VideoNOC: Assessing Video QoE for Network Operators Using Passive Measurements

Tarun Mangla,
Ellen Zegura, Mostafa Ammar
Georgia Institute of Technology

Emir Halepovic, Kyung-Wook Hwang,
Rittwik Jana, Marco Platania
AT&T Labs – Research

ABSTRACT

Video streaming traffic is rapidly growing in mobile networks. Mobile Network Operators (MNOs) are expected to keep up with this growing demand, while maintaining a high video Quality of Experience (QoE). This makes it critical for MNOs to have a solid understanding of users' video QoE with a goal to help with network planning, provisioning and traffic management. However, designing a system to measure video QoE has several challenges: i) large scale of video traffic data and diversity of video streaming services, ii) cross-layer constraints due to complex cellular network architecture, and iii) extracting QoE metrics from network traffic. In this paper, we present *VideoNOC*, a prototype of a flexible and scalable platform to infer objective video QoE metrics (e.g., bitrate, rebuffering) for MNOs. We describe the design and architecture of *VideoNOC*, and outline the methodology to generate a novel data source for fine-grained video QoE monitoring. We then demonstrate some of the use cases of such a monitoring system. *VideoNOC* reveals video demand across the entire network, provides valuable insights on a number of design choices by content providers (e.g., OS-dependent performance, video player parameters like buffer size, range of encoding bitrates, etc.) and helps analyze the impact of network conditions on video QoE (e.g., mobility and high demand).

CCS CONCEPTS

• **Networks** → **Network measurement**; • **Information systems** → **Multimedia streaming**;

KEYWORDS

QoE, Video streaming, Passive measurement, Cellular network

ACM Reference format:
Tarun Mangla, Ellen Zegura, Mostafa Ammar and Emir Halepovic, Kyung-Wook Hwang, Rittwik Jana, Marco Platania. 2018. VideoNOC: Assessing Video QoE for Network Operators Using Passive Measurements. In *Proceedings of 9th ACM Multimedia Systems Conference, Amsterdam, Netherlands, June 12–15, 2018 (MMSys'18),* 12 pages.
https://doi.org/10.1145/3204949.3204956

MMSys'18, June 12–15, 2018, Amsterdam, Netherlands

ACM ISBN 978-1-4503-5192-8/18/06…$15.00
https://doi.org/10.1145/3204949.3204956

1 INTRODUCTION

Over the past several years, HTTP Adaptive Streaming (HAS) has become the *de facto* standard for video streaming over the Internet, with video traffic surging to over 70% of all network data in 2016 [4]. Consequently, network operators are very interested in understanding if today's video services are performing as efficiently as possible and provide a high Quality of Experience (QoE) to the end users. In particular, MNOs are interested in correlating customer QoE with their network's performance to help in resource provisioning and troubleshooting. We focus on Mobile Network Operators (MNOs), whose network architecture complexity, radio spectrum constraints and cross-layer protocol interactions make video QoE monitoring more challenging and critical.

Measuring video QoE, however, is difficult as user experience is subjective and hard to quantify. Recent standardization efforts propose to use objective video QoE metrics such as *video quality* and *video stalls* to model user QoE [8, 9]. The challenge before MNOs and hence our focus is to infer these objective metrics for a diverse set of video streaming services at network scale.

Existing network monitoring and troubleshooting are done within *Network Operations Centers (NOCs)* [3]. NOCs track overall state and performance of the network elements and links, to facilitate detection of, e.g., outages, failures, and security issues, as well as of the operator-managed services (VPNs, VoIP, or IPTV). Cellular NOCs can also use aggregated radio-level Key Performance Indicators (KPIs) generated by networks elements, such as channel availability, link utilization and some form of link-layer throughput measurement. While these low-level metrics provide a high-level picture of the overall network performance, they alone are not adequate for assessing application-layer performance and user QoE. This motivates the design and deployment of *VideoNOC*, a "specialized NOC" focused on analyzing QoE metrics for video streams.

In theory, there can be multiple approaches to design a *VideoNOC*. The simplest approach could be to proactively collect and analyze QoE data from all video clients using the MNO's network. This is clearly infeasible because MNOs do not have direct access to the video clients at the end devices. Alternatively, a *VideoNOC* could attempt to collect QoE information collected by some Content Providers (CPs) using in-client measurements [5, 37]. This data, however, is owned by CPs and is not available to the MNO. Even if made available, this data is video service-specific and may not have the detail or granularity that would be needed by the MNO for its own provisioning and troubleshooting purposes. Another possible approach could be to measure application and network performance through field testing in different locations of the network. However, with the scale of today's networks and plethora of mobile video services, this is simply not scalable. Hence, these approaches are not adequate for network-wide video QoE monitoring.

The focus of this paper is, therefore, on the design and implementation of a practical *VideoNOC*, which needs to passively collect data from the MNO's network elements and analyze it to learn the users' QoE. This alleviates the need to interact directly with client devices, makes correlation with network-level performance more straightforward, and is less expensive than active field testing. In addition to monitoring QoE, a *VideoNOC* may also have elements to perform network troubleshooting based on QoE data. We relegate this aspect of *VideoNOC* to future work.

Our design of *VideoNOC* utilizes passively collected logs of HTTP/S transactions to infer objective QoE metrics, such as *average bitrate* and *re-buffering ratio* (for unencrypted video), and *session throughput* (for all video), as periodic or per-session measurements. In designing *VideoNOC*, we address three key challenges:

Scalability: For network-wide QoE monitoring, *VideoNOC* is designed as a light-weight system using two key approaches: i) it uses a transparent non-caching web proxy for passive measurements thus avoiding packet-level processing, ii) it processes HTTP/S logs in a distributed manner at regional Cellular Data Centers (CDCs), the convenient traffic aggregation points in cellular network, thus avoiding the need to move the raw logs to a central location.

Cross-layer aspects: The data collected by the web proxy lacks network location information as it operates at a higher layer (HTTP/S) in the network stack with no notion of mobile network location. We augment the web proxy data with network location using the radio-level logs collected at a different layer in the network. Further, to reduce the overhead of this step, we aggregate application data from the same video sessions at a time granularity that reduces its volume by three orders of magnitude but at the same time preserves the information for network location augmentation.

QoE inference: It is challenging to identify the video traffic and then infer video QoE from passive network measurements. We identify video traffic using signature vectors composed of various HTTP headers for unencrypted and TLS headers for encrypted traffic. Previous techniques for inferring QoE from passive network measurements have relied on statistical methods that require significant training overhead [12, 25, 27]. In our work, we devise a less burdensome technique that models network dynamics of adaptive video delivery using domain knowledge.

We deploy *VideoNOC* as a prototype in the real network of a large U.S. based MNO. We demonstrate that our implementation of *VideoNOC* can provide an MNO with a spatio-temporal view of video demand across the network. It can help MNOs understand the impact of network factors such as user mobility and demand on the video QoE. Furthermore, *VideoNOC* can also provide unique insights from the network data otherwise not available from QoE monitoring performed at end devices by CPs or on servers and CDNs. These insights motivate both best practices as well as opportunities for cooperation between MNOs and CPs for the benefit of all stakeholders in the mobile video ecosystems.

Our major contributions include:

(1) Design and architecture of *VideoNOC*, outlining specific decisions that lead to building a scalable, light-weight system on top of the existing mobile network infrastructure.
(2) Methodology for creating a novel data source for MNOs, the *Mobile Video QoE Metrics*, which offer insights into the usage and performance of video services across network locations.
(3) Unique insights into the design and behavior of 15 mobile video services from an MNO's perspective.

The remainder of the paper is organized as follows. Section 2 presents the relevant background. Section 3 outlines the system design and goals, while Section 4 describes the implementation. Several use cases and insights are presented in Section 5. Related work follows in Section 6, while Section 7 concludes the paper.

2 BACKGROUND

We present a high-level overview of LTE cellular architecture including different network measurement data sources, key HAS streaming concepts and the relevant video QoE metrics.

LTE Cellular Architecture: Modern cellular network such as Long Term Evolution (LTE) consists of two major components: a Radio Access Network (RAN) and an Evolved Packet Core Network (EPC). The RAN includes various devices or user equipment (UE) and base stations (known as eNodeBs). The EPC consists of Serving Gateway (SGW), Packet Data Network Gateway (PGW), Mobility Management Entity (MME) and other elements. The EPC is located at geographically distributed CDCs (Figure 1). A UE is a mobile device (smartphone, tablet, LTE USB dongle, or an LTE modem card, etc.) that connects to the eNodeB over the radio channel.

Each eNodeB controls a cell site that typically consists of several radio cells. A cell covers a geographic area with a directional antenna using one frequency band. A sector or face can have multiple cells covering the same direction but possibly different range due to the frequency band characteristics. A UE is connected to a single cell at any point in time and can be handed off between cells within a sector, cells in different sectors, or cells on different eNodeBs.

Call Detail Records (CDR): One of the important data sets generated in a cellular network is the Call Detail Records (CDR). These logs contain the RAN information for a UE and are created by MME as a part of its mobility and hand-off management function [2]. When UE connects to the network, a Radio Access Bearer (RAB) is established to allow the UE to exchange data over the RAN, and appropriate state is created, maintained and logged, so that mobility can be managed and paging functions performed. CDRs contain association between the UE and network location, i.e., the cell, sector, and the eNodeB. However, this association is internal to the network and not exposed externally. In addition, data is transmitted inside tunnels internal to the network, which hides the transport and higher layers from the internal network elements. Hence, MME that generates CDRs is unaware of applications transmitting the data, and application-aware elements, such as proxies external to the cellular network, are unaware of the internal topology, making it challenging to correlate the data across layers.

HTTP Adaptive Streaming (HAS): In HAS, the video is split into chunks, usually of equal duration, with each chunk encoded at multiple bitrates chosen from a pre-defined set. The video player on the client decides the quality of chunks to stream based on some adaptation logic. The use of HTTP makes this approach middle-box friendly and enables content providers to use commodity Content Distribution Network (CDN) servers. The bitrate adaptation at the client allows a diverse set of clients to perform well under a variety of network conditions. As a result, an overwhelming majority of video CPs now rely on HAS-based technologies such as

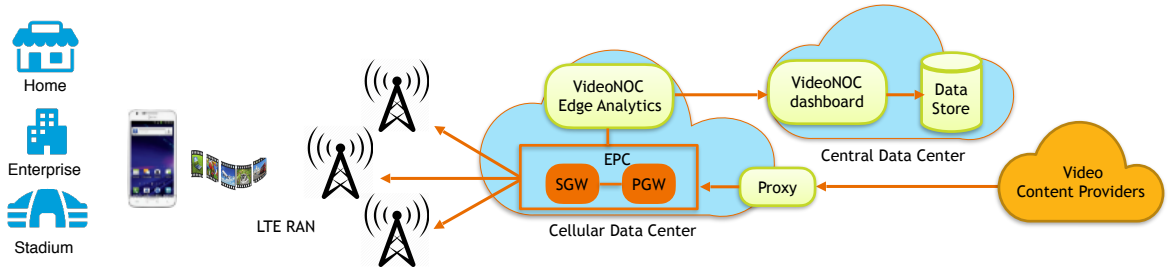


Figure 1: A simplified view of a HAS video delivery with the placement of *VideoNOC* components in the LTE network

HLS [15] and MPEG-DASH [6]. While parts of the framework for these technologies is standardized, they still allow CPs to choose some key video design parameters such as bitrate adaptation algorithms [28, 30, 45], encoding video and audio bitrates, number of bitrate levels, etc. Different choices of these parameters often lead to different video performance under similar network conditions [14].

QoE metrics: According to recent standardization efforts and literature, QoE in HAS can be characterized by multiple objective metrics such as *video quality, re-buffering ratio, quality switches and startup delay* [8, 9, 34]. Average bitrate can be used to reflect the streamed video quality, based on the requirement for more bits to encode a higher quality image. In general, the bitrate of a particular quality level depends on the complexity of the video content being encoded and the encoding efficiency. The selection of quality levels made available to the video player is determined by the each CP. Re-buffering ratio is the proportion of the viewing time the video stalled because of buffer underrun. Bitrate switches represent user-perceived image quality variations in the session. *VideoNOC* gives a detailed estimation of these QoE metrics for unencrypted video services. In addition, QoE is also impacted by video startup time which is the delay it takes for the video to begin playing since the user requested it. *VideoNOC* currently does not estimate video startup delay and we consider it as a part of future work.

3 SYSTEM DESIGN

This section presents the main goals of *VideoNOC* and key design decisions.

Goals: *VideoNOC* is designed to achieve the following goals:

- *Assess video QoE and demand*: It provides an MNO with a comprehensive view of video QoE and relative resource demand imposed by a wide variety of CPs within its network. This allows video QoE-aware capacity planning, traffic management, and other in-network optimizations.
- *Understand cross-layer interactions*: This allows MNOs to understand the impact of network-layer factors such as mobility and radio resource provisioning on video QoE.
- *Detect issues in video service design*: *VideoNOC* can uncover design-related issues that are not apparent to CPs but impact network-related performance of the services. For example, it can detect bandwidth-wasting video services that impact their own and other users' experience. This allows MNOs to recommend best practices of video service design to the CPs with an overall goal of improving user QoE and efficient utilization of network resources.

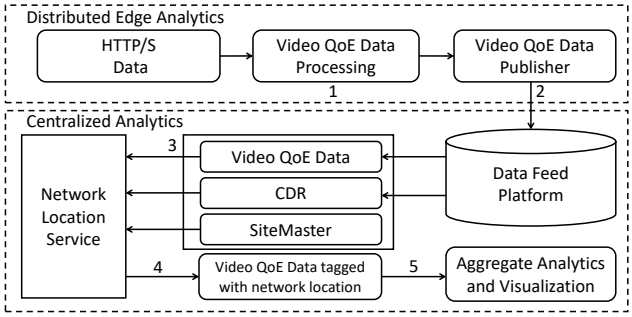


Figure 2: VideoNOC architecture

3.1 Design choices

Data sources: We label any type of network performance or high-level application-layer analysis of QoE in the operator's network as 'network data'. To meet the goals of *VideoNOC*, we have to use network data intelligently to mitigate the lack of access to ground truth from video players or CPs.

In a typical cellular network, the traditional performance data source are radio level KPIs, which do not reflect application-level performance, but provide low-level radio statistics about radio bearers and radio resource usage. Performance of higher layers of networking stack have to be obtained from either transport layer data, or web proxies (HTTP and HTTPS transactions). There are trade-offs, pros and cons with all of these data sources. Transport-layer data (TCP/UDP flow data) provides insights into packet-level transmissions, TCP connection details like throughput, retransmissions, etc., which is fairly granular but requires significant processing power to extract higher-layer information. HTTP data contains protocol-specific information, but any loss, retransmissions, or throughput variation in lower layers is hidden. Finally, radio-level data contains device associations with base stations, sectors, and cells, as well as handover information and timing, but has no notion of transport or higher layer protocols.

Within our *VideoNOC* design, we find a good compromise in using HTTP (including HTTPS) logs and RAN data. HAS video services are based on HTTP and we find key pieces of information in URIs and HTTP headers that allow us to extract QoE metrics to a satisfactory level of accuracy and precision. Obtaining HTTP data is relatively easy with the deployment of a standard web proxy. We have to use RAN data to provide network location (serving radio cell), since higher-level protocols (HTTP/TCP/UDP) have no notion of mobile network location. As explained later, cellular networks have a built-in RAN data reporting capabilities that can be exploited for our purpose. We decide not to use TCP flow data at this time

due to the very high implementation, deployment and processing overhead, but leave it as an opportunity for future work.

There is a cross-layer aspect to the decision on data sources. To relate RAN data and HTTP data, such as locate the serving cell for each HTTP request, we have to use the existing cellular network infrastructure capability to associate IP data traffic with the cellular device. Otherwise, the internal UE identification on the RAN could not be related to HTTP traffic outside the PGW.

Centralized vs. decentralized processing: Another design decision is where to process the HTTP logs collected by the web proxy. With the HTTP log arrival rate of up to 15 GB/min or 2 Gbps in our limited deployment, it is prohibitive to transfer them to a central location. We process this data at the edge using an already available generic Edge Analytics (EA) computation platform. Processing at the edge reduces the data to be transported to the central location. This reduction in data volume is significant and is approximately 3 orders of magnitude. As shown in Figure 2, video QoE data is transported without location identifiers, using an existing Data Feed Platform (DFP) similar to Apache Kafka[1], to be augmented with network location at a central location.

QoE metrics aggregation: Since HTTP log processing extracts video QoE metrics at the edge and then discards the logs, we need to decide which metrics to generate and preserve, and how often to report them, while maintaining the ability to associate network location at a later stage.

The length of video sessions greatly varies depending on the length of videos, loss of interest, and other factors. One option is to generate session-level QoE metrics at the end of the video session, but this has three main drawbacks: (i) aggregate QoE metrics of a long video session (e.g., 1 or 2 hours) cannot reflect impact of the mobility and handoffs, (ii) aggregate QoE does not reflect possible variability throughout the session, (iii) long sessions may prevent *VideoNOC* to report QoE metrics in a timely manner.

Therefore, it is critical to group video chunks appropriately in time and space to enable both timely reporting and account for mobility. We explore several periodic reporting options. Compared to per-session reporting, per-chunk reporting would increase the reported data volume significantly as well as processing overhead, since chunks are typically 2-10 seconds in duration. We tested 5 minutes as a reporting period, but found that when UEs are mobile and stream video, they can handoff between several cell sites with 5 minutes. This makes it impossible to correctly associate the serving cell to the part of the video session, and due to intermittent nature of HAS downloads, time spent in each cell is not reflective of the number of chunks downloaded.

We then explored shorter time intervals for periodic reporting and find that it is extremely rare to find more than one handoff within one minute. This results in a 1 to 1 relationship between the serving cell and the group of video chunks. The cost of 1-minute reporting is certainly much higher than 5-minute and per-session reporting, about 60% in data volume, but we find the trade-off acceptable and establish that establish it as correct compromise[2].

[1] kafka.apache.org/

[2] For handoffs observed within 1-minute period, we assign the first serving cell to that minute. The reason is that HAS player reaction to changing network conditions is often delayed due to throughput smoothing functions and their ability to detect changes generally after the current video chunks have fully downloaded

In addition, the goal is to look for persistent issues in the network or video services, rather than transient effects.

Data analysis: After the video QoE and full network location data are joined, it remains in the Data Store (DS) to be used for reporting and analysis. We use three approaches for this purpose, (i) a dashboard for aggregated periodic reports, (ii) interactive maps for spatio-temporal analysis, and (iii) special studies from raw data.

4 ARCHITECTURE AND IMPLEMENTATION

We next describe the architecture and implementation of the main *VideoNOC* components, including data collection, video QoE metric estimation, and network location tagging.

4.1 HTTP data collection

We use passively collected logs from a transparent web proxy covering a fraction of CDCs, corresponding to several PGWs of the cellular network, where all IP traffic from and to the Internet conveniently aggregates. The web proxy handles HTTP and HTTPS transactions between a subset of UEs and corresponding servers in a standard manner, by mediating HTTP transactions and passing through HTTPS, while logging the activity. This web proxy behaves similar to the well-known `squid` [7].

The web proxy instances generate 4 parallel streams of logs, which are ingested by EA processes. Logs are delivered as flat files by the web proxy and placed in 4 directories of a special file system. Logs are batched in files, with each file covering on the order of 5 minutes of traffic and roughly 250,000-300,000 HTTP transactions each. Multiple files can cover approximately the same time period as load varies. The streams always have consistent UE allocation, as the web proxy places all records for each UE in the same stream, allowing us to process the four streams in parallel.

Logs include fields as outlined in Table 1 with their source and usage in video QoE processing. Note that some fields may be unpopulated or not provide full information in every log. For example, User agent may be improperly populated by the app, URI may obfuscate video chunk information, or Content type field may not contain video but other generic type. *VideoNOC* deals with these issues by either using only available data or deriving it from other fields and/or stream characteristics, depending on the CP. The UE ID is anonymized in this and CDR data so that the actual users cannot be identified. No personally identifiable information (PII) was gathered or used in conducting this study. To the extent any data was analyzed, it was anonymous and/or aggregated data. While the web proxy coverage does not include the entire network, we have no reason to believe that the general methodology and the available data set is not reflective of the overall network traffic. *Even with the modest set of available data collected from a fraction of network locations, we can extract a wealth of valuable insights (see Section 5).*

The volume of the HTTP logs varies depending on traffic load of each CDC, diurnal pattern, etc., with peak ingestion rate of the log streams reaching about 15 GB per minute. Figure 3 shows the ingest rates over several days, indicating two important aspects. Most obviously, some CDCs handle more traffic than others, leading us to resize the processing power per CDC accordingly. Then, diurnal variability can be significant in some CDCs calling for elastic scaling. While some CDCs have a relatively stable traffic load, others exhibit

Table 1: HTTP data logged by a web proxy

HTTP log field	Source	Usage
HTTP request arrival time	Proxy	Time when player requested a video chunk
Transaction end time	Proxy	Time when all data of HTTP(S) transaction were sent to the UE
UE ID	Proxy/EPC	Associates video flows to different UEs
Server IP address	Proxy	Distinguishes some CPs that do not use domain names
Content length	Proxy	Distinguishes video chunks from meta-data
User agent	HTTP	Identifies UE make and model, OS and OS version
Content type	HTTP	Helps identify video, audio, and other content types
HTTP method	HTTP	Distinguishes between video chunks and meta-data vs. app feedback
HTTP response code	HTTP	Detects failed video requests, redirections, etc.
URI	HTTP	Identifies CPs, CDNs, video chunks, tracks, sessions, meta-data, etc.
SNI	HTTPS	Identifies CPs and CDNs

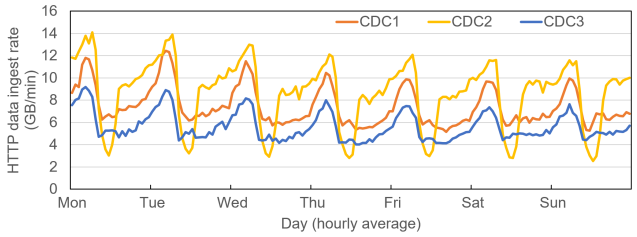


Figure 3: Data ingest rates for a representative CDC subset

high variability (up to a factor of 7), indicating that the underlying distribution of traffic across CDCs will drive the deployment of resources for this type of data processing.

4.2 QoE inference

Once HTTP log files are ready, they are processed to estimate video QoE metrics using the EA platform which is composed of virtual machines and is co-located with the web proxy. Each EA node consist of 16 Intel Xeon E3 CPUs at 2.7 GHz, 4 cores per CPU, 32 GB RAM and runs CentOS 6.5. Additional nodes can be added as needed based on the traffic demand. Each EA node runs 4 parallel processes handling one data stream each, with the outputs merged at the end of the processing.

Our QoE inference approach first reconstructs video sessions from HTTP logs. The QoE of a session is then estimated by extracting useful features from the HTTP logs corresponding to the session. The QoE estimation is done at both session and per-minute level. At the end of the QoE inference process, the EA outputs 3 types of records: (i) S-records contain per-session metrics and are published when session end is detected, (ii) P-records contain periodic aggregates published every minute, and (iii) B-records that are both session and periodic in a sense, which contain aggregate metrics for sessions shorter than 1 minute. Processing overhead of periodic reporting is less than 6%. The total output of our EA turns out to reduce the data volume by 3 orders of magnitude, even with 1-minute reporting. Next we describe each of the steps in detail.

4.2.1 Session reconstruction. In order to reconstruct video sessions, we need to distinguish between video and non-video traffic in the web proxy logs. We identify a record to belong to a certain video CP by inspecting the HTTP request URI. Figure 4 shows an abstract template of the chunk request URI from an anonymized CP that we call *VideoApp*. Content IDs are arbitrary alpha-numeric strings

Figure 4: URI template and the extracted information

meaningful only to CDNs, and often randomized or hashed. They do not allow particular content to be identified from traffic. Since the request URI is not available for encrypted traffic, we rely on the Server Name Identification (SNI) [1] to indicate the CP. Non-video traffic is then filtered out based on the content length. Other useful meta-data about each session is also collected whenever available, such as UE OS, OS version and device type (see Table 1).

The set of required *signatures* to identify a video service is obtained by a combination of two methods: (i) grouping similar URI patterns from HTTP data and (ii) using active measurement and packet trace collection. Note that active measurement reveals signatures that the CPs used for the particular UE in that test, as for example, different CDN might be selected based on UE or external IP address. This information is then augmented by URI patterns from HTTP data to cover all variants, if possible. CPs occasionally change their URI patterns, CDNs, or drop domain names in favor of IP addresses. These changes can be detected by observing data volumes, repeating active tests, or using other techniques. Note that most CPs typically use only a few *templates* for their content.

The detection of session end is done in several ways. For CPs that use session tokens or specific session change indicators in the URI, we use those as triggers. Detecting a video stream with the different content ID or CP from a UE triggers session end of the active session for that UE. Otherwise, the empirically determined timeout without new chunks closes the current session for a UE.

4.2.2 QoE estimation. Once the video sessions have been reconstructed, we estimate the QoE by extracting useful features from the records and feeding them into our QoE estimation algorithms. Since the extracted features are different for encrypted and unencrypted video sessions, we use different QoE estimation methodologies for both of them, leading to different levels of QoE view.

Unencrypted traffic: We estimate the following three key QoE metrics for a session: *average bitrate*, *re-buffering ratio* and *bitrate switches*. The QoE estimation approach in this case is based on two key insights. First, an HAS video session can be abstracted as a sequence of video chunks appearing as separate HTTP GET requests on the network. Second, the request URI in the video

chunks can provide significant information related to the session QoE such as chunk identifier and quality level (see Figure 4). Thus, monitoring the HTTP logs corresponding to video chunks along with relevant information extraction from the request URI allows us to model a client video session.

Specifically, we get request completion time (T_i), chunk quality (Q_i) and chunk size (S_i) for every chunk request i in the video session V. The chunk duration (L) in seconds is obtained from the *manifest* files for a few videos in out-of-band experiments, or sometimes inferred from the URI, if indicated. The total number of chunks downloaded in a session is denoted by N. We use this information to estimate different video quality metrics as follows:

- *Average bitrate*: We estimate the average bitrate by taking the time average of the collected chunk size of the session.

$$\hat{br} = \frac{\sum_{i=1}^{N} S_i}{N \times L} \tag{1}$$

  Average bitrate is a strong indicator of the video quality as CPs typically use a specific encoding profile per discrete bitrate, which maps to a single video quality metric value [34].
- *Number of bitrate switches*: Number of bitrate switches can be calculated simply by calculating the number of times the chunk quality changed between consecutive chunks. Here I is the indicator function and has a value of 1, if the consecutive chunks do not have same quality, zero otherwise.

$$\hat{br\_switch} = \sum_{i=2}^{N} I(Q_i \neq Q_{i-1}) \tag{2}$$

- *Re-buffering ratio*: Intuitively, re-buffering time is estimated by keeping an account of video chunks that have been downloaded and the part of that video content that should have been played so far. The exact details are described in prior work [35].

We also preserve distributions of per-chunk bitrates within sessions, to enable aggregation per bitrate.

Encrypted traffic: For encrypted traffic, we have the TLS transaction statistics with a single TLS connection potentially corresponding to multiple HTTP requests. The QoE estimation methodology for unencrypted traffic can not be applied to encrypted traffic as per-HTTP transaction statistics are not available. Hence, for encrypted traffic, we calculate the session throughput (ST), which is defined as follows:

$$ST = \frac{\sum_{i=1}^{N} S_i}{T}, \tag{3}$$

where S_i is the content length of the i^{th} TLS transaction in a session of N transactions and T is the total session time as observed on the network. We rely on degradation in ST in order to detect degradation in video QoE. From large scale validation (see subsection 4.3), we show that ST is strongly correlated with *average bitrate*.

For a session, we record the overall session throughput as well as per-minute session throughput which is calculated by considering transactions only in the current minute. The latter is done to detect temporary variations in the session QoE. Note that ST is also calculated for unencrypted video services with the TLS connection content length replaced by HTTP transaction content length.

Table 2: Correlation between ST and ground truth.

Content type	$\rho_{(br,ST)}$		$\rho_{(rr,ST)}$	
	OS1	OS2	OS1	OS2
Live	0.89	0.88	-0.18	-0.10
VoD	0.76	0.80	-0.27	-0.37

4.3 QoE metrics validation

We evaluate the accuracy of our QoE inference algorithms by comparing them against the QoE metrics collected by a mainstream 3rd party video analytics SDK built into *VideoApp* (anonymized for confidentiality), a mobile app from a large mobile video service serving unencrypted traffic. The QoE metrics provided by the in-app SDK consist of *average bitrate* and *re-buffering ratio*. This is the typical approach used by commercial video services to monitor QoE.

Data set: We consider the *VideoApp* sessions collected by *VideoNOC* for a period of 12 days in the year 2017, in a small part of the network. The estimated QoE metrics of these collected sessions are compared with the corresponding in-app SDK-collected QoE metrics referred to as ground truth. It is not trivial to match the sessions collected on the network with their corresponding in-app SDK logs. Due to the independent sources and anonymizations of both data sets, we have to resort to the following matching logic. We use the session content identifier, session start time and duration, OS kind and version, and CDN to match sessions. We filter out sessions under 1 minute since it is challenging to match the network session duration with the ground truth playtime for such short sessions. In the case of a single match between data sets, we use it for comparison, while in cases when a single *VideoApp* session matches multiple *VideoNOC* sessions, we discard all of them. For the time period under study, we were able to match 70,214 sessions, which is a small fraction of all *VideoApp* sessions. Both Live and VoD are well-represented in this set.

Detailed QoE metrics: We evaluate the inference accuracy for average bitrate and re-buffering ratio for the above data set (see [35] for details). The key findings are summarized here:

- We predict the average bitrate within a relative error of 10% for up to 90% of sessions.
- We accurately predict re-buffering ratio with an absolute error less than 1% for up to 90% of the sessions.

Video session throughput: In addition, we also analyze the usability of ST to infer video QoE. This is important as we do not have chunk-level statistics for all CPs, especially for those encrypting their data. We correlate the in-app SDK average bitrate and re-buffering ratio with the calculated ST for the matched sessions in *VideoApp*. Table 2 shows the Pearson's correlation coefficient (ρ) values between ST and average bitrate ($\rho_{(br,ST)}$) and between ST and re-buffering ratio ($\rho_{(rr,ST)}$) for each analyzed OS. We find that average bitrate and ST are strongly positively correlated, while there is a weak negative correlation between re-buffering ratio and ST. This shows that session throughput can be used as an indicator of video QoE. Specifically, degradation in session throughput would indicate degradation in average bitrate in most cases and high re-buffering in some cases. Recent work also shows that average downlink throughput maps reasonably well to video QoE [19].

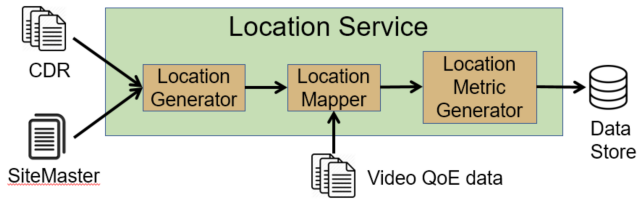


Figure 5: Location Service architecture

4.4 Network Location Service

RAN data containing network location is generated at the edge (in CDC), but we specifically chose to associate it to QoE data in a centralized manner in this implementation of *VideoNOC*. The ideal scenario would be to associate network location in the CDC, but practical considerations led us to the centralized approach. In particular, (i) network location is already available as part of the centralized CDR data and (ii) RAN data is not available for processing in every CDC. It is more efficient to avoid replicating network location extraction at the edge in our current design.

We build a network Location Service that joins video QoE data and network location obtained from two other data sources, CDRs and a master inventory of base stations and cells that the network operator manages, which we call *SiteMaster*. CDRs provide the cell ID for each period that UE was connected, while *SiteMaster* contains the cell site physical location and extended radio-level details, such as frequency, bandwidth, etc.

The design of the Location Service presents two challenges. First, there are three data sources with different volume, locality, and other characteristics. Video QoE data is published to the DFP in flat files covering approximately 5-minutes of real time, similar to its ingest process. This data may include some records older than 5 minutes, due to timeouts used to delimit video sessions. This data arrives on the order of up to 5 Mbps on average in randomly distributed arrivals from multiple CDCs. CDRs arrive at much higher rate, up to about 70 Mbps, and in much larger files, up to several GB to tens of GB in size. They contain radio-level sessions of all network devices. This data source is a very large superset of devices streaming video at any point in time. CDRs arrive from the same DFP to which *VideoNOC* is just a subscriber, as opposed to EA being also a publisher under our control. Location Service has to ingest large amounts of CDR data and maintain longer history to maximize match rate to video QoE data arriving in smaller batches. *SiteMaster* is a static file updated daily, from which extended network location data is retrieved. It contains on the order of hundreds of thousands of records.

Second challenge is that the computation must be fast in order to detect possible network issues as early as possible. The total processing time has to account for the regular computation, plus potential restart of the entire job or parts of it in case of failures.

The architecture of the Location Service is presented in Figure 5. It is composed of three components: *Location Generator*, *Location Mapper*, and *Stats Generator*, forming a multistage pipeline. The Location Generator takes CDR data as input and extracts from each record a subset of fields, such as transaction start and end times, UE ID, and a vector of cell IDs and duration of time spent associated with them. We expand this to have one record per serving cell for a UE. UE IDs are anonymized in a consistent manner for joining with video QoE data. Then, Location Service fetches extended network location data from *SiteMaster* for each cell in the CDRs. The outcome of Location Generator is a set of records of the form: <*cell ID, start time, end time, UE ID, sector ID, tower ID, latitude, longitude*>.

The Location Mapper produces a new set of video QoE data with network location and other cell-level information by joining the records produced by Location Generator and video QoE data on the UE ID field. A new record contains the trajectory of a device during an entire transaction. The total transaction duration is divided in periods, each one with its own duration, cell id, sector id, tower id, and cell site location (latitude and longitude). By looking at these new video QoE data records, an operator can map information about the user QoE to the underlying physical network. This allows the operator to understand QoE differences across network elements.

In case of unjoined records, the Location Mapper tags those video QoE data entries with *unknown_location*. This happens when video QoE data records do not have corresponding entries in the CDR files. Our design shows a trade-off between the number of records without network location and the computation time. Processing more CDR files every hour helps reduce the number of records without network location, but it may take longer to finish. In case of failures, the re-execution of failed tasks may overlap with the beginning of the next computation. While this may not be a problem in large dedicated cluster, it may cause delays in shared clusters.

Finally, the Stats Generator uses the output of the Location Mapper to compute network location-based statistics, such as the number of handover between cells (within the same eNodeB and across), and handover between sectors (within the same eNodeB). These metrics are added to video QoE data, in addition to serving cell for periodic, and cell trajectory for per-session records. The final records are self-contained for reporting and analytics and stored in a central DS. Both DS and the landing area from DFP are large-scale repositories on Hadoop Distributed File System (HDFS).

The Location Service is implemented in Java using Apache Spark [40]. Our motivation for using Spark is that it allows users to efficiently build distributed applications that process a large amount of data. The Location Generator is composed of *map* operations; the Location Mapper uses *map, join, filter* operations, while the Stats Generator has *map* and *reduce* operations.

We distribute the Location Service computation across four servers, each one having 64 CPUs at 2.20 GHz, 8 cores per CPU, 800 GB of RAM, and running CentOS 6.9. These servers are part of a cluster shared by many applications. One server hosts the Spark manager, while the other three host the Spark executors. We configure Spark to use 12 executors, each one with 20 GB of RAM and 4 cores. The Spark manager distributes executors across the three servers based on resource availability. Due to the shared nature of the cluster, we don't allow more than one Spark environment running at the same time. Hence, we choose the amount of input data such that the Location Service completes its execution before the next batch, even in the presence of failures, and the percentage of video QoE data without network location is acceptable (3% to 5%). Finally, we use the Spark capability to automatically restart failed tasks. Currently, a failure of the entire job requires manual intervention from an operator. We leave the Location Service fault tolerance as future work. Location Service is implemented with less than 1,000 LoC, it runs on a modest shared generic computation

infrastructure, and its average computation delay varies between 17 and 25 minutes.

5 EXAMPLE USE CASES AND ANALYSIS

In this section, we demonstrate three examples of use cases for which *VideoNOC* can provide insightful results: *i) relative video service usage and impact, ii) insights into video service design, and iii) understanding the impact of network factors on video QoE, namely mobility and higher video demand per cell.* We also compare our findings to a 2011 study [24], to point out the evolution of video streaming in cellular networks over the years.

Data set: The data set from which we draw sample network locations used for this analysis comprises of logs spanning two weeks in 2017. The data consists of 272 million anonymized sessions from a sample of 15 different CPs amounting to 5,070 TB of data. We filter out sessions shorter than 2 minutes in duration to remove potential skew due to auto-play feature, users that browse or sample videos, and overall startup effects (e.g., initial buffering, which we discuss separately in detail).

We group CPs by the estimated broad type of content served – user-generated content (UGC), premium video-on-demand (VoD) and live TV (Live) content. If a CP uses HTTPS, we refer to it by appending an 's' to the end of the name. UGC services are streaming predominantly user-generated videos with a varying proportion of commercial videos and advertisements in up to 4K resolutions. Most of them are considered online social networks. VoD CPs are large paid streaming services with movies and shows offered in various quality levels up to 4K resolution, either under independent subscription or as a part of the home TV services. Live CPs offer live TV streams from large TV and sports broadcasters.

5.1 Insights into relative video usage

As video services and their delivery evolved over the past several years, one example use case is to consider the relative demand, efficiency and popularity across CP categories and network locations.

Relative contribution to overall video demand: Considering the contribution to the overall number of video sessions streamed, we observe that the top 5 CPs from our sample set generate 95% of video sessions. They consist of UGC and VoD CPs, with majority being UGC. Other VoD CPs generate 4.5% of video sessions, and Live CPs remaining 0.5%. While Live video is currently a minor contributor to video sessions in cellular, this is an emerging category that may grow in the future. The relative contribution in terms of data demand is similar as the same top 5 CPs contribute 93.5% of the overall data demand. This is qualitatively similar to the 2011 study [24]. *Given the significant impact of these top 5 CPs, optimizing these services for cellular networks would have a large overall impact on the efficiency of network resource utilization.*

Understanding efficiency of video content delivery: We consider the delivered video bitrates and their relationship to the overall video session throughput (ST) to understand the efficiency of delivery vs. QoE. We focus on a subset of CPs for which we can estimate the available bitrates using request URIs. Figure 6 shows the medians of average per-session bitrate and ST from a sample of CPs and network locations. The bitrate ranges indicates significant increase compared to the 2011 study, which found that most video

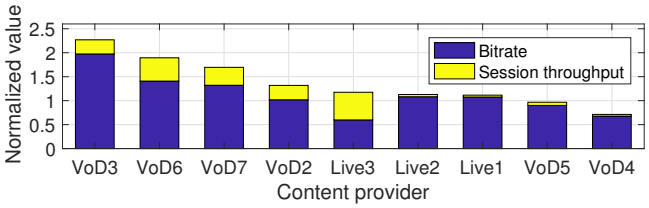


Figure 6: Normalized median per-session average bitrate and session throughput from a sample of CPs and network locations. A value of 1 corresponds to 1.5 Mbps.

sessions were encoded at 255 Kbps. This improvement comes with transition from 3G to LTE and from progressive download (PD) to adaptive bitrate (ABR) streaming technologies. Another aspect of video evolution compared to 2011 study is that nearly all detected video sessions use some form of ABR, with PD diminished.

However, the question arises whether the high bitrates are reasonable. Video sessions in cellular networks are typically streamed to small-screen devices, such as phones. Existing studies show that on small screens, bitrate levels of 1 to 1.5 Mbps with state-of-the-art encoding techniques can provide a high visual quality with a diminishing utility beyond that point [42]. Delivering excessive video bitrates wastes data allotments and network bandwidth, something of utmost importance under limited radio spectrum. The data reveals that most of the sample CPs actually exceed the median per-session bitrates of 1.5 Mbps, shown as the normalized value of 1 in Figure 6, making the excess bandwidth use more of a rule rather than an exception.

Further compounding the problem, ST often exceeds the bitrate by a significant margin, indicating that many services deliver much more data than the useful encoding bitrate, with overhead reaching up to 50% (Figure 6). This comes as a consequence of several possible issues: i) use of less efficient encoding, such as Constant Bit Rate (CBR), as opposed to Variable Bit Rate (VBR), ii) less efficient transport and packaging schemes, which in cases of separate audio and video streams could lead to more overhead, iii) various chunk duplication and replacement behaviors (discussed later in detail), etc. We find one or more of these issues present in many CPs.

On the other hand, we can see that some services have minimal overhead and tend to stream appropriate bitrates for small screens. It is encouraging to see that the service designs conscientious of cellular environment and device context are starting to appear among CPs. *These observations raise a question of whether the surging demand for video bandwidth could be significantly curbed by simply considering screen size of mobile devices, and carefully selecting more efficient methods, thereby reducing the obvious resource waste. The findings present a strong argument in favor of utility-based adaptation for more efficient use of network resources.*

Cell coverage: We observe the proportion of cells with traffic from each CP, out of the total number of cells in our sample dataset. This offers an insight into the *network coverage* across CPs. We observe that a small number of CPs can be observed in a majority of radio cells, with the top top 5 CPs cumulatively covering 99% of cells. *This suggests that thoroughly understanding QoE for those CPs could provide a representative view of video QoE across different parts of the entire network. Combined with the highly skewed demand, it may be possible to use sampling to improve speed and reduce the*

Table 3: Service design aspects across CPs

CP	Bitrate range (Kbps)	Number of bitrate levels	Chunk duration (s)	Chunk IAT (s)	Estimated buffer in OS1 (s)	Estimated buffer in OS2 (s)
UGC1s	-	-	-	4.7	35	35
UGC2	-	-	-	3.2	23	25
UGC3s	-	-	-	3.5	25	25
UGC4s	-	-	-	-	22	22
UGC3	254-1291	4	3.3	0.7	-	-
VoD1	-	-	-	-	60	27
VoD2	130-2664	11	10,5	7.8	20	52
VoD3	266-4681	8	4	3.6	10	16
VoD4	268-2408	6	4	3.9	14	25
VoD5	232-3219	7	4	3.5	45	29
VoD6	194-5901	11	3,10	8	20	35
VoD7	264-3875	8	9	8.7	12	25
Live1	175-3404	7	8	7.4	10	12
Live2	171-3402	7	8	7.7	13	13
Live3	153-2744	15	6	5.6	16	7

processing cost of video QoE data. However, it is still valuable to study other CPs, from the perspective of interaction with most popular ones and general traffic, and because their relative importance to users might be higher, as these are generally premium services.

5.2 Insights into video service design

Another example of *VideoNOC* use is to uncover design choices observed for various CPs, focusing on those that have high impact on QoE and efficient use of resources in cellular network (Table 3).

Bitrate range: We report detected *Bitrate range* across a *Number of bitrate levels* for CPs in the dataset. While most CPs provide a similar lowest bitrate (near 200 Kbps), the range drastically varies, with some CPs exceeding 4 Mbps at the high end. While some CPs deliver all available bitrates over cellular networks, others do not deliver the highest bitrates in cellular networks in any appreciable amount (e.g. Live1, Live2, VoD4, VoD5). We confirm that this is the CP design having nothing to do with network performance by finding that the highest requested bitrates across cells with low, medium, and high load, are generally the same. This points to a form of internal control or cap on bitrates in cellular, which we further confirm by manual inspection of mobile apps. Such controls are implemented either using *CP-tailored manifest files for cellular or application settings (e.g. by settings in user preferences).*

We further find it peculiar that there is no apparent rule followed by CPs on the number and granularity of bitrates offered. For approximately similar range, there could be a large difference in number of bitrates. One reason could be differences in streaming technologies across devices for the same CP leading to different encoding requirements and CDN storage management. Another reason could be differences in bitrate adaptation strategies where more bitrates may achieve some design objective for the CPs, such as smoother visual quality transitions during adaptation.

Chunk duration: *Chunk duration* in Table 3 shows the duration of a video chunk obtained by manually inspecting the manifest file for some of the services. It appears that there is a tendency where Live services use longer chunks than VoD services. These choices might be the remnants of the legacy recommendations in HLS, however, it is not possible to determine the true reason from network traffic. Longer chunks might, however, reduce the flexibility and agility of the player to adapt to rapidly changing network conditions, often found in cellular networks.

For the benefit of deriving automated methodology in *VideoNOC*, we compare these values to the median *Chunk inter-arrival time (IAT)* observed on the network for each video service. The values are very close to the actual chunk duration for the services. This indicates a viable possibility for automatically detecting chunk durations from their inter-arrival times in *VideoNOC*, thereby removing the need to manually inspect these services.

Buffer levels: Another source of network overhead in video streaming can be inadequately sized video buffer. In HAS, players usually fill the video buffer and then enter into the steady state phase where they maintain the buffer level by fetching another chunk once there is space in the buffer. Downloading a large amount of video, especially during the startup buffering phase, can lead to waste of network bandwidth in case of abandonment. Past work has indicated that a large number of users tend to sample video content [17], or do not watch the entire content [24].

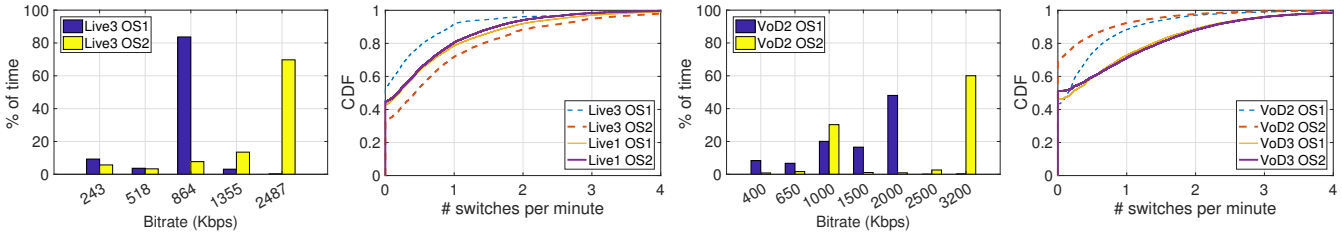
To assess the buffering behavior at large, we use the following methodology, which is intended to estimate average buffering across sessions, as opposed to precisely determine the maximum buffer size in each player. For each CP, we estimate and compare the average amount of content (in seconds) buffered in the first minute of the video session, as we provide per-minute periodic reporting in *VideoNOC*. The buffered content ($\hat{B}_{startup}$) is estimated by dividing the amount of data downloaded in the first minute ($D_{startup}$) of the video session by the average bitrate ($\bar{br}$) of the entire session.

$$\hat{B}_{startup} = \frac{D_{startup}}{\bar{br}} \tag{4}$$

In case of CPs for which we do not have the bitrate, we use *ST* as an estimate of the bitrate. The median of $\hat{B}_{startup}$ across all sessions of a CP is calculated and shown as *Estimated buffer* (Table 3) for OS1 and OS2. For encrypted services, we are not able to detect the OS type and hence the same values are shown for both OS. We exclude UGC3 from this analysis as we found that this CP pre-loads multiple videos in background before users specifically select to play a title.

For the same CP, we find differences in the video buffer content across OS's. This indicates different design choices across OS's. The video buffer for *Live* content is smaller than for *VoD* and *UGC*, which is to some degree expected, as *Live* services are streaming content generated in real time and it is not desirable to have a long lag to real time. For most CPs, the buffer is less than 30s. However, we also find larger buffers for some CPs (e.g., 60s for VoD1 on OS1), indicating non-trivial network overhead due to abandonment in these CPs. *This points to the need for exploring smarter strategies to determine the buffer size, especially in the beginning of the video playback, in order to minimize network overhead due to users sampling videos.*

Bitrate usage and switching variability across OS: Different buffering strategies by OS prompt a closer examination whether other design aspects differ across OS's, resulting in different QoE. We look deeper into variability in bitrates and switching using two example services, Live3 and VoD2.



(a) Bitrate distribution **(b) Number of bitrate switches** **(c) Bitrate distribution** **(d) Number of bitrate switches**

Figure 7: Variability of QoE metrics across OS for Live3 and VoD2

Table 4: Network overhead due to chunk replacement

CP	% sessions w/ non-zero CR overhead	mean CR overhead (% bytes)	% sessions w/ ≥ 20% CR overhead	CR overhead (% data)
UGC3	79.99%	13.68%	26.48%	9.14%
Live1	73.29%	8.68%	13.42%	5.42%
Live2	63.89%	7.75%	12.79%	4.95%
VoD5	60.52%	6.38%	12.66%	7.80%
Live3	60.07%	6.55%	9.30%	3.58%
VoD6	41.14%	8.37%	17.30%	4.62%
VoD7	38.13%	5.46%	10.06%	5.11%
VoD2	25.93%	4.03%	7.65%	4.54%
VoD3	20.07%	2.75%	4.79%	4.75%
VoD4	3.02%	0.31%	0.55%	1.30%

Figure 7a shows the distribution of bitrates played by Live3 on OS1 and OS2. Nearly 70% of Live3 content on OS2 is streamed at an average bitrate of 2,487 Kbps, whereas 84% of the content on OS1 is streamed at an average bitrate of 864 Kbps. This suggests that sessions on OS2 may experience higher video quality than OS1[3]. However, Live3 sessions on OS2 have a higher number of bitrate switches per minute as compared to OS1. *This seems to suggest that higher bitrate switching on OS2 could be because trying to stream video at higher bitrate leads to instability as available bandwidth varies.* We also plot the bitrate switches per minute for Live1 in Figure 7b to show a video service in which the switches per minute are similar on both OS platforms.

Interestingly, we observe a contrasting example in the case of VoD2, where there is similar variation in streamed bitrates across the two OS, but streaming at higher bitrate does not lead to higher bitrate switches. As shown in Figure 7c, VoD2 sessions on OS2 are streamed at higher bitrate than sessions on OS1. However, unlike Live3, the bitrate switches per minute for VoD2 sessions are lower on OS2 as compared to OS1 (see Figure 7d), despite streaming at a higher bitrate on OS2.

These examples suggest that it is not necessarily the network conditions alone, but also player-specific design choices such as default bitrate levels and bitrate adaptation algorithm that significantly impact the QoE of a video session in practice.

Chunk replacement (CR) overhead: It is common to observe multiple chunk requests with the same chunk identifier, but different bitrate. We call this phenomenon *Chunk replacement* and

[3]Video quality is also impacted by the encoding technology used. Some encoding technologies are more efficient than others and can provide higher quality at the same bitrate level

Table 5: Impact of mobility and demand on session throughput for a subset of network locations

CP	Impact of mobility $\Delta_{mobility}$		% mobile sessions		Impact of demand Δ_{higher}	
	OS1	OS2	OS1	OS2	OS1	OS2
UGC1s	-6.5%	-6.5%	13.7%	13.7%	-27.8%	-27.8%
UGC2	2.6%	-0.6%	7.4%	5.7%	-16.2%	-23.2%
UGC3s	-1.9%	-1.9%	7.8%	7.8%	-24.1%	-24.1%
UGC4s	13.8%	13.8%	4.4%	4.4%	-43.6%	-43.6%
UGC3	-1.4%	-1.4%	6.6%	6.6%	-19.7%	-19.7%
VoD1	-7.6%	-2.2%	11.3%	14.4%	-7.2%	-2.9%
VoD2	-1.2%	5%	10.8%	15.5%	-32.6%	-24.6%
VoD3	-5.6%	-5.2%	13.6%	17.1%	-41.2%	-35.3%
VoD4	-1.9%	-0.2%	12.3%	17%	-6.2%	-3.5%
VoD5	0.4%	-1.6%	13.7%	17.6%	-1.4%	-3.1%
VoD6	-5.9%	-6.1%	11.7%	18.6%	-19.6%	-17.8%
VoD7	-1%	-2%	9%	15.3%	-2.5%	-22.3%
Live1	0.2%	0%	10.9%	20.5%	-2.4%	-3.2%
Live2	-0.3%	-0.6%	14.4%	20.3%	-9.2%	-10.4%
Live3	0.2%	-13.8%	10.9%	16.2%	-12.4%	-25.8%

it can happen due to several reasons, including (i) player trying to improve user experience when network conditions allow, (ii) recovering from various errors or overly aggressive but aborted attempts for high-quality chunks, (iii) ensuring consistent transition between bitrates (frame alignment), and possibly others. This behavior has been observed in previous studies [35, 36], but has not been quantified at large scale. Note that replaced, or otherwise duplicated or repeated chunks, are important from the perspective of both the MNO (represent wasted network resources) and end user (represent waste of a limited data plan). We define CR overhead as the percentage of data in a video session transmitted due to replaced chunks.

Table 4 shows the CR overhead for video services in which we could read the chunk identifier and hence detect chunk replacement. A large number of sessions, as high as 80% for UGC3 have non-zero chunk replacement overhead. A non-zero chunk replacement overhead can still be explained by the fact that video players typically start by downloading lower quality chunks to quickly fill up the video buffer and replace them later with high quality chunks, given sufficient network bandwidth. However, a significant number of sessions (26% for UGC3) have chunk replacement overhead greater than 20% which is non-trivial. The overall network overhead due to chunk replacement is quite high (up to 9% for UGC3). This points to the need of looking into optimizing the trade-off between improved user experience and network overhead.

5.3 Impact of mobility and demand

The final example use case shows the impact of two important factors on video QoE, namely mobility and per-cell demand, as both are highly relevant to cellular networks.

Impact of user mobility: In a cellular network, user mobility often leads to hand-offs between cells and eNodeBs. While the duration of the hand-off itself is short, the reduced radio signal strength at the edges of a cell can cause degradation in the video QoE. We study the impact of user mobility on video session throughput (*ST*), which is most directly affected by varying radio signal.

We infer mobility in a video session by observing the number of eNodeBs in the session. We label a session *mobile* if there were at least three different eNodeBs encountered during that session. We use three eNodeBs as an indicator of mobility instead of two since a stationary UE could be handed-off to a cell in an adjecent eNodeB due to variation in received signal strength. We label the sessions with a single eNodeB as *stationary*, and ignore sessions with two eNodeBs in our analysis. We then calculate the relative change in median session throughput of *stationary* and *mobile* video sessions, referred to as $\Delta_{mobility}$, for every CP:

$$\Delta_{mobility} = \frac{\hat{ST}_{mobile} - \hat{ST}_{stationary}}{\hat{ST}_{stationary}} \times 100\% \quad (5)$$

Impact of mobility is shown in Table 5 as $\Delta_{mobility}$ with percentage of mobile sessions for the CPs on the two OS platforms. The key high level observation is that the average values of $\Delta_{mobility}$ are low, indicating that user mobility does not significantly impact video QoE. The reduction in *ST* predominantly in the range of up to 7% should not impact QoE as the distance between adjacent bitrates is typically by the factor of 1.5 to 2. However, the non-negligible percentage of mobile sessions (up to 20%) might warrant closer inspection to determine precise impact. The relatively low impact (and in some cases improvement) in *ST* could be attributed to a combination of potential reasons: (i) the video buffer can compensate for the temporary degradation in throughput during hand-off, and (ii) mobile devices are outdoors or in vehicles where radio signal typically has better quality than indoors.

Impact of higher video demand: The last-mile cellular radio link is one of the most challenging network environments, with frequent fluctuation in signal strength, quality and available bandwidth. Therefore, contention for resources is expected, and we examine its impact on video QoE, again considering change in *ST* ia subset of cells. We use the following heuristic to infer *higher video demand* in a cell. For every cell, we use 15-minute time bins and label the video demand in the cell as either *higher* or *lower*. A cell is considered under higher demand in the current time bin, if:

- *there are at least 10 video sessions during that time bin*, and
- *at least 50% of the video sessions have ST less than the median ST for the corresponding CP*

We compute the relative change in median session throughput of sessions under *higher demand* cells ($\hat{ST}_{higher}$) and sessions under *lower demand* cells ($\hat{ST}_{lower}$).

$$\Delta_{higher} = \frac{\hat{ST}_{higher} - \hat{ST}_{lower}}{\hat{ST}_{lower}} \times 100\% \quad (6)$$

This heuristic allows us to consider relative impact of video sessions on each other, and alleviates the need to consider busy hours, radio characteristics, or resource utilization. However, the total load of those cells is not driven by detected video streams, but overall traffic, Table 5 shows the *Impact of demand* as Δ_{higher} values for different CPs. Most of the CPs are significantly impacted by higher load with *ST* dropping by as high as 40% for UGC4s, but generally in the 20%-30% range. We also observe that the impact is different across CPs. This can be attributed to the difference in encoding bitrates, adaptation algorithms, presence of bitrate control, and known instability issues when adaptive players compete [13]. There is a strong indication that CPs that normally limit the bitrates in cellular such as VoD5 and Live1 are not highly impacted by increased demand.

6 RELATED WORK

QoE inference for operators: Existing work on QoE estimation from network measurements can be broadly divided into two categories. One category correlates network QoS with video QoE metrics using machine learning-based techniques. Consequently, these methods require ground truth on QoE for initial training of models. *OneClick* [21] and *HostView* [31] propose obtaining the subjective ground truth QoE through user feedback. *Prometheus* [12] and Orsolic et al. [38] suggest using instrumented clients to obtain the objective video QoE metrics. Dimopoulos et al. [23] and *BUFFEST* [33] inspect QoE metrics sent by the player to the CP to correlate them with network-level QoS metrics. However, the strong reliance on ground truth QoE metrics makes it difficult for NOs to implement this approach in practice.

Another category of QoE estimation approaches models video sessions based on the network traffic dynamics of the underlying video streaming protocol. This kind of QoE inference has been proposed for HTTP progressive streaming using network logs collected at TCP layer [39] or HTTP layer [22]. In our recent work, we provide a video session modeling approach for HAS video to estimate individual session QoE metrics using HTTP logs [35]. Our underlying QoE inference approach in *VideoNOC* is also based on an approach similar to this technique with addition of using session throughput for inferring encrypted video QoE. In addition to video, several efforts exist to infer QoE from network data for other applications such as web browsing [16, 41].

Video performance characterization: Several active measurement studies exist to characterize performance of the commercial video players [14, 36, 43] and CDNs [10, 11]. Similarly, large-scale measurement efforts to characterize video QoE have also come using performance data collected passively from commercial video players [29], CDNs [44] or both [26]. *VideoNOC* is similar in spirit to these studies but provides a unique MNO's perspective of video performance. Closest to the insights provided by *VideoNOC* is the work by Erman et al. [24] on characterizing cellular video performance. We provide a fresh take on their findings showing the cellular video evolution in the past few years as well as several additional insights such as impact of mobility and high demand on mobile video performance.

In-network optimizations: Chen et al. [20] present an HAS-aware scheduler at eNodeB to improve fairness among different

HAS streams. Existing works [18, 32] suggest network-assisted bitrate adaptation in HAS. We believe that using *VideoNOC* system can help find network locations needing attention and assess the impact of in-network modifications after deployment.

7 CONCLUSION AND FUTURE WORK

We present the design and implementation of *VideoNOC*, a prototype platform for video QoE monitoring in a cellular network. By addressing the key challenges, we demonstrate feasibility of large-scale video QoE inference and its benefits in understanding and improving adaptive video streaming in cellular environments. While our QoE metric estimation techniques are applicable to any type of network where passive measurements of HTTP/S traffic are available, we address specific scalability and cross-layer challenges to build a practical and efficient system on top of the existing cellular network architecture. The resulting data set, *Mobile Video QoE Metrics*, can be used to analyze interactions between the network and video streams, leading to insights on how to improve current and build better networks and video services in the future. Our future work will explore enriching the existing QoE data with transport-layer and relevant RAN KPIs (e.g., signal quality, channel utilization) for a deeper investigation into the root causes of performance problems, explore edge processing of RAN data, QoE inference on encrypted video traffic, and cooperation between MNOs and CPs.

ACKNOWLEDGMENTS

We thank our shepherd and the anonymous reviewers for their comments and feedback. This work is funded in part by NSF grant NETS 1409589.

REFERENCES

[1] 2008. The Transport Layer Security (TLS) Protocol Version 1.2. IETF RFC 3954. (August 2008).
[2] 2014. 3GPP TR 26.938 version 12.0.0 Release 12. (2014).
[3] 2017. A look inside ATTs global network operations center. (2017). http://fortune.com/att-global-network-operations-center/
[4] 2017. Cisco Visual Networking Index: Forecast and Methodology, 2016-2021. (2017).
[5] 2017. Conviva: in-app QoE monitoring. (2017). https://www.conviva.com/
[6] 2017. DASH. (2017). http://dashif.org/mpeg-dash/
[7] 2017. Squid: Optimising Web Delivery. (2017). http://www.squid-cache.org/
[8] 2018. ITU-T P.1203: Objective video QoE standard. (2018). https://www.itu.int/rec/T-REC-P.1203
[9] 2018. VQEG: Objective video quality assessment. (2018). https://www.its.bldrdoc.gov/vqeg/projects/audiovisual-hd.aspx
[10] V. K. Adhikari, Y. Guo, F. Hao, V. Hilt, Z. L. Zhang, M. Varvello, and M. Steiner. 2015. Measurement Study of Netflix, Hulu, and a Tale of Three CDNs. *Proc. of IEEE/ACM Transactions on Networking* (2015).
[11] V. K. Adhikari, Y. Guo, F. Hao, M. Varvello, V. Hilt, M. Steiner, and Z. L. Zhang. 2012. Unreeling netflix: Understanding and improving multi-CDN movie delivery. In *Proc. of IEEE INFOCOM*.
[12] V. Aggarwal, E. Halepovic, J. Pang, S. Venkataraman, and H. Yan. 2014. Prometheus: Toward Quality-of-Experience Estimation for Mobile Apps from Passive Network Measurements. In *Proc. of ACM HotMobile*.
[13] S. Akhshabi, L. Anantakrishnan, A. C. Begen, and C. Dovrolis. 2012. What Happens when HTTP Adaptive Streaming Players Compete for Bandwidth?. In *Proc. of ACM NOSSDAV*.
[14] S. Akhshabi, A. C. Begen, and C. Dovrolis. 2011. An experimental evaluation of rate-adaptation algorithms in adaptive streaming over HTTP. In *Proc. of ACM MMSys*.
[15] Apple. 2018. HLS. (2018). https://developer.apple.com/streaming
[16] A. Balachandran, V. Aggarwal, E. Halepovic, J. Pang, S. Seshan, S. Venkataraman, and H. Yan. 2014. Modeling Web Quality-of-Experience on Cellular Networks. In *Proc. of ACM MobiCom*.
[17] A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, and H. Zhang. 2013. Developing a Predictive Model of Quality of Experience for Internet Video. In *Proc. of ACM SIGCOMM*.
[18] S. Benno, J. O. Esteban, and I. Rimac. 2011. Adaptive streaming: The network HAS to help. *Bell Labs Technical Journal* (2011).
[19] P. Casas, B. Gardlo, R. Schatz, and M. Mellia. 2016. An Educated Guess on QoE in Operational Networks Through Large-Scale Measurements. In *Proc. of ACM Internet-QoE*.
[20] J. Chen, R. Mahindra, M. A. Khojastepour, S. Rangarajan, and M. Chiang. 2013. A Scheduling Framework for Adaptive Video Delivery over Cellular Networks. In *Proc. of ACM MobiCom*.
[21] K. T. Chen, C. C. Tu, and W. C. Xiao. 2009. OneClick: A Framework for Measuring Network Quality of Experience. In *Proc. of IEEE INFOCOM*.
[22] G. Dimopoulos, P. Barlet-Ros, and J. SanjuÃăs-Cuxart. 2013. Analysis of YouTube user experience from passive measurements. In *Proc. of CNSM*.
[23] G. Dimopoulos, I. Leontiadis, P. Barlet-Ros, and K. Papagiannaki. 2016. Measuring Video QoE from Encrypted Traffic. In *Proc. of IMC*.
[24] J. Erman, A. Gerber, K. K. Ramakrishnan, S. Sen, and O. Spatscheck. 2011. Over the top video: The gorilla in cellular networks. In *Proc. of ACM IMC*.
[25] A. Gerber, J. Pang, O. Spatscheck, and S. Venkataraman. 2010. Speed testing without speed tests: Estimating achievable download speed from passive measurements. In *Proc. of ACM IMC*.
[26] M. Ghasemi, P. Kanuparthy, A. Mansy, T. Benson, and J. Rexford. 2016. Performance Characterization of a Commercial Video Streaming Service. In *Proc. of ACM IMC*.
[27] E. Halepovic, J. Pang, and O. Spatscheck. 2012. Can you GET me now?: Estimating the time-to-first-byte of HTTP transactions with passive measurements. In *Proc. of ACM IMC*.
[28] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson. 2014. A Buffer-Based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service. In *Proc. of ACM SIGCOMM*.
[29] J. Jiang, V. Sekar, I. Stoica, and H. Zhang. 2013. Shedding Light on the Structure of Internet Video Quality Problems in the Wild. In *Proc. of ACM CoNEXT*.
[30] J. Jiang, V. Sekar, and H. Zhang. 2012. Improving Fairness, Efficiency, and Stability in HTTP-based Adaptive Video Streaming with FESTIVE. In *Proc. of ACM CoNEXT*.
[31] D. Joumblatt, J. Chandrashekar, B. Kveton, N. Taft, and R. Teixeira. 2013. Predicting User Dissatisfaction with Internet Application Performance at End-Hosts. In *Proc. of IEEE INFOCOM*.
[32] V. Krishnamoorthi, N. Carlsson, D. Eager, A. Mahanti, and N. Shahmehri. 2013. Helping hand or hidden hurdle: Proxy-assisted HTTP-based adaptive streaming performance. In *Proc. of IEEE MASCOTS*.
[33] V. Krishnamoorthi, N. Carlsson, E. Halepovic, and E. Petajan. 2017. BUFFEST: Predicting Buffer Conditions and Real-time Requirements of HTTP(S) Adaptive Streaming Clients. In *Proc. of ACM MMSys*.
[34] Y. Liu, S. Dey, F. Ulupinar, M. Luby, and Y. Mao. 2015. Deriving and Validating User Experience Model for DASH Video Streaming. *IEEE Transactions on Broadcasting* (2015).
[35] T. Mangla, E. Halepovic, M. Ammar, and E. Zegura. 2017. MIMIC: Using passive network measurements to estimate HTTP-based adaptive video QoE metrics. In *Proc. of IEEE TMA/MNM*.
[36] A. Mansy, M. Ammar, J. Chandrashekar, and A. Sheth. 2013. Characterizing Client Behavior of Commercial Mobile Video Streaming Services. In *Proc. of ACM MoVID*.
[37] New Relic. 2017. Mobile APM Features. (2017). https://newrelic.com
[38] I. Orsolic, D. Pevec, M. Suznjevic, and L. Skorin-Kapov. 2017. A machine learning approach to classifying YouTube QoE based on encrypted network traffic. *Multimedia Tools and Applications* (2017).
[39] R. Schatz, T. Hossfeld, and P. Casas. 2012. Passive YouTube QoE Monitoring for ISPs. In *Proc. of IMIS*.
[40] Spark. 2016. Apache Spark: Lightning-fast cluster computing. (2016).
[41] M. Trevisan, I. Drago, and M. Mellia. 2017. PAIN: A Passive Web Speed Indicator for ISPs. In *Proc. of ACM Internet QoE*.
[42] D. D. Vleeschauwer, H. Viswanathan, A. Beck, S. Benno, G. Li, and R. Miller. 2013. Optimization of HTTP adaptive streaming over mobile cellular networks. In *Proc. of IEEE INFOCOM*.
[43] S. Xu, S. Sen, Z. M. Mao, and Y. Jia. 2017. Dissecting VOD Services for Cellular: Performance, Root Causes and Best Practices. In *Proc. of ACM IMC*.
[44] H. Yin, X. Liu, F. Qiu, N. Xia, C. Lin, H. Zhang, V. Sekar, and G. Min. 2009. Inside the Bird's Nest: Measurements of Large-scale Live VoD from the 2008 Olympics. In *Proc. of ACM IMC*.
[45] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli. 2015. A control-theoretic approach for dynamic adaptive video streaming over HTTP. *ACM SIGCOMM Computer Communication Review* (2015).