Swipe Along: A Measurement Study of Short Video Services

Shangyue Zhu
University of Notre Dame
United States

Theo Karagioules
AT&T Labs — Research
United States

Emir Halepovic
AT&T Labs — Research
United States

Alamin Mohammed
University of Notre Dame
United States

Aaron D. Striegel
University of Notre Dame
United States

ABSTRACT

Short videos have recently emerged as a popular form of short-duration *User Generated Content* (UGC) within modern social media. Short video content is generally less than a minute long and predominantly produced in vertical orientation on smartphones. While still fundamentally being streaming, short video delivery is distinctly characterized by the deployment of a mechanism that pre-loads ahead of user request. Background pre-loading aims to eliminate start-up time, which is now prioritized higher in *Quality of Experience* (QoE) objectives, given that the application design facilitates instant 'swiping' to the next video in a recommended sequence. In this work, we provide a comprehensive comparison of four popular short video services. In particular, we explore content characteristics and evaluate the video quality across resolutions for each service. We next characterize the pre-loading policy adopted by each service. Last, we conduct an experimental study to investigate data consumption and evaluate achieved QoE under different network scenarios and application configurations.

CCS CONCEPTS

• **Information systems → Multimedia streaming**.

KEYWORDS

Video streaming, short-form, measurement

ACM Reference Format:
Shangyue Zhu, Theo Karagioules, Emir Halepovic, Alamin Mohammed, and Aaron D. Striegel. 2022. Swipe Along: A Measurement Study of Short Video Services. In *Proceedings of Multimedia Systems (MMSys)*. ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3524273.3528186

1 INTRODUCTION

Over the past decade, significant growth of online video sharing has been spurred by novel streaming techniques such as *HTTP Adaptive Streaming* (HAS) and the sustained adoption of smartphone devices. Concurrently, wireless network capacity and speed improvements lend support for a variety of video offerings that may appeal to larger user-bases, such as 360 videos, Augmented Reality, live and game streaming, and teleconferencing. One such video offering is positioned at the intersection of UGC and social media, namely that of continuous short-duration video streaming (referred to as simply *short videos* in this paper), which was initially popularized by TikTok and then adopted by many existing video streaming services [34]. The scope of this paper is to characterize and experimentally compare four of the most popular short video services: *YouTube Shorts*, *TikTok*, *Instagram Reels*, and *Facebook Watch*.

Short videos typically do not exceed one or a few minutes in duration and are primarily UGC. Short video content is most often recorded on smartphones (vertical orientation), contrary to long-form content that is typically recorded with dedicated camera equipment and is often professionally produced. Short videos are delivered over HTTP and largely under the adaptive streaming principle, yet exhibit some new properties when compared with long-form streaming, in regard to encoding characteristics and the prioritization of the various QoE objectives.

Short video content is typically encoded in a multitude of variants or quality levels (bitrates and resolutions), offering video clients granularity in bitrate switching to adapt to varying network conditions. Adaptation may occur between or within videos, depending on the individual design of each service. Such content organization resembles long-form videos, although these typically have more encoding variants [22]. In addition, each short video service has its own proprietary system design, which may pose an additional challenge in comparative perceptual *Video Quality* (VQ) analysis. To this end, we statistically analyze large video metadata samples per service, focusing on the distribution of encoding bitrate and video duration per quality variant. Moreover, to conduct a systematic VQ comparison for all variants offered per service, we create and curate a set of diverse complexity videos, which we upload individually to each studied service. We find that there exist significant differences in design decisions between services, with some offering a better trade-off between perceptual VQ and bandwidth requirements than others. Our analysis shows that encoding bitrates across quality variants may not always conclusively correlate to perceptual quality.

Generally, in terms of overall QoE, offering high quality uninterrupted video streaming has been a primary streaming objective, particularly in the more volatile mobile networks. Yet now, as the shorter video duration creates the user expectation of fast access to content, start-up and wait time between videos become crucial to overall short video user experience. Moreover, since short videos are consumed as a sequence of videos accessed via instant 'user swiping', start-up and wait time become even more stringent QoE

Multimedia Systems (MMSys), Jun 14 - 17, 2022, Athlone, Ireland
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9283-9/22/06…$15.00
https://doi.org/10.1145/3524273.3528186

objectives. Therefore, to amplify user-engagement through seamless video consumption, short video services have adjusted their clients to implement a pre-loading mechanism. Parts of several videos are downloaded in parallel to the current video playout, ensuring that the client's buffer contains a number of individual videos at any given time. This constitutes a fundamental design difference from long-form video streaming, where the buffer would typically host future content of only the playing video. Thus, in short videos the download process, given the preloading operation, becomes parallel as opposed to serial (long videos), creating characteristically different traffic patterns.

Pre-loading policies vary across services with some offering constant data-size pre-loading, while others employ time-based approaches, each with different parameterization. Thus, we have focused on characterizing the different pre-loading properties for each of the studied services. Moreover, as the impact of such pre-loading operations on bandwidth requirements and QoE is not well understood, we provide experimental insights under a series of test configurations. We find that pre-loading can indeed improve QoE under certain circumstances, such as long user-sessions. Nonetheless, careful implementations are required since aggressive pre-loading may not only produce high data wastage, but can also create significant strain on the network, ultimately eliminating any prior QoE gain. The main contributions of this work are:

- *Service design.* We provide a statistical analysis of video duration and encoding bitrate, which we relate to quality level labeling. We find significant differences in bitrate and resolution selections, as well as quality labeling, making direct comparison challenging.
- *Video quality analysis.* We evaluate the video quality and encoding ladder offered by each video service. Results indicate large differences in VQ between services, as well as trade-off inefficiencies between VQ and encoding bitrates.
- *Pre-loading characterization.* We characterize the pre-loading policy for each of the studied services and unveil related parameters such as the size, number and duration of pre-loaded videos. Vastly different policies are detected, resulting in varying risks of wasted data vs. achieving uninterrupted playback.
- *Data consumption and QoE evaluation.* We conduct experiments under a series of network scenarios and application configurations ('Data Saver' mode ON - OFF) to gain insights on behavior, data usage, and overall QoE. Stalls and wait time are significantly affected by data saving settings and overall service designs, with some services achieving minimal disruption, while others facing major challenges in competitive and constrained conditions.

The paper is organized as follows. Section 2 reviews recent research in video streaming and initial work on short videos. Section 3 specifies the methodology and tools used, while Section 4 provides insights on content characteristics and the video quality offered by each service. Section 5 characterizes the pre-loading policy for each studied service. Section 6 presents experimental results of our investigation on data consumption and achieved QoE, under a series of network conditions and application configurations.

2 RELATED WORK

Video streaming research spans a wide set of works that includes techniques that optimize QoE, frameworks to evaluate perceptual video quality, and new video applications. Optimizing QoE is typically the main objective of any development around video streaming. Streaming protocols [17, 27] employ client-side techniques to provision against fluctuating network conditions, that may take various input signals to perform a bitrate adaptation decision. Throughput-based adaptation algorithms use bandwidth estimates based on measured throughput [41], while buffer-based strategies adjust bitrate based on the instantaneous buffer level [16]. Most commonly, a combination of inputs is considered [31], while the algorithmic approach may employ heuristics [32], optimization [8] or machine learning [23]. Adaptation may alternatively occur server-side [7], or be network-assisted [13].

To facilitate the evaluation of video delivery and specifically the perceived VQ, multiple assessment frameworks have been proposed [10, 20]. In particular, Tu *et al.* [35] proposed a model based on frame-level quality scores. Robitza *et al.* [29] provide tools for audiovisual quality evaluation via the QoE standard ITU-T Rec. P.1203 [18]. Recently, *Video Multimethod Assessment Fusion* (VMAF) [21] has emerged as a widely-adopted quality assessment method, that has shown a high correlation with human perception.

The study presented in this paper aims to provide a comprehensive video quality comparison and video delivery performance evaluation among four popular short video services. A similar investigation was conducted by Chen *et al.* [9], who focused on the characterization of only one service, namely that of Douyin (a variant of TikTok available exclusively in the Chinese market), providing data consumption insights and available content statistics, such as the average video size and length. LiveClip [15] is another work that focuses on the system design of Douyin, but also explores alternative download strategies, primarily in regard to bitrate selection during the preloading operation, in an effort to provision against data wastage caused by fast swiping. While pre-loading policies [19, 37, 38] are primarily designed to reduce wait time between consecutive video playouts, video QoE may exhibit substantial fluctuations over different short video sessions. Thus, Guo *et al.* [14] employed reinforcement learning to generate quality-driven bitrate decisions for short video sessions. Zhang *et al.* [39] analyzed one of the top short video services in China and present the distribution of short video length and bitrate based on the content provider's database. Our work aims to complement such existing research by analyzing international large-scale services, providing a direct comparison of their system design.

3 METHODOLOGY

In this section, we introduce several experimental methodologies and tools that we employ to collect insights into the design and operation of short video services. Our work covers four popular services, which we refer to as Service 1-4 in the remainder of the paper. Due to the varying designs and closed nature of these services, we generally used black-box approaches for measurement and evaluation. Our methodologies could be grouped as follows: 1) API and browser-based analysis, 2) custom video creation and upload, and 3) active experiments under different network conditions.

In addition to creating custom videos, we also analyzed larger samples of videos already available on each video service. This combination of video sources calls for various tools to be employed, broadly classified as 1) packet trace collection tools, 2) screen recording tools, 3) tools for scripting and automation of app invocation and video playback, 4) API and web application tools, etc.

3.1 API and browser-based analysis

To obtain ground truth information on encoded variants, such as resolutions and bitrates from each video service, as well as to download the variants for analysis, we applied several approaches, which allowed us to inspect the design decisions of services without having internal access.

Selenium [25] is a web-browser automation framework, which can scrape web pages and provide information about the objects, including downloading video variants and their metadata, to the extent provided by each service. We fed Selenium with video URLs to get the web page content in HTML or scripted form and use custom parsers to extract quality labels (360p, 720p, etc) and bitrates. This process involved multiple iterations due to the daily limit some services impose (~100 video URLs) when using Selenium, ultimately obtaining a large number of video URLs per service.

Tools such as youtube-dl [6] and pafy [4] were used as plugins in Selenium for video metadata extraction (listing all audio and video variants, including resolution, size, bitrate, codec and duration).

Browser Developer Tools [33] are web testing components of some browsers (e.g., Chrome, Firefox). We used them to: i) observe the *Server Name Indication* (SNI) or domain names of video servers, ii) data consumption (video object sizes), and iii) validate the captured data in experiments.

3.2 Custom video set

One of our goals was to gain insights into the service design of each video service, which includes selections of VQ levels, with associated resolutions, frame rates and bitrates, buffering or preloading, and data savings methods. Our approach was to create a set of custom videos, upload them to every service, and then download all variants generated by each service. We recorded videos using a representative popular smartphone (iPhone XR) in a fixed video format with 1080p resolution (1080x1920 pixels) and 30 FPS (frames per second), which is the most popular upload format for short video services [11]. Videos were encoded using *Constant Bit Rate* (CBR) in the H.264/MPEG-4 format. We created a set of 15 videos and uploaded them to each service twice in the same sequence within a single user account. Uploading twice creates a sequence of 30 videos that will become useful in the analysis of pre-loading.

Our custom videos were divided into 3 groups consisting of 5 videos each based on motion levels, so we could analyze data usage and VQ depending on this feature. Low-motion videos had static backgrounds (e.g., furniture) and characters. The average encoding bitrate of this group was 5.1 Mbps. Medium-motion had generally static background and moving characters (e.g., a person dancing or walking), with an average bitrate of 12 Mbps. The high-motion group had both dynamic background and characters (e.g., driving), with an average bitrate of 20 Mbps. Overall average video duration was 32 seconds, in the range of 15 to 44 seconds. This approach allowed us to compare and contrast design choices made by video services in providing users with different VQ levels, efficiency in encoding and data usage, as overall delivery QoE.

To evaluate the perceptual VQ of the video variants offered by different services (Section 4), we used the VMAF score [40], which was obtained by comparing the transcoded video to the original. Degradation to the original video comes from reduced resolutions and bitrates. VMAF scores were computed by the open-source library `libvmaf` (we used v0.6.1) via execution of the `phone_model`. VMAF score has a range of 0 to 100, where 100 signifies the highest quality (equal to the original content), whereas 0 indicates the lowest possible quality, while a 6 point difference is just noticeable [1]. VMAF is a relative metric, so it is best suited for assessing degradation level of the original video, rather than subjective quality. While non-reference metrics could also be useful for VQ evaluation, they may be more appropriate for non-UGC services. For our scope, VMAF was a more suitable choice due to its high accuracy and the relative ease of its application on all generated variants, after uploading original content.

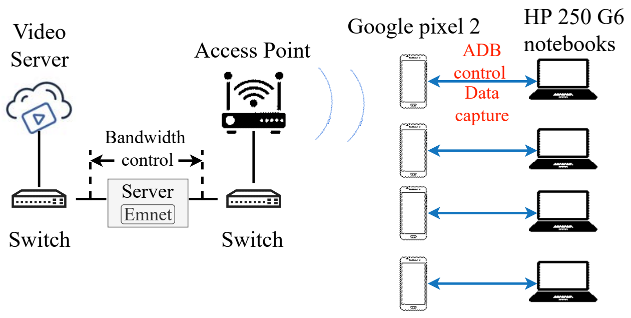


Figure 1: Active experimentation test setup.

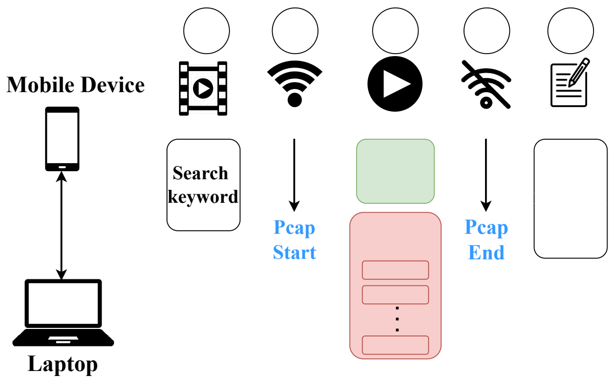


Figure 2: Test design for pre-loading analysis.

3.3 Active Experiments

Important goals of this work included evaluating pre-loading policies and data consumption in varying network conditions for which we set up a controlled network environment (Fig. 1). Video applications ran on four Android phones (Google pixel 2, Android 11) which were connected to four HP 250 G6 laptops that monitored and collected packet traces using Video Optimizer [3] and controlled by *Android Debug Bridge* (*ADB*) commands [12]. ADB was used to simulate user actions such as automatically invoking video apps and swiping the screen. We used an iOS device (iPhone XR, iOS 13.5) connected to a Macbook Pro (OS 10.13), which implements a *Remote Virtual Interface* (RVI) to enable packet capture with tcpdump [5]. Since iOS devices do not allow for automatically simulating user actions, we resorted to manual control for a subset of experiments. Smartphones are connected to the network via a NETGEAR R6230 WiFi access point, which is connected to the internet and video servers via switches. We used EmNet [24] running on the switch to control the network bandwidth. In addition, we also ran experiments in a live cellular network in the Unites States.

Monitoring logs were used to synchronize application actions with packet trace data. The monitoring logs recorded timestamped information about swiping actions, that helped identify and define download periods per video sequence. To evaluate the wait time between videos and stalls, we used the screen recording function of Android devices to capture the entire experiment visually as ground truth data and manually extracted the metrics. While some automated solutions for single video analysis and stall detection are available [36], short videos have the additional requirement of handling sequences of videos with swiping events in addition to wait time measurements. Thus, automation was not applicable for our experimental evaluation of short video systems.

3.3.1 Pre-loading characterization methodology. Our active measurement environment allowed us to design a custom experiment to observe the pre-loading behavior for all studied video services. The tests were operated manually as a typical user would use the app. As shown in Fig. 2, we started by a keyword search (Step 1) to reach our user page with the custom video set. In Step 2 we started the packet and screen captures just before starting the first (main) video playback. We played the first video in full, while each service pre-loaded additional videos (pre-loaded videos) in the background (Step 3). Once the video finished playing, we turned off the internet connection and stopped capturing the packet trace (Step 4). We swiped up the screen, while the network connection was disabled, as many times as needed to capture the number and duration of pre-loaded videos (Step 5). We performed 10 iterations per service.

With screen recording we could observe the number and duration of pre-loaded videos, while from packet traces we could compute the data consumption for the main and pre-loaded videos. We started by extracting the total video data from the packet traces using SNIs from the TLS headers corresponding to video flows. SNIs were obtained by inspecting URLs of video objects inside the browser developer tools. The total data could be partitioned according to the timestamp of the main video playback completion. All data delivered after that point were for pre-loads, while the data prior were a sum of the main and pre-loaded videos. We could further identify the main video size from the browser tools or API calls. All remaining data indicate the size of pre-loads.

4 SERVICE DESIGN CHARACTERIZATION

In this section, we characterize the aspects of service design relating to VQ, resolution, and encoding bitrates. We use both a large sample of available videos on each service and a custom video set. Metadata for each video, obtained using Selenium, reveal the following key features of each variant:

- Label: A “quality label” typically represented by resolution notation (e.g., 360p, 720p, 1080p for video)
- Resolution: Actual spatial resolution of the variant (e.g., 360x640, 720x1280)
- Bitrate: Average bitrate of the variant (e.g., 1.5 Mbps)
- Codec: Encoding standard (e.g., AVC, VP9, AV1 for video; AAC, AC3 for audio)
- Size: Data size of the variant

Other common features across variants are also included, such as video URL and duration. Spatial resolutions also reveal vertical orientation of videos, compared to long-form videos. We focus on video variants only in our analysis. We anonymize service names in the presentation of results to preserve content provider privacy. As our analysis pertains to a relative comparison of the service design for the different short video workflows, all findings remain both valid and valuable even without revealing identity.

4.1 Service design characterization via sampling

We explore service design by collecting large samples of video metadata from all studied services between August and October 2021. The summary of the sample set is shown in Table 1. We notice a large number of resolutions associated with a smaller number of labels in most services, suggesting precise VQ tuning of variants. We note that the number of labels used by two of the studied services was exceptionally large. We indicate with a “+” instances that we believe may carry an even larger number of labels and resolutions in the wild. We also observe that some labels and resolutions are dominant in some services, while more uniformly distributed in others. For example, Service3 has a relatively balanced usage of 270p, 260p, and 1080p (38%-43% of samples), while in Service4, 240p and 640p dominate in 72% and 51% of samples, respectively. However, Service3 and Service4 only use up to 6 variants per video. This variety may pose a challenge in comparing quality levels across services. In particular, Table 1 indicates the portion of each codec in the sample. Content providers typically opt to use multiple codecs per video but may not use every codec for every video.

We further record the proportion of samples where most popular codecs are used. Notably, Service1 only uses AVC, while the other three use VP9 substantially or predominantly. We assume that the prevalent variants generated by each service are the ones served to users by default, and thus we do not seek to analyze all variants in detail. We also see anecdotal evidence that Service2 delivers VP9 format to both iOS and Android devices in our tests, as both major mobile platforms support it [28]. Therefore, we proceed to compare AVC from Service1 and VP9 from the other services. Our sample contains a collection of currently popular videos, as identified by a ‘trending’ tag or via homepage recommendations. In addition, we searched for and accessed content by popular creators per service.

Table 1: Encoding information per service.

	Samples (#)	Labels (#)	Resol. (#)	VP9	AVC
Service1	2,472	1	8+	0%	100%
Service2	3,583	8	12+	98.3%	100%
Service3	2,264	18+	42+	88.7%	69.4%
Service4	1,821	57+	215+	86.1%	67.3%

Table 2: Common resolutions per service.

	360p	720p	Nearest resolutions
Service1	-	576x1024	576x1024 (720p)
Service2	406x720	810x1440	608x1080 (480p)
Service3	360x640	720x1280	682x854 (640p)
Service4	364x680	720x1280	640x1138 (640p)

We use two common labels (360p and 720p) to explore the associated resolutions and bitrates. We consider labels merely as logical groups and a coarse proxy for VQ rather than as an indicator of exact VQ or subjective experience. Table 2 shows the most common resolution for each label. We observe that the 360p spatial resolutions of the three services are similar (Table 2), while for 720p some services tend to deviate from the label-implied resolutions. We also include the set of the nearest resolutions to the single one we detected for Service1, and the labels associated with those resolutions in the same table. Here we also observe significant variety of resolutions and labels.

We further explore the range of bitrates per label across services, to understand how quality labels relate to data usage and bandwidth requirements, based on encoding bitrates. Fig. 3a compares 360p bitrates for three services that offer it (thin lines), and 720p bitrates for all four services (thick lines). We use the sub-sample where every video has both 360p and 720p labels for the three services offering multiple labels, so that comparison between labels becomes feasible. For 360p, Service2 and Service3 show a narrow range of encoding bitrates resulting in tight distributions with over 90% of samples under 500 Kbps and with about 70% of samples falling between approximately 400 and 500 Kbps. In contrast, Service4 has a wider range, with 20% of samples between 600 Kbps and 1.6 Mbps.

Generally, 720p bitrates exhibit wide ranges (as expected), except for Service3, whose bitrates are in in $[0.3, 2.5]$ Mbps, with more than 90% of samples encoded below 1.4 Mbps. This service uses a constant bitrate difference between the 360p and 720p labels for most samples. Service2 appears to aim for under 2.5 Mbps at 720p, but without tight ranges and the largest difference between bitrates used for 360p and 720p. Less than 10% of samples in several services exceed 2.5 Mbps for 720p.

We next analyze the set of nearest resolutions, i.e. resolutions that have similar height and width dimensions, to see if similar resolutions relate to similar bitrates, and therefore data usage and bandwidth requirements (Fig. 3b). Service2 exhibits a tighter range than others, and peaks at 1.5 Mbps, with most samples encoded under 1 Mbps. Other services have a wider set of bitrate ranges, with potential bandwidth requirement implications. Seemingly services with tight distributions aim to ensure that certain resolutions can reliably play under certain network conditions (such as under 1.5 Mbps, where distributions either end or have a distinct knee).

Amongst all differences, similarity in bitrate distributions of Service1 and Service4 is notable and suggests similar encoding parameters and optimization approaches, with no strict limits and allowing much higher bitrates in the tails.

The duration of the video content is one of the most significant differentiators when comparing short videos to traditional on-demand streaming. Most short video services generally limit the duration, as seen in Fig. 3c. Service2 and Service4 have strict limits at about 30 and 60 seconds, respectively. Service1 appears to have a small number of videos longer than 60 seconds, while Service3 offers a wider range in duration (not an exclusively short video service). We show a truncated *Cumulative Distribution Function* (CDF) for Service3 up to 180 seconds.

In summary, large sample analysis reveals significant differences in design decisions between short video services that look very similar to users. Findings suggest that in short video systems, multiple resolutions are grouped in only a few quality labels, which prohibits their use as conclusive indicators of bandwidth requirements. The differences in encoding bitrates for the same labels and resolutions also raise questions on how they all relate to perceptual quality.

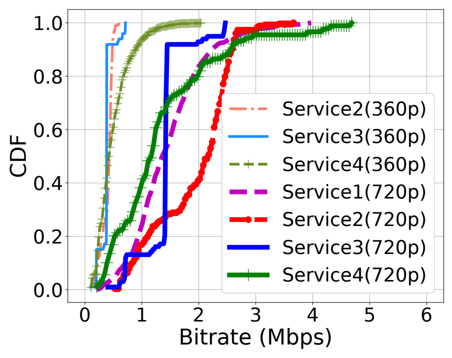
4.2 Perceptual quality analysis using custom videos

As outlined in Section. 3, we upload a custom video set to each service and then download all generated variants. This allows us to measure degradation of the original video and obtain a form of VQ analysis that labels and resolutions could not conclusively provide.

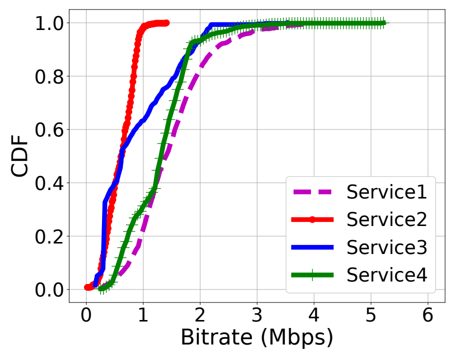
VQ vs. labels. Findings from the large sample indicate that different VQ optimization approaches are used by services. To relate labels with perceptual quality, we plot VMAF vs. labels in Fig. 4 for the three motion-based video groups of Section 3.2. Average VMAF is plotted with error bars indicating standard deviation. For clarity, most common labels are plotted, although some services offer additional labels.

We first observe that generally for all services, VMAF drops as motion increases, most noticeably for high-motion videos. Diminishing returns in VMAF are also observed, as expected, since small phone screens do not offer better VQ perception beyond a 720p resolution. These are general qualitative characteristics we would expect to see, meaning there are no obvious and large aberrations in the overall service designs related to VQ within each service.

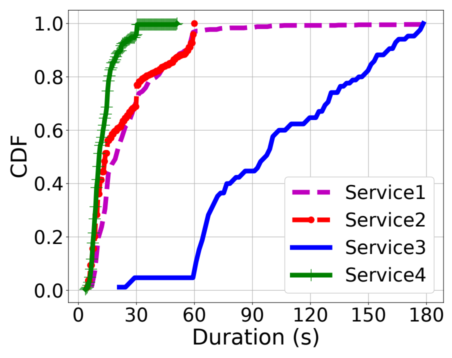
However, we observe numerous differences in some aspects of system design from Fig. 4. At the high end, a clear distinction of Service2 is that it offers 1080p, but with negligible VQ improvement, as expected. Recall that the original video is 1080p, but with all these services being smartphone-oriented, most services opt to not go past the VQ where returns significantly diminish, which corresponds to 720p. At the low end, we note that 2 services offer variants at extremely low quality (VMAF of 20 and below), which may be due to either default encoding settings or an attempt to facilitate ability to deliver content even at the most adverse network conditions. Service1 further stands out by offering only one variant for all motion levels that we could detect and compare, and although it uses only AVC, VMAF indicates that it compares extremely well



(a) Video bitrates for 360p and 720p labels.

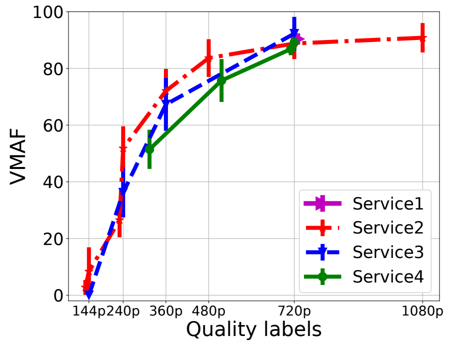


(b) Video bitrates for nearest resolutions.

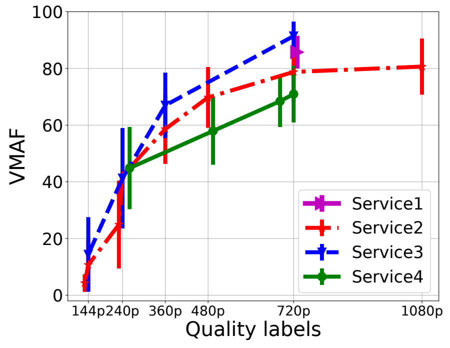


(c) Video duration.

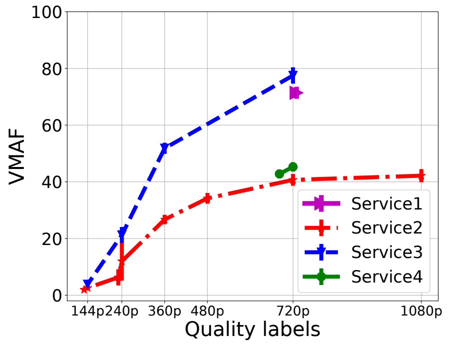
Figure 3: Statistical analysis for video bitrate and duration.



(a) Low-motion videos.



(b) Medium-motion videos.



(c) High-motion videos.

Figure 4: VMAF for various quality labels.

with VP9 encodings of other services. Finally, Service4 limits high-motion videos to a very small range of labels.

It can be noted that labels generally relate well in perceptual VQ within all services. However, between services, labels do not necessarily correspond to the same or similar VQ. Ideally, we should see the same or similar VQ (VMAF score) for the same label, somewhere within 6 VMAF points, as the threshold of noticeable difference. We note significantly larger gaps in multiple instances, while the closest VQ is observed for low-motion videos across all labels.

For medium-motion videos, the gap in VQ widens for 360p and higher labels, with a 20 VMAF point difference seen between Service2 and Service3 for 720p. For the high-motion video group, more than 30 VMAF points are measured between Service2 and Service3 for 720p. We also did not expect to see that similar VMAF values would be represented by very different labels across services (e.g., VMAF of near 70 comes from 360p label of Service3 and from 720p of Service4 in Fig. 4b).

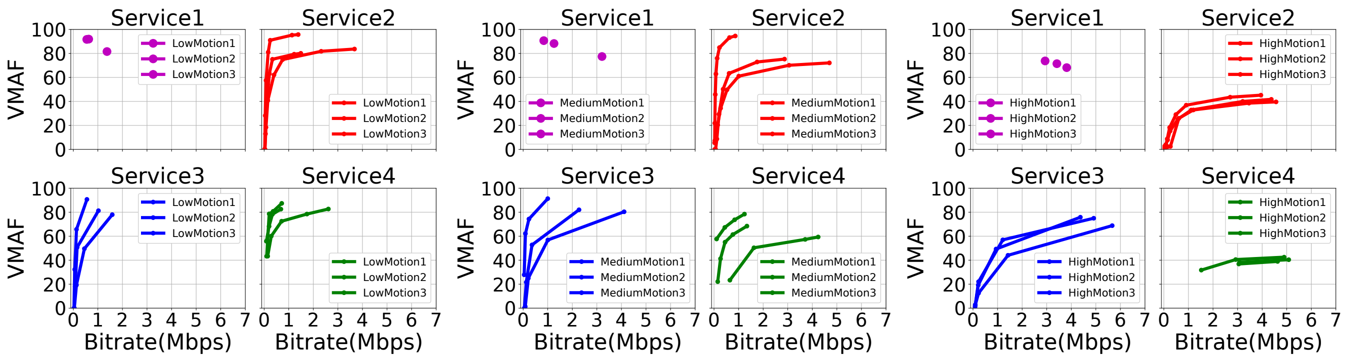
VQ vs. encoding bitrates. Some of the observed differences in VMAF and labels of offered variants prompt us to delve into more detail regarding encoding efficiency across multiple services. Different VQ for the same type of videos raises questions about encoding efficiency, in particular of the bitrates used to produce

Table 3: Average encoding bitrates for Service3 in Mbps.

	720p	360p	240p	144p
Low-motion bitrate	0.52	0.13	0.04	0.01
Medium-motion bitrate	1.60	0.32	0.10	0.04
High-motion bitrate	5.10	1.50	0.17	0.07

variants of certain qualities. Hence, we randomly select three videos from each set of motion levels to observe the relationship between VMAF and encoding bitrates, where each bitrate is associated with a quality label. The bitrate directly corresponds to data usage and bandwidth requirement to deliver a video under various network conditions and when combined with other encoding parameters, it directly impacts the VQ of that variant. We provide an illustrative example of generated average bitrates in VP9 for several labels used by Service3 across motion levels in Table 3.

Fig. 5 compares VMAF and its corresponding bitrate for the same videos across services. First, we observe similar curve shapes for three services that produce multiple detected bitrates (as opposed to Service1 that has only one), as we observed for labels. This is expected, as labels still relatively correspond to VQ and encoding bitrate requirements within each service. Diminishing returns on



(a) Low-motion videos. (b) Medium-motion videos. (c) High-motion videos.

Figure 5: VMAF for various encoding bitrates.

VQ are noticeable, sometimes appearing very quickly, as the curves sharply flatten past certain bitrates.

As motion levels increase, significantly more bits are needed to maintain VQ, sometimes ineffectively. Some medium-motion samples prove to be a challenge for Service4, while both Service2 and Service4 struggle with high-motion videos (degraded to VMAF of below 50). We further note that some services expend significant amount of data to produce variants with negligible VQ benefit (e.g., Service2 and Service4 for all motion levels), while Service3 appears to aim to expend high bitrate for one variant that significantly improves VQ. Service1 manages to maintain relatively high VMAF scores at moderate bitrates.

In summary, different video services maintain a relatively high VMAF score of at least 70 for most low and medium-motion videos at bitrates up to 2 Mbps. However, the high-motion videos reveal significant differences, where similar VQ could result in 50% higher data usage in some services (e.g., 3-4 Mbps vs. 5-6.5 Mbps for Service1 and Service3, respectively).

5 PRE-LOADING POLICIES

As noted earlier, short video services are commonly characterized by the introduction of pre-loading, which makes user experience seamless by facilitating instant playout of consecutive videos in a playlist. Thereby, interactivity and choice are encouraged by making the browsing process (swiping) a nearly zero cost activity. In reality the associated data cost has simply been "pre-paid" and shifted back in time, due to pre-loading. To wit, Fig. 6 provides a brief glimpse into one of the pre-loading policies under such conditions where network connectivity is disabled just as the first video completes playing. In spite of the network being disabled, portions of the next three videos are still able to be played (via swiping) with eventually the fifth and sixth videos having no content.

However, there are many subtle design choices that emerge with respect to the employed pre-loading policy, such as the number of videos that should be pre-loaded. Too few may result in the user swiping ahead of network performance but too many may cause wasted data. The number of pre-loads also draws the likelihood of engagement for the video selection of the user. When services recommend videos well, users are more likely to watch all the videos thus being less likely to rapidly swipe. Conversely, poor recommendations may result in rapid browsing or forced content refresh, ultimately leading to further data wasting.

Similarly, what criterion should be used in terms of how much content to buffer? Should the length of pre-loading be size dependent or time (playback) dependent? One provides a consistent data cost while the other ensures a more consistent resilience and playback amount. That cost in turn can be impacted by choices with respect to coding and resolution that effectively may be attempting to guess network conditions significantly forward in time. Taking the third video from Fig. 6, it could be played after the full playback of videos 1 and video 2 or it could be quickly swiped to by the user with network conditions varying dramatically from when the pre-loading occurred to when the actual playback and subsequent buffering need to occur.

In this section, we explore, compare and contrast the operating principles, design choices and attributes of the pre-loading policy of each video service. We apply a consistent testing methodology across all services as described earlier and offer observations on each policy, summarized in Table 4.

5.1 Service1

We begin by conducting tests following the methodology of Section 3.3.1 in a network environment with sufficient bandwidth (20 Mbps), where speeds are confirmed by SpeedTest [2]. Fig. 7 plots the time series of data bursts following each swipe indicated by

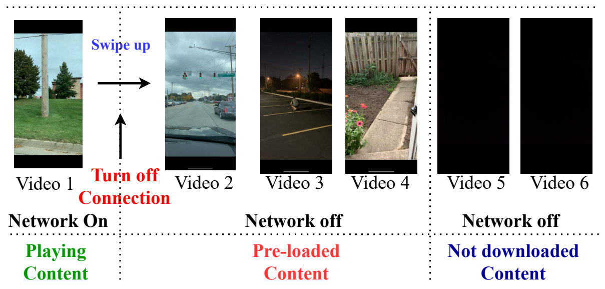


Figure 6: Methodology for measuring pre-loading length.

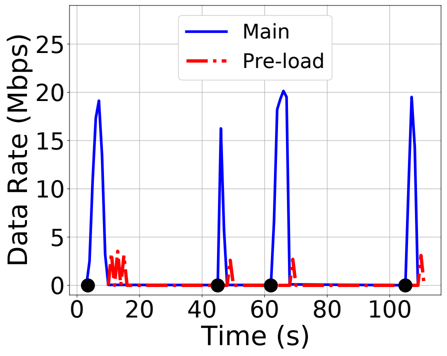


Figure 7: Data rate at 20 Mbps, Service1

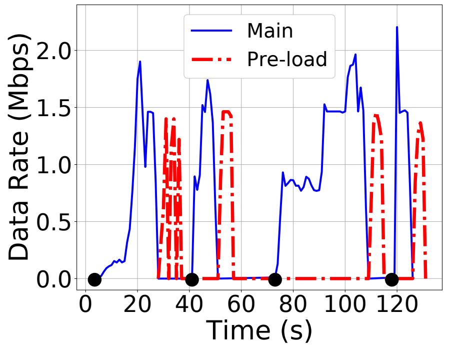


Figure 8: Data rate at 2 Mbps, Service1

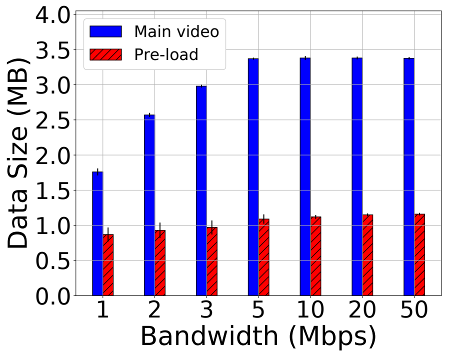


Figure 9: Data size vs. bandwidth, Service1

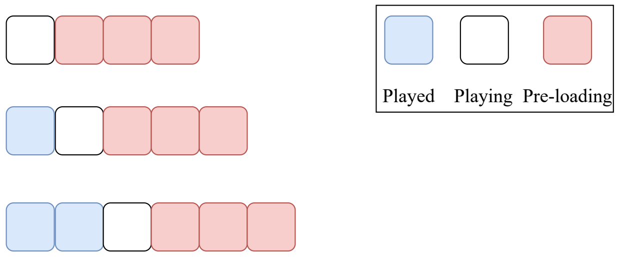


Figure 10: Service1 pre-loading sequence.

dots on the x-axis. A single test run has 10 swipes, but we plot fewer for clarity. The y-axis denotes the downloaded data rate, and the x-axis denotes time. Each video is watched completely before the swipe occurs. The viewed video ('Main') is denoted in blue, while pre-loading data in red. We distinguish the main video and pre-loading data based on our recorded playing time and video data (size, duration) extracted using Selenium. We leverage the gaps in data bursts, TCP sequence numbers and timing of packets to confirm the distinction between the main and pre-loaded videos when delivered over the same TCP connection.

Notably, in the sufficient network environment (20 Mbps), Service1 quickly pre-loads videos and shows idle connection periods, after each pre-load and before the next video needs to be played (following a swipe). In contrast, in Fig. 8, we plot the impact of alow bandwidth of 2 Mbps which has a significant impact on downloading and total time to view all of the videos (last video is swiped to start nearly 25 seconds later). The time series of the low bandwidth case illustrates that the app has to adapt to the conditions of minimal idle time between download bursts, and that pre-loading does not appear to compete with the main video download; they are instead managed to run consecutively. Initial pre-loading is more challenging, until steady-state is reached, where the app maintains the number of pre-loaded videos. The intended pre-loading sequence following the described policy is illustrated in Fig. 10.

Size versus Time: One notable design choice for Service1 is the apparent reliance on a very tight range of pre-load sizes rather than targeting a specific duration for each video. To demonstrate this phenomenon, we select two videos from the previous experiment, with one video set as the watched video and another video to observe with regards to pre-loading. Both videos are sufficiently large (around 3 MB) and long (over 40 seconds). Fig. 9 plots the average size of the watched video versus the average size of pre-loaded video as the bandwidth is varied in a very wide range (1-50 Mbps) with each data point sampled ten times. While the watched video download size varies significantly, the pre-loaded size is in the close range of 1 MB. One would expect that any adaptation potentially employed for the time-based pre-loaded video would make the pre-loaded size vary significantly across the various network bandwidths. Overall, Service1 preloads 1MB for each of the next three or five videos, depending on the OS (iOS vs. Android). We summarize all characteristics of pre-loading policies in Table 4.

5.2 Service2

Service2 employs a relatively simple pre-loading policy by pre-loading only one video. Hence, we omit a figure. We find that Service2 uses a time-based approach, pre-loading 20 seconds of video data for iOS and 30 seconds for Android. This often results in pre-loading videos fully, given the duration distribution of Fig. 3c.

Our investigation yields that Service2 may employ either TCP or QUIC as the transport protocol for individual sessions. Analysis of this and other services that use QUIC is more challenging due to lack of TCP sequence numbers, so we resort to delineating data bursts using a 100 ms timeout. We detect that pre-loads are always in the first burst, which is large, while subsequent small bursts load the remainder of the main video. The size of the pre-load in this case is simply the total size of the consecutive bursts less the video size retrieved from metadata.

We further discover that pre-loading starts near concurrently with the main video load or with a short delay, as established by swiping immediately after the main video starts playing and finding pre-loaded data already available.

5.3 Service3

Fig. 11 illustrates the pre-loading sequence for Service3. In contrast to Service1 and Service2, pre-loading in Service3 does not happen on every swipe, but rather occurs in trains or batches, when the number of buffered but not yet watched videos is less than four. For instance, four videos (v2-v5) are pre-loaded in a train while the

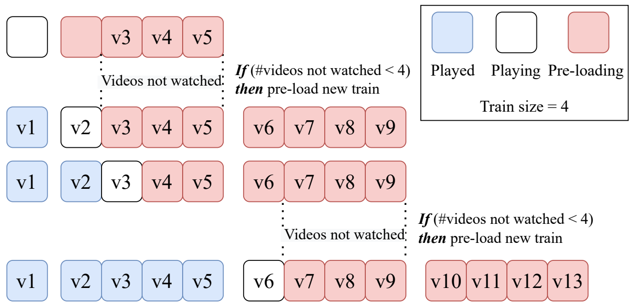


Figure 11: Service3 pre-loading sequence using trains.

first video (v1) is playing. When the second video (v2) is played, an additional batch of four videos are pre-loaded (v6-v9). Subsequent pre-loading does not occur again until the sixth video (v6). Though not shown, the pattern repeats after every 4 videos.

The impact of this policy decision is potentially substantial during the early viewing period in a user session. Whereas Service1 could be relatively busy with respect to pre-loading during the first video (pre-loading three or five videos), subsequent swipes only incur one additional pre-loading demand (maintaining a fixed number of pre-loaded videos). Service2 exhibits a similar behavior, albeit with only one pre-loaded video. In contrast, Service3 can potentially be more bursty in terms of demand and therefore wasteful in the event of an extremely short session, such as when watching only two videos (v1, v2) but pre-loading an additional seven (v3-v9). In contrast though to Service2 that pre-loaded tens of seconds of content, Service3 pre-loads are much shorter (2-10 s). This tempers the impact and burstiness though also dampens the robustness against stalls in the face of changing network conditions.

We further experiment with swiping time after the main video starts playing to determine that there is a consistent delay in the start of pre-loading. This delay is variable and influenced by network bandwidth and video bitrates, but it corresponds to approximately 10 s of main video content. This is determined as follows. We start playing the main video for increasing number of seconds starting from 1 s, then turning off the network. Swiping will reveal if there is a pre-load available. Once pre-loaded video plays, we can swipe back to the prior (main) video and measure the available play time. Repeated trials reveal a 10-second delay. Service3 has been observed to employ both TCP and QUIC interchangeably, as the transport protocol.

5.4 Service4

Service4 adopts the train-based pre-loading policy like Service3 (figure omitted), although videos are generally much shorter. In fact, similar policies are employed by two video services, each at the extremes of the duration spectrum (Figure 3c). Furthermore, Service4 can be extremely aggressive in train sizes, electing to pre-load up to 17 videos, resulting in a pre-load buffer of up to 22 videos. In tandem with this significant pre-load length, Service4 uses short pre-loaded durations of 1-3 seconds per video, which ultimately may not result in an excessive amount of data. We also find that the pre-loading starts about 3 s after the main video download initiates, using steps similar to those outlined for Service3.

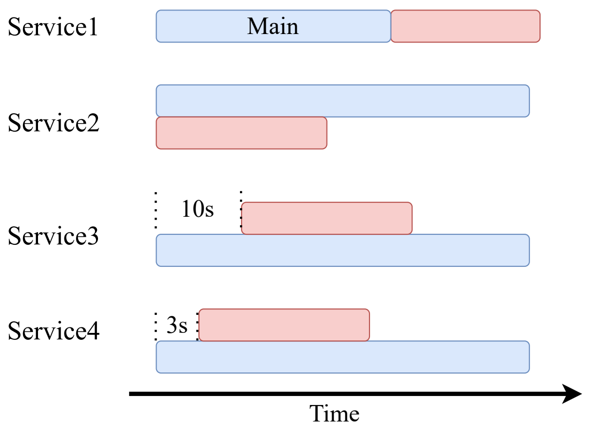


Figure 12: Pre-loading relative to main video download.

We conclude the pre-loading policy analysis by comparing the differences in relative positioning of the pre-load, in reference to the main video download, in Figure 12. Service1 downloads the main video completely before pre-load starts, while Service2 generally initiates the main video and pre-load downloads concurrently. For Service3 and Service4 a short delay is introduced for the pre-load (10 s and 3 s, respectively) after initiating the main video download. Similar to Service2 and Service3, Service4 employs both TCP and QUIC interchangeably, as the transport protocol.

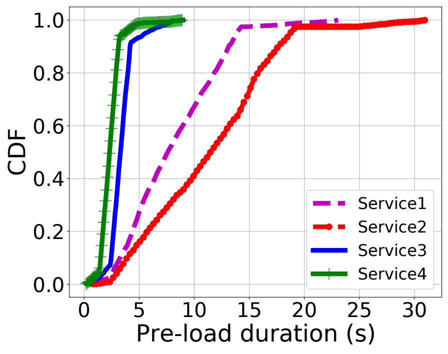
5.5 Statistical analysis of pre-loading characteristics

Next, we compare and contrast the expected distributions of the various pre-loading policy characteristics when watching a large number of videos in practice. We compare the pre-load duration, size, and duration percentage with respect to the currently playing (main) video. The evaluations were conducted using ample network bandwidth (20 Mbps) and several hundred videos from each service (i.e. {789, 885, 738, 643} for Service1 to Service4 respectively). The pre-load duration is the number of seconds of video downloaded for each pre-load. This value is inferred by turning the network connection off and measuring the time of available video at the client per instance in the sequence, ensuring that in the case of the first video in the sequence, no swiping to the consecutive videos happens for at least 60 seconds. The pre-load size represents the observed data volume per pre-loaded video. Finally, the pre-load percentage represents the ratio of pre-load duration over actual video duration. CDFs per respective parameter are plotted in Figs. 13a-13c.

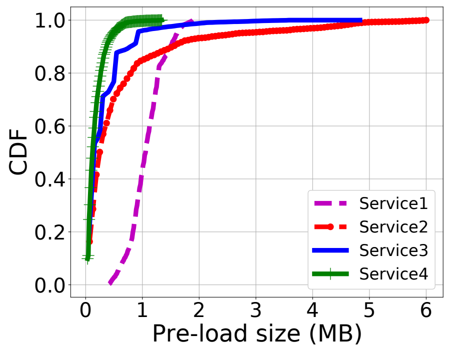
Fig. 13a shows the duration per pre-load for Service4 and Service3 remains in the more narrow range of 1-3 s and 2-4 s, respectively, while each service may pre-load a substantial number of videos (up to 7 and 22, respectively). Having a large number of pre-loaded videos may have a significant impact on the network demand and would require a large buffer at the application side. Thus, partially downloading only a short portion for each video, may alleviate both challenges. Contrary, Service2 that pre-loads only one video presents a wider distribution in pre-load duration (up to 20 s on iOS and 30 s on Android). Notably, with an average

Table 4: Attributes of studied video services.

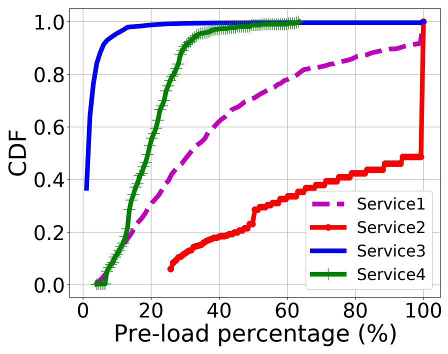
Attribute	Service1	Service2	Service3	Service4
Number of pre-loads	3 (iOS)/5 (Android)	1	7	22
Pre-load size per video	0.5 - 1.8 MB	0.1 - 6 MB	0.1 – 5 MB	0.1 – 1.6 MB
Pre-load duration per video	1 - 25 s	20 s (iOS)/30 s (Android)	2 - 4 s (iOS)/2 - 10 s (Android)	1 - 3 s
Average video duration	25 s	28 s	101 s	14 s
Average video size	4.9 MB	1.7 MB	19 MB	1.1 MB
Transport protocol	TCP	QUIC/TCP	QUIC/TCP	QUIC/TCP
Quality levels	1+	Up to 8	Up to 6	Up to 6



(a) CDF of pre-load duration.



(b) CDF of pre-load size.



(c) CDF of pre-load duration percentage.

Figure 13: Pre-loading duration and size characteristics.

video duration of 28 s, Service2 may pre-load videos in their entirety as shown in Fig. 13c, where roughly half of the observed instances for Service2 are pre-loaded at 90% of content duration or more.

As Service1 employs a size-based pre-loading policy, its pre-load size is near 1 MB in most cases (Fig. 9). In contrast, other services that may pre-load either only a single video (Service2) or short portions of multiple videos (Service3 and Service4) in a time-based manner, tend to utilize less data for the pre-loading operation of each video. Nonetheless, Service1 is clearly associated with a higher data cost per pre-load, that does not necessarily translate to higher pre-load percentage (median is 30% as per Fig. 13c).

Table 4 summarizes all pre-loading attributes such as the different policies observed for each service, along with the number, size and duration of pre-loads. In particular, we appropriately note instances where values are crucially different between iOS and Android devices. We have also gathered relevant service design information such as the transport protocol observed and number of quality levels offered.

6 DATA CONSUMPTION AND QOE EVALUATION

In this section, we analyze overall data consumption and QoE per service, under different network and application configurations. As short videos are primarily consumed on mobile devices and are often downloaded via the cellular network, the incurred data cost may be deducted from the subscriber's monthly data budget. Therefore, evaluating the bandwidth requirements generated by each service can provide two valuable insights. The first pertains to the data requirements associated with the pre-loading operation, while the second concerns the extent to which higher amount of downloaded data (i.e. higher encoding bitrates) reflect on VQ gains. To measure QoE, we evaluate individually 3 of the most relevant metrics [30], i.e VQ (via VMAF), wait time between videos, and stall time. Wait time is defined as the time difference between swiping on the app interface and the time of the first video frame is rendered on the screen, while stall time is defined as the duration of playout interruption, not including wait time. Wait and stall time may have a significant impact on QoE, especially in the context of short video streaming, given the frequent 'swiping' user-behavior, that is commonly associated with related services.

We note that the scope of our analysis in Section 4 and this section is to evaluate and compare the studied services from a content preparation and delivery point of view of the system design respectively, excluding user-specific design. Thus in the following, we allow the player to execute its pre-loading method unimpaired to assess QoE in different network conditions.

6.1 Experimental design

According to the methodology of Section 3.3, we perform automated (via ADB scripting) measurements for this study, while capturing both packet traces and ground-truth screen-recording. We explore 4 distinct experimental scenarios (scenarios 1-4). Initially we setup a competitive network scenario on WiFi, where four clients running the same service simultaneously stream a sequence of 15 videos, as specified in Section 3.2, while sharing a bandwidth of 6 Mbps (scenario 1) and then of 20 Mbps (scenario 2).

Following, we conduct measurements on a cellular network. We use only one client for testing in the cellular scenarios, where for each service we explore the 'Data Saver' mode. Data saver is a built-in function accessible in the settings option of each studied video application and is meant to reduce overall data consumption. As the specifics of the precise data-saving logic are not available for any of the services, we resort to comparing cumulative data consumption after streaming the sequence of 15 videos, first with the 'Data Saver' mode activated (ON) (scenario 3) and then deactivated (OFF) (scenario 4). During cellular experimentation, bandwidth was measured at 18 - 20 Mbps via SpeedTest [26], prior to each measurement. For reference, we provide the cumulative data volume associated with the highest quality for all 15 videos, as the theoretical maximum data required per service: 105 MB for Service1, 122 MB for Service2, 158 MB for Service3, and 173 MB for Service4.

Computing VMAF for playing videos is challenging, due to potential adaptation, stalls, and decoding and rendering impacts. For this evaluation, we resort to indirect measurement by converting bitrates into VMAF. We first generate a mapping between VMAF and bitrates for all 15 videos presented in Fig. 5. Video and audio bitrates are directly known from metadata or media file inspection. The average VMAF for a test is obtained by extracting the data rate for each video flow, excluding idle times. Known audio bitrate is subtracted, and the remainder approximately represents video encoding bitrates (modulo small overheads), which we convert to VMAF by interpolation using the aforementioned mapping (Fig. 5). For reference, the average audio bitrates for the 15 videos used are: 32 Kbps for Service1, 128 Kbps for Service2, 75 Kbps for Service3, and 83 Kbps for Service4. This is an approximate method, since VMAF score normally corresponds to an encoding variant, and in our experiments, adaptation may cause switching between variants.

As studied services prevent access to application-layer information, difficulty in obtaining ground-truth for wait and stall time is mitigated by visual inspection via Android's screen recording function, automatically operated via ADB for time precision. Nonetheless, according to Fig. 5, VMAF scores for adjacent (even 3 consecutive) quality levels are very close, rendering adaptation events hard to detect by visual inspection. While anecdotal evidence of adaptation within video was observed for a subset of the studied services, in this study we resort to data consumption to infer adaptation.

6.2 Data consumption vs. perceptual quality

Fig. 14 presents averages (over 10 iterations), of VMAF scores against the cumulative consumed data, generated by streaming the 15 videos per service and per investigated scenario.

Service1 achieves close range of VMAF in both competitive WiFi and real-world cellular scenarios, with moderate data consumption (50-80 MB), as expected from Fig. 4, previously analyzed in Section 4. A point worth noting is that according to our metadata analysis and Table 1, Service1 provides only one resolution (720p), while Fig. 14 suggests some data preservation mechanism, given the data consumption difference (~35%) between the WiFi scenarios or between the Data Saver 'ON' and 'OFF' scenarios. This would

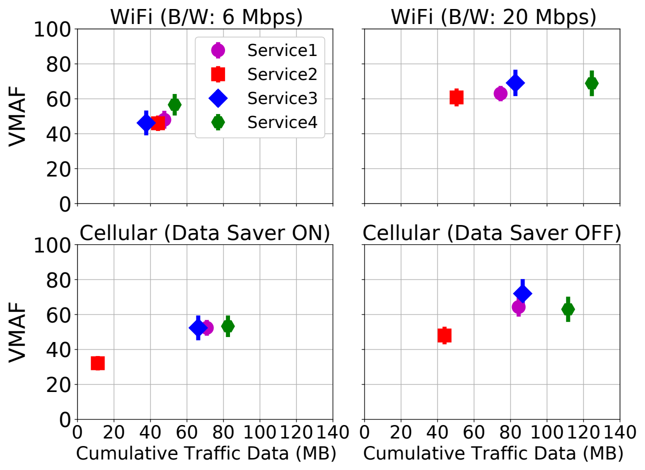


Figure 14: VMAF vs. cumulative data consumption.

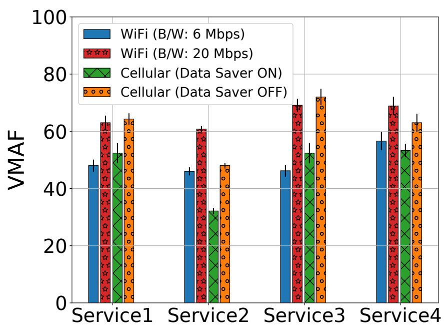
be attributed to either a more conservative pre-loading policy or employment of bitrate adaptation.

Service2, which according to Table 1 offers up to 8 resolutions, shows similar or at times lower VMAF scores, when compared with the other services in Fig. 4. Fig. 14 indicates that in fact Service2 scores lower in average VMAF score across all services in all scenarios, while requiring the least data consumption. This indicates that Service2 prioritizes data savings over VQ, which seems a reasonable design choice for cellular connectivity scenarios. Additionally, its Data Saver mode is more efficient than other services, reducing data requirements by more than 50%.

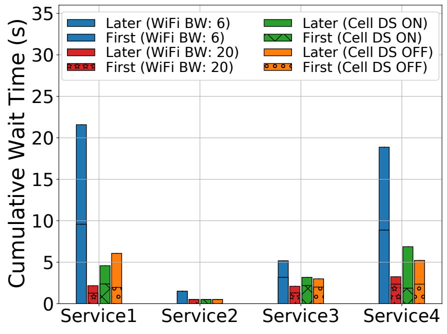
In Fig. 4, Service3 scores highest or equally high in VMAF when compared to other services in most scenarios, which is also supported by Fig. 14. In terms of data consumption, Service3 performs similarly to Service1 and at least 25% better than Service4. While Service4 shows the highest data consumption of all services, it is not present equivalently high VMAF scores in most scenarios, except in the constrained WiFi (6 Mbps).

Comparing cellular and WiFi networks between the least constrained scenarios (WiFi 20 Mbps and cellular without 'Data Saver'), we observe that data usage is similar across all services (within 10% and lower in cellular for 3 out of 4 services), while VMAF is nearly the same across three services. Only Service2 shows a reduction in VMAF by about 20% (60 vs. 48), in cellular.

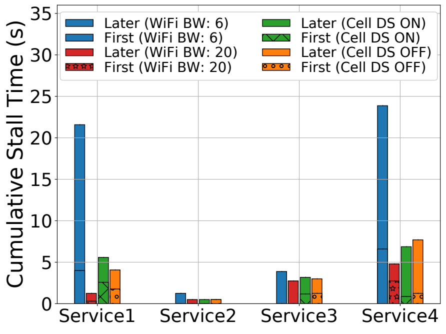
In summary, Service1 and Service3 manage to strike a better trade-off between VQ and data consumption, consistently ranking in the "middle", with Service4 using the most data but failing to provide significantly better VQ than other services. Service2 provides the lowest data consumption, but is associated with lower VMAF scores. The 'Data Saver' reduces data consumption by a minimum of 15% for most services, reaching 50% for Service2. Such data savings are associated with VQ reduction of about 15-25% for Service1, Service3 and Service4, but more (~33%) for Service2. Network type does not appear to significantly impact the behavior and performance of most services, unless constrained by bandwidth or by the 'Data Saver' setting.



(a) Average VMAF score.



(b) Cumulative wait time per client.



(c) Cumulative stall time per client.

Figure 15: QoE evaluation

6.3 QoE evaluation

Fig. 15 presents the average (over 10 iterations) QoE metrics, i.e. VMAF score for VQ, cumulative wait time and cumulative stall time, measured while streaming the same list of 15 videos in four experimental scenarios. Fig. 15a allows us to conclude that Service3 manages to perform equally well or outperform, in terms of VMAF scores, all services in all scenarios, except Service4 in the constrained WiFi scenario (6 Mbps). It is worth noting that any gains in VQ are more appropriately evaluated in contrast to the incurred cost in terms of data consumption, something that was addressed in Section 6.2.

Fig. 15b shows wait time results, where we separate 'First' wait time (for the first video), and 'Later' (for all subsequent videos cumulatively). Wait time for the first video is also commonly referred to as 'start-up time'. All services maintain a relatively low average start-up time of less than 2s ('First' video) in all studied scenarios, except the constrained WiFi (6 Mbps) scenario, where Service1 and Service4 show relatively higher values of 9 and 8 s, respectively. The average cumulative wait times for 'Later' are significantly reduced on a per video basis, when compared with the start-up time of the 'First' video. This insight suggests that the effectiveness of pre-loading on reducing wait times depends on the length of the user-session. User-sessions that consist of consuming a large number of videos consecutively, benefit more in the long-term from pre-loading, in terms of cumulative wait time. The higher start-up time incurred for the first video is ultimately countered by pre-loading more data of subsequent videos, thus eliminating wait time further in the session. Meanwhile, we also consider the impact of *Round Trip Time* (RTT) on each service. We observed 25-35 ms of RTT on WiFi and 71-107 ms on cellular. Under the same network type, each service's RTT is comparable, which suggests that higher start-up and wait time occurrences are more likely throughput-driven rather than attributed to RTT.

In regard to stall time, Fig. 15c shows that in all scenarios, except the constrained WiFi (6 Mbps), the average stall time is less than 3 s, for the first video of the playlist (1 of 15). The cumulative stall time for the remaining playlist (2-15 of 15) is kept below 5 s for all services, except for Service4 which is slightly higher at about 7 s.

The first video may account for about 25% to 50% of the total stall time in some of the cases, such as for Service1 and Service3 in the cellular scenarios. High stall times for the first video can be an indication that parallel pre-loading of multiple videos in a congested network may quickly saturate the network link, resulting in poor QoE during the initiation of a short video session. Thus, the choice of the pre-loading parameters, such as the amount of video data pre-loaded, is an important design feature for short-video services.

Service2 achieved the lowest wait and stall times, which indicate good QoE, coupled with lower bitrate requests (with related implications on VQ). This hypothesis is also supported from Service2's low bandwidth requirements (Fig. 14) and VMAF scores (Fig. 15a).

In summary, our results, suggest that pre-loading - employed in all studied short video systems - has the potential to improve start-up or wait times, especially for longer user-sessions. Given that short video users access a new video more frequently per user session, when compared to long videos users, short video user experience is enhanced by seamless content transitions without delay. Nonetheless, the amount of pre-loaded data is an important design feature, since more aggressive pre-loading may create significant strain on the network, ultimately hindering QoE in terms of playout interruptions (stalls).

7 CONCLUSION

We present a detailed investigation of the characteristics of four short video platforms. We outline insights into design decisions on encoding quality levels, resulting bitrates and perceptual quality, using large samples and custom-generated videos that were uploaded and then downloaded for analysis. We further reveal the principles of pre-loading mechanisms, such as the number of videos, duration, and size. Finally, we compare the data traffic consumption and QoE impact under multiple network conditions and present the performance results for each service. Our work complements initial research on short videos. Our methodologies can be used by researchers and service providers to gain further insights and improve system designs for the benefit of all participants in this new and exciting application ecosystem.

ACKNOWLEDGMENT

We thank all reviewers for their critical comments, and all the participants. This material is based upon work supported by the National Science Foundation under Grant No.CNS-1718405.

REFERENCES

[1] 2017. Finding the Just Noticeable Difference with Netflix VMAF. https://www.linkedin.com/pulse/finding-just-noticeable-difference-netflix-vmaf-jan-ozer/.

[2] 2020. Speedtest by Ookla. https://www.speedtest.net

[3] 2021. AT&T Video Optimizer. https://developer.att.com/video-optimizer.

[4] 2021. pafy. https://pypi.org/project/pafy/.

[5] 2021. TCPDUMP/LIBPCAP. https://www.tcpdump.org/index.html

[6] 2021. youtube-dl. https://github.com/ytdl-org/youtube-dl

[7] Sa'di Altamimi and Shervin Shirmohammadi. 2019. Client-Server Cooperative and Fair DASH Video Streaming. In *NOSSDAV '19* (Amherst, Massachusetts). 1–6.

[8] Abdelhak Bentaleb, Ali C. Begen, Saad Harous, and Roger Zimmermann. 2018. Want to Play DASH? A Game Theoretic Approach for Adaptive Streaming over HTTP. In *In Proc of MMSys'18* (Amsterdam, Netherlands). 13–26.

[9] Zhuang Chen, Qian He, Zhifei Mao, Hwei-Ming Chung, and Sabita Maharjan. 2019. A study on the characteristics of douyin short videos and implications for edge caching. In *TURC'19* (China). 1–6.

[10] Petr Cika and Dominik Starha. 2018. Video Quality Assessment on Mobile Devices. In *IEEE IWSSIP'18* (Maribor, Slovenia). 1–4.

[11] CreatorKit. 2020. *Video Length Guide: Facebook, Instagram, TikTok & More (2020).* https://creatorkit.com/blog/video-length-guide/

[12] Android Developers. 2021. *ANDROID STUDIO, Android Debug Bridge (adb).* https://developer.android.com/studio/command-line/adb

[13] Reza Farahani, Farzad Tashtarian, Alireza Erfanian, Christian Timmerer, Mohammad Ghanbari, and Hermann Hellwagner. 2021. ES-HAS: An Edge- and SDN-Assisted Framework for HTTP Adaptive Video Streaming. In *NOSSDAV '21* (Istanbul, Turkey). 50–57.

[14] Jing Guo and Guanghui Zhang. 2021. A Video-Quality Driven Strategy in Short Video Streaming. In *MSWiM '21* (Alicante, Spain). 221–228.

[15] Jianchao He, Miao Hu, Yipeng Zhou, and Di Wu. 2020. LiveClip: towards intelligent mobile short-form video streaming with deep reinforcement learning. In *NOSSDAV '20* (Istanbul, Turkey). 54–59.

[16] Te-Yuan Huang, Ramesh Johari, Nick McKeown, Matthew Trunnell, and Mark Watson. 2014. A Buffer-based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service. In *SIGCOMM'14* (Chicago, USA). 187–198.

[17] ISO/IEC. 2014. Dynamic adaptive streaming over HTTP (DASH). *International Standard 23009-1:2014* (May 2014).

[18] ITU-T. 2017. P.1203 : Parametric bitstream-based quality assessment of progressive download and adaptive audiovisual streaming services over reliable transport. *International Telecommunications Union* (Oct. 2017).

[19] Vengatanathan Krishnamoorthi, Niklas Carlsson, Derek Eager, Anirban Mahanti, and Nahid Shahmehri. 2015. Bandwidth-aware prefetching for proactive multi-video preloading and improved HAS performance. In *ACM MM'15* (Brisbane). 551–560.

[20] T Senthil Kumar. 2019. A novel method for HDR video encoding, compression and quality evaluation. *JIIP'19* 1, 02, 71–80.

[21] Zhi Li, Anne Aaron, Ioannis Katsavounidis, Anush Moorthy, and Megha Manohara. 2016. Toward a practical perceptual video quality metric. *The Netflix Tech Blog* 6, 2 (2016).

[22] Tarun Mangla, Ellen Zegura, Mostafa Ammar, Emir Halepovic, Kyung-Wook Hwang, Rittwik Jana, and Marco Platania. 2018. VideoNOC: Assessing Video QoE for Network Operators Using Passive Measurements. In *MMSys '18* (Amsterdam, Netherlands). 101–112.

[23] Hongzi Mao, Ravi Netravali, and Mohammad Alizadeh. 2017. Neural adaptive video streaming with pensieve. In *In Proc of SIGCOMM '17* (Los Angeles, USA). 197–210.

[24] SEGGER Microcontroller. 2007. *EmNet User Guide & Reference Manual.* https://www.segger.com/doc/UM07001_emNet.html

[25] Baiju Muthukadan. 2011. *Selenium with Python.* https://selenium-python.readthedocs.io/

[26] Ookla. 2020. *Speedtest by Ookla - The Global Broadband Speed Test.* https://www.speedtest.net/

[27] R. Pantos and W. May. 2011. HTTP live streaming. *IETF, Informational Internet-Draft 2582* (Sept. 2011). https://datatracker.ietf.org/doc/html/draft-pantos-http-live-streaming-10

[28] Mike Peterson. 2020. iPhones, iPads can now stream 4K YouTube videos in iOS 14. https://appleinsider.com/articles/20/06/23/iphones-ipads-can-now-stream-4k-youtube-videos-in-ios-14

[29] Werner Robitza, Steve Göring, Alexander Raake, David Lindegren, Gunnar Heikkilä, Jörgen Gustafsson, Peter List, Bernhard Feiten, Ulf Wüstenhagen, Marie-Neige Garcia, Kazuhisa Yamagishi, and Simon Broom. 2018. HTTP Adaptive Streaming QoE Estimation with ITU-T Rec. P. 1203: Open Databases and Software. In *Proc of MMSys '18* (Amsterdam, Netherlands). 466–471.

[30] Michael Seufert, Sebastian Egger, Martin Slanina, Thomas Zinner, Tobias Hoßfeld, and Phuoc Tran-Gia. 2015. A Survey on Quality of Experience of HTTP Adaptive Streaming. *IEEE ComST'15* 17, 1 (2015), 469–492.

[31] Kevin Spiteri, Ramesh Sitaraman, and Daniel Sparacio. 2018. From Theory to Practice: Improving Bitrate Adaptation in the DASH Reference Player. In *Proc of MMSys '18* (Amsterdam, Netherlands). 123–137.

[32] Babak Taraghi, Abdelhak Bentaleb, Christian Timmerer, Roger Zimmermann, and Hermann Hellwagner. 2021. Understanding Quality of Experience of Heuristic-Based HTTP Adaptive Bitrate Algorithms. In *NOSSDAV '21* (Istanbul, Turkey). 82–89.

[33] Google Team. 2014. *Google Chrome DevTools.* https://developer.chrome.com/docs/devtools/

[34] Kevin Tran. 2021. *TikTok Fights Off Copycat Competition Data Suggests.* https://variety.com/vip/tiktok-fights-off-copycat-competition-data-suggests-1235050064/

[35] Zhengzhong Tu, Chia-Ju Chen, Li-Heng Chen, Neil Birkbeck, Balu Adsumilli, and Alan C Bovik. 2020. A comparative evaluation of temporal pooling methods for blind video quality assessment. In *IEEE ICIP'20* (Abu Dhabi, UAE). 141–145.

[36] Shichang Xu, Eric Petajan, Subhabrata Sen, and Z. Morley Mao. 2020. What You See is What You Get: Measure ABR Video Streaming QoE via on-Device Screen Recording. In *NOSSDAV '20* (Istanbul, Turkey). 60–66.

[37] Guanghui Zhang, Ke Liu, Haibo Hu, and Jing Guo. 2021. Short Video Streaming With Data Wastage Awareness. In *IEEE ICME'21* (Shenzhen, China). 1–6.

[38] Haodan Zhang, Yixuan Ban, Xinggong Zhang, Zongming Guo, Zhimin Xu, Shengbin Meng, Junlin Li, and Yue Wang. 2020. APL: Adaptive Preloading of Short Video with Lyapunov Optimization. In *VCIP'20* (Macau, China). 13–16.

[39] Yuming Zhang, Yan Liu, Lingfeng Guo, and Jack YB Lee. 2022. Measurement of a Large-Scale Short-Video Service Over Mobile and Wireless Networks. In *IEEE TMC'22* (Orlando, USA). 1–1.

[40] Kyle Swanson Zhi Li. 2016. *Netflix, VMAF - Video Multi-Method Assessment Fusion.* https://github.com/Netflix/vmaf

[41] Zhi Li, Xiaoqing Zhu, Joshua Gahm, Rong Pan, Hao Hu, Ali.C Begen and David Oran. 2014. Probe and Adapt: Rate Adaptation for HTTP Video Streaming At Scale. In *IEEE JSAC'14* (USA). 719 – 733.