High Complexity and Bad Quality? Efficiency Assessment for Video QoE Prediction Approaches

Frank Loh[1], Gülnaziye Bingöl[2], Reza Farahani[3], Andrea Pimpinella[4], Radu Prodan[3], Luigi Atzori[2], Tobias Hoßfeld[1]
[1] *University of Würzburg, Institute of Computer Science, Würzburg, Germany*
[2] *DIEE, University of Cagliari, Italy and CNIT, University of Cagliari, Italy*
[3] *Institute of Information Technology, Alpen-Adria-Universität Klagenfurt, Austria*
[4] *DIGIP, University of Bergamo, Dalmine (BG), Italy*
Contact: frank.loh@uni-wuerzburg.de

***Abstract*—Video streaming has dominated Internet traffic, pushing network providers to ensure high-quality services to avoid customer churn. However, predicting streaming quality is challenging due to traffic encryption, requiring extensive network monitoring. While several prediction approaches have been studied, they often overlook resource and energy demands. To address this, we analyze existing methods, quantifying monitoring efficiency to predict video quality degradation. Finally, we highlight significant differences in efficiency, driven by data requirements and the prediction approach, offering insights for providers to select a suitable method for their needs.**

***Index Terms*—Video Streaming; Network Monitoring; QoE Prediction; Resource Demands.**

I. Introduction

The surge in online activities, multimedia consumption, and social media engagement has significantly increased network traffic. Internet traffic exceeded 100 billion gigabytes in 2022, with video streaming accounting for 65 % of general and over 67 % of global mobile traffic [1]. For end users, minimal initial delay and consistent video quality are crucial for a high *Quality of Experience* (QoE). Service providers must ensure robust network quality to meet *Service Level Agreements* (SLA) and deliver optimal streaming quality. Identifying factors that degrade network quality through accurate measurements is essential to develop tailored monitoring solutions, reduce costs, and efficiently allocate resources. However, increasing traffic complexity requires sophisticated monitoring systems, challenging efforts to reduce energy and resource demands.

Currently, various methods to predict key QoE degradation factors like re-buffering, playback quality, and initial delays are available [2], [3], [4]. These approaches examine each packet in both uplink and downlink, leading to high requirements for data analysis and increased hardware and energy consumption. Other methods use partial data, such as uplink requests only [5], [6], raising questions about their efficiency in balancing resource demands with prediction quality.

In this paper, we identify key concepts to detect QoE degradation factors during video streaming and assess the resource and energy demands of such approaches. We analyze the number of packets and data volume needed for prediction and find that lightweight methods using partial data maintain high prediction quality while reducing monitoring efficiency. Our contribution is threefold: *(i)* We examine monitoring effort to identify efficiency of well-studied QoE degradation factor prediction approaches using a large-scale dataset. *(ii)* We differentiate the effort based on prediction methods, required data, and video resolution. *(iii)* We discuss trade-offs between effort and prediction accuracy, offering guidelines to help providers in choosing the appropriate approach. To this end, the paper identifies and answers the following research questions (*RQs*):

RQ1: How much data need to be monitored for QoE degradation factor prediction approaches from the literature?

RQ2: What is the most appropriate approach type with respect to monitoring and processing effort?

RQ3: Are there trade-offs between monitoring and processing effort, approach complexity, and prediction accuracy?

In the remainder, Sec. II summarizes background and related literature and in Sec. III, we present our methodology. We then perform an evaluation in Sec. IV and conclude in Sec. V.

II. Background and Related Work

First, we provide background and related work on video streaming and streaming monitoring.

A. Video Streaming and Streaming Monitoring

Video streaming involves downloading and playing audio and video content simultaneously. When downlink bandwidth is insufficient, video quality may drop or interrupt, impairing the user's QoE. Key QoE degradation factors include initial delay, video quality, frequency of quality switches, and the number and duration of stallings [9]. Thus, accurate monitoring of these factors is essential. Note, our focus is on network-detectable QoE degradation, excluding factors like video fragments or blurriness, covered in other studies [10].

1) Streaming Monitoring and Quality Prediction: For comprehensive streaming monitoring, it is essential to detect all relevant data without influencing the performance of data transmission or affecting QoE. The following steps are typical for a high-quality monitoring procedure [5], [11].

978-3-903176-66-9 ©2024 IFIP

Table I: Overview of select related work.

Reference		Approach	Prediction goal	Data	# Features	Window size	Quality
Wassermann'20	[2]	random forest	QoE	packets	20 – 208	1 s	up to 86 % – 91 % recall and precision
Orsolic'20	[7]	tree-based	QoE/KPI	packets	10 – 228	1 s – 20 s	up to 90 % precision and recall
Shen'20	[3]	CNN	initial delay, resolution, stalling	packets	16	10 s	> 90 % precision and recall
Gutterman'20	[6]	random forest	buffer warning, video state, video quality	requests	127	20 windows of 10 s	video dependent: 67 % – 92 %
Loh'21	[5]	random forest, NN, LSTM	QoE	requests	10	10 requests	78 % – 96 %
Madanapalli'21	[4]	random forest	QoE	requests	11 every 0.5 s	5 s – 30 s	88 % – 93 % recall and accuracy
Loh'23	[8]	uplink model	quality change, stalling, buffering issues	requests	3 – 10	10 requests	up to 60 % – 90 % precision and recall

Flow Monitoring: First, all flows associated with video streams are identified, usually using the four tuples of source and destination IP addresses and ports to distinguish flows and identify the correct sender and receiver. Ports can sometimes help separate video from audio content or different resolutions [8], but this can be error-prone and requires verification [8]. Additionally, actual video traffic must be distinguished from other streaming platform traffic, such as ads or video suggestions [11], [12]. Since video streams dominate a session's flows [13], various methods for identifying video flows are well-established [14], [15].

Network Traffic Post-Processing: After monitoring, extensive post-processing is needed, including data extraction, labeling, and calculating additional parameters and features [11], [16]. Key data includes IP addresses, port information, protocols used, and traffic size in both uplink and downlink directions [5]. This data is often aggregated into time windows [2] or requests [5], [6]. Other relevant parameters include packet inter-arrival times, inter-request times, and total uplink and downlink volume within the aggregation period [13].

Quality of Experience Prediction: Finally, model- and ML-based approaches are commonly used to predict QoE degradation factors. Model-based methods describe the streaming session using incoming data [17] or inter-request times to assess data sufficiency [5]. ML-based methods require feature selection and training to predict QoE degradation factors, with feature sets ranging from around ten [5] to over 200 [2]. The complexity of approaches varies, from simple tree-based models [5], [2] to neural networks [2] and LSTM solutions [5]. Once all factors are predicted, expected QoE is determined via QoE models, as detailed in [18].

B. Related Work

Nowadays, QoE prediction approaches can be categorized into real-time or non-real-time methods, with or without ML. The first non-real-time session modeling approach was introduced by Dimopoulos in 2016 [19], while real-time methods, like those published by Mazhar et al. in 2018 [20], are preferred for timely quality degradation identification. Table I summarizes key literature. Tree-based methods use fewer than 20 features from full packet traces, achieving over 90 % precision and recall for QoE degradation factors [2], [7]. CNN-based approaches use 16 features for similar accuracy [3], while request-based methods, like [6], rely on 127 features. Newer methods use fewer features with comparable quality for on-demand [8], [4] and live streaming [12]. Session models based on request counts also reduce data needs [17]. However, a comprehensive evaluation of the monitoring effort for these approaches is lacking, which this work aims to address.

III. Methodology

This section introduces the dataset and methodology used to quantify the monitoring and processing efficiency for QoE degradation prediction approaches.

A. Dataset Summary

We analyze a large-scale dataset [11] featuring 14,057 video runs and over 1,000 hours of playtime, measured using the native YouTube app. The dataset covers more than 80 network scenarios and 170 bandwidth settings, providing a wide range of realistic and constructed network conditions [11]. It includes video resolutions from 144p to 1080p, using both TCP and UDP protocols. This allows for a general evaluation across typical video streaming scenarios on YouTube, one of the largest contributor to global streaming traffic.

B. Influencing Factors on Monitoring and Processing Effort

To quantify efficiency of QoE degradation factor prediction approaches, we analyze the monitoring and data processing, discussing the prediction approach at the end of this paper.

1) Monitoring Effort: A key factor influencing QoE prediction is the data volume. Since video flows must be separated from cross-traffic and platform-specific data like ads or recommendations regardless of the method used, this separation is not considered here.

Monitored Packets: We quantify the number of transmitted packets that must be monitored during video streaming. Given varying video lengths in our dataset, we calculate the number of packets per second as a CDF in Figure 1. The x-axis shows the average packets per second per video, with different lines representing packet types: black for all packets, brown for downlink, orange for uplink, and yellow for uplink packets with payload. As expected, more packets are monitored when downlink or both uplink and downlink are considered. On average, all packets total 173 per second, downlink alone 114, and uplink 58. For uplink payload packets used in uplink-based prediction approaches (e.g., [5], [6]), only 0.36 packets per second need monitoring.

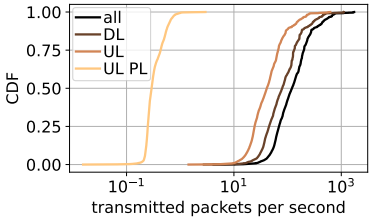

Figure 1: Transmitted packets per second when streaming video.

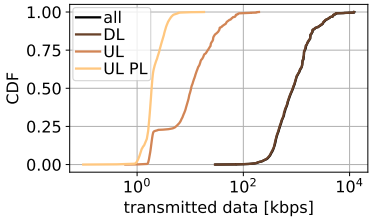

Figure 2: Transmitted data in kbps when streaming video.

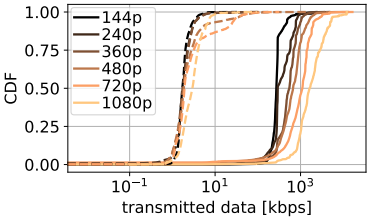

Figure 3: Data per resolution (solid: downlink, dashed: uplink payload).

Monitored Bytes: Considering monitored kilobyte per second in Figure 2, the difference in average traffic between all packets and downlink packets is minimal at 1.25 Mbps, as the downlink carries most of the video data. In contrast, uplink traffic averages 16.8 Kbps overall and 2.2 Kbps for packets containing actual payload. Uplink traffic shows a multi-modal distribution, with 25 % slightly exceeding the payload data in yellow, while the remaining 75 % requires significantly more.

Figure 3 finally plots the data needed to stream videos by resolution. The x-axis shows kilobytes per second on a logarithmic scale, with solid lines for downlink and dashed lines for uplink. More data is needed for downlink than uplink, with higher resolutions increasing the downlink data requirement. Specifically, 144p needs 60 % less data than 360p, 720p requires twice as much, and 1080p needs four times more. Uplink data is less affected by resolution changes, with 144p needing slightly more and 1080p and about 50 % more than 360p. The packet count follows a similar pattern. A more detailed analysis is omitted because of space reasons. To this end, we can answer the first research question (RQ1) as follows: *The monitoring effort depends on the data type. While we need to monitor more than 100 packets per second if the full packet trace or downlink packets are considered, less than one packet per second is required using uplink packets containing payload only. A similar behavior is visible if we consider the required amount of data in byte by comparing uplink and downlink. Furthermore, the monitoring effort is heavily increasing with resolutions considering downlink data, in contrast to only a slight increase using uplink. As a result, for larger resolutions streamed in the future, uplink-based monitoring is suggested because it scales better.*

2) Data Processing: After monitoring, it is essential to post-process the data and extract the required features.

Processing based on Prediction Approach: Different QoE degradation factor prediction approaches use either real-time data or time windows with historical data. Real-time or simple calculation methods require minimal processing overhead. In contrast, approaches that consider specific time frames or windows around the current playback position need to retain data for a certain period, which increases the data volume and the number of packets or features that must be processed.

Raw Data Post-Processing: The raw data post-processing steps determines which and how much data must be kept, which information should be extracted from each packet, and which additional post-processing operations are required. Typically parameters such as packet arrival times, source and destination IP addresses and ports, the protocol, and the payload size are kept and extracted as features [5], [8], [2].

Processing Effort Calculation: To determine the processing effort, we can categorize features into single event features and time window-based features. Single event features, such as packet arrivals or sending timestamps, require minimal effort, as only one event is monitored at a time. In contrast, time window-based features depend on the window size and packet count within it. This requires monitoring all packets and updating variables for each new packet, increasing processing overhead based on the number of packet types per time window. Additionally, computing features like sums, averages, or standard deviations involves moderate effort. Advanced features, such as distributions or results from iterative algorithms, require more extensive processing.

C. Efficiency for QoE Influencing Factor Prediction

Finally, we can quantify the monitoring and processing effort required to predict key QoE degradation factors using approaches from the literature. Based on the aforementioned considerations and the different approaches existing in the literature, summarized in Section II-B and Table I, we can differentiate the required effort according to four categories, data input, data usage duration, required features, and prediction approach type. In the following, we briefly introduce the categories that are evaluated later in the paper with regard to the potentially required effort.

Data Input: First, the required data input is essential to quantify the efficiency for the prediction approach. Typical prediction approaches distinguish between uplink traffic [6], [8] and full packet traces [2], [7] as input. We have already seen large differences in the amount of data that need to be monitored between both approaches in Section III-B. Consequently, this is not evaluated in detail again in the evaluation but is included in the assessment of the other categories, as the number of required data commonly influences the data usage, the number of features, or the general approach complexity.

Data Usage Duration: Besides the amount of data, the duration, specific data are required for analysis is usually of key interest. If data need to be stored longer in the memory, more resources are required for a longer period of time. This leads to a higher resource and energy demand. Furthermore, usually more data are then required to calculate essential

features. The duration of data usage is determined by the prediction approach, which means whether QoE degradation factors are directly predicted based on the data arriving at the monitoring instance or if a time window is required. Consequently, we can study direct approaches, as suggested in [5], [8], time window approaches with small time windows of 1 s as investigated in [2], [7], and up to time windows of 20 s [7], or 30 s [4]. Furthermore, several time windows of a specific duration can be kept, as done in [6], keeping 20 windows of 10 s length.

Required Features: The complexity to assess the required features can be determined by the count of raw features and the effort to calculate the features. As summarized in Table I, the count of features ranges from three to 228 in the literature. The effort for feature calculation is dependent on the input data for the calculation and the calculation complexity itself, as discussed in the processing effort calculation above.

Prediction Approach Type: Finally, the complexity of the approach must be considered. We can distinguish between a simple session reconstruction model without complex training or testing procedures [8], classical ML approaches including tree-based solutions [2], [5], and more complex solutions using neural networks [3]. As the required resources for the different solutions highly depend on, among others, the required data, hardware, model design, and training duration, we will briefly discuss these influences at the end of this paper.

IV. Evaluation

We could identify different settings that impact the effort for QoE degradation factor prediction approaches. These settings are evaluated in the following according to their categories defined above. As we already identified large differences using only uplink data or full packet traces, the influence of the input on the effort with a different data usage duration and varying effort for feature assessment is discussed in the following.

A. Data Usage Duration

The window size for the prediction is essential considering the data usage duration. Monitored data are, for example, held for this duration in the memory to calculate features for the complete window size. We compare this effort to calculate a single feature in Figure 4 as CDF for all videos from our dataset to demonstrate the effort as a result of using different window sizes and input data. The x-axis represents the average memory size needed to hold the required data, and the colors show different approaches. The black line is the request-based approach using a minimal window size of ten requests proposed in [5]. The brown line shows the effort for a request-based approach with a maximum window size of 20 windows and 10 s duration [6]. The window size has a significant influence on the average required memory, although both uplink-based approaches use the same data. Although the approach illustrated in black has an average required memory of 7.85 KB, it is more than 55 KB on average for the maximal request-based approach in brown and a seven-fold increase. In contrast, if a full packet trace is monitored and a window of 1 s is used, as presented by Orsolic in [7], a further three-fold increase in the required memory on average is visible, shown in orange. Finally, if we consider a 20 s window from the same approach, 20 times more memory is required. Consequently, we see that the selection of the approach and the required amount of data have a significant effect on the effort. Similarly, we can compare the average number of active packets, that is the average number of packets that must be kept for feature calculation, per video for different approaches. Obviously, it is highly dependent on the input data and the window size. We plot the result in Figure 5 as CDF and keep the colors as above. The x-axis shows the average number of active packets. Especially when we compare the usage of only uplink requests with minimal data requirements in black and the full packet trace with a window size of 20 s from Orsolic [7] in yellow, a more than 1,000-fold increase is visible.

B. Required Features

As the raw comparison of the number of features is trivial and available in Table I, we include the amount of required data in our assessment. We plot the effort increase for different approaches in Figure 6 on the x-axis compared to the average effort for the most lightweight approach using only requests and a minimal window of ten requests from the previous consideration (black line Figure 5). The colors for the other approaches are kept as above, the solid lines show the increase in number of operations that have to be performed in comparison to the most lightweight approach. Each operation is assumed to generate the same effort while n operations represent the calculation of one feature using n packets. Thus, this effort changes with more packets or features. The dashed lines indicate the required memory as described in the data usage section above and is influenced by the required data size and the time the data need to be kept in the memory. Here, we assume that all features can be calculated at once and that no additional memory is required to calculate multiple features. We see an effort increase with regard to the number of operations for all approaches against the minimum. However, although the orange line shows the effort increase using a full packet trace as data source, it performs better than the brown line showing the request-based approach using 127 features [6]. More packets are required using a full packet trace but only a window of 1 s is used with 10 features. The request-based approach [6] requires 127 features in the worst case with 20 10 s windows. We see a contrary behavior if we compare the dashed brown and orange line which show the required resources that are influenced by the amount of data that must be kept. Furthermore, for all approaches, the full packet trace approach from Orsolic [7] using 228 features and a 20 s window as worst case performs the worst, shown in yellow. To this end, we can answer our second research question (RQ2), as follows: *The effort required to predict key QoE degradation factors increases with more data, features, or longer time windows. However, the most lightweight approach with regard to the required number of data that need to be monitored does not always show the least effort. Consequently,*

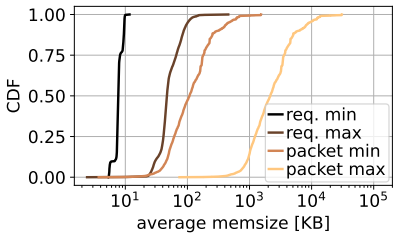

Figure 4: Memory for processing.

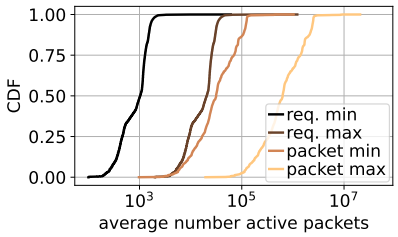

Figure 5: Average number used packets.

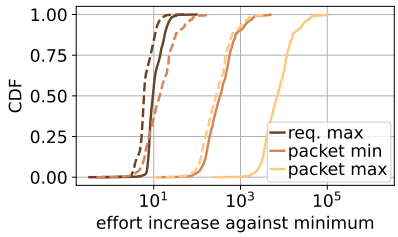

Figure 6: Effort comparison.

the requirement of additional features must be studied to reduce effort and thus, resource and energy consumption.

C. Discussion: Comprehensive View on Approaches

We can identify two key results with respect to the monitoring and processing effort for the approaches from literature. First, using partial data requires less effort for monitoring and feature assessment, and available resources could be saved for the actual prediction model. If a similar complex model is operated, the total pipeline from monitoring to actual prediction is cheaper on partial data than on the full packet trace if the number of features and the window size are not excessively increased. Second, there are models available in the literature that use only requests and operate a very simple model (e.g., [8]). If the main focus is the required effort, and since we assume that a simple heuristic requires the least effort compared to ML-based solutions, such an approach can be suggested. Most ML solutions require more input data and thus, more effort to obtain prediction results. However, simple random forest approaches achieve acceptable prediction accuracy with moderate effort. Finally, we can answer our third research question (RQ3): *To keep an approach lightweight when many packets need to be monitored, it is essential to limit the number of features. This holds in particular true if much data is required to calculate single features using long time windows. Consequently, we suggest to limit the amount of required data, e.g., to uplink requests, short time windows, and to few features. Moreover, literature shows that such approaches can perform with similar prediction accuracy as high resource-consuming solutions, as we highlight in Table I.*

V. Conclusion

We pinpoint crucial elements within a streaming monitoring approach that collectively contribute to the total effort and influence efficiency of a QoE degradation factor prediction approach. Such effort can be quantified by necessary memory, storage, CPU requirements, or energy consumption, making it a fundamental quality indicator. We see that lightweight monitoring approaches, which only take partial data such as uplink requests into account and require a small number of features, are considerably more cost-effective compared to full packet trace solutions. Furthermore, the duration arriving packets are kept in any form of memory for feature calculation plays an important role. We see from the literature that more complex models often do not significantly improve prediction accuracy or that a minor improvement can require significantly more resources. Thus, we recommend to select the methods to predict QoE degradation factors in a wise way dependent on the prediction goal and to prefer lightweight solutions. In the future, comprehensive measurements for different approaches are required to quantify the memory, storage, and CPU requirements for the investigated procedures in detail.

References

[1] Sandvine, "2023 Global Internet Phenomena Report," 2022, accessed: 2023-08-08. [Online]. Available: https://www.sandvine.com/global-internet-phenomena-report-2023

[2] S. Wassermann, M. Seufert, P. Casas, L. Gang, and K. Li, "Vicrypt to the Rescue:Real-time, Machine-Learning-Driven Video-QoE Monitoring for Encrypted Streaming Traffic," *Trans on Netw and Service Mgmt*, 2020.

[3] M. Shen, J. Zhang, K. Xu, L. Zhu, J. Liu, and X. Du, "DeepQoE: Real-Time Measurement of Video QoE from Encrypted Traffic with Deep Learning," in *Int. Symposium on Quality of Service*. IEEE, 2020.

[4] S. C. Madanapalli *et al.*, "Modeling Live Video Streaming: Real-Time Classification, QoE Inference, and Field Evaluation," *arXiv preprint arXiv:2112.02637*, 2021.

[5] F. Loh *et al.*, "Uplink vs. Downlink: Machine Learning-Based Quality Prediction for HTTP Adaptive Video Streaming," *Sensors*, 2021.

[6] C. Gutterman *et al.*, "Requet: Real-Time QoE Detection for Encrypted YouTube Traffic," in *Multimedia Systems Conference*. ACM, 2019.

[7] I. Orsolic and L. Skorin-Kapov, "A Framework for in-Network QoE Monitoring of Encrypted Video Streaming," *IEEE Access*, 2020.

[8] F. Loh *et al.*, "Uplink-based Live Session Model for Stalling Prediction in Video Streaming," in *Network Op. and Mngmt Symp.* IEEE, 2023.

[9] M. Seufert *et al.*, "A Survey on Quality of Experience of HTTP Adaptive Streaming," *IEEE Communications Surveys & Tutorials*, 2014.

[10] W. Sun *et al.*, "Analysis of Video Quality Datasets via Design of Minimalistic Video Quality Models," *arXiv:2307.13981*, 2023.

[11] F. Loh *et al.*, "YouTube Dataset on Mobile Streaming for Internet Traffic Modeling and Streaming Analysis," *Scientific Data*, 2022.

[12] ——, "Machine Learning Based Study of QoE Metrics in Twitch.tv Live Streaming," in *Network Operations and Mngmt Symp.* IEEE, 2023.

[13] A. Pimpinella, A. Redondi, F. Loh, and M. Seufert, "Machine-Learning Based Prediction of Next HTTP Request Arrival Time in Adaptive Video Streaming," in *Int. Conf. on Netw. and Serv. Mgmt*, 2021.

[14] D. Tsilimantos, T. Karagkioules, and S. Valentin, "Classifying Flows and Buffer State for YouTube's HTTP Adaptive Streaming Service in Mobile Networks," in *Multimedia Systems Conference*. ACM, 2018.

[15] Y.-n. Dong *et al.*, "Novel Feature Selection and Classification of Internet Video Traffic based on a Hierarchical Scheme," *Comp. Networks*, 2017.

[16] M. Seufert *et al.*, "A Wrapper for Automatic Measurements with YouTube's Native Android App," in *Network Traffic Measurement and Analysis Conference*. IEEE, 2018.

[17] F. Loh *et al.*, "Is the Uplink Enough? Estimating Video Stalls from Encrypted Network Traffic," in *Netw. Op. and Managem. Symp.*, 2020.

[18] A. Seufert *et al.*, "QoE Models in the Wild: Comparing Video QoE Models Using a Crowdsourced Data Set," in *Int. Conf. on Quality of Multim. Exp.*, 2021.

[19] G. Dimopoulos *et al.*, "Measuring Video QoE from Encrypted Traffic," in *Internet Measurement Conference*, 2016.

[20] M. H. Mazhar and Z. Shafiq, "Real-Time Video Quality of Experience Monitoring for HTTP and QUIC," in *Conf on Comp Comm*, 2018.