Towards a Causal Analysis of Video QoE from Network and Application QoS

Michalis Katsarakis[1], Renata Teixeira[1], Maria Papadopouli[2,3], Vassilis Christophides[1,2]
[1]Inria, Paris, France
[2]Department of Computer Science, University of Crete, Heraklion, Greece
[3]Institute of Computer Science-FORTH, Heraklion, Crete, Greece
{michail.katsarakis, renata.teixeira, vassilis.christophides}@inria.fr, mgp@ics.forth.gr

ABSTRACT

The relationship between the user perceived Quality of Experience (QoE) with Internet applications and the Quality of Service (QoS) of the underlying network and applications is complex. Unveiling statistical relations between QoE and QoS can boost the prediction and diagnosis of QoE. In this paper, we shed light on the relationship between QoE and QoS for a popular application: YouTube video streaming. We conducted a controlled study where we asked users to rate their perceived quality of YouTube videos under different network conditions. During this experiments, we also captured network QoS and application QoS. We then analyze the resulting dataset with SES, a feature selection algorithm that identifies minimal-size, statistically-equivalent signatures with maximal predictive power for a target variable (e.g., QoE). We found that we can build optimal QoE predictors using a minimal signature of only three features from application or network QoS metrics compared to four when we consider features from both layers.

1. INTRODUCTION

With the proliferation of media delivery services (e.g., VoD, LiveTV) and devices (e.g., tablets, smartphones) more and more users share and consume video content in their everyday activities, for example for education or entertainment. A high *Quality of Service* (QoS) is essential for sustaining the revenue of service providers, carriers, and device manufactures. Yet, the perceived *Quality of Experience* (QoE) of users is far from perfect—e.g., videos that get stalled or that take a long time to load. Dissatisfied users may change Internet Service Providers (ISPs) or video streaming services. Hence, the incentives for measuring and improving QoE are high. Video streaming services can instrument the player to directly measure *application QoS* metrics, such as startup delay or buffering events. However, ISPs can only monitor *network QoS* metrics, such as throughput or delay. Mapping network and application QoS to QoE is challenging.

The literature is rife with studies that aim at mapping QoS to QoE. Some studies use *subjective* metrics of QoE to capture experience scores users give explicitly and then build models of QoS to QoE [1, 2, 3, 4]. Subjective QoE studies are costly in terms of required human effort. Moreover, it is often hard to obtain user feedback in all possible scenarios that may arise in practice. Another approach is to infer quality of experience implicitly from *user engagement* metrics (e.g., viewing time, abandonment ratio, number of visits) [5, 6, 7, 8]. Studies based on user engagement require an extremely large number of users to control for all possible variations of human viewing behaviors and other confounding factors. Instead of measuring QoE, some previous studies proposed quality models to assess application QoS (e.g., join time or buffered playtime) from the underlying network conditions (e.g., bandwidth, packet losses) [9, 4]. These studies work with the assumption that application QoS is a good proxy for QoE.

Clearly, different actors in the online video delivery chain (e.g., video streaming services, ISPs) have different incentives and means to measure and affect the user QoE. Uncovering statistically equivalent subsets of QoS metrics across and within levels provides actionable knowledge for building QoE predictors. To achieve this goal, we leverage recent advances on feature selection algorithms [10, 11] to exploit available experimental evidence of the joint probability distributions of QoE/QoS metrics. This type of statistical reasoning will enable us to determine *local causal relationships* between a target QoE variable, seen as *effect*, and multiple QoS metrics across or within levels, seen as *causes*. Such *data-driven analysis* is justified by the multiplicity of dependencies that exist between network or application QoS metrics as different adaptation mechanisms (e.g., *TCP congestion avoidance*, *HTTP bitrate adaptation*) are activated at each level in real life. Building optimal predictors based on (eventually several) probabilistically minimal subsets of features opens the way for a principled comparison of the predictors. To the best of our knowledge this is the first work in this direction that mines local

Internet-QoE, August 22-26, 2016, Florianopolis , Brazil
© 2016 ACM. ISBN 978-1-4503-4425-8/16/08…$15.00
DOI: http://dx.doi.org/10.1145/2940136.2940142

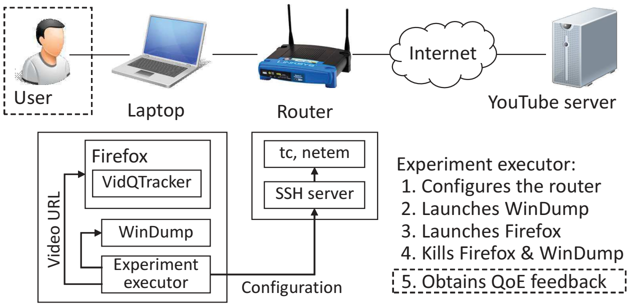


Figure 1: Testbed for collecting our ground-truth dataset. Dashed tasks require user participation.

causal relationships between concrete sets of features presented in Fig. 2.

The rest of the paper is organized as follows: Section 2 describes our experimental testbed and instrumentation software. Section 3 presents our experimental design in terms of network configuration scenarios and representative conditions of user experience that we tested with online users. In Section 4, we discuss the statistically equivalent QoS signatures extracted by our experimental data using the SES [11] feature selection method. Section 5 summarizes our work and future work plans.

2. EXPERIMENT SETUP

Our statistical analysis requires a ground-truth dataset that includes for each video-streaming session the underlying network and application QoS, annotated with the corresponding QoE score provided by the user. We present in this section how we generate this dataset for YouTube and detail in the next section the conducted user study for obtaining QoE annotations.

Testbed. We have set up a testbed for measuring the QoS of streaming YouTube videos while emulating various network conditions (Fig. 1). The testbed consists of a router and a laptop. The router (TL-WDR3600) runs the OpenWRT 15.05, a Linux-based operating system for embedded devices. The router connects to the Internet with a Gigabit Ethernet link. We use `tc` and `netem` on the router to limit the bandwidth of both the LAN and WAN network interfaces of the router as well as to introduce delays and packet losses. The `tc` and `netem` commands are applied on each direction (i.e., upstream and downstream) separately. The screen of the Windows 8.1 laptop (Dell Inspiron 17R SE 7720) is configured at a maximum brightness and resolution (1920 x 1080 pixels). The volume level is set at 24%, using the built-in speakers of the device. The laptop is configured with all network interfaces down, except the one connected to the router. We use the Firefox Web browser (v45.0.1) to stream YouTube videos. Our experimental setup also includes a non-controlled network part, namely, the path from the router to the YouTube server. This non-controlled part has a varying QoS and may introduce additional, unpredictable network impairments which are captured by our monitoring instrumentation.

Monitoring network QoS. We instrument the laptop to capture network packet traces with WinDump,[1] the

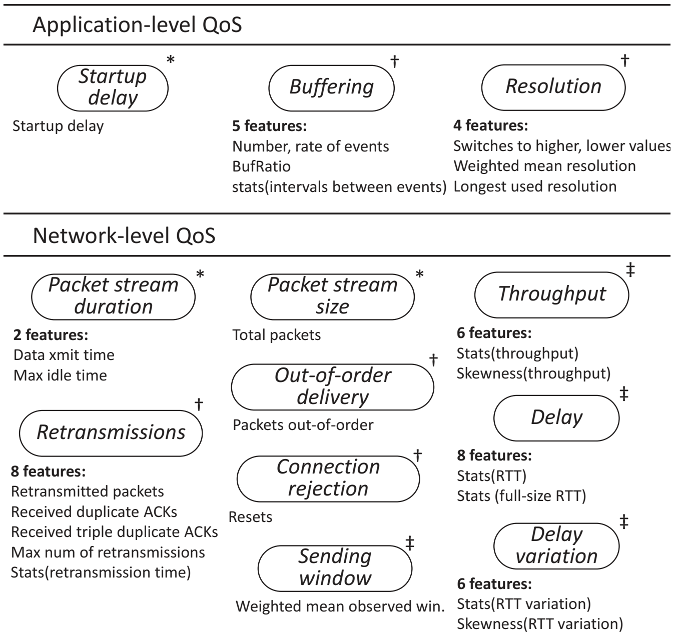


Figure 2: Overview of *QoS metrics* and their features (single-valued statistics). A QoS metric can be single-valued (*), event list (†), or time series (‡).

Windows version of tcpdump. We then extract network QoS metrics from the packet traces using `tcptrace`.[2] Note that we can only extract the RTT metrics of data packets towards the YouTube server.

Monitoring video QoS. We developed VidQTracker, a Firefox extension that monitors streaming video QoS and relies on the HTML5 video element and its API. VidQTracker logs the start and end of a video session together with a number of events that capture when the video starts loading, starts playing, and is terminated (with the reason for termination: playback completion, abandonment, or error), as well as the video duration. Additionally, buffering, pause, and off-screen events can be detected. We also capture changes in the video resolution. Information not available through events is polled from available variables or parsed from the parent HTML document (e.g., the number of parsed, decoded, presented, and painted video frames are sampled with sampling period of 1 second). The *video framerate* [12, 6] is a timeseries derived from sampling the rate at which video frames are painted on the screen. *Video framerate* samples obtained during the startup delay and buffering events are 0-valued. The lowest adaptive format of YouTube videos has reduced framerate, in addition to low resolution. By including in our dataset features about the *video framerate*, we may miss the indirect dependencies of *startup delay*, *buffering events*, and *video resolution* on the QoE.

From QoS metrics to QoS features. We compute a vector of network and application QoS features for every video session. These QoS features are essentially statistics (i.e., min, max, mean, median, standard deviation, and skewness) regarding network or applica-

[1] https://www.winpcap.org/windump/

[2] http://www.tcptrace.org/tcptrace-manual/manual/

tion QoS metrics. We then remove some features (e.g., that are redundant, or about fixed parameters of our controlled experiments). Fig. 2 presents 10 application-level and 34 network-level features that we consider in our study. Each network-level QoS feature is computed for both downstream and upstream. The notation ↑, ↓, and ↕ is used to indicate the direction from the laptop to YouTube and vise versa. Instead of the average throughput feature that `tcptrace` provides, we extract features from the raw throughput timeseries sampled with period of 1 sec. The grouping of the QoS features shown in Fig. 2 is for presentation purposes only and does not affect our analysis.

Experiment setting. During our experiments, we stream a set of videos under a set of network configurations. A *network configuration* is a combination of bandwidth, delay, and packet loss values for the router configuration, which we set using `tc` and `netem`. For each network configuration, we launch Firefox in private-browsing mode with the video URL as argument (Fig. 1). We capture the network traffic with WinDump and the application QoS with VidQTracker. The space of network conditions is large, but not all settings will trigger perceivable changes in video QoS metrics. For constant video QoS metrics, it is unlikely that the user QoE would vary dramatically. To reduce the number of configurations for testing, we deployed two experimental settings: The first without user participation, aiming to narrow down the number of network configurations, and the second with users who rate the QoE for video sessions under different conditions.

- *Without user participation*: We tested all combinations when we set bandwidth to 1, 5, 10, and 30 Mbps, delay to 0, 30, 100, 200, and 1000 ms, and loss to 0%, 0.1%, and 0.5%. The next section presents the results of this analysis, which we use to set the network configurations for the user study.

- *With user participation*: We employ a GUI to synchronize the user actions and the configuration of the experiment. When the user clicks the button to view a video, our script sets the network configuration and launches Firefox. When the JSON file appears in the filesystem or a maximum video session time has elapsed, the script kills Firefox and prompts the user to rate the quality of the video.

3. USER STUDY

This section first analyzes the outcome of our experiments without users to select the network configurations to use in the user study. Then, we describe our method for recruiting users and conducting the study. Finally, we present a brief characterization of the dataset resulting from this study.

Table 1: Adaptive formats of test videos

	Video 1			Video 2		
Resolution (pixels)	**Bitrate (bps)**		**Framerate (fps)**	**Bitrate (bps)**		**Framerate (fps)**
	MP4	**WEBM**		**MP4**	**WEBM**	
256x144	111620	117846	12	110316	133378	13
426x240	254106	266924	24	247749	276596	25
640x360	661263	489218	24	628814	520312	25
854x480	1239085	914945	24	1155270	975176	25
1280x720	2386422	1812522	24	2315971	1918178	25
1920x1080	4364609	3137856	24	4356456	3358321	25
2560x1440	–	–	–	10348977	11055642	25
3840x2160	–	–	–	22128661	22293096	25
Duration	2:33			2:29		
URL	https://youtu.be/ncvFAm4kYCo			https://youtu.be/31vUX88BE6E		

3.1 Videos and network configurations

We select two test videos for the study. We consider a relatively small number of video sessions to keep the number of tests that each user has to rate low. To avoid any bias due to the video content itself, we pick two similar videos—both action movie previews. Table 1 lists the two videos as well as the bitrates and frames per second for each video under different resolutions. Both videos are available on high resolution (i.e., 1080p, or higher), and as shown in the table, at each resolution level, they exhibit similar bitrates. Volunteers participate in two 20-minute rating sessions, where they rate the same video four times, considering also various delays (e.g., the startup delay and buffering events).

The experiments without user participation covered 60 combinations of network configurations. We tested each network configuration 10 times using Video 2 (Table 1), each time obtaining a vector of application-level QoS features. The video was chosen because of its availability on even higher-quality adaptive streaming formats (i.e., resolution of 2560x1440 and 3840x2160 px). Choosing three easily-understood features, namely the sum of buffering durations, the weighted mean resolution, and the startup delay, we ran the K-means clustering algorithm. We tried multiple values for the input parameter k that indicates the number of clusters. In particular, we tried each value of $k \in \{5, 6, ..., 12\}$ 100 times, each time randomly choosing the initial cluster centroid positions. The best clustering based on the silhouette method, reported 8 clusters. Then we traced back the network configurations that resulted in these 8 YouTube QoS conditions. Table 2 reports the centroid of each cluster and the most frequently observed configuration among the cluster elements. Within most clusters, all packet loss configuration values were present, and therefore we decided to ignore the packet loss configuration column. The laptop screen provided an upper bound on the video resolution, which did not increase over 1920x1080 when we tested 30-Mbps network configurations.

3.2 Recruitment and method

Table 2: Clusters of application QoS and network configuration that produced them.

Cluster		Configuration			Application QoS		
ID	**size**	**BW**	**delay**	**loss**	**Sum Buff. Dur.**	**WMean res.**	**Startup delay**
1	201	10 Mbps	30 ms	0%	0.0 sec	911 px	0.7 sec
2	103	10 Mbps	100 ms	0.1%	0.0 sec	951 px	2.3 sec
3	58	1 Mbps	100 ms	0.5%	0.0 sec	377 px	8.6 sec
4	27	5 Mbps	200 ms	0.5%	0.3 sec	734 px	3.8 sec
5	39	1 Mbps	30 ms	0.1%	0.8 sec	364 px	9.0 sec
6	129	10 Mbps	1 s	0%	12.6 sec	335 px	22.5 sec
7	15	1 Mbps	1 s	0.5%	25.3 sec	281 px	21.7 sec
8	28	5 Mbps	1 s	0.5%	27.6 sec	298 px	22.0 sec

Table 3: Network configurations and test videos.

Session 1			Session 2		
Bandwidth	**Delay**	**Video**	**Bandwidth**	**Delay**	**Video**
10 Mbps	100 ms	Video 1	5 Mbps	200 ms	Video 2
5 Mbps	1 s	Video 1	1 Mbps	1 s	Video 2
1 Mbps	100 ms	Video 1	10 Mbps	30 ms	Video 2
10 Mbps	1 s	Video 1	1 Mbps	30 ms	Video 2

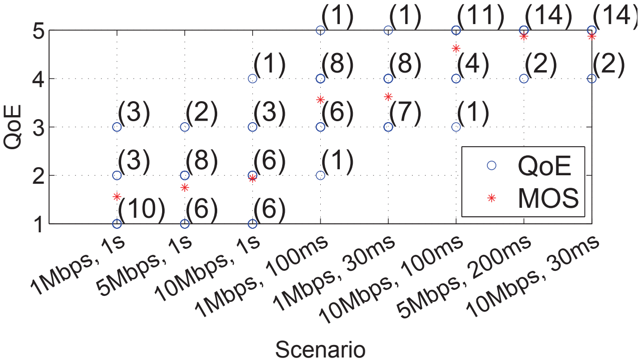


Figure 3: Distribution of QoE scores for each network scenario (coinciding scores are shown in parentheses).

We recruited sixteen volunteers among our colleagues at LINCS (mostly masters and doctoral students in computer science), excluding the authors of this paper. The study followed the guidelines for the subjective assessment of video quality of Internet video [13], subject to few modifications pertaining to HTTP adaptive video streaming, and the minimization of the user participation time. Each user attended two experiment sessions of 20 minutes each. During a session, a user watches one video (source stimulus) streamed four times, under different network configurations (as presented in Table 3). The study did not include any training session for the demonstration of the range and type of impairments to be assessed.

3.3 Dataset characteristics

We analyze the distribution of QoE scores on each scenario (Fig. 3). Fig. 4 (a–d) presents the measured application QoS features and Fig. 4 (e–h)) the underlying network QoS features. We observe no significant variations in the scores users provided for each tested scenario. Interestingly, users evaluated better the scenario with 5 Mbps bandwidth and 200 ms delay, compared to the one with 10 Mbps bandwidth and 100 ms delay. These two scenarios are the first tested in each rating session (Table 3). It seams that the “10 Mbps bandwidth and 100 ms delay” scenario suffers from the user score calibration effect since it was the first scenario tested in both sessions. The effect of this discrepancy on our statistical analysis is not important.

There are QoS variations within the video sessions performed under the same scenario (e.g., scenarios with 1 Mbps bandwidth and 30 or 100 ms delay exhibit high variance of startup delay). These variations can be attributed to the path from the router to the YouTube server. We observe that scenarios with a network delay of 1 sec have a startup delay of 20 sec or more and are the only ones that suffer from buffering events. Scenarios with 1 sec delay also had a low resolution, with some exceptions at the 10 Mbps scenario.

The mean throughput↓ is notably lower than the bandwidth configuration of each scenario (Fig. 4 (f)). On the other hand, the max throughput↓ (Fig. 4 (f)) is usually very close to the bandwidth configuration, except from the scenarios with 1 sec delay. Between the downloading of two successive chucks, a long period without any downloaded may occur[14], which reduces the mean throughput↓. In the case of high bandwidth (e.g., 5 or 10 Mbps) and 1 sec delay, both the mean and the max throughput↓ are lower than the configured bandwidth due to the low TCP congestion window.

4. QoS-QoE CAUSAL RELATIONS

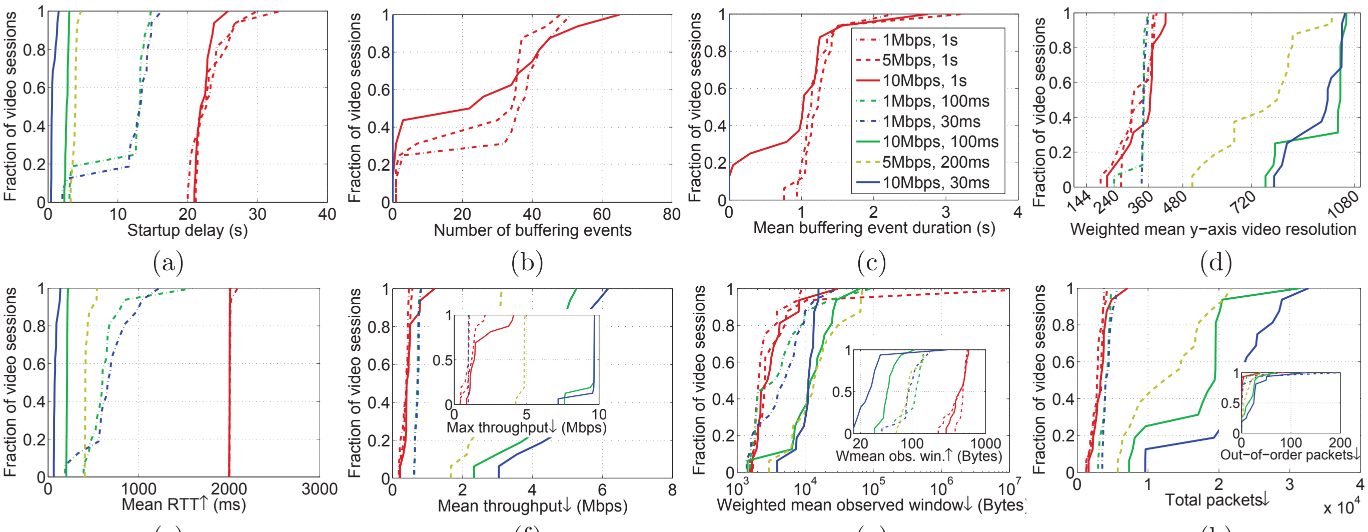
We rely on the SES [11] method to identify multiple minimal-size sets of QoS features with maximal predictive power for the target QoE. SES accepts as input a dataset D with network and application QoS features annotated with user QoE, seen as random variables. It reports a number of variable sets $Q_i, i = 1...n$ such that each set Q_i contains variables that are ‘equivalent’ to each other, w.r.t. a target variable T, such as the QoE. One can then build a predictive signature by choosing one variable from each set Q_i. Two signatures A and B contain equivalent information about T if and only if the following conditions hold: $T \not\perp A$, $T \not\perp B$, $T \perp A|B$ and $T \perp B|A$. $X \perp Y|Z$ denotes that two sets of variables X and Y are conditionally independent given a set of variables Z. Conditional dependence is defined as absence of conditional independence and denoted as $X \not\perp Y|Z$. Two variable sets X and Y are conditionally independent given Z if and only if $P(X \cap Y|Z) = P(X|Z) * P(Y|Z)$.

An *ordinal logistic regression test* has been employed for the assessment of conditional independences which is suitable when the target is an ordinal categorical variable, such as QoE scores, and the set of predictor variables is mixed (i.e., both continuous and categorical), such as QoS metrics. Using this test, we compute a p-value for the null hypothesis that a feature x is independent from QoE given a conditioning set cs. When the p-value is lower than a significant threshold $\alpha = 0.05$, we reject the null hypothesis and consider x and QoE conditionally dependent, given the conditioning set cs.

Initially, the algorithm creates an empty set S of selected variables, all variables are considered for inclusion in S, and each variable is considered equivalent only to itself ($Q_i \leftarrow i$). During the main loop the algorithm alternatively attempts to (a) include in S the variable maximally associated[3] with T conditioned on any possible subset of the variables already selected and (b) exclude from S any variable x that is not any more associated with T given any subset Z of other variables in S. Once a variable x is excluded from S, it cannot be inserted any more. However, before eliminating it, the algorithm tries to identify any variable y in Z that is equivalent to x, by verifying whether $y \not\perp T|Z'$ where $Z' \leftarrow (Z \cup \{x\}) \setminus \{y\}$. If such a variable exists, the list of x-equivalent variables Q_x is added to Q_y. Finally, all equivalence sets $Qi, i \in S$, are returned as output.

We ran SES 3 times, providing as input dataset a) all QoS features, b) only network-, and c) only application-level QoS features. Fig. 5 presents the reported sets of equivalent QoS features Q_i annotated with the respective QoS metrics from which they were derived. QoS metrics are represented in *italics* in both the text and figures to distinguish them from QoS features. All fea-

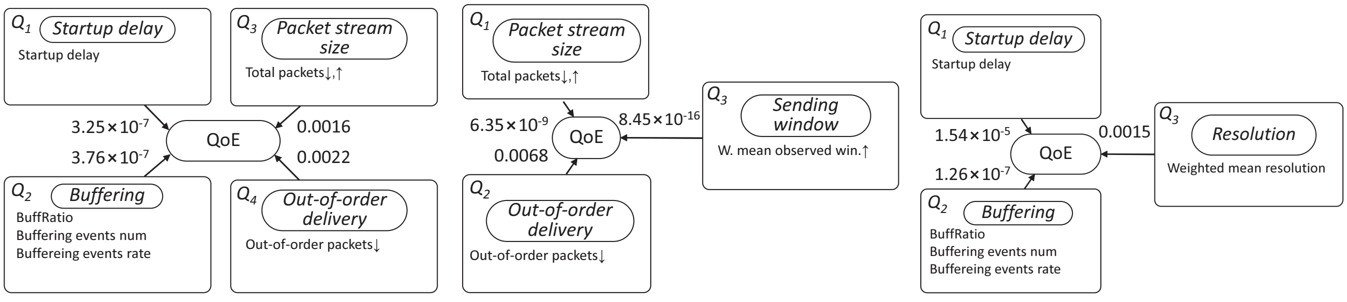
[3] Maximal association stands for statistical dependence with minimal p-value.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 4: Application-level (a–d), and network-level(e–h) QoS features, grouped by network scenario. All sub-figures share the legend of (c). Entries in the legend indicate the bandwidth limitation and introduced delay parameters of the router configuration.



(a) (b) (c)

Figure 5: Multiple equivalent QoS-feature signatures for the prediction of QoE, when considering a) application and network, b) only network, and c) only application QoS features.

tures belonging to a set Q_i are equivalent with each other for the prediction of the target QoE. A directed edge from an equivalence set Q_i to the target T (i.e., QoE) represents the conditional dependence between the variables in Q_i and T (i.e., $V \not\perp T, \forall V \in Q_i$). The weight assigned to a directed edge is the p-value of the conditional independence test between the first feature of the specific equivalence set Q_i and the target T, given the first element of every other equivalence set. The lower the p-value, the stronger the association between the first feature of the specific equivalence set Q_i and the target T.

As expected from previous studies, the features about *buffering events* would have the strongest association with QoE, and that features about *startup delay* and *resolution* will be present in the signatures, too, albeit with weaker associations. This expectation is confirmed when only application-level QoS features are available (Fig. 5 (c)). When all features are available (Fig. 5 (a)), the minimal-size signature contains 4 features related to both application and network QoS metrics: a) the startup delay, b) one feature about buffering events, c) one feature about the packet stream size, and d) the out-of-order packets↓ feature. As application QoS metrics are directly perceived by the users, SES successfully found that the p-values for the conditional independence tests about the startup delay and the BuffRatio are 3 orders of magnitude lower that the ones regarding the total packets↓ and the out-of-order packets↓.

When features from only one QoS level (e.g., network, or application) are available, SES consistently selects the features of Fig. 5 (a) that are still available, augmented by additional features from the available QoS level. More precisely, for QoS features only from the application level (Fig. 5 (c)), SES outputs the same equivalence sets regarding *startup delay* and *buffering* evens, augmented with the "weighted mean resolution". For QoS features only from the network level (Fig. 5 (b)), SES retains the equivalence sets about the *packet stream size* and *out-of-order delivery* in addition to the feature "weighted mean observed window↑".

We argue that the total packets↓ is a possible network-level proxy for the video resolution—although not equivalent. The feature total packets↓ is influenced by the amount of data that the application layer transfers, as well as by a number of network-level factors (e.g., retransmissions, and window probe packets). The different formats of the videos come with different bitrates (Table 1). Using a higher-resolution format of the video requires more data and more packets to be sent to the client. When all features are available, SES finds that both the total packets↓ and the weighted mean resolution are statistically dependent with QoE (p-values are 5.63×10^{-25} and 2.15×10^{-24}) given the empty set. Additionally, SES finds that the total packets↓ is statistically dependent with QoE given the weighted mean resolution (p-value is 0.0144), but the weighted mean resolution is independent with QoE given the total packets↓ (p-value is 0.0678). This is the reason why SES reports the weighted mean resolution when the packet stream size is not available. The fact that the total packets↓ remains dependent with QoE when conditioning with the weighted mean resolution prohibits SES to include these two features into the same equivalence set Q_i.

Interestingly, the feature out-of-order packets↓ appears both in Fig. 5 (a) and Fig. 5 (b). An out-of-order packet can be caused either by a retransmission, or by in-network reordering. In the first case, the sender infers that a packet has been lost and retransmits the packet. The sequence number of the retransmitted packet is smaller than previously observed packets of the TCP connection, and hence it is considered "out-of-order". In-network reordering can occur because of parallelism within a router, a route change, or if the network itself creates a duplicate copy of the packet [15]. The feature out-of-order packets↓ is conditionally dependent with QoE given the empty set (p-value is 6.15×10^{-8}) and SES does not find any conditioning set that renders out-of-order packets↓ independent with QoE. This can be attribute to the fact that the various network scenarios we tested produce the same stochastic ordering for both the total packets↓ and the out-of-order packets↓ (Fig. 4 (h)). We speculate that the worse the network conditions are, the higher the probability a packet is delivered out-of-order. At the same time, transmitting more packets increases the number of packets delivered out-of-order.

Finally, when application QoS features are not available, SES uses the weighted mean observed window↑ instead of the startup delay and a feature regarding *buffering* (Fig. 5 (b)). The weighted mean observed window↑ takes higher values in video sessions performed under worse network scenarios (Fig. 4 (g)). The scenarios with the highest weighted mean observed window↑ are the ones with 1 sec. delay. We believe that the congestion window of the upstream packet flow increases to higher values due to duplicate ACKs being sent to the YouTube server to trigger fast retransmissions.

5. SUMMARY

In this work we exploited an original framework for mining causal relationships among QoE and various QoS metrics at network and application level. In particular, we have analysed QoE scores provided by a set of users for YouTube video streaming applications under different network conditions. This work is the first step towards our ambition to assess QoE directly from network QoS metrics obtained via passive measurements of real traffic generated by online users. We will rely on the extracted minimal QoE/QoS signatures to build real-time predictors and compare their accuracy when using only network, only application or both QoS metrics. Last but not least, we plan to extend our experimental setting for other online applications such as teleconferencing services.

Acknowledgement

We thank the students at the LINCS, who participated in our experiments. This work was supported by the EC Seventh Framework Programme (FP7/2007-2013) no. 611001 (User-Centric Networking) and the French ANR Project no. ANR-15-CE25-0013-01 (BottleNet).

6. REFERENCES

[1] R. K. Mok, E. W. Chan, and R. K. Chang, "Measuring the quality of experience of http video streaming," in *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*. IEEE, 2011.

[2] R. K. Mok, E. W. Chan, X. Luo, and R. K. Chang, "Inferring the qoe of http video streaming from user-viewing activities," in *Proceedings of the first ACM SIGCOMM workshop on Measurements up the stack*. ACM, 2011.

[3] T. Hoßfeld, R. Schatz, E. Biersack, and L. Plissonneau, "Internet video delivery in youtube: from traffic measurements to quality of experience," in *Data Traffic Monitoring and Analysis*. Springer, 2013.

[4] F. Wamser, M. Seufert, P. Casas, R. Irmer, P. Tran-Gia, and R. Schatz, "Yomoapp: A tool for analyzing qoe of youtube http adaptive streaming in mobile networks," in *Networks and Communications (EuCNC), 2015 European Conference on*. IEEE, 2015.

[5] A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, and H. Zhang, "Developing a predictive model of quality of experience for internet video," in *ACM SIGCOMM Computer Communication Review*. ACM, 2013.

[6] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. Joseph, A. Ganjam, J. Zhan, and H. Zhang, "Understanding the impact of video quality on user engagement," *ACM SIGCOMM Computer Communication Review*, 2011.

[7] S. S. Krishnan and R. K. Sitaraman, "Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs," *Networking, IEEE/ACM Transactions on*, 2013.

[8] Q. A. Chen, H. Luo, S. Rosen, Z. M. Mao, K. Iyer, J. Hui, K. Sontineni, and K. Lau, "Qoe doctor: Diagnosing mobile app qoe with automated ui control and cross-layer analysis," in *Proceedings of the 2014 Conference on Internet Measurement Conference*. ACM, 2014.

[9] B. Staehle, M. Hirth, R. Pries, F. Wamser, and D. Staehle, "Yomo: a youtube application comfort monitoring tool," *New Dimensions in the Assessment and Support of Quality of Experience for Multimedia Applications, Tampere, Finland*, 2010.

[10] I. Tsamardinos, L. E. Brown, and C. F. Aliferis, "The max-min hill-climbing bayesian network structure learning algorithm," *Machine learning*, 2006.

[11] I. Tsamardinos, V. Lagani, and D. Pappas, "Discovering multiple, equivalent biomarker signatures," in *7th Conference of the Hellenic Society for Computational Biology and Bioinformatics (HSCBB12)*, 2012.

[12] Q. Huynh-Thu and M. Ghanbari, "Temporal aspect of perceived quality in mobile video broadcasting," *Broadcasting, IEEE Transactions on*, vol. 54, no. 3, pp. 641–651, 2008.

[13] I.-T. Rec, "P. 913, "methods for the subjective assessment of video quality, audio quality and audiovisual quality of internet video and distribution quality television in any environment"," *Int. Telecommun. Union, Geneva, Switzerland*, 2014.

[14] F. Wamser, P. Casas, M. Seufert, C. Moldovan, P. Tran-Gia, and T. Hossfeld, "Modeling the youtube stack: From packets to quality of experience," *Computer Networks*, 2016.

[15] S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, and D. Towsley, "Measurement and classification of out-of-sequence packets in a tier-1 ip backbone," *IEEE/ACM Transactions on Networking (ToN)*, 2007.