Are you on Mobile or Desktop? On the Impact of End-User Device on Web QoE Inference from Encrypted Traffic

Sarah Wassermann*, Pedro Casas*, Zied Ben Houidi†, Alexis Huet†, Michael Seufert‡
Nikolas Wehner‡, Joshua Schüler‡, Shengming Cai†, Hao Shi†, Jinchun Xu†, Tobias Hossfeld‡, Dario Rossi†
*AIT Austrian Institute of Technology, †Huawei Technologies Co. Ltd, ‡University of Würzburg

***Abstract*—Web browsing is one of the key applications of the Internet, if not the most important. Due to the proliferation of web-based services, Web Quality of Experience (QoE) monitoring has gained strong momentum for Internet Service Providers (ISPs). In this paper, we address the problem of Web QoE monitoring from the ISP perspective, relying on in-network, passive measurements. As a proxy to Web QoE, we focus on the analysis of the well-known SpeedIndex (SI) metric. Given the wide adoption of end-to-end encryption, we resort to machine-learning models to infer the SI of individual web page loading sessions, using as input only packet level data. Our study particularly targets the impact of different end-user device types (e.g., smartphone, desktop, tablet) on the performance of such inference models. Empirical evaluations on a large, multi-device, heterogeneous corpus of Web QoE measurements for top popular websites demonstrate that (i) the proposed solution can infer the SI as well as estimate QoE ranges from in-network traffic measurements with high accuracy, and (ii) the device type introduces a strong bias in the feasibility of Web QoE inference models, causing models trained for single device types to badly generalize to other devices. As a consequence, models for Web QoE monitoring in real network deployments must be trained on multi-device measurements to achieve proper inference performance, putting into question the applicability of previously conceived approaches for encrypted Web QoE monitoring.**

***Index Terms*—Web QoE; Smartphone vs. Desktop; Network Monitoring; Machine Learning; SpeedIndex; Encrypted Traffic.**

I. Introduction

The Web is one of the most relevant components of the Internet, both in terms of popularity as well as ubiquity. The performance of the Web is highly relevant for the success of every on-line service, as it severely impacts the engagement and churn of users. The overall performance of a web service as perceived by the end user can be measured by the corresponding Web Quality of Experience (QoE). From a practical perspective, reliably measuring Web QoE is challenging.

Different from other services, such as video streaming or gaming, web browsing is a composite of multiple multimedia components and embedded services; loading a single web page today requires tens of flows to get the various page resources located in diverse servers from different content providers. In this complex process, the network plays a key role impacting users' Web QoE, forcing Internet Service Providers (ISPs) to deploy effective means to monitor Web QoE as perceived by their customers.

The literature on web performance analysis is rife with objective metrics capturing the performance of web pages, including metrics such as Page Load Time (PLT), SpeedIndex (SI), Above the Fold Time (AFT), etc. However, all these metrics require access to the application layer, which is hidden from the eyes of the ISP by the wide deployment of end-to-end network traffic encryption.

The analysis of Web QoE from purely in-network, encrypted traffic measurements is yet an under-explored problem; in fact, we have been the first ones recently addressing it [1], [2], for the specific scenario of desktop web browsing. By using page load controlled experiments, where network data is simultaneously collected with ground-truth Web QoE metrics such as SI, AFT, etc., we have shown the potential of using supervised Machine Learning (ML) to infer these metrics from features computed on the encrypted stream of packets.

In this paper, we follow a similar approach to [1], [2], extending the analysis in multiple new directions. Firstly and most important, we consider Web QoE not only for desktop devices, but include smartphone and tablet Web QoE. The lion's share of Internet-access devices today is smartphones, with nearly three quarters of the world population using just their smartphones to access the Internet by 2025 [3]. As we find in our results, a model trained only on desktop browsing data provides poor Web QoE estimation performance when applied to smartphone and tablet measurements. We refer to this problem as cross-device training and validation issues. Secondly, besides training regression models to estimate Web QoE objective metrics – in particular SI, we rely on real end-user data from previous studies [4] to build Web QoE classification models for subjective metrics – e.g., MOS scores. Third, we present an extensive benchmark comparing the performance of different ML models for both estimation tasks – regression for SI inference, and classification for QoE estimation. Last, we compare the performance of a single, multi-device Web QoE inference model against individual, per device type models, and show that the simplest, single-model approach provides similar results to the per-device specialization scenario, further supporting the practical application of the proposed multi-device approach for passive Web QoE monitoring.

The remainder of the paper is organized as follows. Sec. II overviews the related work on Web QoE monitoring and analysis. Sec. III presents the overall modeling and data generation approach, including a characterization of the produced datasets for this study. In Sec. IV, we introduce and evaluate the proposed ML models for Web QoE inference and QoE classification; starting from a single device type – desktop, we additionally present evidence of the aforementioned cross-device training and validation issues. Sec. V extends the models to the multi-device scenario, training individual, per-device models to highly improve inference performance. In Sec. VI, we show that a single model trained on all devices data can still offer good performance for both inference and classification tasks, laying the basis for a generalizable, multi-device, Web QoE passive monitoring system. Finally, Sec. VII concludes this paper.

II. Related Work

Initial Web QoE models were based on plain Page Load Times (PLT) [5], [6], and are still broadly used in practice to infer user satisfaction in web-browsing, e.g, under ITU-T [7]. However, research in the field demonstrated that PLT is not the most accurate proxy to user perception of web page loading times. Indeed, the actual web content visible to the user is usually displayed much earlier, as most web pages often stretch beyond the browser's viewport. Additional in-browser metrics have been accordingly devised to better suit the page display on the screen. An approach is the so-called Above the Fold Time (AFT), i.e., the time until the visible portion of a web page has been fully loaded, which has been also tested within traditional Web QoE models [8]. Newer Web QoE metrics have been proposed recently, such as the SI, which takes into account the whole visual progress of the page loading, by processing a video capture of the screen. Besides single metric modeling, ML-based approaches have been presented [4], [9] to model Web QoE from a combination of metrics.

Another direction in the literature proposes to understand how external components influence Web QoE. Prior work [10], [11] has studied the impact of network quality fluctuations and outages on user Web QoE. But besides network quality, other components influence Web QoE. They are linked to the specific web page content – usability [12], aesthetics [13], etc., as well as device type – desktop, smartphone, tablets [14]. Important to our study, these papers show that smartphones and tablets have their own characteristics, not only regarding screen sizes, but also in terms of content rendering and web designs. Most of these papers are based on the analysis of Web QoE in controlled, small-scale lab environments.

Others directly rely on in-browser metrics as a proxy to infer Web QoE, conducting large-scale active measurement campaigns. For example, the impact of multiple features such as transport protocols, network connections, visible portion, etc., on PLT and AFT is studied in [15], based on a set of 244 million measurements collected during 6 months for the top-10000 Alexa websites. Other papers also measured the impact of similar features on PLT and SI or AFT in different countries and different types of networks [16], including mobile ones [17].

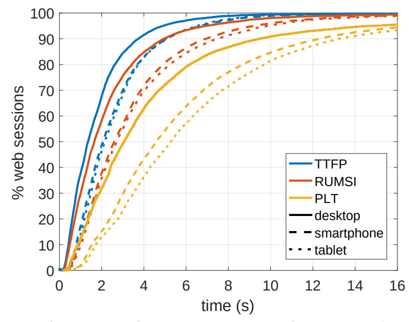
While useful, most of prior work stayed at the application-level. This is problematic for ISPs, which have no direct access to in-browser metrics, but only to network traffic. In recent years, TLS encryption has even narrowed the information that ISPs can collect from the network side, and previous approaches [18] based on DPI and HTTP traffic analysis are no longer applicable. Other papers [19], [20] developed *correlated approximations* to the SI metric, such as Byte/Object-Index [19] and Pain-Index [20], which can be computed from packet- and flow-level measurements, with no need to access traffic payload, thus seamlessly operating with encrypted traffic. In recent work [1], [2], we took a step further to directly infer the SI metric, using ML techniques mapping network (encrypted) traffic features to SI.

When it comes to the specific case of Web QoE monitoring in mobile devices, there have been multiple papers using ML [21]–[24] or simple modeling approaches [25] to map application layer metrics [23], [24] or network QoS metrics [21], [22], [25] into QoE-related metrics. From these, two papers [21], [22] are the closest to our work, but either propose analysis approaches which are no longer applicable due to HTTP traffic encryption [22], or do not address the specific problem of web browsing [21].

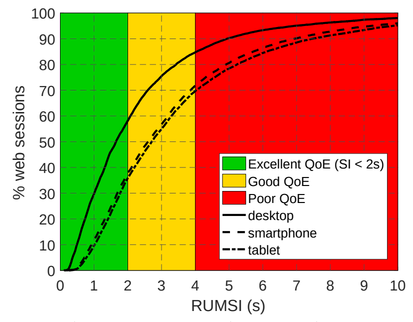
In this paper, we deal with an unexplored and so far neglected issue: understanding the impact of the end-user device type on the analysis of Web QoE from in-network traffic measurements. To the best of our knowledge, this is the first paper addressing this issue.

III. Web QoE Datasets & Modeling Approach

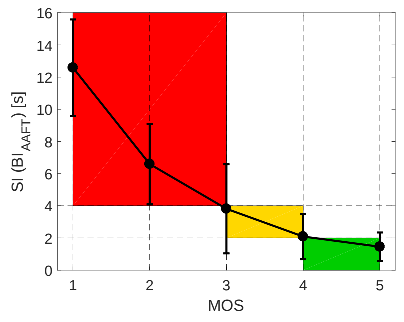
The proposed solution to the Web QoE monitoring problem consists of training supervised ML models to map network traffic features, extracted from the encrypted network web page traffic, into relevant Web QoE metrics. The approach is data-driven, and thus needs datasets containing both the collected traffic traces – the *input*, and the targeted Web QoE metric – the *ground truth*. To fully control the generation of such datasets, we built a measurement testbed based on multiple private instances of WepPageTest (WPT) (https://www.webpagetest.org/), a well-known open-source web performance analysis tool. WPT is today the default tool used for web page performance testing, both in the industry and the academia. Different from previous studies [1], [2], [15]–[17], [19], which have studied Web QoE for exclusively desktop browsers and desktop devices (or in some exceptional cases, emulating mobile devices), our measurement testbed consists of three different, non-emulated types of devices, including smartphones, tablets, and desktop, using WPT agents for Android and Linux. Chrome (last stable version) is used as browser. Instead of leveraging in-device WPT traffic shaping capabilities, devices are connected to the open Internet through independent network emulators, which allows for more realistic network access performance configurations in terms of bandwidth, latency, packet loss rate, etc. This allows for heterogeneity in the generated measurements. Configurations



(a) Time performance metrics per device.



(b) RUMSI and QoE classes.



(c) SI vs. MOS scores.

Fig. 1: Distribution of (a) TTFP, RUMSI, and PLT values, (b) QoE classes per device type, based on (c) real-user MOS scores.

used in the study include access downlink bandwidth up to 10 Mbps, packet loss rates up to 10%, and RTTs up to 100 ms.

Using WPT measurements, the platform extracts about 90 different KPIs and Web QoE metrics – such as PLT, SI, AFT, Byte Index [19], Time to Interactive (TTI), etc., as well as content characteristics of the visited pages. Network traffic is captured at an intermediate proxy and stored as .pcap traces, from where model input features are extracted. For this study, we generate a fully balanced dataset – 1/3 desktop, 1/3 smartphone, 1/3 tablet, of more than 50.000 web page *loading sessions* (i.e., the loading of a single page), targeting the top 500 websites according to Alexa top sites list (https://www.alexa.com/topsites). The same pages are visited multiple times for each device type, using the same access network setups. We do not consider the effect of caching. All tests correspond as such to a first-view loading session.

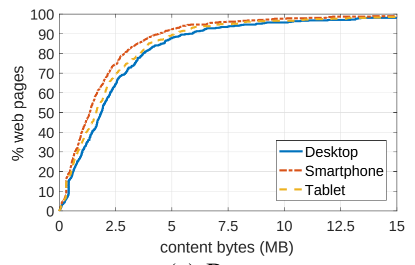
Without loss of generality, we focus on the inference of one particular Web QoE metric, the SI, which is today one of the most accepted metrics reflecting Web QoE. Nevertheless, the methodology applies to any other similar Web QoE metric. As shown in [19], measuring the SI is cumbersome in terms of computational resources and might introduce bias in the data capturing/processing, mainly due to the video capturing and analysis. This is particularly critical in smartphones and tablets, which are generally resource-constrained; therefore, instead of focusing on the SI metric, we collect the so-called RUM SpeedIndex (RUMSI) metric [26], which is a passive approximation to the SI, computed from the analysis of web page resource timings. Finally, we assume that the monitoring system takes as input network traffic from a single web session. In the case of concurrent web sessions, a classification methodology from the literature [20] could be applied to disentangle them. Indeed, while the Web traffic identification step is out of the scope of this paper, we have multiple techniques addressing this step.

A. Data Characterization

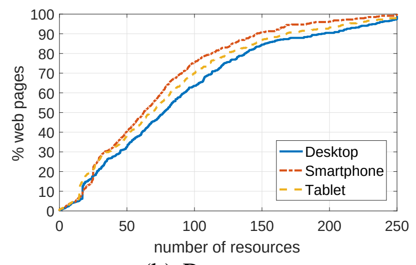
The list of top 500 Alexa web pages is assorted in terms of contents, and as we show next, the type of device being used has a visible impact on web page characteristics and timing performance. Fig. 1(a) depicts the distribution of three relevant Web QoE metrics, per device type. These include: the Time to First Paint (TTFP), which accounts for the time at which the first object is painted on the browser, the RUMSI, and the PLT. Note how the values are significantly higher for both smartphone and tablet devices as compared to desktop devices, pointing to a more complex rendering process in mobile devices. This is most probably linked to the specific hardware limitations of smartphone and tablet devices, as well as the particular characteristics of the OS and browser combination – native Chrome in Android. In addition, the way pages are optimized (most times dynamically) and rendered in mobile devices impacts the loading times. Worse loading performance in mobile devices is a commonly known issue in practice – e.g., see page speed stats at https://backlinko.com/page-speed-stats. Interestingly, when comparing Android devices, TTFP values are almost identical for smartphone and tablet, RUMSI is slightly higher for tablet, while PLT is significantly higher for tablet. As we see next, this is most probably explained by the fact that webpages have more content to load in tablets. It is also interesting to note how PLT significantly overestimates the perceived loading time of web pages, represented by the (RUM)SI metric.

Fig. 2 characterizes the 500 web pages per device type, in terms of (a) page size, (b) number of resources, (c) number of root domains, number of (d) image and (e) java-script resources, (f) share of static content served by CDNs, (g) number of DNS requests and (h) established connections to retrieve these pages, and (i) RUMSI to PLT ratio. The latter reflects the complexity of the web page in terms of visible content (SI) and full content downloading (PLT).

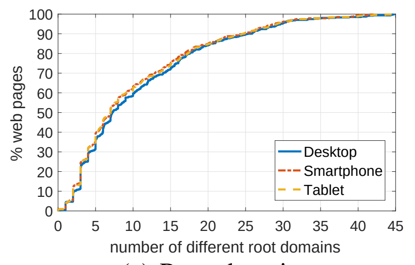
As expected, web pages browsed in desktop devices are bigger than those browsed in smartphone or tablet devices, which are optimized for smaller screen sizes. The average page size is 2.7MB in desktop, 2.4MB in tablet, and 2.1MB in smartphone. Figs. 2(b) and 2(c) further illustrate the richness and complexity of the web pages in terms of number of embedded contents and their location at different root domains, with more than 30% to 35% of the web pages consisting of more than 100 resources, and about 40% of the web pages fetching resources from more than 10 different root domains. The screen size probably plays a key role in terms of page characteristics, as the number of resources is higher for desktop, followed by tablet, and finally by smartphone.



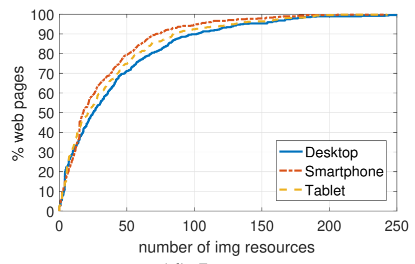
(a) Bytes.



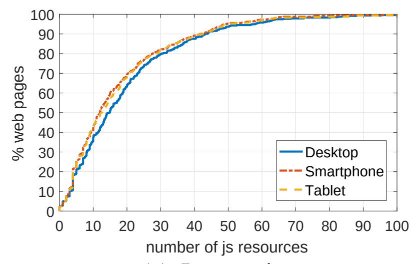
(b) Resources.



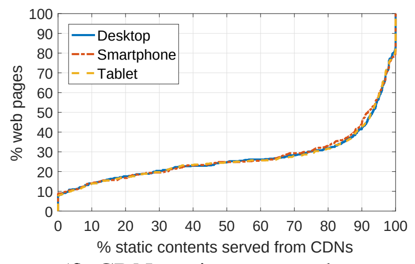
(c) Root domains.



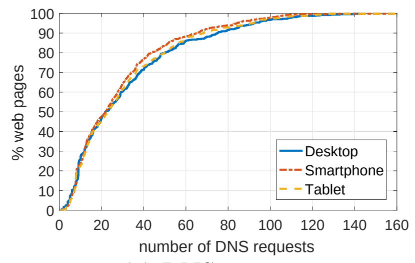
(d) Images.



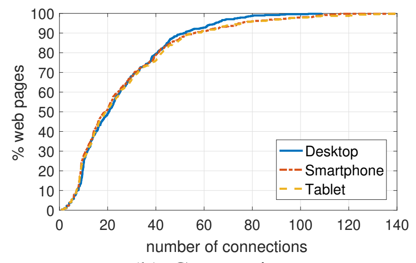
(e) Java-script.



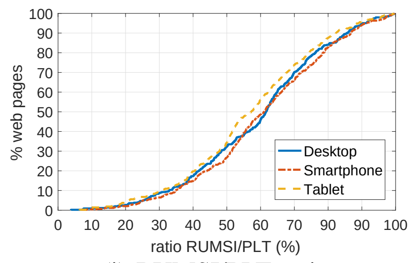
(f) CDN static content share.



(g) DNS requests.

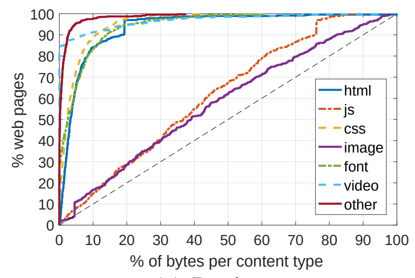


(h) Connections.

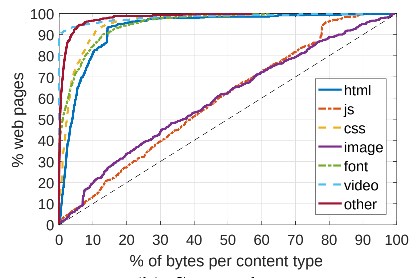


(i) RUMSI/PLT ratio.

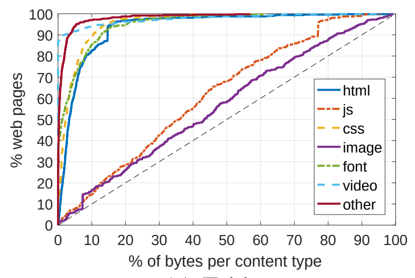
Fig. 2: Web pages characterization, per device type, including page sizes, number of resources, number of root domains, share of static content served by CDNs, number of DNS requests and established connections to retrieve these pages, and SI/PLT ratio. The latter reflects the complexity of the web page in terms of visible content (SI) and full content downloading (PLT).



(a) Desktop.



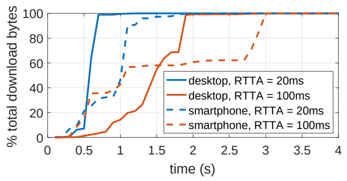
(b) Smartphone.



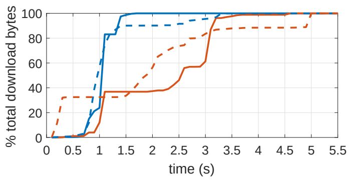
(c) Tablet.

Fig. 3: Shares of web page contents (bytes), per device type, split by html, JS, images, CSS, fonts, video, and others.

This is specially noticeable in terms of (d) images. CDNs are used by default to host static contents, with about 70% of the web pages hosting more than 70% of the static contents in CDNs, and about 20% hosting all static contents in CDNs. The number of DNS requests observed for a single page loading session can be as high as 160, and for more than 50% of the web pages, it takes more than 20 DNS requests to fetch the content. Note that desktop pages generate slightly higher number of requests, linked to the higher number of resources to fetch. In terms of connections, both smartphone and tablet establish a slightly higher number of connections, and more than 30% of the pages involved more than 30 connections to download the full page. Finally, the RUMSI/PLT ratio shows not only how large is the overestimation introduced by PLT in terms of perceived page load times, but also how different this is for the different web pages. Indeed, about 10% of the pages have a ratio below 0.3 – the visible content loads way faster than the full content, and 30% of the pages have a ratio above 0.7 – the visible content corresponds to the full web page content. In terms of web page content, Fig. 3 presents the shares of bytes per content type and device. Content is split in *html*, *java-script*, *CSS*, *image*, *font*, *video*, and *other* content



(a) `https://zoom.us`



(b) `https://cnn.com`

Fig. 4: Examples of CBD features or loading curves, using ΔT = 100 ms, and different Access RTT (RTTA) setups.

types. Images and java-script contents make the majority of the bytes, with 50% of the pages having a share of either images or java-script above 40%. Video contents are limited, only present in about 10% to 15% of the pages, with a slightly higher presence in Desktop. Overall, desktop and tablet pages have a higher share of image content, whereas smartphone pages have more java-script.

Conclusion: while most of the pages are very similar for every device type in terms of size, share of content types, and retrieved contents from external servers, differences can be significant for a small share of the pages. In terms of performance, loading times in Android devices (smartphone and tablet) are significantly higher than for desktop, a common trend observed in the practice of web page speed analysis.

B. Subjective QoE Analysis

While the SI is a good objective metric reasonably capturing the Web QoE of real users [8], we resort to previous subjective Web QoE studies to better understand the expected QoE for the generated dataset. In particular, prior work [4] conducted a subjective study where about 240 participants rated their browsing experience – loading of individual pages using desktop browser, according to a 5-level Absolute Category Rating (ACR) MOS score (1 - *bad* to 5 - *excellent* QoE). We rely on their publicly available dataset to identify QoE-related timing thresholds which could translate the RUMSI in our dataset to broad QoE classes. In Fig. 1(c) we depict the relationship between SI and MOS scores obtained in that study. While the SI metric was not directly measured, additional metrics such as the Byte Index where computed, which can be used as a good proxy to the real SI [19]. Based on these subjective QoE results, we define 3 Web QoE classes: **(e)**xcellent – MOS $\geq$ 4, **(g)**ood – 3 $\leq$ MOS $<$ 4, and **(p)**oor – MOS $<$ 3, resulting in SI thresholds of 2 and 4 seconds. Interestingly, the SI thresholds recommended in the industry as target for excellent Web performance vary between 1 second (desktop) and 3 seconds (mobile), which are in line with the proposed higher QoE class threshold of 2 seconds. Important to note is that our thresholds are derived for the case of browsing on desktop devices, and one would expect higher thresholds for Web QoE in mobile devices. Nevertheless, for this study, we assume the same thresholds apply to the three device types alike. Using these thresholds, Fig. 1(b) shows that about 60%/40% of the loading sessions correspond to excellent QoE in desktop/mobile (smartphone and tablet) respectively,

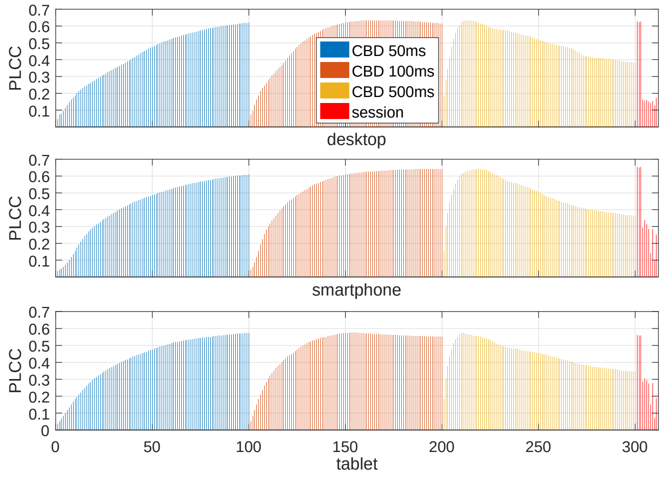


Fig. 5: Correlation between input features and RUMSI.

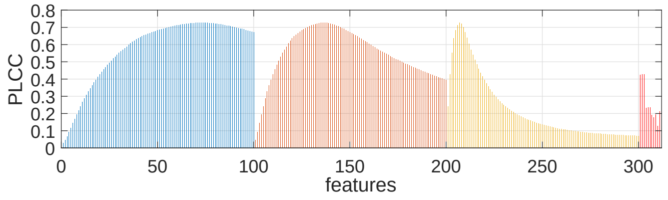


Fig. 6: Correlation between input features and QoE class.

25%/30% to good QoE, and the remaining 15%/30% result in poor QoE.

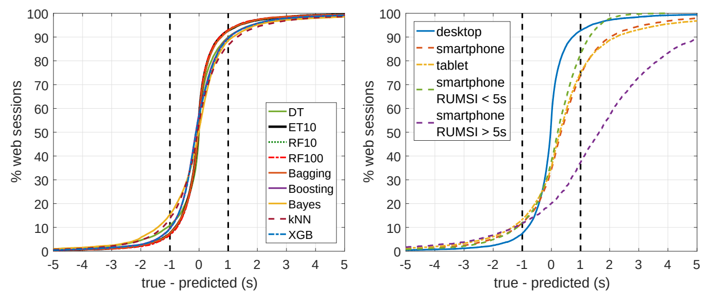
C. Targets and Input Features

We realize the Web QoE monitoring solution through two different prediction tasks: (i) inference of the RUMSI metric, which corresponds to a regression task, and (ii) prediction of the Web QoE class {e,g,p}, which corresponds to a 3-classes classification task. We use exactly the same input features in both tasks, derived from the stream of encrypted packets. To define input features, we follow the rationale behind the computation of the SI metric itself, which considers the whole progress of the page loading. We define the Cumulative Bytes Downloaded features $CBD(i)_{\Delta T}$, as the (normalized) cumulative number of bytes downloaded from the first collected byte at time t_0 up to time $t = t_0 + i \times \Delta T$, with $i = 1, \ldots, m$. The CBD features track the download progress of the page bytes, using a time resolution ΔT. Fig. 4 depicts examples of CBD features for different network configurations, both for desktop and smartphone devices, using $m = 100$ and $\Delta T = 100$ ms. Pages loading faster have a CBD *loading curve* rising sharper and arriving to full loading earlier.

For this study, we take $m = 100$ samples, and 3 different resolutions for the features computation, using ΔT = 50 ms, 100 ms, and 500 ms, for a total of 300 CBD features. Using different resolutions helps capturing different phenomena in the downloading progress, which potentially affect the SI, as well as allowing to track different page load durations, in this case up to 5, 10, and 50 seconds, respectively. We consider $n = 11$ additional input features, related to the

model	MAE-mAE (ms)	MRE-mRE (%)	PLCC
DT	766 – 288	37 – 18	0.817
ET10	598 – 260	31 – 16	0.879
RF10	602 – 262	31 – 16	0.860
RF100	564 – 249	30 – 15	0.885
Bagging	600 – 266	31 – 16	0.857
Boosting	767 – 426	48 – 26	0.861
Bayes	976 – 491	64 – 29	0.727
kNN	940 – 496	54 – 29	0.752
XGB	774 – 429	48 – 26	0.849

TABLE I: Benchmarking of different ML models for RUMSI inference, for desktop.



(a) Desktop. (b) Cross-device testing.

Fig. 7: RUMSI inference performance. Models are trained using exclusively desktop measurements.

complete page loading session; these include: full session duration (first to last packet), downlink/uplink session duration (first to last packet in downlink/uplink direction), total packets downlink/uplink/full, total bytes downlink/uplink/full, and session mean throughput downlink/uplink.

Fig. 5 depicts the linear correlation between these input features and the RUMSI metric, for different device types. Correlation values are rather high for all devices, with stronger correlations observed for CBD features between 5 seconds and 10 seconds, as well as for session-duration features. Fig. 6 shows correlation values for the QoE classification problem, for all devices together; as expected, based on the considered time-thresholds, higher correlations are observed between 2 and 5 seconds.

IV. Desktop Models' Lack of Generalization

We now focus now on the Web QoE estimation tasks. As a reference for performance evaluation, we start by training and evaluating different ML models for the specific case of desktop measurements. Recall that desktop measurements represent the most common data source so far used in the Web QoE literature [1], [2], [15]–[17], [19]. We then apply the trained models to both smartphone and tablet data, to evidence the cross-device lack of generalization issue. As a general note on the evaluations in this paper, all performance results correspond to 5-fold cross validation.

model	desktop						
	ACC	R{e}	R{g}	R{p}	P{e}	P{g}	P{p}
DT	80.3	88.8	66.1	72.8	88.4	66.2	73.8
ET10	84.4	93.1	70.5	75.6	89.6	73.7	81.7
RF10	84.6	93.1	71.6	74.6	90.1	73.1	82.1
RF100	86.9	93.3	77.4	79.1	92.3	76.2	84.7
Bagging	85.7	93.2	74.3	76.8	90.8	74.6	84.8
Boosting	82.9	91.1	70.3	73.7	90.1	69.1	79.3
Bayes	60.1	93.0	11.6	19.5	63.2	38.3	46.7
kNN	74.9	87.3	55.1	62.1	81.8	59.1	72.0
XGB	82.2	90.9	69.0	72.1	89.8	67.9	77.6

TABLE II: Benchmarking of different ML models for Web QoE prediction, in desktop. The three levels of QoE correspond to **excellent**{e}, **good**{g}, and **poor**{p} QoE.

A. *RUMSI Inference in Desktop*

Tab. I reports the RUMSI inference performance achieved by 9 different ML models, most of them based on decision trees. The tested models include single decision tree (DT), multiple types of ensembles using different numbers of trees, such as randomized trees (ET), random forest (RF), bagging trees, and boosting - including XGB optimizations. The listed is completed by a plain bayesian approach, and by the standard k nearest neighbors (kNN). We assess performance using three standard performance metrics for regression problems, including the absolute error (AE), the relative error (RE), and the linear correlation (PLCC). We take both mean (M) and median (m) values for the error metrics, to filter out significantly large errors. Fig. 7(a) additionally depicts the distribution of the prediction errors.

RF100 attains the best inference performance, with a median absolute error of 249 ms, and a median relative error of 15%. Absolute prediction errors are below 500 ms for more than 70% of the sessions, and more than 85% of the session RUMSI values are inferred with an error below 1 second. Similar performance is realized by smaller ensembles - e.g., RF10, ET10, and bagging, using 10 instead of 100 trees. Given the training speed improvements attained by the ET10 model, we take it as the underlying prediction model in subsequent evaluations.

B. *QoE Classification in Desktop*

Tab. II reports the classification benchmark results obtained for desktop. Again, RF100 provides the best results, with an overall accuracy (ACC) close to 87%. Recall (R) and precision (P) are above 90% for the excellent QoE class prediction, but good and poor performance classes tend to be confused by the predictor, and recall/precision drop close to 80% for these QoE classes. Nevertheless, similar to the inference of the RUMSI metric, results are highly accurate for an overall Web QoE monitoring solution.

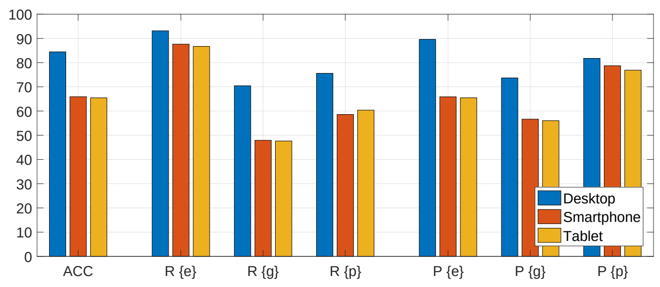


Fig. 8: Cross-device QoE classification performance. Models are trained on desktop measurements.

C. *Lack of Generalization for Mobile Devices*

Now that we have built the models for desktop, a natural question is, how good would these models perform in data collected from other device types? This is critical in practice, as a significant – and ever growing, share of web browsing activity comes from mobile devices. Fig. 7(b) depicts the distribution of inference errors per device type, using the ET10 model, trained exclusively on desktop data. Tab. III summarizes the corresponding performance metrics. There is a strong inference performance degradation when applying the desktop model to both smartphone and tablet data. Median absolute errors almost triple as compared to desktop performance. The distribution of errors shows that the desktop model tends to underestimate the RUMSI metric when applied to other devices, not present at training time. One could argue that RUMSI in smartphone and tablet is significantly higher than in desktop (cf. Fig. 1), thus performance degradation could be linked exclusively to this mismatch. However, performance degradation is also significant when only considering smaller RUMSI values, with median errors more than doubling for the example case of smartphone – from 260 ms to 592 ms, testing only for RUMSI below 5 seconds.

Performance degradation is also significant for the QoE classification problem. Fig. 8 reports the classification performance per device type, again using the ET10 model, trained on desktop data. Overall classification accuracy drops from 84% in desktop to 67% in smartphone and tablet. Recall for excellent QoE degrades only slightly, but strongly for the other classes, and most important, precision for excellent QoE also drops strongly, meaning that the model can not correctly track the classification problem.

Conclusion: RUMSI inference and QoE prediction can be properly realized using CBD and session-based features, extracted directly from the stream of encrypted bytes. However, models so far proposed in the literature for single device types [1], [2], [15]–[17], [19] might not perform properly in the wild, where other than desktop devices are used for web browsing. In particular, the evaluated models trained exclusively in desktop data realized strongly degraded performance when applied to smartphone and desktop devices.

device	**MAE-mAE (ms)**	**MRE-mRE (%)**	**PLCC**
desktop	598 – 260	31 – 16	0.879
smartphone	1245 – 721	41 – 28	0.728
tablet	1434 – 724	44 – 28	0.618
smartphone RUMSI $<$ 5s	867 – 592	43 – 29	0.455
smartphone RUMSI $>$ 5s	2812 – 2000	32 – 27	0.667

TABLE III: Inference performance per device type. The ET10 model is trained using desktop data.

model	**device**	**MAE-mAE (ms)**	**MRE-mRE (%)**	**PLCC**
DT	S	1021 – 372	34 – 14	0.770
	T	1082 – 298	31 – 11	0.731
ET10	S	**788 – 354**	**28 – 13**	**0.859**
	T	**804 – 314**	**25 – 11**	**0.867**
RF10	S	813 – 383	29 – 14	0.856
	T	867 – 357	27 – 12	0.852
RF100	S	**764 – 360**	**27 – 13**	**0.876**
	T	**815 – 334**	**26 – 12**	**0.866**
Bagging	S	820 – 380	29 – 14	0.855
	T	874 – 362	27 – 13	0.853
Boosting	S	1067 – 598	42 – 23	0.834
	T	1206 – 642	43 – 24	0.813
Bayes	S	1245 – 668	48 – 26	0.749
	T	1337 – 626	47 – 25	0.697
kNN	S	1205 – 639	46 – 23	0.724
	T	1284 – 592	44 – 21	0.709
XGB	S	1068 – 601	42 – 23	0.831
	T	1207 – 652	43 – 24	0.811

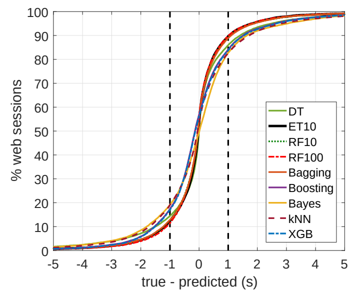
TABLE IV: Benchmarking of different ML models for RUMSI inference, for (S)martphone and (T)ablet. Training and validation is done on each specific device data-set.

model	**device**	**ACC**	**R{e}**	**R{g}**	**R{p}**	**P{e}**	**P{g}**	**P{p}**
ET10	S	**80.5**	**85.7**	**74.6**	**80.5**	**84.4**	**73.6**	**83.6**
	T	**83.5**	**87.9**	**77.9**	**84.4**	**87.7**	**77.8**	**84.9**
RF100	S	**83.1**	**86.6**	**78.0**	**84.5**	**87.5**	**76.9**	**84.7**
	T	**85.8**	**88.7**	**80.8**	**87.9**	**90.6**	**81.0**	**85.6**
Bagging	S	81.3	86.9	75.1	81.2	85.1	74.8	83.9
	T	83.8	89.3	78.4	83.4	88.5	77.8	85.1

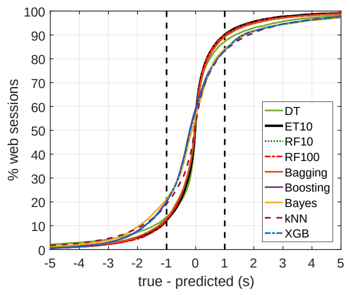
TABLE V: Benchmarking of selected ML models for Web QoE prediction, specialized per mobile device type.

V. Improving Performance by Specialization

As a second approach to the problem, we now investigate the performance improvements that can be realized by training per-device models. Such per-device models are of limited applicability in the wild, as one would need to first infer the specific device type being used for web-browsing to apply the correct model. Nevertheless, the exercise serves as a *what-if*, best performance benchmark, in terms of access device type, providing a baseline for a more general, multi-device model.

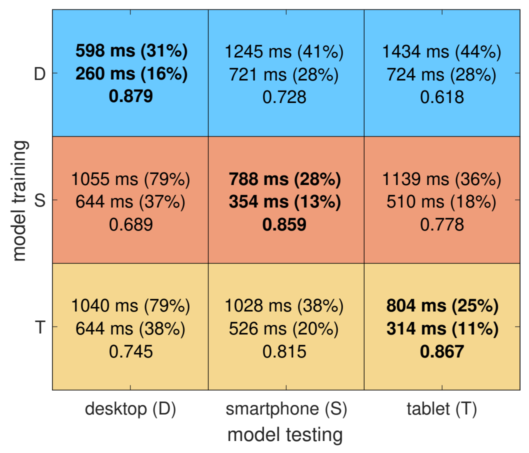


(a) Smartphone.



(b) Tablet.

Fig. 9: RUMSI inference performance, training/validating models on each specific mobile device dataset.



model training \ model testing	desktop (D)	smartphone (S)	tablet (T)
D	**598 ms (31%)** **260 ms (16%)** **0.879**	1245 ms (41%) 721 ms (28%) 0.728	1434 ms (44%) 724 ms (28%) 0.618
S	1055 ms (79%) 644 ms (37%) 0.689	**788 ms (28%)** **354 ms (13%)** **0.859**	1139 ms (36%) 510 ms (18%) 0.778
T	1040 ms (79%) 644 ms (38%) 0.745	1028 ms (38%) 526 ms (20%) 0.815	**804 ms (25%)** **314 ms (11%)** **0.867**

Fig. 10: Cross-device inference performance, using specialized, per-device ET10 as underlying model.

A. *Per-Device Analysis*

Tab. IV reports the RUMSI inference performance realized by the benchmarked ML models, for both smartphone and tablet devices; here, training and validation is done for each specific device data-set. The corresponding distribution of inference errors is depicted in Fig. 9. As expected, per-device specialized models significantly improve inference performance, obtaining results comparable to desktop (cf. Tab. I). Note in particular how specialized models for both smartphone and tablet provide even lower relative errors as compared to desktop. The same improvement observations apply to the Web QoE prediction/classification task: Tab. V reports the obtained results for three selected models, showing similar to desktop classification performance.

B. *Cross-Device Analysis*

Finally, Fig. 10 shows how the aforementioned cross-device training and validation issues also hold when considering different device types, using the RUMSI inference as example. The figure reports the usual AE, RE and PLCC metrics arranged as a training/testing matrix, where rows correspond to the device type data used for training, and columns to the device type data used for testing. While specialization improves inference performance – the matrix diagonal, training a model on measurements from a particular device type and applying the resulting model on measurements from a different device type results in poor inference performance, for all device type combinations. Note also how the cross-device lack of generalization applies to mobile devices, which are closer in terms of characteristics (cf. Fig. 1 and Fig. 2); while the performance degradation is lower when considering cross-data from smartphone/tablet devices, it is still significant.

Conclusion: training per device type models significantly improves performance for both Web QoE inference and prediction tasks, but the gains can be easily turned into strong performance degradation, when there is no certainty on the specific device type being monitored. As a consequence, per-device specialization still suffers from the same lack of generalization and practical limitations observed for the specific case of desktop, limiting its usage in the wild.

device	MAE-mAE (ms)	MRE-mRE (%)	PLCC
desktop	649 – 300	37 – 18	0.874
smartphone	804 – 363	27 – 14	0.855
tablet	798 – 318	25 – 11	0.868
all	750 – 327	30 – 14	0.869

TABLE VI: Multi-device RUMSI inference performance.

VI. A Multi-Device Model

Having shown the lack of generalization and the cross-device issues introduced by per-device specialized models, we take the most natural step to conceive multi-device Web QoE models. Possible approaches include the usage of stacking/ensembles of specialized models [27], or the inclusion of a pre-processing device-type classification task, preceding the inference/prediction task. However, the simplest approach of exposing models to all devices at training time already provides high accuracy and generalizes well across devices – which has high practical appeal as it simplifies deployment.

Considering multi-device RUMSI inference first, Tab. VI and Fig. 11(a) summarize performance attained by a single ET10 model, trained on all-devices data. Compared to per-device specialized models (cf. the matrix diagonal in Fig. 10), there is a marginal degradation for the corresponding multi-device model, and mainly observed for desktop, with an error increase close to 10%. Still, performance is consistent across the three device types, with an overall median absolute (relative) error of 327 ms (14%). Overall, the generalization capabilities of the multi-device model outweigh the accuracy of the specialized models, making it a preferred choice for Web QoE monitoring in real deployments.

Considering multi-device QoE classification next, Fig. 11(b) depicts performance obtained with a single ET10 model, trained on all-devices data: again, a slight performance degradation compared to the specialized desktop model (cf. Tab. II), yields to significant gain in terms of generalization to mobile devices (cf. Fig. 8). The overall model accuracy is 82.2%, with recall and precision values for excellent, good, and poor QoE of about $R = \{89\%, 73\%, 80\%\}$ and $P = \{86\%, 74\%, 82\%\}$, respectively.

Conclusion: multi-device models significantly improve generalization of the Web QoE inference and prediction approach across different devices, with only slight under-performance as compared to specialized models. As such, multi-device models provide simple, more accurate and more reliable monitoring capabilities in realistic web browsing scenarios.

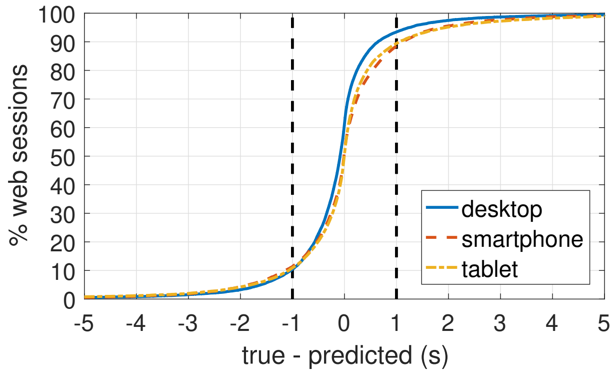
VII. Concluding Remarks

The last decade has seen a resurgence of Internet services offered through the Web, which are accessed by end users through an ever increasing plethora of end-devices, including desktop, laptops, tablets, and mobiles. In this context, Web QoE monitoring has gained strong momentum for ISPs, motivating this study on Web QoE monitoring from in-network, passive measurements, using machine learning models. Our study particularly targeted the impact that different end-user device types have in the performance of such models, considering both regression – inference of RUMSI, and classification tasks – Web QoE prediction.

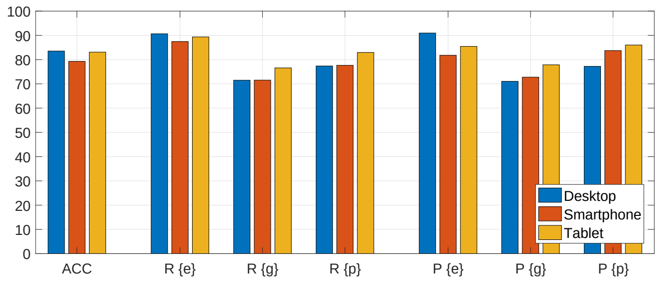
In a nutshell, our findings raise awareness of the fact that models for Web QoE monitoring must be exposed to multi-device measurements to achieve proper inference and prediction performance in real network deployments. To the best of our knowledge, we are the first to unveil the strong impact that the device type has in such models.

In more details, our empirical evaluations on a large, multi-device, heterogeneous corpus of Web QoE measurements for top popular websites shows that the proposed solution can infer the (RUM)SI as well as estimate Web QoE ranges from in-network traffic measurements with high accuracy. At the same time, the device type introduces a strong bias in the feasibility of Web QoE inference and prediction models, causing models trained for single device types to badly generalize to other devices. Finally, we showed that this cross-device lack of generalization can be solved by properly training on a multitude of devices.

Additional contributions of our study include the measurement and analysis of Web QoE metrics and web page characteristics for mobile devices – smartphones and tablets, which has been generally neglected in the literature, mostly due to the lack of tools and the additional complexities and limitations of the underlying mobile OS – Android in this paper. The characterization of these measurements revealed that, while most of the pages have similar characteristics for every device type in terms of size, share of content types, and retrieved contents from external servers, differences can be significant in some cases, potentially exacerbating the lack of generalization issues observed in the Web QoE monitoring models. Last, we have also confirmed that loading times in Android devices (smartphone and tablet) are significantly higher than for desktop, a common trend observed in the practice of web page speed analysis.



(a) RUMSI Inference.



(b) QoE Classification.

Fig. 11: (a) RUMSI inference and (b) QoE classification performance, ET10 model training on all-devices data.

References

[1] A. Huet, et al., "Revealing qoe of web users from encrypted network traffic," in *IFIP Networking Conference*, 2020.

[2] A. Huet, et al., "Web quality of experience from encrypted packets," in *ACM SIGCOMM Posters and Demos*, 2019.

[3] Cisco, "Cisco Annual Internet Report (2018-2023) White Paper, Updated March 2020," Cisco, Tech. Rep., 2020.

[4] D. N. da Hora, et al., "Narrowing the gap between qos metrics and web qoe using above-the-fold metrics," in *PAM*, 2018.

[5] E. Ibarrola, et al., "Web qoe evaluation in multi-agent networks: Validation of itu-t g. 1030," in *ICAS*, 2009.

[6] S. Egger, et al., "Waiting Times in Quality of Experience for Web Based Services," in *QoMEX*, 2012.

[7] "G.1030 : Estimating End-to-end Performance in IP Networks for Data Applications," https://www.itu.int/rec/T-REC-G.1030.

[8] T. Hoßfeld, et al., "Speed Index: Relating the Industrial Standard for User Perceived Web Performance to Web QoE," in *QoMEX*, 2018.

[9] Q. Gao, et al., "Perceived Performance of Top Retail Webpages in the Wild: Insights from Large-scale Crowdsourcing of Above-the-fold QoE," in *Internet-QoE*, 2017.

[10] A. Sackl, et al., "The influence of network quality fluctuations on web qoe," in *QoMEX*, 2014.

[11] A. Sackl, et al., "Quantifying the impact of network bandwidth fluctuations and outages on web qoe," in *QoMEX*, 2015.

[12] M. Varela, et al., "QoE in the web: A dance of design and performance," in *QoMEX*, 2015.

[13] M. Varela, et al., "Towards an understanding of visual appeal in website design," in *QoMEX*, 2013.

[14] S. Baraković and L. Skorin-Kapov, "Survey of research on quality of experience modelling for web browsing," *Quality and User Experience*, vol. 2, no. 1, p. 6, 2017.

[15] A. Saverimoutou, et al., "A 6-month analysis of factors impacting web browsing quality for QoE prediction," *Computer Networks*, vol. 164, 2019.

[16] A. S. Asrese, et al., "Measuring web latency and rendering performance: method, tools & longitudinal dataset," *IEEE Transactions on Network and Service Management*, 2019.

[17] M. Rajiullah, et al., "Web experience in mobile networks: Lessons from two million page visits," in *WWW*, 2019.

[18] S. Ihm, et al., "Towards understanding modern web traffic," in *ACM IMC*, 2011.

[19] E. Bocchi, et al., "Measuring the Quality of Experience of Web Users," *ACM SIGCOMM Computer Communication Review*, vol. 46, no. 4, 2016.

[20] M. Trevisan, et al., "PAIN: A Passive Web performance indicator for ISPs," *Computer Networks*, vol. 149, 2019.

[21] V. Aggarwal, et al., “Prometheus: Toward Quality-of-Experience Estimation for Mobile Apps from Passive Network Measurements,” in *ACM HotMobile*, 2014.
[22] A. Balachandran, et al., “Modeling Web Quality of Experience on Cellular Networks,” in *ACM MobiCom*, 2015.
[23] P. Casas, et al., “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.
[24] S. Wassermann, et al., “Machine learning models for YouTube QoE and user engagement prediction in smartphones,” *SIGMETRICS Perform. Eval. Rev.*, 2019.
[25] A. Nikravesh, et al., “Qoe inference and improvement without end-host control,” in *2018 IEEE/ACM Symposium on Edge Computing (SEC)*, 2018.
[26] P. Meenan, “Real User Monitoring SpeedIndex (RUMSI) - SpeedIndex Measurements from the Field using Resource Timings,” 2020. [Online]. Available: https://github.com/WPO-Foundation/RUM-SpeedIndex
[27] P. Casas, et al., “GML Learning, A Generic Machine Learning Model for Network Measurements Analysis”, in *International Conference on Network and Service Management*, 2017.