RMSE, and the oscillation level of inferences can also be used as an indicator of sub-optimal user experience.

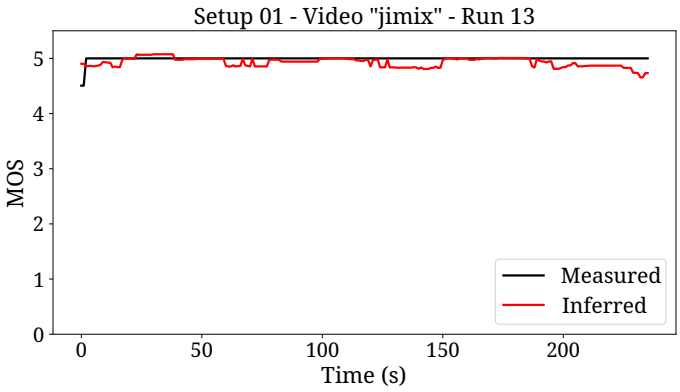

Fig. 6. Sample session on setup 1 and video "jimix".

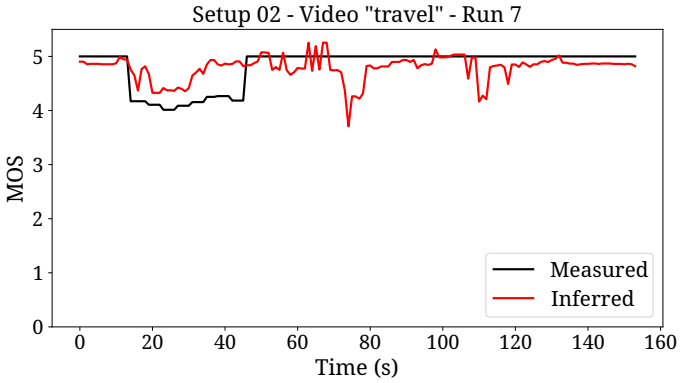

Fig. 7. Sample session on setup 2 and video "travel".

On setup 4 the mean MOS is stable across all experiments but at lower values (2.02 with 0.23 of standard deviation). We also observed the highest mean RMSE values, being the worst performance obtained with our method, as shown in the session of Figure 9. Similar to the previous case, the high oscillation of the inferred values can also indicate QoE issues during the video session. As we can see from Table VI the inferences on setups 3 and 4 had the highest RMSE. Both cases illustrate situations in which the method presented the lowest accuracy. Nevertheless, a combined analysis of the mean inferred values with the standard deviation of the inferences can indicate that the session is ongoing with sub-optimal MOS.

The inferred values shown in Figures 7, 8, and 9 show higher oscillations than the measured values along time. This is due to the fact that the measured value only changes when the DASH adaptation algorithm switches to a different quality level (or a stall occurs), while the inferred value is updated every second, based on the most recent network measurements. This makes the inferred values more sensitive to network quality fluctuations.

The results from the model training phase, described in Table V, show how the model accuracy could be improved by selecting more relevant features and adding a second-layer model. With the emulated setup, even though we used virtual connections between containers with network impairments

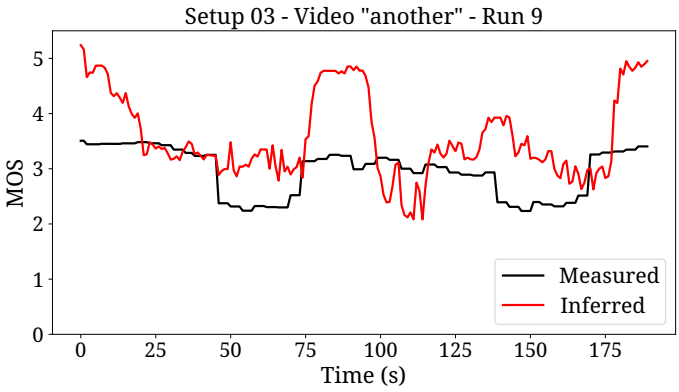

Fig. 8. One video session of the ”another” video on setup 3.

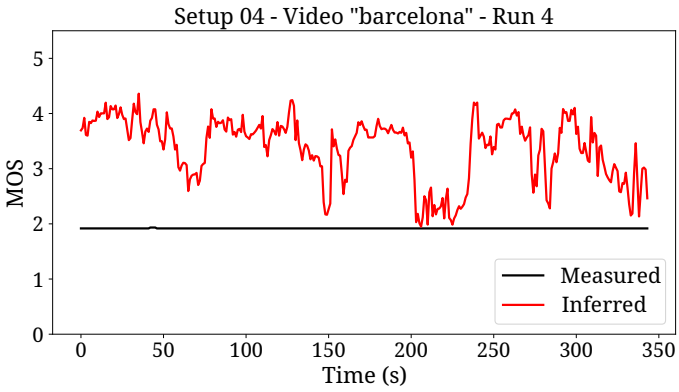

Fig. 9. One video session of the ”barcelona” video on setup 4.

generated by the TC tool, we were able to build a dataset and train models that were applicable in real wireless deployments. Based on this observation, further improvements can be made on the method by increasing the complexity of the network impairments inserted with TC (e.g. adding packet corruption probability, packet duplication, reordering), or adding more inference layers encompassing specific features of the technology used (e.g. Ethernet, WiFi, 4G, 5G).

VI. Conclusions

In this work, we propose and experimentally evaluate a method for inference of QoE for DASH video based on ICMP probing and Machine Learning. We improve our previous method by adding a secondary model that reduces the inference error. We validate our results using the Virtual Wall and CityLab testbeds, with five different wireless deployments. Results show that the model is effective in identifying situations in which the user receives sub-optimal QoE, and therefore, that the provider should take action. Such a system can be employed, for example, for automated network management based on QoE. On setups with more challenging network conditions we observed higher inference errors, however, the average MOS inferences and the high level of oscillations showed by the standard deviation indicate a degraded QoE situation. In future work, we aim to apply our method to network management tools and optimize networks based on inferred QoE.

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 projects Fed4FIRE+ (Grant Agreement No. 723638), 5G-Blueprint (Grant Agreement No. 952189), and by the FLEXNET project: “Flexible IoT Networks for Value Creators” (Celtic 2016/3), in the Eureka Celtic-Next Cluster.

References

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

[2] M. Yang, S. Wang, R. N. Calheiros, and F. Yang, “Survey on QoE assessment approach for network service,” *IEEE Access*, vol. 6, pp. 48 374–48 390, 2018. [Online]. Available: doi.org/10.1109/ACCESS.2018.2867253

[3] M. Fiedler, T. Hossfeld, and P. Tran-Gia, “A generic quantitative relationship between quality of experience and quality of service,” *IEEE Network*, vol. 24, no. 2, pp. 36–41, 2010. [Online]. Available: doi.org/10.1109/MNET.2010.5430142

[4] I. T. U. ITU, “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: www.itu.int/rec/T-REC-P.10

[5] P. Juluri, V. Tamarapalli, and D. Medhi, “Measurement of quality of experience of video-on-demand services: A survey,” *IEEE Communications Surveys and Tutorials*, vol. 18, no. 1, pp. 401–418, 2016. [Online]. Available: doi.org/10.1109/COMST.2015.2401424

[6] Y. Chen, K. Wu, and Q. Zhang, “From qos to qoe: A tutorial on video quality assessment,” *IEEE Communications Surveys Tutorials*, vol. 17, no. 2, pp. 1126–1165, 2015. [Online]. Available: doi.org/10.1109/COMST.2014.2363139

[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: doi.org/10.1109/TNSM.2016.2537645

[8] I. T. U. ITU, “ITU-T Rec P.1203: Parametric bitstream-based quality assessment of progressive download and adaptive audiovisual streaming services over reliable transport,” 2017. [Online]. Available: 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: doi.org/10.1109/ACCESS.2019.2901778

[10] A. Khan, L. Sun, E. Ifeachor, J. O. Fajardo, F. Liberal, and H. Koumaras, “Video quality prediction models based on video content dynamics for H.264 video over UMTS networks,” *International Journal of Digital Multimedia Broadcasting*, 2010. [Online]. Available: doi.org/10.1155/2010/608138

[11] L. Qian, H. Chen, and L. Xie, “SVM-based QoE estimation model for video streaming service over wireless networks,” *International Conference on Wireless Communications and Signal Processing, (WCSP)*, pp. 1–6, 2015. [Online]. Available: doi.org/10.1109/WCSP.2015.7341066

[12] I. Paudel, J. Pokhrel, B. Wehbi, A. Cavalli, and B. Jouaber, "Estimation of video QoE from MAC parameters in wireless network: A Random Neural Network approach," *International Symposium on Communications and Information Technologies, (ISCIT)*, pp. 51–55, 2014. [Online]. Available: doi.org/10.1109/ISCIT.2014.7011868

[13] 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: doi.org/10.23919/CNSM50824.2020.9269120

[14] 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)*. IEEE, 2018, pp. 529–534. [Online]. Available: doi.org/10.1109/INFCOMW.2018.8407018

[15] 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: doi.org/10.1109/IWQoS.2012.6245987

[16] 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: doi.org/10.1109/ICIN.2019.8685901

[17] 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: doi.org/10.23919/TMA.2019.8784589

[18] 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: doi.org/10.1109/COMST.2014.2360940

[19] 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: doi.org/10.23919/IFIPNetworking.2019.8816854

[20] R. Ul Mustafa, D. Moura, and C. E. Rothenberg, "Machine learning approach to estimate video qoe of encrypted dash traffic in 5g networks," in *2021 IEEE Statistical Signal Processing Workshop (SSP)*, 2021, pp. 586–589. [Online]. Available: doi.org/10.1109/SSP49050.2021.9513804

[21] 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: doi.org/10.1109/GLOCOM.2016.7842022

[22] 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: doi.org/10.1145/2939672.2939785

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

[24] J. Mambretti, J. Chen, and F. Yeh, "Next Generation Clouds, the Chameleon Cloud Testbed, and Software Defined Networking (SDN)," in *2015 International Conference on Cloud Computing Research and Innovation (ICCCRI)*, 2015, pp. 73–79. [Online]. Available: doi.org/10.1109/ICCCRI.2015.10

[25] C. Adjih, E. Baccelli, E. Fleury, G. Harter, N. Mitton, T. Noel, R. Pissard-Gibollet, F. Saint-Marcel, G. Schreiner, J. Vandaele, and T. Watteyne, "FIT IoT-LAB: A large scale open experimental IoT testbed," in *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, 2015, pp. 459–464. [Online]. Available: doi.org/10.1109/WF-IoT.2015.7389098

[26] J. Marquez-Barja, B. Lannoo, D. Naudts, B. Braem, C. Donato, V. Maglogiannis, S. Mercelis, R. Berkvens, P. Hellinckx, M. Weyn *et al.*, "Smart Highway: ITS-G5 and C2VX based testbed for vehicular communications in real environments enhanced by edge/cloud technologies," in *EuCNC2019, the European Conference on Networks and Communications*. IEEE, 2019. [Online]. Available: biblio.ugent.be/publication/8642435/file/8656511

[27] 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: doi.org/10.1109/INFCOMW.2018.8407018

[28] S. Latre, P. Leroux, T. Coenen, B. Braem, P. Ballon, and P. Demeester, "City of things: An integrated and multi-technology testbed for iot smart city experiments," in *2016 IEEE International Smart Cities Conference (ISC2)*, 2016, pp. 1–8. [Online]. Available: doi.org/10.1109/ISC2.2016.7580875

[29] L. Sanchez, J. A. Galache, V. Gutierrez, J. M. Hernandez, J. Bernat, A. Gluhak, and T. Garcia, "SmartSantander: The meeting point between Future Internet research and experimentation and the smart cities," in *2011 Future Network Mobile Summit*, 2011, pp. 1–8. [Online]. Available: ieeexplore.ieee.org/abstract/document/6095264

[30] 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: doi.org/10.1145/3204949.3208124

[31] 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: doi.org/10.1109/QoMEX.2017.7965631

[32] 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: doi.org/10.5555/2188385.2188395

[33] W. Reese, "Nginx: The High-Performance Web Server and Reverse Proxy," *Linux J.*, vol. 2008, no. 173, Sep. 2008. [Online]. Available: doi.org/10.5555/1412202.1412204