

Predicting the effect of home Wi-Fi quality on QoE

Diego da Hora, Karel van Doorselaer, Koen van Oost, Renata Teixeira

▶ **To cite this version:**

Diego da Hora, Karel van Doorselaer, Koen van Oost, Renata Teixeira. Predicting the effect of home Wi-Fi quality on QoE. INFOCOM 2018 - IEEE International Conference on Computer Communications, Apr 2018, Honolulu, United States. pp.1-10. hal-01677214

HAL Id: hal-01677214
https://inria.hal.science/hal-01677214

Submitted on 8 Jan 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Predicting the effect of home Wi-Fi quality on QoE

Diego da Hora
Telecom Paristech
Paris, France

Karel van Doorselaer, Koen van Oost
Technicolor
Edegem, Belgium

Renata Teixeira
INRIA
Paris, France

***Abstract*—Poor Wi-Fi quality can disrupt home users' internet experience, or the Quality of Experience (QoE). Detecting when Wi-Fi degrades QoE is extremely valuable for residential Internet Service Providers (ISPs) as home users often hold the ISP responsible whenever QoE degrades. Yet, ISPs have little visibility within the home to assist users. Our goal is to develop a system that runs on commodity access points (APs) to assist ISPs in detecting when Wi-Fi degrades QoE. Our first contribution is to develop a method to detect instances of poor QoE based on the passive observation of Wi-Fi quality metrics available in commodity APs (e.g., PHY rate). We use support vector regression to build predictors of QoE given Wi-Fi quality for popular internet applications. We then use K-means clustering to combine per-application predictors to identify regions of Wi-Fi quality where QoE is poor across applications. We call samples in these regions as poor QoE samples. Our second contribution is to apply our predictors to Wi-Fi metrics collected over one month from 3479 APs of customers of a large residential ISP. Our results show that QoE is good most of the time, still we find 11.6% of poor QoE samples. Worse, approximately 21% of stations have more than 25% poor QoE samples. In some cases, we estimate that Wi-Fi quality causes poor QoE for many hours, though in most cases poor QoE events are short.**

I. Introduction

Wi-Fi is the preferred technology for accessing the internet from home. Home Wi-Fi networks can, however, disrupt end-to-end application performance. For example, in dense urban neighborhoods it is typical to see tens of competing Wi-Fi networks [1], alongside many non Wi-Fi RF devices [2]. Also, a poorly located AP will leave stations with weak signal. In these cases, Wi-Fi quality can degrade users' internet experience, or the Quality of Experience (QoE).

When QoE is poor, users are often helpless and just call their residential ISP for support. In fact, the "administrators" of home networks likely have limited to no network management expertise [3]. The problem is that the ISP has little visibility into what is happening within the home Wi-Fi, which leads to painful helpdesk calls or, worse, customers who simply change provider. Our discussions with ISPs revealed that technical support calls represent a considerable fraction of their operations cost and they claim that often the root cause of the problems is the home Wi-Fi (recent results confirm that the home Wi-Fi often bottlenecks end-to-end performance [4], [5]).

Our goal is to build a system for ISPs to detect when home Wi-Fi quality degrades QoE *before* users call. Such a system would enable ISPs to proactively fix recurrent problems and consequently reduce customer churn. This system can also help to reduce the length of helpdesk calls, which in turn would help reduce operational costs. In many cases, the ISP provides and controls the home AP, so we leverage the home AP as monitoring point within the home. For such a system to scale to all the customers of an ISP, it must work with commodity APs already deployed by ISPs. This restriction excludes solutions that rely on APs with specialized hardware or multiple Wi-Fi interfaces [2], [6]–[8]. In addition, the system cannot disrupt user activity, so we exclude solutions that require installation on end users' devices [9] and that employ active measurements [10], [11], since continuous active measurements may disrupt user traffic and drain devices' battery. Instead, we rely on periodic polling of Wi-Fi metrics typically available on commodity APs (e.g., PHY rate and RSSI). Wi-Fi quality, however, is highly variable [6] and reporting every single Wi-Fi degradation would lead to too many alarms; most often for events imperceptible to users.

The first contribution of this paper is to **develop a method to detect instances of poor QoE from Wi-Fi quality metrics**. Detecting when Wi-Fi quality degrades QoE is challenging as we can only obtain a limited number of Wi-Fi metrics in commodity APs and we have no information about the applications that end users are running at a given time.[1] We rely on regression models to build predictors that estimate the effect of Wi-Fi quality on QoE for four popular applications: web browsing, YouTube, audio and video real time communication (RTC) (§IV). We generate training samples in a controlled environment, where we sweep a large set of Wi-Fi conditions, while observing application behavior (§II). To reduce the tests with real users, we measure application quality metrics and use state of the art methods to translate these into QoE as captured by the degradation of mean opinion scores (DMOS), in range [1-5](§III). Finally, we develop a method to combine the per-application predictors into a detector of poor QoE samples. For each possible combination of Wi-Fi quality parameters, we generate a sample with the predicted DMOS for each application and then we apply K-means clustering in the resulting dataset. Our analysis of the resulting clusters identifies ranges of Wi-Fi quality where we predict poor QoE for most applications, which we define as *poor QoE samples*.

Our second contribution is to **characterize the effect of Wi-Fi quality on QoE in the wild**. We apply our detection method on Wi-Fi metrics collected from 3,479 APs of customers of a large Asian-Pacific residential ISP over a period

[1]ISPs avoid running per-packet capture on APs due to privacy concerns and to avoid overloading the AP.

of one month in 2016 (§V). Our results show that Wi-Fi quality is often good enough for the four applications we study. We classify 11.6% of samples as poor QoE samples, and we find that 21% of stations have more than 25% poor QoE samples. Then, we group consecutive poor QoE samples from a single station into *poor QoE events*, which allow us to distinguish among short, intermittent, and consistent poor QoE events. We find that over 78% of poor QoE events are short, although we observe poor QoE events stretching over many hours. Finally, we analyze poor QoE events by verifying the underlying Wi-Fi metrics. We find that the majority of poor QoE events have average PHY rate $<$ 15Mbps. In the other cases, we observe an indication of interference. In particular, poor QoE events caused by non Wi-Fi interference are more likely intermittent than consistent.

II. Experimental setup

To understand how Wi-Fi quality affects user QoE, we perform controlled experiments with four applications: web browsing, YouTube streaming, and audio / video RTC. These are popular applications that represent a large fraction of users' traffic [6]. The typical method to assess QoE is to request users' explicit feedback and then average individual users' scores into a mean opinion score (MOS) [12]. This approach, however, is impractical for us given the wide range of Wi-Fi configurations we must cover. For example, our tests generate over ten thousand data points across the four applications and we require multiple users to rate each point. Instead, during each experiment we measure application-specific QoS metrics and use state-of-the-art models that map application QoS to QoE. This section describes our testbed, the Wi-Fi parameter space we test, and our method to generate traffic and collect per-application QoS.

A. Testbed

We emulate a Wi-Fi home network on the testbed shown in Figure 1. We use an AP with a Broadcom BCM6362 NIC with 802.11n 2x2 technology, and two MacBook pro stations (STA1 and STA2) with 802.11n 2x2 technology. We choose an AP with a Broadcom NIC because these are often deployed by residential ISPs. In particular, this model of AP is used in the deployment we study in §V. We only use metrics that are commonly exposed by Wi-Fi drivers and our predictors should generalize well for other 802.11n NICs.

The Wi-Fi testbed is located in Belgium, and it connects to the internet through an Ubuntu 12 gateway using TC to emulate bandwidth restrictions of a home network access link, with 16 Mbps available bandwidth for download traffic. Our Web and iperf servers are Ubuntu 12 computers. We also use an Android tablet with 802.11n 1x1 technology, using iperf in server mode to receive traffic that interferes with the AP under test. We use the default AP Wi-Fi configurations: 20MHz channel width and long guard interval PHY rates only. We restrict Wi-Fi physical layer (PHY) rates to only use one-spatial-stream, due to the difficulty of reliably enabling MIMO communication in shielded boxes. Although we only study PHY rates up to 65 Mbps, results for higher PHY rates should be similar to those where PHY rate is 65 Mbps, since in both cases the transmission rate is unlikely to degrade QoE. We expect results for 802.11ac NICs to be similar to those of 802.11n for PHY rates and modulation schemes covered in this study.

B. Experimental parameters

We vary the Wi-Fi conditions over two axis:

Link speed. We vary the link speed by introducing 6, 12, 15, 18, 19, and 20 dBs of attenuation in the path between AP and STA using a programmable attenuator.

Medium availability. We vary the medium availability at the AP by introducing interference from Wi-Fi or non-Wi-Fi sources. To introduce Wi-Fi interference, we use the iperf client to generate competing Wi-Fi traffic in the "interfering link", as shown in Figure 1. The AP from the interfering link is configured to only use PHY rate 5.5 Mbps, which blocks the medium longer per medium access. We found that by generating constant bit rate UDP traffic of 0.7 Mbps, 1.1 Mbps, 2.1 Mbps and 2.9 Mbps we obtain, respectively, 70%, 50%, 30% and 15% medium availability, measured by the clear channel assessment (CCA) counters on the AP.

To generate interference from a non Wi-Fi source, we use a signal generator to inject a narrowband sinewave to block the AP's CCA. We sweep the interfering sinewave in and out of the spectral CCA range of the AP every 200ms. We vary the percentage of time the sinewave is inside and outside this range to create scenarios with medium availability of 100%, 75%, 50%, 25%, and 12.5%. AP's Wi-Fi counters confirm that we obtained the intended availability with less than 5% error.

C. Applications

For each Wi-Fi scenario, we execute automated tests with the following applications.

Web browsing. STA1 uses PhantomJS, a headless browser, to sequentially access 10 times a set of 10 pages on the internet. We chose a mix of pages from the Alexa TOP 20 pages in Belgium. We access the front pages of the following domains: bing.com, google.be, twitter.com, live.com, wikipedia.org, facebook.com, yahoo.com, amazon.fr, nieuwsblad.be, and hln.be. We use JavaScript to record the page load time (PLT) triggered by the *onload* event.

YouTube. STA1 uses Google Chrome to access an instrumented YouTube page, which downloads and plays each YouTube video using DASH, during two minutes. We access three YouTube videos per experiment: a politician speech, a movie trailer, and a music clip. We choose these clips to obtain diversity in video content. We use JavaScript to record join time, buffering events, and video resolution changes.

Audio and video RTC. We use WebRTC to perform audio and video RTC tests. We implemented a simple WebRTC application, stored on the web server, that allows two peers to communicate in an audio and/or video call. We use the default OPUS plugin on WebRTC to encode audio and the VP8 plugin to encode video with default options. We use

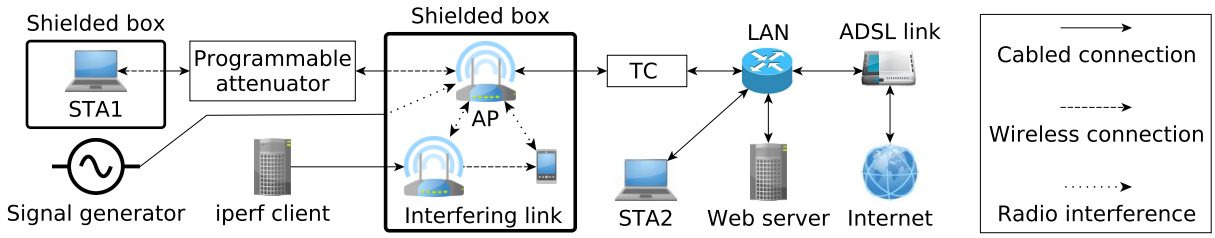


Fig. 1: Wi-Fi testbed used in the study.

the *–use-fake-device-for-media-stream* command line option to perform RTC tests with a pre-recorded audio/video sample. This setup closely resembles a real user environment, since the audio/video processing will go through the same processing stack as a real microphone/camera. We record the received audio/video streams by each WebRTC peer.

We use a set of 20 different audio samples recommended by ITU-T for speech quality assessment [13], each with an approximate duration of eight seconds. We create a single audio file with all the audio samples, which is then sent to Google Chrome's fake audio input device. To prevent echo canceling from interfering with the audio samples, we only examine cases where STA2 sends audio and STA1 is silent. After the experiment, we manually extract the individual audio samples from the received audio feed. For video, we use standard reference videos featuring human subjects from the Xiph collection [14], namely: *FourPeople*, *Johnny*, *KristenAndSara*, *Vidyo1*, *Vidyo3* and *Vidyo4*. We downscaled the original video format to 640×480, 30 fps to match common webcam capabilities. We merged all the video samples, each with 300 frames (10 seconds), into a single video file, separated by 15 "black" frames to mark the transition between samples. After the experiment, we extract individual video samples from the received video using the "black" frames to detect sample transition, as well as manual verification (in rare cases, none of the "black" frames were received).

III. Training Set

We use the testbed and controlled experiments presented in the previous section to build a training set where each sample is a vector that contains Wi-Fi metrics we passively collect from the AP labeled with the estimated QoE for each application. This section describes our approach to map application QoS metrics into an estimated QoE and how we build the training set.

A. QoE metric: Degradation MOS

For each application, we denote the application QoS to QoE model as a function, $f_{app}^{MOS} : (a, x_a) \rightarrow y^{MOS}$, where x_a is a vector of application QoS metrics for an application a and y^{MOS} is an absolute MOS score. Although the models in the literature output an estimate of the absolute MOS, our goal is to estimate Wi-Fi's contribution to MOS degradation. For example, the simple act of encoding a video for real-time transmission will reduce its quality and the absolute MOS will capture this degradation. Yet, users today are used to the quality loss due to encoding. Hence, we measure QoE with the *degradation MOS* as the normalized output of f_{app}^{MOS} in range $[1-5]$, using Equation 1. Note that we call this function f_{app}^{DMOS} to emphasize that it estimates DMOS based on application QoS metrics, in contrast to the models we introduce in §IV, denoted by f_{Wi-Fi}^{DMOS}, which estimate DMOS from Wi-Fi quality metrics.

$$f_{app}^{DMOS}(a, x_a) = 4.0 \times \frac{f_{app}^{MOS}(a, x_a) - MOS_{min}(a)}{MOS_{max}(a) - MOS_{min}(a)} + 1.0 \quad (1)$$

We set $MOS_{min}(a)$ as $\min\{f_{app}^{MOS}(a, x_a)\}$ for all x_a in the experiments, and $MOS_{max}(a)$ as the average $f_{app}^{MOS}(a, x_a)$ for x_a during baseline experiments (i.e., no Wi-Fi impairment). The estimated DMOS is always 5 in baseline scenarios, and always 1 in the worst scenario. It is possible to find $MOS_{max}(a)$ by maximizing $f_{app}^{MOS}(a, x_a)$ for the set of application QoS parameters expected to be found in the target scenario, and $MOS_{min}(a)$ by minimizing $f_{app}^{MOS}(a, x_a)$ instead. We use Equation 1 to obtain estimated DMOS for each application model. We interpret DMOS quality impairments using the degradation category rating scale [15], where: 5, Imperceptible; 4, Perceptible but not annoying; 3, Slightly annoying; 2, Annoying; and 1, Very annoying.

B. QoS-to-QoE models

We select state-of-the-art QoS-to-QoE models to implement $f_{app}^{MOS}(a, x_a)$ for each application, a, we study as summarized below. For more details on the models as well as our validation through user studies, refer to our tech report [16].

Video RTC. There is extensive work on objective video quality assessment. Since we have access to the original video signal in our testbed, we use full reference analysis to calculate the video structural similarity (SSIM). We compare each video clip received by STA1 with the original sent by STA2 and compute per-frame SSIM. We calculate the video's average SSIM (s) and map it to the video MOS score using the model proposed by Wang et. al [17].

We noticed that under low network capacity, WebRTC often suppresses frames to reduce bandwidth, sending as little as 10 frames per second in extreme conditions. We deal with skipped frames using the strategy employed by Zinner et. al [18], by comparing reference skipped frames to the last received frame.

This reduces SSIM whenever the transmitted sample presents skipped frames or freezes.

On some experiments, we observed instances where STA1 was able to associate to the AP and communicate to STA2 through WebRTC, but could not sustain the video call due to high packet loss. Since WebRTC prematurely terminates the call, we could not obtain transmitted samples. For every experiment where we observe a WebRTC connection for 10 seconds but no video traffic, we generate a sample with DMOS = 1.0, since there is no service.

Audio RTC. There are many methods to estimate the perceived voice quality. We use the Perceptual Speech Quality Measure (PESQ) [19] to detect audio distortions and the E-model [20] to account for one way delays, as proposed by Lingfen et. al [21]. We obtain PESQ MOS by comparing the reference and received test samples using the reference PESQ implementation by ITU-T, and one way delay by monitoring the median latency of WebRTC packets. Similarly to video WebRTC experiments, we observed instances where STA1 was able to associate to the AP and communicate to STA2 through WebRTC, but could not sustain the audio call. For every experiment where we observe a WebRTC connection for 10 seconds but no audio traffic, we generate a sample with DMOS = 1.0.

Web browsing. While there is extensive work showing that web browsing QoE is a result of many influence factors, user waiting time is consistently identified as the main system influence factor and the page load time (PLT) as the main application QoS metric. We use the PLT as measured by the *onload* event, since for the majority of pages, the *onload* event has a strong correlation with the user page load time [22]. We employ the ITU-T G 1030 single-page web-QoE model [23], considering the medium network context. The ITU-T G 1030 proposes a logarithmic relationship between PLT and MOS, which is in line with most recent studies on web QoE [24], [25]. During web browsing experiments, we limit PLT to a maximum of 10s. When the web page takes more than 10s to load, but the main HTML finished loading, we consider PLT = 10s. If the main HTML never loads, we consider that the service failed and set DMOS = 1.0.

YouTube. YouTube uses a playback buffer to smoothly play the video. *Buffering events* occur when the playback buffer empties. YouTube uses adaptive bitrate streaming in default configuration. In this scenario, the video player dynamically selects video bitrate during playback to avoid buffering events. There are three main factors influencing QoE: average bitrate, join time, and buffering events [26], [27]. Most studies agree that users are more tolerant to initial delays. Therefore, we model YouTube QoE as a function of buffering events and video bitrate. For buffering events, we use the MOS model proposed by Wamser et. al, which estimates MOS values given the number and duration of buffering events [27]. In cases of alterations on the video bitrate, we use SSIM to quantify the impact of video resolution on QoE similarly to Zinner et. al [18]. Finally, we model YouTube QoE as the minimum MOS between the estimated MOS of the two models. By using the minimum MOS, we remain faithful to each model when only one type of impairment is present. We note that both types of impairments happen together in less than 10% of samples in our testbed dataset.

TABLE I: Wi-Fi metrics measured on the access point.

Metric	Description	Period
BUSY	% of time the medium is busy	2 s
WiFi	% of time busy due to Wi-Fi traffic	2 s
nonWiFi	% of time busy due to non Wi-Fi traffic	2 s
TxPhy	PHY rate of last frame sent	1 s
FDR	Frames sent / retransmitted to STA	1 s
RSSI	Received signal strength indicator	1 s

C. Building the training set

During each experiment, we passively measure the Wi-Fi metrics shown in Table I on the AP. We calculate features describing the Wi-Fi metrics considering Wi-Fi samples measured over an interval T during the application execution. The training set consists of one feature vector per application sample containing the mean, std, min, max, 25%-ile, and 75%-ile for each metric in Table I, labeled with $f_{app}^{DMOS}(a, x_a)$, where x_a captures a's application QoS measured directly from each application.

We choose T based on the duration of the application execution. For audio and video experiments, we consider one application sample per audio / video sample, with $T = 10s$. For web browsing, we create one sample per web page access, using $T = 10s$ since this is the maximum page load time. For YouTube, we create one sample per video playback, using $T = 120s$ since we play each video for two minutes. We obtain a total of 3175 video samples, 3062 audio samples, 153 YouTube samples, and 5370 Web samples in the training set.

IV. Predicting the effect of Wi-Fi quality on QoE

This section shows how we predict the effect of Wi-Fi quality on QoE. We must learn a function, $f_{Wi-Fi}^{DMOS} : (a, x_{Wi-Fi}) \rightarrow y^{DMOS}$ that for each application, a, predicts DMOS based on a feature vector computed from measured Wi-Fi quality, x_{Wi-Fi}. We formulate the problem of predicting the effect of Wi-Fi quality on QoE as a regression problem.

A. Building the predictor

We select support vector regression (SVR) as regression algorithm as it outperformed linear regression, Gaussian naive Bayes, and decision tree regression in terms of prediction accuracy and model generality during preliminary tests. Since SVR is not scale invariant, we normalize features based on minimum / maximum values from the dataset. We evaluate prediction accuracy using root mean squared errors (RMSE), as it is common practice on regression models.

Feature selection. We perform feature selection with stepwise regression, a method which iteratively adds features to the feature vector in order to minimize prediction error. Figure 2 shows prediction accuracy as we increase the number of features. We find that using more than six features is

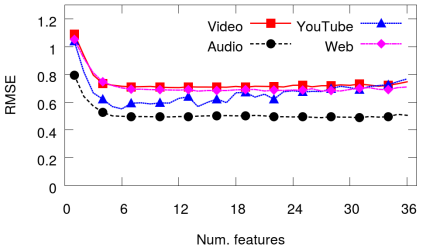


Fig. 2: Error for predictors with different number of features.

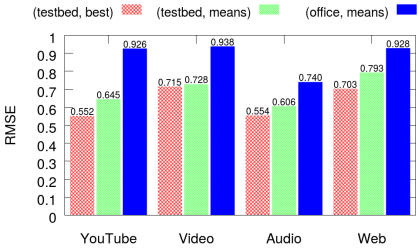


Fig. 3: Predictor evaluation.

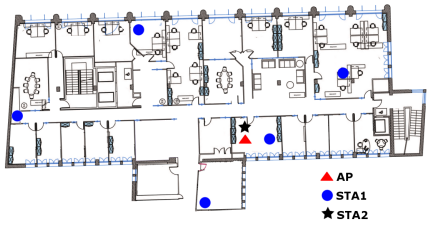


Fig. 4: AP / STA1 position on validation dataset

TABLE II: Feature selection output per application.

Application	Feature vector		
Video	$\text{TxPhy}_{25\%}$	$\text{BUSY}_{25\%}$	BUSY_{max}
	RSSI_{mean}	$\text{RSSI}_{75\%}$	$\text{WiFi}_{25\%}$
Audio	TxPhy_{min}	RSSI_{std}	$\text{WiFi}_{25\%}$,
	WiFi_{max}	nonWiFi_{max}	FDR_{mean}
YouTube	TxPhy_{mean}	$\text{BUSY}_{75\%}$	RSSI_{mean}
	$\text{RSSI}_{25\%}$	$\text{WiFi}_{25\%}$	nonWiFi_{min}
Web	TxPhy_{max}	BUSY_{std}	RSSI_{min}
	WiFi_{max}	nonWiFi_{max}	FDR_{mean}

unnecessary, as prediction accuracy flattens out (or reduces in the case of YouTube). Table II shows the best feature subset per application. We show results for predictors using two set of features: $f^{DMOS}_{Wi-Fi,best}$, a predictor that uses the features from Table II; and $f^{DMOS}_{Wi-Fi,means}$, a predictor that considers the features available in the deployment dataset (§V), namely the means of TxPHY, RSSI, BUSY, WiFi, and nonWiFi.

Parameter selection. We configure three SVR parameters. SVR uses the C parameter to penalize misclassified samples and γ parameter to set the range of influence of each support vector in the model. High values of C and γ are known to over-fit the dataset. The ϵ parameter regulates the error margin over which predictions incur into no penalty. We find the best combination of parameter through grid optimization, with tests where $C \in [1, 1000]$, $\epsilon \in [0.01, 1]$, and $\gamma \in [0.1, 100]$ at regular intervals. First, we use RMSE from ten-fold cross-validation to track which combination of SVR parameters works best for each predictor. We found that $\epsilon = 0.3$ works well for all applications, but several combinations of γ and C produce low RMSE values. Since SVR models with high γ tend to generate high-variance models, we choose a low γ parameter (in our case, $\gamma = 3$) alongside the C which minimizes RMSE.

Web page complexity. For web browsing, we build one predictor per page, because the baseline PLT of different web pages vary significantly (e.g., 0.287s for *bing.com*, 2.849s for *facebook.com*). For simplicity, we summarize web QoE results using three different pages with different levels of complexity: google.be (web simple), facebook.com (web average), and amazon.fr (web complex).

Model visualization. To understand how Wi-Fi metrics influence the decision of predictors, we show the decision curves for SVR models using two features: TxPhy_{mean} and BUSY_{mean} in Figure 5. The SVR models learn similar boundary conditions. When Wi-Fi conditions are perfect, with BUSY_{mean} near 0 and TxPhy_{mean} at maximum, we observe a predicted DMOS above 4.5 on all predictors. Similarly, when either TxPhy_{mean} is close to 0 or BUSY_{mean} close to 100%, we observe a predicted DMOS below 1.5.

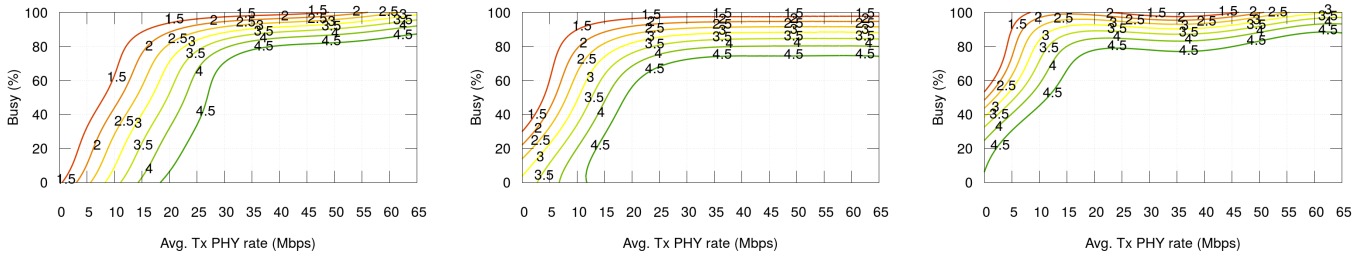
Figures 5a and 5b show the SVR models for video RTC and YouTube QoE. We observe that both models predict reduced QoE when TxPhy_{mean} is below 15 Mbps. This indicates that these applications are sensitive to Wi-Fi bottlenecks. We also see that YouTube is more resilient to Wi-Fi impairments. This is due to the adaptive bitrate selection, which dynamically adjusts video bitrate to match available bandwidth, avoiding buffering events. Figure 5c shows the SVR model for predicting audio QoE. Audio impairments only happen when Wi-Fi conditions are extremely poor. We also observe less cases where predicted QoE is between 1.5 and 4.5. This is because the audio RTC application has very little room for adaptation, so either Wi-Fi quality supports audio, in which case it works well, or it does not and audio quality is very poor or the call drops.

B. Validation

To validate our predictors, we generate the "Office" dataset in an uncontrolled environment (i.e. our lab), over four weeks in 2016. We vary the position of STA1 according to Figure 4 and automatically execute experiments every 30 minutes, from 7am to 10pm, in order to observe different levels of link speed and interference. Each experiment consists of multiple runs as described in §II-C. Then, we select 22 experiments where we observe the most diverse set of Wi-Fi parameters. We follow the same procedure as to generate the testbed dataset to obtain the estimated DMOS per application, f^{DMOS}_{app}.

We evaluate prediction errors of the two predictors, $f^{DMOS}_{Wi-Fi,best}$ and $f^{DMOS}_{Wi-Fi,means}$ on the Wi-Fi testbed dataset through ten-fold cross-validation. We also evaluate $f^{DMOS}_{Wi-Fi,means}$ using the Office dataset, i.e., in these results we learned the predictor using the testbed dataset, and evaluate the accuracy using the office dataset.

Figure 3 shows prediction errors for each of these cases for each application. We observe that the error of $f^{DMOS}_{Wi-Fi,means}$ is only slightly higher than that of $f^{DMOS}_{Wi-Fi,best}$. This result indicates that $f^{DMOS}_{Wi-Fi,means}$, which uses simple means, is sufficient to estimate the impact of Wi-Fi quality on QoE. We



(a) Video RTC (b) YouTube (c) Audio RTC

Fig. 5: Visualization of YouTube, audio and video RTC SVR models with two features: mean TxPhy and BUSY.

TABLE III: Cluster centroids in QoE space.

	C0	C1	C2	C3	C4	C5
Video	1.42	1.42	2.98	3.63	4.55	4.71
Audio	1.62	2.62	3.61	4.27	4.73	4.83
YouTube	1.35	1.49	2.51	3.37	4.43	4.87
Web complex	1.14	1.36	1.55	2.18	2.98	4.29
Web average	1.19	1.82	1.52	2.74	3.90	4.91
Web simple	1.74	4.48	2.70	4.30	4.74	4.98

observe higher prediction errors on the Office dataset. These errors are closer to what we expect in a real user environment, given that the predictor was not learned using this dataset.

C. Detection of Poor QoE

Our predictors so far work on a per-application basis, but we have no information on the specific applications that home users are running. Instead, our goal is to focus on the worst cases, i.e., when we predict that multiple applications experience poor QoE. To identify these poor QoE samples in our Wi-Fi quality parameter space, we generate a synthetic set by sweeping all combinations of Wi-Fi parameters. For this analysis, we consider the predictor $f^{DMOS}_{Wi-Fi,means}$. We only generate samples where BUSY < 100 and BUSY = WiFi + nonWiFi. For each Wi-Fi sample, x_{Wi-Fi}, we obtain a six-tuple where each element is the output of one of our per-application predictors (YouTube, Video, Audio, and Web complex, average, and simple), $f^{DMOS}_{Wi-Fi,means}(a, x_{Wi-Fi})$. Then, we apply a clustering algorithm to discover groups of related samples.

We find clusters using the K-means algorithm, which is a simple and efficient clustering algorithm. One challenge with K-means is to select the number of clusters, K. We test K values between 2 and 30, and find that the fraction of explained variance, used to quantify intra-cluster similarity, increases very slowly for $K > 6$. Furthermore, a small number of clusters eases our manual analysis to identify the meaning of each cluster.

Table III shows cluster centroids for K=6, ordered by the Euclidean norm. We manually analyze each cluster. Clusters C5 and C4 generally present high DMOS on most applications, with DMOS ≥ 4.0 on all but two applications. Clusters C0, C1, and C2 present DMOS < 3.0 on the majority of applications. Figures 6 and 7 show clusters disposition on Wi-Fi QoS space, considering that only Wi-Fi interference or non Wi-Fi interference generates medium occupancy, respectively.

TABLE IV: Station Wi-Fi technology on deployment.

Technology	Before filtering		After filtering	
	stations	samples	stations	samples
.11n 1x1	17784	58.5M	13496	58.1M
.11n 2x2	12272	72.1M	9729	71.8M
.11g	964	3.3M	636	3.3M
Total	**31020**	**133.9M**	**23861**	**133.2M**

It is interesting to note that clusters are cohesive in the Wi-Fi QoS space. Also, clusters associated with low DMOS (e.g. C0) are located in regions of known poor Wi-Fi configurations (e.g. low PHY rate and high medium occupation).

Poor QoE samples. Clusters C3, C2, C1, and C0 contain samples where predicted DMOS < 4.0 on multiple applications. Therefore, we classify samples from clusters C3, C2, C1 and C0 as *poor QoE* samples. We call samples from cluster C4 and C5 as *good QoE* samples.

V. Effect of Wi-Fi Quality on QoE in the Wild

In this section, we analyze the effects of Wi-Fi quality on QoE in the wild. We collect Wi-Fi measurements from 3,479 APs of customers of a large Asian-Pacific residential ISP, with 31,020 stations connected to these APs, during the month of September, 2016. We analyze 133.9 million samples.

A. Deployment dataset

The AP logs Wi-Fi performance every second for all stations connected to it. This AP collects all the metrics from Table I except for FDR. Our backend system polls the APs to collect the mean metrics every 30s. Since station metrics such as TxPhy need traffic to reflect the link quality, we filter out samples from inactive stations.

First, we look at deployment characteristics regarding station usage and technology. We find some stations with very little activity: we have less than 10 samples on 6.5% of stations. This is because some home Wi-Fi appliances seldom transfer data, such as “smart scales”, and there is not enough traffic to provide valid samples. There are also APs with a small number of samples (2.99% had less than one hour of data). We consider one station *active* if we observe at least 5 minutes of activity over 24 hours (similarly for the AP). 23.6% of stations have one or less active days, which indicates a device belonging to visitors or sporadically used.

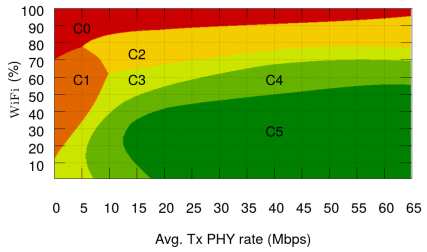


Fig. 6: Cluster classification: Wi-Fi interference

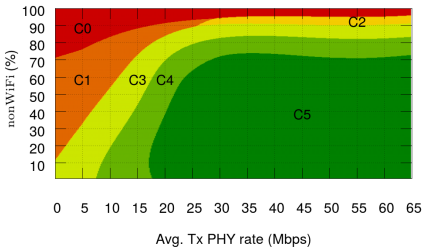


Fig. 7: Cluster classification: non Wi-Fi interference

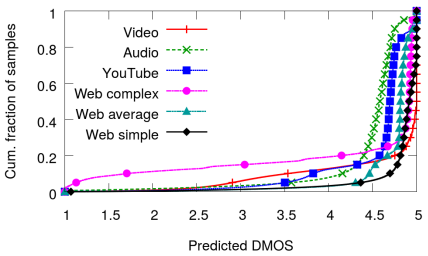


Fig. 8: Predicted DMOS frequency in the wild.

TABLE V: Cluster frequency in deployment.

	C0	C1	C2	C3	C4	C5
Frequency	0.26%	0.59%	0.68%	9.93%	5.47%	83.1%

We focus our study on APs and stations that actively use the Wi-Fi network. We filter stations with less than two active days per week (leaving 76.9% of stations for our analysis) and APs with less than five active days per week (we analyze 92.29% of APs). Table IV shows the number of stations observed per technology before and after filtering, as well as the number of samples. Notice that we retain 99.4% of all samples, although we filter 23.1% of stations. Over 34% of .11g stations have less than 2 active days per week, which suggests that stations with legacy technology are less frequently used.

B. QoE predictions per application

Figure 8 shows the cumulative fraction of predicted DMOS per application in the deployment. We observe that over 68% of samples contain predicted DMOS above 4.5 for all applications. This is reassuring as most of the time, Wi-Fi works well for all applications. We observe instances where predicted DMOS < 3.0 for all applications. We predict DMOS below 3.0 for web complex and video RTC on 14.8% and 5.8% of samples, for YouTube and web average on 3.3% and 2.5% of samples, and for audio RTC and web simple on 1.2% and 1.1% of samples. The difference in frequency between applications is due to the application sensitivity to the Wi-Fi impairments. All samples with mean TxPHY up to 6.5 Mbps show predicted DMOS below 3.0 for web complex, on a total of 9.14% of all deployment samples. On other predictors, this only occurs when we also observe interference.

C. QoE across applications

Some applications require better Wi-Fi quality than others, with a larger fraction of samples where predicted DMOS $<$ 3.0. Now, we would like to investigate how the same Wi-Fi quality affects QoE for different applications. We rely on the clusters from §IV-C. Table V shows the fraction of the samples in the deployment dataset in each cluster. Fortunately, we find that nearly 88.4% of samples in the deployment fall in Clusters C4 and C5, which have good QoE. We classify 11.6% of deployment samples as poor QoE samples.

Figure 9 shows the fraction of poor QoE samples per station. ISPs can use the fraction of poor QoE samples to identify stations with frequent Wi-Fi-related QoE issues. The majority of stations (53%) have less than 5% poor QoE samples, therefore, Wi-Fi is unlikely to impair user experience in those cases. 21% of stations, however, have more than 25% poor QoE samples, which represent frequent Wi-Fi impairments. For the rest of this analysis, we focus on stations with at least 5% poor QoE samples.

D. Characterization of poor QoE events

We define a *poor QoE event* as a series of poor QoE samples spaced by no more than a time threshold T. We select T based on the distribution of the inter-arrival times of samples (not shown for brevity). We use $T = 2$ minutes, because it represents the knee of the curve; 74% of all samples have inter-arrival time ≤ 2 minutes. Notice that our definition of poor QoE events allow for some good QoE samples to occur in-between poor QoE samples. Even in these cases, users are likely to experience poor quality during the poor QoE event, since Wi-Fi quality is intermittent.

Figure 10 shows the duration of poor QoE events, for T = 2 minutes. The majority of poor QoE events are short: 78% have duration of two minutes or lower. Mobile stations going in or out of Wi-Fi range or temporary Wi-Fi impairments, such as those caused by a peak of Wi-Fi interference, are some of many possible reasons for short-lived poor QoE events. We also observe a number of long poor QoE events, some stretching over several hours. These more persistent Wi-Fi problems can happen when either the station is poorly located or the AP is using a busy Wi-Fi channel.

We categorize poor QoE events into three distinct classes. Short-lived poor QoE events are those with duration below or equal 2 minutes We divide other events between consistent and intermittent. Intermittent events have good QoE samples in-between poor QoE samples. If the fraction of poor QoE samples during a poor QoE event is greater or equal to 80%, we call it a consistent poor QoE event, otherwise it is an intermittent poor QoE event.

Figure 11 shows the division between short, intermittent, and consistent poor QoE events. We group stations based on their fraction of poor QoE samples. For stations with fraction of poor QoE samples below 5%, 51% of poor QoE samples belong to short or intermittent poor QoE events. Intermittent and short-lived poor QoE events are difficult to detect and

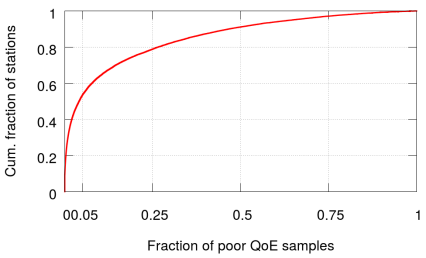


Fig. 9: Fraction of poor QoE samples per station

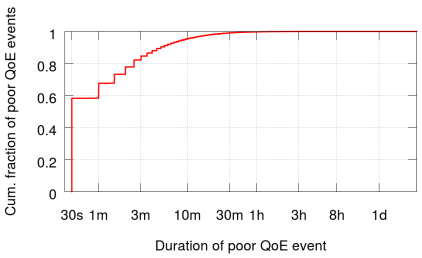


Fig. 10: Duration of poor QoE events for $T = 2$ minutes.

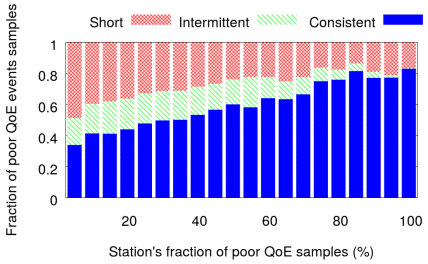


Fig. 11: Classification of poor QoE samples per group of station.

TABLE VI: Characterization of poor QoE events

Wi-Fi metrics		Consistent	Intermittent
$\text{TxPHY}_{mean} < 15$	$\text{BUSY}_{mean} > 60$		
✓	✓	14.2%	5.0%
✓	✗	74.3%	73.0%
✗	✓	10.6%	18.6%
✗	✗	0.8%	3.4%

diagnose, since when we detect a good QoE samples after a series of poor QoE samples it is unclear if the root cause was fixed or is only temporarily absent. These stations require a long-term monitoring approach to detect and diagnose Wi-Fi problems. We observe that stations with a larger fraction of poor QoE samples have less short and intermittent events, and more consistent poor QoE events. For stations with fraction of poor QoE samples above 50%, we observe, on average, 69% of poor QoE samples on consistent poor QoE events. While this holds true for the majority of stations, there are some exceptions. Some stations with a very high fraction of poor QoE samples have nearly all samples on short poor QoE events. In these cases, we found that this occurs because they generate traffic for very short periods, with long inactive intervals in-between. This prevents us from classifying poor QoE samples into long poor QoE events.

Analysis of long poor QoE events. Next, we investigate what is the most likely cause for intermittent and consistent poor QoE events. We use simple thresholds on the Wi-Fi metrics to infer the cause of poor QoE events.

We look at mean Wi-Fi metrics along all samples of a poor QoE event to identify its most likely cause. We find that, in general, poor QoE events show either $\text{TxPHY}_{mean} < 15$ Mbps or $\text{BUSY}_{mean} > 60\%$. We diagnose events where $\text{TxPHY}_{mean} < 15$ Mbps as having *low PHY rate*, and events where $\text{BUSY}_{mean} > 60\%$ as having *high medium occupation*. On poor QoE events with high medium occupation, we get further insight on the root cause by comparing WiFi_{mean} and nonWiFi_{mean}.

Table VI summarizes this analysis. We find that 74.3% of consistent poor QoE events have only low PHY rate and 10.6% have only high medium occupation (73.0% and 18.6% for intermittent events). We find both low PHY rate and high medium occupation on 14.2% of consistent poor QoE events and on 5.0% of intermittent poor QoE events. When we observe both low PHY rate and high medium occupation, we more often have consistent poor QoE events. We also observe 0.8% and 3.4% of consistent and intermittent poor QoE events where TxPHY_{mean} is > 15 Mbps and $\text{BUSY}_{mean} < 60\%$. These cases occur on the prediction boundary of poor QoE samples, and they occur more often for intermittent events.

We further investigate cases where we only observe high medium occupation. In these cases, we typically observe either high WiFi_{mean} or nonWiFi_{mean}. We find that 62.5% of consistent poor QoE events have high medium occupation due to Wi-Fi interference, since WiFi_{mean} is $> 50\%$. This percentage is only 33.8% for intermittent poor QoE events, which indicates that non Wi-Fi interference is more likely to produce intermittent poor QoE events.

VI. Related Work

Quality of Experience. Subjective quality assessment uses human subjects to explicitly evaluate user experience [12], [27]–[29]. Objective quality assessment proposes quality metrics that correlate with user opinion, but can be calculated without user involvement [27], [30], [31]. We use objective quality metrics to assess the effect of Wi-Fi quality on QoE. Mapping network QoS to QoE for a number of applications has been a popular research topic [32]–[35], but none of these studies consider the mapping of Wi-Fi QoS to QoE. One exception is the work of Chakraborty et al. [36]. They propose ExBox, a system that relies on predicting the impact of an incoming flow on QoE of a set of flows to make flow admission control decisions in Wi-Fi enterprise networks. ExBox assumes that some of the stations in the Wi-Fi network are instrumented to report application QoS metrics for online learning of a binary classifier that decides whether incoming flows can be accepted or not. The only Wi-Fi metric that ExBox considers is SNR. In contrast, our goal is detection and diagnosis in home Wi-Fi, which brings different operational constraints (e.g., no knowledge of applications, no instrumentation of user devices). Although we study a similar set of applications, we offer a more in-depth study of the relationship between multiple Wi-Fi metrics and QoE. This work extends our earlier workshop paper [37], which focused only on Web QoE.

Wi-Fi performance characterization. Pei et al. [4] deploy 47 APs in a university campus and show that the Wi-Fi link significantly contributes to end-to-end latency. Ioannis et al. [1] characterize the Wi-Fi performance with passive measurements from 167 homes, over a period of four months. Biswas

et al. [6] characterize Wi-Fi network usage and performance on over 20 thousand industrial networks. These studies improve our understanding of Wi-Fi quality in different settings, but none of them addresses the issue of how Wi-Fi quality affects QoE as we do.

VII. Conclusion

This paper was the first to shed light on the effects of home Wi-Fi quality on QoE of different applications. Our first contribution was to develop a method to detect instances of poor QoE from Wi-Fi quality metrics. We showed that we can predict application QoE from Wi-Fi metrics available on commodity APs with low errors. We built on these predictors to identify poor QoE samples. ISPs can use this method to detect when Wi-Fi quality is likely to degrade customers' QoE to proactively fix Wi-Fi problems. In fact, we have interest from ISPs to incorporate these predictors with the Wi-Fi monitoring system, which is already deployed in field trials. Our second contribution was to characterize the effect of Wi-Fi on QoE in the wild. We apply our predictors on Wi-Fi metrics collected from 3,479 APs over one month. Our results were reassuring in that in the vast majority of samples Wi-Fi quality can sustain good QoE; still we found 21% of stations have a fraction of poor QoE samples above 25%. ISPs can use the fraction of poor QoE samples to identify stations with frequent poor QoE due to Wi-Fi. In particular, intermittent events are challenging to troubleshoot and require a long-term monitoring approach, as we propose here.

Acknowledgements

We thank Luc Gyselinck and Nick Godman for helping with the Wi-Fi testbed and deployment data analysis. This work was carried out at LINCS while Diego worked for Technicolor and benefited from the support of the European Community Seventh Framework Programme (FP7/2007-2013) no. 611001 (User-Centric Networking), the French ANR Project no. ANR-15-CE25-0013-01 (BottleNet), and NewNet@Paris, Cisco's Chair "Networks for the Future" at Telecom ParisTech (http://newnet.telecom-paristech.fr).

References

[1] I. Pefkianakis, H. Lundgren *et al.*, "Characterizing home wireless performance: The gateway view," in *Proc. IEEE INFOCOM*, 2015.

[2] S. Rayanchu, A. Patro, and S. Banerjee, "Airshark: detecting non-wifi rf devices using commodity wifi hardware," in *Proc. IMC*. ACM, 2011.

[3] W. K. Edwards and R. E. Grinter, "At home with ubiquitous computing: Seven challenges," in *International Conference on Ubiquitous Computing*. Springer Berlin Heidelberg, 2001.

[4] C. Pei, Y. Zhao *et al.*, "Wifi can be the weakestlink of round trip network latency in the wild," in *Proc. IEEE INFOCOM*, 2016.

[5] S. Sundaresan, N. Feamster, and R. Teixeira, "Home network or access link? locating last-mile downstream throughput bottlenecks," in *PAM*. Springer, 2016.

[6] S. Biswas, J. Bicket *et al.*, "Large-scale measurements of wireless network behavior," in *Proc. ACM SIGCOMM*, 2015.

[7] A. Patro, S. Govindan, and S. Banerjee, "Observing home wireless experience through wifi aps," in *Proc. international conference on Mobile computing & networking*. ACM, 2013.

[8] K. Lakshminarayanan, S. Seshan, and P. Steenkiste, "Understanding 802.11 performance in heterogeneous environments," in *Proc. SIGCOMM workshop on Home networks*. ACM, 2011.

[9] L. DiCioccio, R. Teixeira *et al.*, "Pinpointing home and access network delays and losses using wifi neighbors," UPMC, Tech. Rep., 2012.

[10] P. Kanuparthy, C. Dovrolis *et al.*, "Can user-level probing detect and diagnose common home-wlan pathologies," *ACM SIGCOMM CCR*, 2012.

[11] I. Syrigos, S. Keranidis *et al.*, "Enabling wireless lan troubleshooting," in *Proc. PAM*. Springer, 2015.

[12] R. Schatz, T. Hoßfeld *et al.*, "From packets to people: quality of experience as a new measurement challenge," in *Data traffic monitoring and analysis*. Springer, 2013.

[13] ITU-T, "Recommendation P.862 annex a: Reference implementations and conformance testing for ITU-T Recs P.862, P.862.1 and P.862.2," 2005.

[14] Xiph.org media, http://media.xiph.org/video/derf/.

[15] ITU-T, "Methods for subjective determination of transmission quality," Aug 1996.

[16] D. N. d. Hora, R. Teixeira *et al.*, "Technical report: Predicting the effect of home wi-fi quality on web qoe," Tech. Rep., available at: https://perso.telecom-paristech.fr/dneves/docs/techreport-homewifi.pdf.

[17] Z. Wang, L. Lu, and A. C. Bovik, "Video quality assessment based on structural distortion measurement," *Signal processing: Image communication*, vol. 19, no. 2, 2004.

[18] T. Zinner, O. Hohlfeld *et al.*, "Impact of frame rate and resolution on objective QoE metrics," in *Quality of Multimedia Experience (QoMEX), 2010 Second International Workshop on*. IEEE, 2010.

[19] ITU-T, "Recommendation P.862: Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs," 2001.

[20] ITU-T, "Recommendation G.107: The e-model: a computational model for use in transmission planning," 2015.

[21] L. Sun and E. C. Ifeachor, "Voice quality prediction models and their application in voip networks," *Transactions on multimedia*, 2006.

[22] M. Varvello, J. Blackburn *et al.*, "Eyeorg: A platform for crowdsourcing web quality of experience measurements," in *Proc. ACM CoNEXT*, 2016.

[23] ITU-T Recommendation G.1030, "Estimating end-to-end performance in ip networks for data applications," 2005.

[24] S. Egger, T. Hossfeld *et al.*, "Waiting times in quality of experience for web based services," in *QoMEX*. IEEE, 2012.

[25] D. Strohmeier, S. Egger *et al.*, "Web browsing," in *Quality of Experience*. Springer, 2014.

[26] A. Balachandran, V. Sekar *et al.*, "Developing a predictive model of quality of experience for internet video," in *ACM SIGCOMM CCR*, 2013.

[27] F. Wamser, P. Casas *et al.*, "Modeling the YouTube stack: From packets to quality of experience," *Computer Networks*, 2016.

[28] M. Katsarakis, R. Teixeira *et al.*, "Towards a Causal Analysis of Video QoE from Network and Application QoS," in *Proc. ACM SIGCOMM Internet-QoE workshop*, 2016.

[29] D. Joumblatt, J. Chandrashekar *et al.*, "Predicting user dissatisfaction with internet application performance at end-hosts," in *Proceedings of IEEE INFOCOM*, 2013.

[30] Z. Wang, A. C. Bovik *et al.*, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, 2004.

[31] T. Wang, A. Pervez, and H. Zou, "VQM-based QoS/QoE mapping for streaming video," in *Proc. IC-BNMT*. IEEE, 2010.

[32] P. Casas, M. Seufert, and R. Schatz, "YOUQMON: A system for online monitoring of YouTube QoE in operational 3G networks," *ACM SIGMETRICS Performance Evaluation Review*, 2013.

[33] G. Dimopoulos, I. Leontiadis *et al.*, "Measuring video QoE from encrypted traffic," in *Proc. IMC*. ACM, 2016.

[34] T. Spetebroot, S. Afra *et al.*, "From network-level measurements to expected quality of experience: the skype use case," in *M & N workshop*, 2015.

[35] M. Papadopouli, P. Charonyktakis *et al.*, "On user-centric modular QoE prediction for VoIP based on machine-learning algorithms," *IEEE TMC*, 2015.

[36] A. Chakraborty, S. Sanadhya *et al.*, "Exbox: Experience management middlebox for wireless networks." in *CoNEXT*, 2016.

[37] D. N. d. Hora, R. Teixeira *et al.*, "Predicting the effect of home Wi-Fi quality on Web QoE," in *Proc. ACM SIGCOMM Internet-QoE workshop*, 2016.

©IEEE 2018. This is the author's version of the work. This work has been submitted to the IEEE for possible publication. Copyright may be transferred without notice, after which this version may no longer be accessible. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.