Energy Considerations for ABR Video Streaming to Smartphones: Measurements, Models and Insights

Chaoqun Yue[1], Subhabrata Sen[2], Bing Wang[1], Yanyuan Qin[1], and Feng Qian[3*]
[1]University of Connecticut [2]AT&T Labs - Research [3]University of Minnesota

ABSTRACT

Adaptive Bitrate (ABR) streaming is widely used in commercial video services. In this paper, we profile energy consumption of ABR streaming on mobile devices. This profiling is important, since the insights can help developing more energy-efficient ABR streaming pipelines and techniques. We first develop component power models that provide online estimation of the power draw for each component involved in ABR streaming. Using these models, we then quantify the power breakdown in ABR streaming for both regular videos and the emerging 360° panoramic videos. Our measurements validate the accuracy of the power models and provide a number of insights. We discuss use cases of the developed power models, and explore two energy reduction strategies for ABR streaming. Evaluation demonstrates that these simple strategies can provide up to 30% energy savings, with little degradation in viewing quality.

CCS CONCEPTS

• **Information systems** → Multimedia streaming;

KEYWORDS

Adaptive Video Streaming; Energy; Measurements; Models.

ACM Reference Format:
Chaoqun Yue, Subhabrata Sen, Bing Wang, Yanyuan Qin, and Feng Qian. 2020. Energy Considerations for ABR Video Streaming to Smartphones: Measurements, Models and Insights. In *11th ACM Multimedia Systems Conference (MMSys'20), June 8–11, 2020, Istanbul, Turkey.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3339825.3391867

1 INTRODUCTION

Video streaming is one of the most energy hungry applications on mobile devices. The current de facto video streaming technology in the industry is Adaptive Bitrate (ABR) streaming, where the server stores multiple *tracks* (also referred to as representations or renditions) of a video, each specifying the same content but with a different bitrate/quality. A track is further divided into a series of *segments*, each containing data for a few seconds' worth of playback. The ABR adaptation logic at the client dynamically determines which quality (i.e., from which track) to fetch for each segment position in the video.

While the literature on energy-efficient video streaming is extensive (see §9), most existing studies focus on single-track streaming. Little is known about energy consumption of ABR streaming, which involves multiple tracks and dynamic track changes over time. Specifically, *what is the component-wise power breakdown in ABR streaming? What are the dominant sources of power consumption? What strategies can be used to reduce energy consumption in ABR streaming?* Such insights into the energy profile of ABR streaming is important as it can help the development of more energy-efficient ABR streaming pipelines and techniques.

To answer the above questions, we first develop component-wise power models that allow us to estimate the power draw of each individual component on the fly using easily accessible information on commodity phones, without the need of an external power meter. Using these models, we quantify the power breakdown in ABR streaming. We consider both regular videos and the emerging 360° panoramic videos (§2), and identify the key differences and similarities between these two types of videos. In addition, to demonstrate the generality of the results, we use two state-of-the-art devices (LG V20 and Moto G5) and two popular ABR players (ExoPlayer and YouTube player). Our main contributions are as follows:

• *Power models (§3 and §5).* To model ABR streaming power draw, a critical task is modeling *per-segment* decoding power draw, since ABR streaming involves playback of segments dynamically chosen from different tracks. Modern phones use a set of specialized hardware units for video decoding. We propose using the *residual power*, i.e., the total power subtracted by the power of the other components (CPU, screen, network), to represent the aggregate power draw of this set of hardware units, and develop a methodology for measuring per-segment residual power. Based on the insights from the measurements, we further propose two scalable methods for obtaining per-segment residual power. In addition, we develop a CPU power model that can be used for multiple heterogeneous processor cores (commonly found on latest smartphones), and network models that specifically consider the characteristics of video streaming. Extensive validation demonstrates that the ABR streaming power draw estimated from our models (i.e., summing up the power draws from the individual component power models) is highly accurate: the error compared to the measurement from a power meter is below 8% across all the settings we explore.

• *Measurement findings (§4 and §6).* We conduct extensive measurements using a diverse set of videos to characterize the residual power. Our findings include: (i) the residual power increases with the frame rate and resolution (the increase is particularly significant for very high resolutions), while is insensitive to the content, encoding quality and codec under the same frame rate and resolution; and (ii) the residual power for 360° videos is significantly higher than that of regular videos due to projection and sensor measurements

* The work of Feng Qian was done when he was at Indiana University.

MMSys'20, June 8–11, 2020, Istanbul, Turkey

ACM ISBN 978-1-4503-6845-2/20/06.
https://doi.org/10.1145/3339825.3391867

that are required for rendering 360° videos. We further characterize the component-wise power breakdown of ABR streaming by conducting in-the-wild experiments in both WiFi and LTE networks under a wide range of network conditions. Our findings include: (i) the residual power only accounts for a small percentage (up to 18%) of the total power draw for regular videos since the specialized hardware units are highly efficient; for 360° videos, the percentage becomes much larger (up to 41%); (ii) due to the complex behavior of ABR streaming, the network power draw can be similar under low to medium network bandwidth settings, despite their significantly different profiles, while is much lower when the network bandwidth is high; (iii) the CPU power draw is stable, insensitive to various video and network characteristics, and is significantly larger for 360° videos than that for regular videos due to projection needed for rendering 360° videos; and (iv) we observe clear device heterogeneity: one phone has significantly lower network power draw, while significantly higher residual power for 360° videos than the other phone.

- *Insights in saving energy and other use cases (§7 and §8).* Based on the insights of our measurement study, we explore two simple energy reduction strategies that can be easily incorporated in existing ABR systems. We implement both strategies in ExoPlayer and quantify their energy savings over commercial LTE networks. Our evaluation demonstrates up to 30% energy savings with little degradation in viewing quality in various LTE network conditions. We further discuss other use cases of our developed power models, ranging from designing new energy-aware ABR schemes to guiding track design for mobile platforms.

2 METHODOLOGY

Our high-level methodology is to derive component-wise power models for online power estimation on commodity smartphones, using information that can be easily obtained via light-weight logging mechanism on the phones. We consider multiple phones and video players, and both regular and 360° videos to verify the generality of the models.

Power measurement. We use a Monsoon power monitor [15] to measure the real-time power drain of a phone with the sampling frequency of 1 kHz. The power measurements are used to derive the power models for individual components involved in ABR streaming by selectively turning on and off certain components.

Phones. We use 2 smartphones from 2 different vendors, LG V20 (4 GB RAM) and Moto G5 (3 GB RAM), both with Qualcomm Snapdragon processor (820 and 430, respectively) and running Android 7.0. The reason for choosing these two phones is that they are top recommended phones with removable batteries [20], allowing easy power measurements using the power monitor. The LG V20 represents a high-end phone with faster CPU and larger memory, and the Moto G5 represents a middle-range phone with slower CPU and less memory. Both phones have LTE (Snapdragon X12 LTE modem for the LG phone and X6 LTE modem the Moto phone) and WiFi (802.11a/b/g, 802.11n) network interfaces.

Baseline power draw. Since our goal is to measure and model the power draw involved in ABR streaming, it is important to first understand the power consumption in the absence of any video streaming workload. To that end, we disable all the other apps and all the components not needed for video streaming (e.g., GPS, camera, Bluetooth) before running an experiment. In this setting, the total power draw is 620 mW for the LG phone (the screen brightness is set to 50% of the full brightness, and the screen power draw is 440 mW, see §3). For the Moto phone, the corresponding total power is 862 mW (573 mW for the screen). The total power draw for each device above represents the power consumption without ABR streaming; running an ABR streaming app leads to additional power draws beyond this baseline value (see §6).

Logging. We have crafted C programs and shell scripts to log the required information needed for our power models, including the amount of time spent in each frequency for each CPU core (`/sys/devices/system/cpu/cpu[id]/cpufreq/stats/time_in_state`), CPU load (`/proc/stat`), and signal strength of the WiFi and LTE network interface cards (`dumpsys telephony.registry`). The above information is logged every second, sufficient for our goal of obtaining an accurate component-wise power breakdown, while adding very little overhead (see below). Our LTE and WiFi power models require knowing network throughput in short time intervals (100 ms). For simplicity, we log the network traffic using tcpdump directly on the phone. While tcpdump can only run on rooted phones, we emphasize that this is not a limitation since many methods can be used to capture fine-granularity network traffic on phones (e.g., a local man-in-the-middle proxy, or a VPN service as used in the ARO tool [8]). We verified that the logging overhead is low – it leads to no more than 1% of the total power draw when downloading data under various network speed in WiFi/LTE networks.

Video Players. Most of the measurement results reported in the paper use ExoPlayer (version r2.9.2), a popular open-source media player for Android that is used in more than 10,000 apps [9], and provides convenient ways to log various playback information (e.g., the tracks selected, player buffer level). To further validate our models, we use another popular player, the YouTube app (version 14.06.56). The track selected for each segment position by this app is obtained using nerds-stats [53] through an automation tool that we developed.

Videos. Table 1 lists nine regular videos. The raw footage of the first three videos are publicly available from [10]. We uploaded these raw videos to YouTube and then download the encoded tracks using `youtube-dl` [14]; the other six videos were downloaded from YouTube directly. Table 2 lists ten 360° videos, all downloaded from YouTube. They are panoramic videos, i.e., each track contains the entire 360-degree scenes. We do not consider more efficient delivery techniques (e.g., tile-based encoding [40, 56, 59, 62]), since such approaches are much more complex and have not been deployed in commercial services. All the regular videos have six tracks, with resolutions from 144p to 1080p; the 360° videos have an additional track of resolution 2160p. All the above videos are around 10 minutes long, encoded in H.264 codec [12]. The frame rate of each track is 30 frames per second. All the tracks are Variable Bitrate (VBR) encoded; the peak rate of a track is 1.0×-3.0× of the average bitrate of the track. All the videos use Fragmented MP4 packaging format [3]. We only consider this format since it is more efficient than MPEG-2 TS [13] and the trend is moving towards this format. In addition to the above encodings, we further use other encodings to investigate the impact of frame rate, encoding quality and codec on residual power (§4).

Table 1: Average bitrate (in Mbps) and peak bitrate to average bitrate ratio (in parenthesis) of nine regular videos.

	Fiction	Sports	Animation	Family	Comedy	Documentary	Action	Nature	Animal
144p	0.08 (1.3)	0.07 (1.4)	0.09 (1.4)	0.08 (1.5)	0.06 (2.0)	0.05 (2.4)	0.06 (2.0)	0.06 (2.0)	0.06 (2.0)
240p	0.19 (1.2)	0.16 (1.4)	0.20 (1.5)	0.15 (1.8)	0.11 (2.4)	0.09 (2.8)	0.11 (2.4)	0.13 (2.0)	0.18 (1.6)
360p	0.34 (1.4)	0.25 (1.9)	0.35 (1.8)	0.40 (1.7)	0.26 (2.8)	0.23 (2.9)	0.27 (2.4)	0.40 (1.9)	0.36 (2.1)
480p	0.76 (1.3)	0.63 (1.7)	0.77 (1.8)	0.78 (1.6)	0.51 (2.7)	0.45 (2.8)	0.55 (2.3)	0.85 (1.6)	0.73 (1.9)
720p	1.42 (1.4)	1.24 (1.7)	1.47 (1.8)	1.47 (1.7)	0.92 (3.0)	0.86 (2.9)	1.02 (2.4)	1.74 (1.6)	1.45 (1.9)
1080p	2.29 (1.4)	2.17 (1.7)	2.50 (1.8)	2.74 (1.7)	2.07 (2.2)	1.58 (2.9)	1.88 (2.4)	3.24 (1.6)	2.97 (1.5)

Table 2: Average bitrate (in Mbps) and peak bitrate to average bitrate ratio (in parenthesis) of ten 360° videos.

	Skydiving	Sci-Fi	Animals	Sea	R. coaster	Island	Racing	House	Concert	Tennis
144p	0.08 (1.1)	0.11 (1.0)	0.10 (1.0)	0.09 (1.2)	0.08 (1.1)	0.10 (1.0)	0.08 (1.0)	0.09 (1.1)	0.09 (1.0)	0.10 (1.0)
240p	0.18 (1.1)	0.24 (1.0)	0.21 (1.1)	0.19 (1.1)	0.22 (1.0)	0.2 (1.1)	0.2 (1.3)	0.23 (1.1)	0.22 (1.1)	0.19 (1.3)
360p	0.48 (1.4)	0.45 (1.4)	0.44 (1.6)	0.51 (1.2)	0.49 (1.3)	0.41 (1.6)	0.42 (1.4)	0.49 (1.2)	0.50 (1.4)	0.51 (1.3)
480p	0.90 (1.3)	0.91 (1.3)	0.97 (1.3)	0.84 (1.4)	1.00 (1.2)	0.96 (1.1)	0.8 (1.4)	0.99 (1.3)	0.93 (1.4)	0.96 (1.3)
720p	1.75 (1.3)	1.76 (1.3)	1.61 (1.5)	1.91 (1.1)	1.93 (1.2)	1.74 (1.2)	1.84 (1.3)	1.95 (1.1)	1.84 (1.4)	1.81 (1.4)
1080p	3.15 (1.3)	3.34 (1.3)	3.06 (1.4)	2.88 (1.4)	3.49 (1.3)	3.16 (1.5)	3.58 (1.4)	3.60 (1.1)	3.24 (1.3)	3.45 (1.1)
2160p	13.73 (1.6)	14.47 (1.6)	12.35 (2.0)	13.13 (1.8)	15.86 (1.4)	15.37 (1.4)	13.69 (1.8)	15.85 (1.6)	14.13 (1.5)	13.34 (1.6)

Network conditions. We conduct in-the-wild experiments in both WiFi and LTE networks. In addition, we use controlled network conditions (by controlling the network bandwidth at the server using `tc` [1] and replaying various network traces collected from WiFi/LTE networks) to investigate the power draw under various network conditions, or for apple-to-apple comparison.

3 COMPONENT POWER MODELS

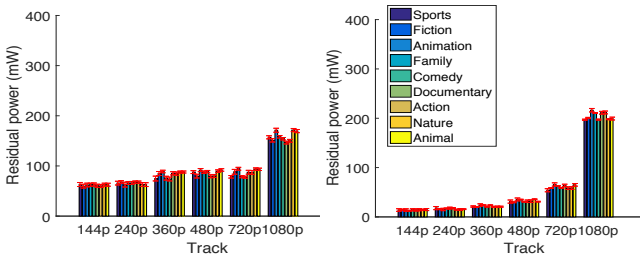
CPU power model. Both the LG and Moto phones have multiple heterogeneous processor cores. One set of cores has lower clock frequency ranges, while the other set has higher clock frequency ranges; we refer to them as *small* and *large* cores, respectively. The LG phone has two small and two large cores, supporting frequencies 307-1594 and 307-2150 MHz, respectively [17]. The Moto phone has four small and four large cores, supporting frequencies 768-1094 and 960-1401 MHz, respectively [16]. The CPU frequencies are managed by CPUFreq Governors [21] in the Android OS. The default governor is Interactive Governor (it sets the CPU speed based on usage, and can aggressively scale up the CPU speed in response to CPU-intensive activities), which is used throughout this study. We observe that during video playback, all the cores are being used. For the LG phone, the two small cores use 653 and 730 MHz primarily, and the two large cores use 653 and 1037 MHz primarily; other frequencies are used in less than 1% of time. For the Moto phone, only the lowest frequency is being used (i.e., 768 MHz on the four small cores and 960 MHz on the four large cores). We confirm experimentally that the multiple cores of the same category use the same frequency at the same time. On the other hand, at a given time, the small and large cores may use different frequencies, leading to heterogeneous frequency usage. While the study in [73] has proposed a power model for smartphones with multiple processor cores (which was adopted in [27]), the model only considers homogeneous cores that use the same frequency at the same time. We propose a new power model and show that it leads to significantly lower errors than the model in [73] (see Appendix). The proposed CPU power model is

$$P_{CPU} = \sum_{k=1}^{N_c} \sum_{i=1}^{M_k} \alpha_{k,i} \left(P_{B,N_c,k}(f_{k,i}) + u_k P_{\Delta,k}(f_{k,i}) \right), \quad (1)$$

where N_c is the number of enabled cores, M_k is the number of frequencies that core k uses, $\alpha_{k,i}$ is the fraction of time that core k spends at frequency $f_{k,i}$, $P_{B,N_c,k}(f_{k,i})$ is the baseline power of core k at frequency $f_{k,i}$ when N_c cores are enabled, u_k is the utilization of core k, and $P_{\Delta,k}(f_{k,i})$ is the power increment of core k at frequency $f_{k,i}$. The coefficients of the model can be obtained using linear regression (see Appendix).

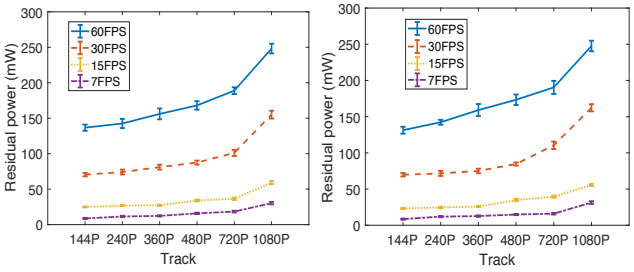
Screen power model. Existing studies have proposed power models for different display technologies [35, 54, 72]. Both the LG and Moto phones use IPS (in-plane switching) liquid-crystal display (LCD) panels. For such panels, the screen power mainly depends on the brightness level of the screen. For a given brightness level, we obtain the power draw of the screen as the difference of the power draws (measured by the power monitor) in two settings: (i) when both the screen and CPU are on, and (ii) when the screen is off and the CPU is on. The Appendix shows the screen power draw for three brightness levels for the two phones. The experiments in the rest of the paper use a medium screen brightness of 50%.

LTE and WiFi power models. The power draw of the LTE interface card of a phone depends on the current Radio Resource Control (RRC) state of the phone [27, 42]. In LTE networks, the two RRC states are Idle and Connected. Within the Connected state, there are three sub-states: Active, short DRX and long DRX, with Active as the only sub-state in which the device sends or receives data. The power draw of a WiFi interface card also depends on its state (either Active or Idle). While a number of studies have characterized the energy consumption of cellular networks [19, 27, 33, 42, 54, 57, 60] and WiFi networks [19, 27, 33, 46, 54, 58], they are not for video streaming, which is long-lived and predominantly in the downlink direction. We propose and validate LTE and WiFi power models that account for the characteristics of video streaming (see Appendix). For LTE, both Active power (during data transfer) and Tail power (no data transfer and not yet in the Idle state) are modeled. For



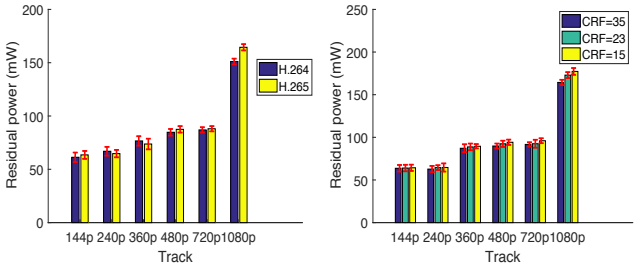
(a) LG phone. (b) Moto phone.

Figure 1: Residual power for 9 regular videos.



(a) Fiction. (b) Football.

Figure 2: Impact of frame rate (LG phone).



(a) Varying codec. (b) Varying encoding quality.

Figure 3: Impact of codec and encoding quality (LG phone).

WiFi, only Active power is modeled since Tail time (for transiting from Active to Idle state) is only hundreds of milliseconds.

Residual power. An important part of power draw in video streaming is due to video decoding. For both the LG and Moto phones, based on the specs [4, 6], a suite of hardware units, e.g., GPU, Video Processing Unit (VPU), Digital Signal Processor (DSP), on the Qualcomm Snapdragon chipset supports video processing. We did not find easily accessible ways to obtain the power draw of each of these hardware units. On the other hand, since our goal is to identify application-level actionable items that can improve the energy efficiency in video streaming, it is sufficient to model the aggregate energy consumption of these hardware units, which we refer to as *residual power*, i.e., the total power subtracted by the power of the other components (CPU, screen, NIC). In ABR streaming, since the playback involves segments dynamically chosen from different tracks, we need to obtain per-segment residual power. We present detailed measurement results on residual power in §4. Based on the insights, we present models for per-segment residual power in §5.

4 RESIDUAL POWER MEASUREMENTS

In this section, we measure residual power for both regular and 360° videos. All the results are obtained by playing back videos locally on a phone. The residual power is obtained as the total power draw (measured by the power meter) subtracted by the screen (screen brightness set to 50%) and CPU power draws.

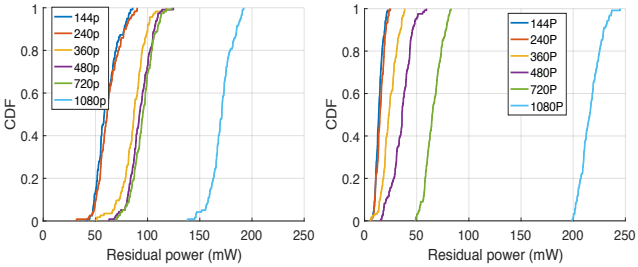
4.1 Regular Videos

While our ultimate goal is to obtain per-segment residual power, we first present per-track residual power to gain insights on the factors that impact the residual power most significantly. After that, we present per-segment residual power measurement results.

4.1.1 Per-track Residual Power. **Impact of resolution and content type.** Fig. 1 shows the residual power for 9 regular videos (Table 1) on the two phones; both the mean and standard deviation over five runs are plotted in the figure. All the videos have six tracks (144p to 1080p) with frame rate of 30 frames per second. We see that the residual power increases with the resolution, since higher resolutions lead to more pixels and naturally higher decoding overhead. The residual power is fairly low for the lower tracks (144p-720p), and increases significantly for the highest track (1080p). This might be due to the characteristics of the specialized hardware units – they are highly efficient for the lower resolution tracks, but less efficient for 1080p. For each of the lower tracks (144p-720p), the residual power of the Moto phone is lower than that of the LG phone; for the 1080p track, the observation is the opposite. Furthermore, for both phones, the residual power is not sensitive to the video content and bitrate variations within the same resolution – for each resolution, the residual power is similar across all the videos, even though the content and bitrate of the video differ significantly (see Table 1), a point that we will return to in §5.

Impact of frame rate. Intuitively, residual power increases with frame rate since more frames need to be decoded per second. To investigate the impact of frame rate, we use `FFmpeg` to encode six videos (each of 30 frames per second) downloaded from YouTube into three additional frame rates: 7, 15, and 60 frames per second. The six videos include three in Table 1 (Fiction, Sports, Animation) and three others in the categories of football, TV show, and gaming. Fig. 2 shows the results for two videos on the LG phone; the results for other types of videos and the Moto phone show similar trends. For a given resolution, when doubling the frame rate, the power draw is approximately doubled when the frame rate is increased from 30 to 60 frames per second, and is more than doubled in the other two cases (i.e., increased from 7 to 15, and from 15 to 30 frames per second). This approximately linear increase of the residual power with the frame rate confirms that the residual power for regular videos indeed comes primarily from decoding.

Impact of codec and encoding quality. We next investigate two other factors, codec and encoding quality, on per-track residual power. H.265 is an emerging codec that is more efficient than H.264. We use `FFmpeg` to encode two raw videos using both H.264 and H.265. For each resolution, we use the constant rate factor (CRF) of 23 for both codecs; for the same video and resolution, the bitrate in H.265 encoding is about 30% lower than that of H.264 encoding. Fig. 3 shows per-track residual power for one video under these two codecs on the LG phone. For the same resolution, we observe similar



(a) LG phone. (b) Moto phone.

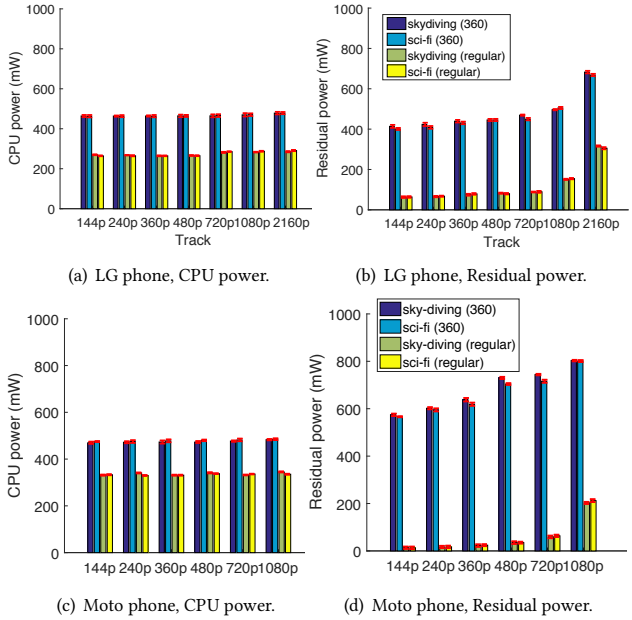
Figure 4: Distribution of per-segment residual power of a regular video (Animation).

residual power between the two codecs, which might be because the Qualcomm Snapdragon chipset used by the two phones have built-in hardware support for both H.264 and H.265 encodings [2].

To investigate the impact of encoding quality, we encode two raw videos (§2) using `FFmpeg` [11], following the per-title "three-pass" encoding procedure from Netflix [31]. We use three CRF values, 15, 23 and 35; lower CRF leads to higher bitrate and quality. For each video, we choose six tracks (144p to 1080p) as used by YouTube. Each track is segmented into 5-sec segments, consistent with the segment duration used by YouTube. For the same resolution, the average bitrate differs significantly under the three CRF values (the average bitrate when CRF is 15 is approximately twice and four times of that when CRF is 23 and 35, respectively), while the residual power, perhaps surprisingly, is similar (see one example in Fig. 3(b)). This might be because the specialized hardware decoding units are highly efficient and are insensitive to encoding quality under the same resolution.

4.1.2 Per-segment Residual Power. Our per-track measurement results above show that for the same resolution and frame rate, the residual power is insensitive to the content and encoding quality. The above observations indicate that the segments in a track (each a few seconds long, of the same resolution and frame rate) may incur similar residual power draw, which is confirmed by our measurements below.

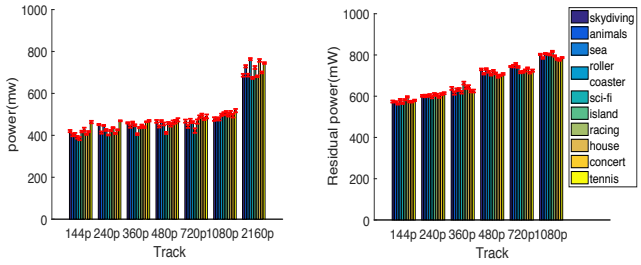
One way to obtain the residual power for a segment is to play the segment multiple times, obtain the power draw each time and then use the average as the residual power for the segment. Our measurements show that this approach leads to large variance among the multiple runs, due to the small segment length, which is sensitive to measurement noises. We design a novel methodology as follows to overcome the above problem. For a particular segment, we repeat the segment for 10 times to construct a "video" with 10 identical segments. The average residual power draw of this "video" is then used as the residual power draw of the segment. Since this "video" is much longer than a segment, the variance across multiple runs is significantly lower than that when playing a single segment multiple times. Fig. 4 plots the CDF of the residual power of a segment (obtained from five runs) for all the segments in each track of one video. We see that for both phones, segments in the same track indeed lead to similar residual power draw.



(a) LG phone, CPU power. (b) LG phone, Residual power.

(c) Moto phone, CPU power. (d) Moto phone, Residual power.

Figure 5: Additional CPU and residual power draws when playing back 360° videos.



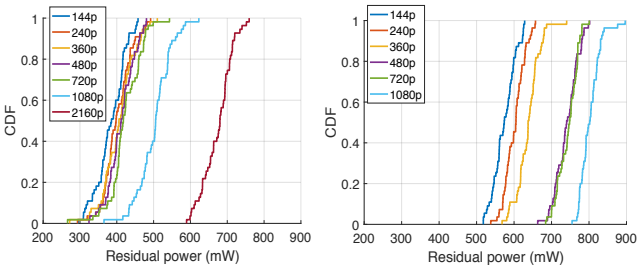
(a) LG phone. (b) Moto phone.

Figure 6: Residual power for ten 360° videos.

4.2 360° Videos

A user may move around while playing a 360° video. We compare the CPU and residual powers when holding a phone stationary and moving it around (horizontally or vertically), and find negligible differences in the power draws. This might be because the same set of operations is performed while playing back a panoramic 360° video (all 360° videos used in this paper are panoramic), whether a phone is stationary or moving. In the following, all the results are obtained in the stationary setting. No results for 2160p are shown for the Moto phone (it cannot play such videos).

Additional power draws in playing back 360° videos. A 360° video can be played back as a regular video (in that case, distorted frames will be shown due to the lack of processings specific to 360° videos). Fig. 5 compares the CPU and residual powers when playing back two 360° videos versus playing them back as regular videos. We see that, for both phones, the former leads to significantly higher CPU and residual power draws, which is due to two additional operations when playing back a 360° video: (i) sensor measurements that are used to determine the viewpoint, and (ii)



(a) LG phone. (b) Moto phone.

Figure 7: Distribution of per-segment residual power of a 360° video (Skydiving).

projection that is required to render a 360° video in 3D in undistorted forms. To measure the power draw due to the sensors, we develop an app that does the sensor measurements in the same way as that when ExoPlayer plays a 360° video (ExoPlayer uses `TYPE_GAME_ROTATION_VECTOR` sensor as input; the frequency of the sensor is set to `SENSOR_DELAY_FASTEST`, i.e., getting sensor data as fast as possible [7]). We run the app and obtain the power draws due to CPU (obtained from our model) and sensors (obtained as the total power draw from the power monitor subtracted by the CPU power; the other components are turned off). We observe that the CPU power before and after starting the sensor measurements is similar, indicating that sensor measurements does not lead to additional CPU operations. Therefore, the power draw due to sensors falls into the residual power. Specifically, based on our measurements, the power draw due to sensors is 247±1.9 mW on the LG phone and 352 ±2.3 mW on the Moto phone, which is a significant portion of the residual power draw (see Figures 5(b) and (d)). After taking account of power draw due to sensors, the rest of the additional CPU and residual powers (compared to playing a 360° video as a regular video) is likely due to projection. Quantifying the precise power draw due to projection is left as future work.

Per-track residual power. Fig. 6 shows the residual power for ten 360° videos (Table 2) on both phones. For the same video, we see that the residual power increases with the resolution; the increase is particularly significant for 2160p on the LG phone (the Moto phone cannot play 2160p). For the same resolution, we again see similar residual powers across the videos (the variance is slightly larger than that for regular videos), indicating that the residual power is not sensitive to video content. For each resolution from 144p to 1080p, the specific residual power is different for the two device models, with the values being lower for the V20 device.

Per-segment residual power. We use the same methodology as that in §4.1 to obtain per-segment residual power for two 360° videos. Fig. 7 plot the results for one video for both phones; the results for the other video is similar. We again observe that for both phones, the segments in the same track have similar residual power.

5 ABR STREAMING: POWER MODELS

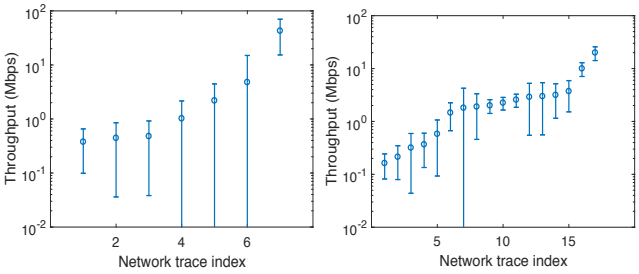
In this section, we first present per-segment residual power model based on the measurement results in §4, and then the overall power model for ABR streaming.

Per-segment residual power model. One straightforward method for obtaining per-segment residual power is following the methodology in §4.1.2 and store the information in a lookup table. During subsequent ABR streaming, when a segment is selected, we can simply look up the table to obtain the residual power for that segment. The above approach requires detailed profiling of per-segment power draw of a video beforehand. We next present two more scalable approaches that are based on the two insights from our measurement results for both regular and 360° videos: (i) the segments in the same track have similar residual power, and (ii) for the same resolution and frame rate, the residual power of a track is similar across videos of a wide range of categories. Specifically, consider a video, let $P_{R,\ell,i}$ denote the residual power of segment i in track ℓ, and let $P_{R,\ell}$ denote the average residual power of all the segments in track ℓ, i.e., $P_{R,\ell} = \left(\sum_{i=1}^{n} P_{R,\ell,i}\right)/n$, where n is the number of segments in a track. The first insight indicates that the difference of $P_{R,\ell,i}$ and $P_{R,\ell}$ is small – it is within ±20 and ±30 mW for regular and 360° videos for the LG phone; the corresponding values for the Moto phone are ±20 and ±50 mW (see Figures 4 and 7), which are only a small fraction of the total power draw during ABR streaming (see §6). Therefore, the first scalable approach is to approximate segment residual power $P_{R,\ell,i}$ as track residual power $P_{R,\ell}$, which can be obtained by playing a track locally and is more scalable than obtaining $P_{R,\ell,i}$ individually.

We now describe the second scalable approach. Let S be a set of representative videos that are encoded using similar encoding pipelines (for scalability, a commercial service, e.g., YouTube or Netflix, uses the same encoding ladder and pipeline for a large number of videos [5, 25, 29, 30, 50]). Therefore, these videos have the same number of tracks and the resolutions across the tracks are consistent, even though they may have significantly different bitrate for the same track as shown in Tables 1 and 2. Let $\overline{P}_{R,\ell}$ represent the average power draw of track ℓ for all the videos in S. The second insight from our measurements indicates that the difference between $P_{R,\ell}$ for one video and $\overline{P}_{R,\ell}$ is small – it is within ±9% and ±8% for the regular and 360° videos for the LG phone; the corresponding values for the Moto phone are ±8% and ±7% (see Figures 1 and 6). Therefore, combining the first and second insights, the second scalable approach is to approximate $P_{R,\ell,i}$ as $\overline{P}_{R,\ell}$, $\forall \ell, \forall i$.

The above three approaches (i.e., the straightforward and two more scalable approaches) for estimating per-segment residual power are increasingly more scalable at the cost of slightly higher errors. As we shall see in §6, even under the third approach, the error during ABR streaming is low (below 8%).

Putting it all together. To estimate the power draw of ABR streaming on a given phone, we first need to obtain the measurement-based models for CPU, screen, network interface cards, as presented in §3. Per-segment residual power draw can be obtained using one of the three approaches described above. After that, during ABR streaming, the power draws from these components can be obtained online using the respective models with real-time collected system and player information (i.e., CPU frequency, load, screen brightness, downlink throughput and the selected track for a segment position) as input, and the sum of the component power draws is the power draw of ABR streaming.



(a) WiFi traces. (b) LTE traces.

Figure 8: WiFi and LTE traces.

6 ABR STREAMING: ENERGY BREAKDOWN

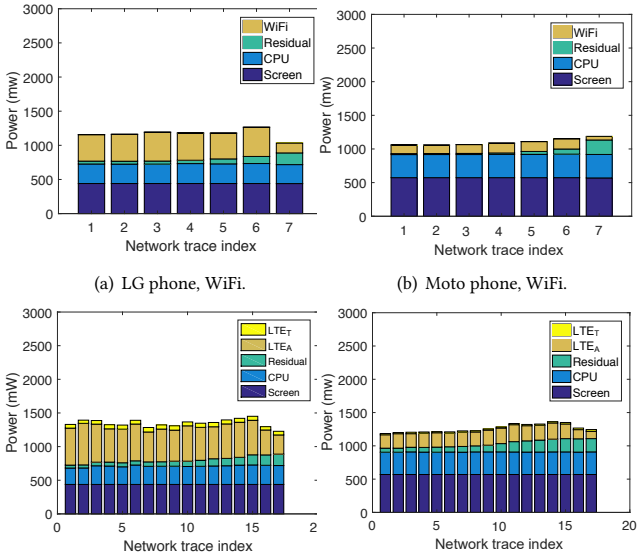
In this section, we conduct experiments in WiFi and LTE networks using both regular and 360° videos to validate the power models for ABR streaming. We further present the component-wise power breakdown of ABR streaming in various scenarios.

6.1 Experiment Setup

Unless otherwise stated, the results are obtained using ExoPlayer with an HTTP server that we set up. All the experiments are conducted in the wild in WiFi and LTE networks, at locations with good signal strength so that the network bandwidth is high and stable. For repeatable experiments and apple-to-apple comparisons, we emulate varying real-world network conditions by running `tc` at the server to "play back" network bandwidth traces (each recording per-second network bandwidth) that were collected from WiFi and LTE networks under varying conditions (controlled experiments for YouTube player are described in §6.6). For WiFi, we use 7 traces collected at a public location on a university campus. Fig. 8(a) plots the average and standard deviation of the network bandwidth for these traces, sorted in the increasing order of the average network bandwidth. The first 6 traces have low to medium average bandwidth (0.3 to 4.7 Mbps); the last one has high average bandwidth (42.7 Mbps). For LTE, we use 17 traces collected from commercial LTE networks, as shown in Fig. 8(b). The first 15 traces have low to medium average bandwidth (0.3 to 3.7 Mbps); the last two have high average bandwidth (9.9 and 19.7 Mbps).

6.2 Model Validation

We validate the ABR streaming power model in §5 as follows. For a given video and network trace, the CPU, screen and network power draws are obtained in real time using our models; the residual power of a selected segment is obtained using one of the three approaches described earlier: (i) use $P_{R,\ell,i}$ directly, (ii) approximate $P_{R,\ell,i}$ as $P_{R,\ell}$, and (iii) approximate $P_{R,\ell,i}$ as $\overline{P}_{R,\ell}$, where $P_{R,\ell,i}$ is the residual power of segment i in track ℓ, $P_{R,\ell}$ is the average residual power of track ℓ of the video, and $\overline{P}_{R,\ell}$ is the average power draw of track ℓ across a set of representative videos (i.e., the nine regular and ten 360° videos in Tables 1 and 2, respectively). To investigate the impact of the length of the video on the accuracy of the model, we assume the length is 2, 4, 6, 8, or 10 minutes. The error of the ABR streaming power model is the total power draw from our model (sum of component power draws) compared with that from the power monitor.



(a) LG phone, WiFi. (b) Moto phone, WiFi.

(c) LG phone, LTE (d) Moto phone, LTE.

Figure 9: Power breakdown of ABR streaming for a regular video (Animation) in WiFi and LTE networks.

Our evaluation shows that, as expected, the errors are the lowest when using the first approach and the highest when using the third approach for per-segment residual power draw. On the other hand, all three approaches lead to low errors. We only report the results for one regular and 360° video (i.e., Animation and Sci-Fi) when the video length is 10 minutes on the LG phone; the results for the other settings (video lengths and Moto phone) are similar. For the regular video, the relative error under the three approaches is below 3%, 4% and 5% in WiFi networks, and below 5%, 6% and 7% in LTE networks, respectively. For the 360° video, the corresponding error bounds are 3%, 3%, and 4% in WiFi networks, and 3%, 3%, and 5% in LTE networks. The measurement results in the rest of the section use the third approach to obtain per-segment residual power.

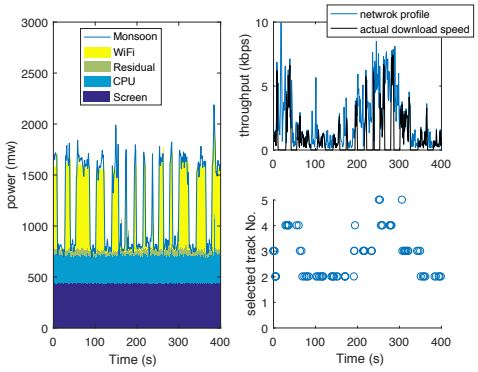


Figure 10: Power draw over time for a regular video under one network trace (Animation, WiFi network).

6.3 Results for Regular Videos

In each setting, we plot the measurement results in increasing order of the average network bandwidth of the network traces. The

results for each network trace are obtained from three runs. Only the average values are plotted (the standard deviations are low and omitted). The power draw of a component is the total energy consumed by the component divided by the duration of the run (from starting the player until the end of the video playback).

WiFi networks. The top row of Fig. 9 plots the component-wise power breakdown under the 7 WiFi traces for a regular video on the LG and Moto phones. For both phones, the screen power is a constant and the CPU power is close to a constant across the traces (CPU is not involved in decoding and is not sensitive to the tracks that are chosen in ABR streaming). As expected, the residual power increases with the average network bandwidth, since higher resolution segments are downloaded under higher network bandwidth; the increase is particularly significant for the last trace, where the highest track (1080p) is selected most of the time due to the high network bandwidth. Maybe surprisingly, the WiFi power draw under the first 6 traces is similar, despite their dramatically different profiles; the WiFi power draw under the last trace is noticeably lower. The above observations on WiFi power draw are due to the complex behavior of ABR streaming, as to be explained later. The power breakdown on these two phones has significant difference. Specifically, for the first 6 traces, the screen, CPU, residual, and WiFi powers take 35-38%, 23-25%, 4-8%, and 32-35% of the total power draw on the LG phone; the percentages on the Moto phone are 50-54%, 30-33%, 1-7%, and 12-13%. For the last trace, the percentages on the LG phone are 43%, 27%, 17%, and 14%, while the percentages are on the Moto phone are 48%, 29%, 18%, and 4%. Compared to the Moto phone, the LG phone has a higher percentage of power draw due to WiFi, and lower percentages due to screen and CPU.

We now explain the reasons behind the WiFi power draw as observed above. The WiFi power draw depends on two main factors: (i) the amount of time when the network interface card is in the Active state, and (ii) the amount of power draw during the Active state. The first factor is affected by the ratio of the network bandwidth to the average bitrate of a segment, which we term as *"slackness"*. A larger "slackness" leads to a faster buffer filling rate, and hence a shorter amount of time in the Active state (the downloading stops when the buffer reaches a certain threshold and resumes again when it is below another threshold; by default, these two thresholds are 50 and 15 seconds in the ExoPlayer that we use). In ABR streaming, the rate adaption logic adapts to the network bandwidth – under a higher network bandwidth, the ABR logic may choose a segment with a higher quality/bitrate and vice versa, and the track selection is based on the *declared* bitrate of a track (which is close to the peak bitrate, instead of the average bitrate). For the video in Fig. 9, since the ratio of the peak over average bitrate is larger for the higher tracks, the "slackness" under higher tracks tends to be larger than that under lower tracks, leading to less time in the Active state, and less average power draw. The impact of the second factor (i.e., the amount of power draw during the Active state) is in the opposite direction: it increases with network bandwidth (see Appendix), and hence higher bandwidth leads to higher power draw during the Active state. The interaction of the above two factors leads to similar WiFi power draw for the first six network traces. For the last network trace, since the bandwidth is significantly higher, the amount of "slackness" is significantly larger, and hence the first factor plays a more dominant role (i.e., the WiFi interface spends most time in the Idle state), leading to lower overall WiFi power draw.

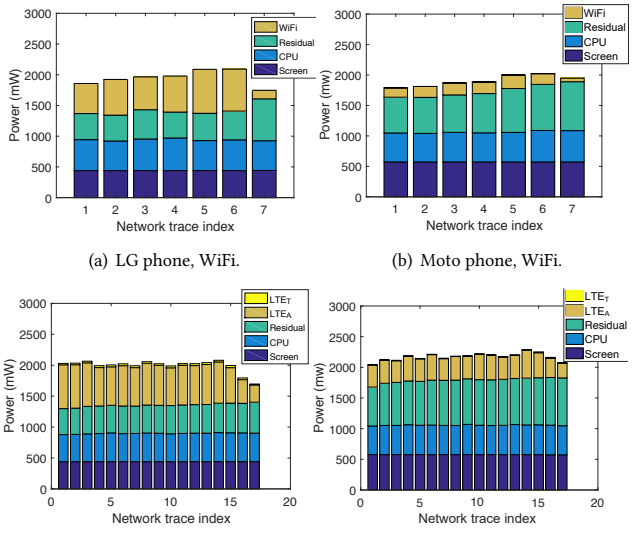
We illustrate the above complex interaction in Fig. 10 for one network profile (with the average bandwidth of 0.6 Mbps). It includes three sub-plots: the component-wise power draw, the network bandwidth and downloading throughput, and the selected tracks over time. We see that the downloading follows an on-off pattern, causing the network power draw to follow an on-off pattern as well. Higher tracks indeed tend to be selected when the network bandwidth is higher, during which the power draw is higher. The first sub-plot in Fig. 10 also shows the power measurement from the power monitor over time. Overall, it matches well with the total power draw calculated from our models.

LTE networks. The bottom row of Fig. 9 plots the results under the LTE traces for both phones. Compared to the results under WiFi networks, the overall power under LTE networks is higher for both phones, primarily due to higher LTE power draw (sum of Active and Tail power) compared to WiFi power draw. As a result, the percentages of screen and CPU power draws over the total power draw in LTE networks are lower than those in WiFi networks. For similar reasons as described earlier, for both phones, the LTE power draw under the first 15 traces is similar despite the significantly different profiles of these traces; the LTE power draw under the last two traces is significantly lower due to their significantly higher network bandwidth. For the first 15 traces, the screen, CPU, residual, and LTE powers account for 30-34%, 18-21%, 3-10%, and 39-48% of the total power draw on the LG phone; the percentages on the Moto phone are 42-48%, 25-28%, 5-15%, and 18-20%. For the last two traces, the percentages of the LTE power are 32% and 28% for the LG phone, and 11% and 9% for the Moto phone. The LG phone has a larger percentage of power draw due to network than the Moto phone.

6.4 Measurement Results for 360° Videos

In our experiments with 360° videos, since the Moto phone cannot play 2160p resolution, we prevent the 2160p track from being selected by removing it from the manifest files. For the LG phone, all seven tracks (144p to 2160p) can be selected.

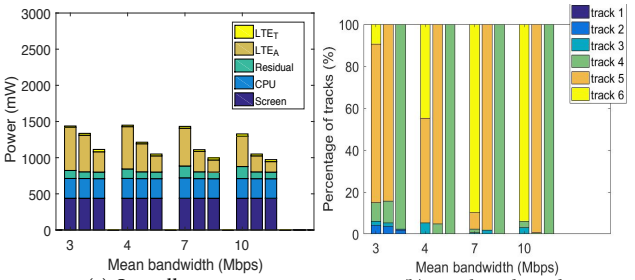
WiFi networks. The top row of Fig. 11 plots the power breakdown for a 360° video under the 7 WiFi traces on the two phones. For both phones, the CPU and residual power draws are significantly higher than those for the regular videos due to the additional operations required for rendering 360° videos (see §4.2). For the WiFi power draw, the trend is similar as that for the regular videos: the network power draw is similar under the first 6 traces, and lower under the last trace compared to other traces. For the first 6 traces, the screen, CPU, residual, and WiFi powers take 21-24%, 24-27%, 21-24%, and 26-34% of the total power draw on the LG phone; the percentages on the Moto phone are 28-32%, 24-27%, 33-37%, and 9-11%. For the last trace, the percentages are on the LG phone are 25%, 28%, 39%, and 8%; the percentages are on the Moto phone are 29%, 26%, 41%, and 3%. Different from the observations for regular videos, for both phones, the screen power becomes less dominant, while the residual power becomes much more significant. For the Moto phone, the



(a) LG phone, WiFi. (b) Moto phone, WiFi.

(c) LG phone, LTE (d) Moto phone, LTE.

Figure 11: Power breakdown of ABR streaming for a 360° video (Sci-Fi) over WiFi and LTE networks.



(a) Overall power. (b) Tracks selected.

Figure 12: Energy saving by capping the top track (LG phone). For each setting, the three bars represent the results for not capping, and capping to 720p and 480p, respectively.

percentage of power draw due to residual power is higher than that for the LG phone; their difference for the last trace is less significant, where the LG phone selects 2160p track most of the time, while the Moto phone selects 1080p most of the time.

LTE networks. The bottom row of Fig. 11 plots the results for a 360° video under the 17 LTE traces. We again observe that, for both phones, the residual power is a significant part of the total power draw despite the specialized hardware units. For the first 15 traces, the screen, CPU, residual, and LTE powers account for 21-22%, 22-23%, 22-23%, and 31-36% of the total power draw on the LG phone; the percentages on the Moto phone are 25-28%, 21-23%, 31-35%, and 17-20%. For the last two traces, the LTE power are lower than that under the first 15 traces: the percentages of LTE power draw is 23% and 17% for the LG phone, and 15% and 12% for the Moto phone.

6.5 Summary of Main Results

The above results are for one regular and one 360° video; the results for other videos show similar trend. Summarizing the above, the key observations are as follows.

• Due to significantly higher CPU and residual powers, 360° videos consume much more energy than regular videos: for regular videos, the total power draw under various WiFi and LTE network conditions is 1.7-2.4× of the baseline power draw (i.e., power draw with no ABR streaming, see §2) for the LG phone and 1.2-1.6× for the Moto phone, while for 360° videos, the corresponding ratios are 3.4-4.2× and 2.1-3.4×.

• For regular videos, the residual power is low due to the specialized hardware units (taking 4-17% and 1-18% of the total power draw under various network conditions for the LG and Moto phones, respectively); for 360° videos, the residual power is significant for both phones (the percentages being 17-39% and 26-41% for the two phones). Therefore, reducing residual power draw for 360° videos is an important future direction.

• We observe clear device heterogeneity: one phone has significantly lower network power draw, while significantly higher residual power for 360° videos than the other phone. Therefore, different energy reduction strategies need to be developed for different phones.

6.6 YouTube Player

The results reported so far are for ExoPlayer. We now briefly report the results for YouTube player, with the focus on power model validation. In the interest of space, we only present the validation results for ABR streaming over LTE networks. We run a YouTube app (version 14.06.56) that requests a regular video (Elephant Dream) from the YouTube server. The video has 6 tracks with resolution from 144p to 1080p. For controlled experiments, we use the set of LTE traces in §6.1. To control the network bandwidth, we run a proxy (`ProxyDroid`) on the phone; the proxy connects to a sever that we set up through a commercial LTE network, and the network bandwidth is controlled at the server using `tc`. The CPU, screen and network (WiFi and LTE) power draws are estimated using our models, and the per-segment residual power is obtained using the three approaches in §6.2. The error is the total power draw from our models compared with that from the power monitor. We again observe that the relative error is low (below 6%, 7% and 8% under the three approaches, respectively).

7 ENERGY REDUCTION STRATEGIES

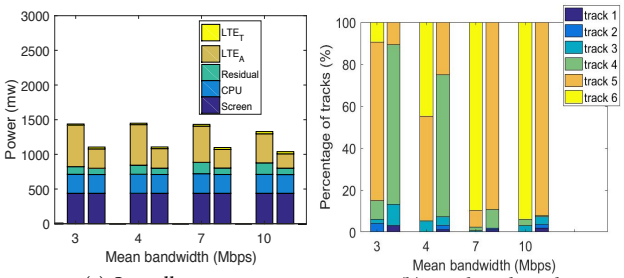
Our measurement results indicate that, for existing ABR schemes that are not energy-cognizant, a simple strategy to reduce their energy consumption is to lower the selected tracks so as to increase the amount of "slackness", i.e., the ratio of the network bandwidth to the bitrate of the selected track. In particular, when the original scheme selects a high-resolution track, it can be lowered, which reduces both network and residual power draw, while not degrading viewing quality much for small screens (such as those on phones). In the following, we explore two such strategies. Our goal is not to derive optimal strategies, rather, is to demonstrate that simple energy-aware enhancement to existing ABR schemes can already provide substantial energy savings.

Consider a video with n tracks. The first strategy *caps the top track* to $\bar{\ell} < n$. That is, if the rate adaptation logic selects a track

$\ell > \bar{\ell}$, then the track is set to $\bar{\ell}$. The second strategy uses *per-segment track reduction*: for a segment position, if the rate adaptation logic selects a track ℓ, then the track is reduced to $\ell - 1$ as long as $\ell \geq \underline{\ell}$. Both strategies can be easily deployed at the server (by modifying the manifest file) or the player (by including a prefilter or in the adaptation logic).

We implement the above two strategies in ExoPlayer. In the following, we quantify their energy savings using two regular videos (Fiction and Sports in Table 1, both with six tracks from 144p to 1080p), and three network profiles collected from a commercial LTE network. For each network profile, we scale it to create four traces with the average network bandwidth as 3, 4, 7 and 10 Mbps, respectively. All the experiments are conducted in the wild in a commercial LTE network, following the methodology in §6.1. The resulting viewing quality is quantified using Video Multimethod Assessment Fusion (VMAF) [48], a state-of-the-art perceptual quality metric that correlates quality strongly with subjective scores. Specifically, for the segments selected by a strategy under a network trace, we use $\sum_{i=1}^{n} w_i q_i$ to quantify the viewing quality, where w_i is the percentage of the time when track i is selected and q_i is the average VMAF score of all the segments in track i (the VMAF for a segment is the mean VMAF value of all the frames in the segment; using median leads to similar values). In the following, we only report the results of one video under one network profile (the results under the other two network profiles and for the other video show similar trends) for the LG phone.

For the capping top track strategy, we explore setting $\bar{\ell}$ to 5 or 4 (i.e., the highest track selected is 720p or 480p). Figures 12(a) and (b) plot respectively the power breakdown and the percentage that each track is selected under three cases (no capping, and capping the top track to 720p and 480p). We observe significant energy savings: compared to no capping, the percentage of savings is 8-22% and 23-30% when capping the top track to 720p and 480p. When the average network bandwidth is low (3 or 4 Mpbs), the saving is primarily from lower network power draw; when the network bandwidth is high (7 or 10 Mpbs), the saving is due to both lower residual and network power draws. We further see that capping the top track can lead to less low quality tracks, since avoiding high resolution tracks can lead to less buffer drainage and hence higher buffer level (under which ExoPlayer is less likely to choose low quality tracks). The VMAF is 92-99, 92-95 and 86-87 under the above three cases, showing that capping the top track only leads to slight degradation in viewing quality (VMAF score is between 0 and 100, and above 80 is considered as good quality [48]). Fig. 13 plots the results for the per-segment track reduction strategy, where $\underline{\ell} = 2$. We again observe significant energy savings (22-24%), with only slight viewing quality degradation (the VMAF is 92-99 and 84-93 for the two cases, respectively).



(a) Overall power. (b) Tracks selected.

Figure 13: Energy saving by per-segment track reduction (LG phone). For each setting, the two bars represent the result of the original scheme and that after applying the strategy.

8 DISCUSSION

Use cases. We next briefly outline several use cases of our power models for ABR streaming.

• **Evaluating the energy efficiency of existing ABR schemes.** Existing ABR schemes have been primarily evaluated in terms of QoE. Our power models allow easy evaluation of the energy consumption of these schemes on commodity phones. Specifically, when running an ABR scheme, the power breakdown of the various components can be obtained using our component-wise models based on system and player information that can be easily collected.

• **Towards energy-efficient ABR schemes.** Based on the evaluation of energy efficiency, existing ABR schemes that were not energy-cognizant can be improved to be more energy-efficient. The two simple energy reduction strategies in §7 are examples of such use cases. A more elaborate strategy can be based on a systematic what-if analysis. Our measurement results further show that, when the amount of slackness is high, the network power draw is only a small fraction of the total power draw. Therefore, another aspect is investigating how much the slackness needs to be under which there is no need to reduce the track selection. In addition to improving the energy efficiency of existing ABR schemes, our power models can be explicitly incorporated in designing new energy-aware ABR schemes. For example, suppose that a phone's battery level is B, and we would like to consume at most $B/2$ of the battery while watching a video. The ABR logic can then take account of the total energy constraint and the estimated energy consumption obtained online from our models to make track selections in order to maximize the QoE under the energy constraint.

• **Guiding track ladder design for mobile platform.** Our measurements results show that the residual power is more sensitive to resolution (particularly for very high resolution) while is insensitive to the bitrate for a given track resolution. The network power draw, on the other hand, is naturally related to average bitrate of a track. These results indicate that track design for ABR streaming needs to consider both the residual power and network power when determining the resolution and quality/bitrate of the tracks.

Generality of models and measurement findings. Our study focuses on two phones that are among the best phones with removable batteries for easy power measurement [20]. The coefficients of the component-wise models are specific to these two phones. On the other hand, our methodologies in deriving these models are applicable to other phones. Our measurement results are obtained by running popular ABR players on these two phones. The specific measurement findings may differ for newer phones. For instance, for newer phones, the residual power for higher resolution tracks may be reduced due to advances in hardware, and the CPU and residual power draws of 360° videos may reduce over time. On the other hand, our measurement methodologies, combined with new

component-wise models that are derived for new phones, can be used to obtain new measurement findings.

9 RELATED WORK

Power models and measurements for smartphones. In addition to those reviewed in §3, additional studies include power breakdown of a phone's main hardware components [23], energy drains in the wild [27], energy consumed by background applications [26, 43], energy profiling on smartphones [57, 60], energy emulation tool [54], overall and component power draw of real users [63], and power characteristics of LTE networks [42]. None of the above studies focuses on ABR streaming as in this study.

Energy consumption measurement for video streaming. Existing studies primarily focus on single-track regular videos, e.g., [39, 67] measure the energy consumption of commercial mobile video services, [69, 70] measure and analyze the power consumption of video streaming in LTE networks. While several studies present power measurements (e.g., aggregate power draw measured by power meters) for ABR streaming; none of them presents power breakdown as in our study. Power consumption of 360° streaming on phones is investigated in [44], which does not consider multi-track ABR streaming or propose component-wise power models.

Energy saving techniques for video streaming. The literature on reducing energy consumption in video streaming is extensive, see survey in [38]. Existing techniques include dynamic caching [47], traffic scheduling [37, 41, 66, 71], multipath delivery [36], coding/transcoding techniques [18, 55], traffic shaping on wireless devices [34, 51], managing radio states by predicting traffic patterns [32], and various techniques on reducing display energy [24, 28, 35, 49, 52]. Most techniques are designed for single-track video streaming. Energy saving techniques for ABR streaming are much sparser: [45, 65] propose both video adaptation and screen brightness techniques for energy-efficient ABR streaming; [68] presents a novel end-to-end adaptation framework that considers display and delivery jointly to reduce energy consumption; and [61] develops quality-aware strategies while reducing data usage, which can lead to reduced energy consumption. We propose energy saving strategies for ABR streaming based on the insights from our measurements, which have not been explored in existing studies.

10 CONCLUSION

In this paper, we have developed component-wise power models for ABR streaming. Using these models, we quantify the power breakdown in ABR streaming for both regular and 360° videos in a wide range of network settings. Our measurements validated the accuracy of the power models and provided a number of insights. Based on the insights, we explored two simple energy reduction strategies for ABR streaming, and demonstrated that they can provide up to 30% energy savings, with little degradation in quality in various network conditions.

ACKNOWLEDGEMENTS

We thank the anonymous reviewers who gave valuable feedback to improve this work, and our shepherd, Jörg Ott, for guiding us through the revisions. We also thank Shuai Hao for helpful discussion and suggestions, and his help on YouTube player measurement.

REFERENCES

[1] Linux tc. https://linux.die.net/man/8/tc, 2014.
[2] Snapdragon 4k Datasheet, Qualcomm. https://www.qualcomm.com/media/documents/files/snapdragon-4k-datasheet.pdf, 2014.
[3] ISO/IEC 14496-12 Information technology – Coding of audio-visual objects – Part 12: ISO base media file format. International Organization for Standardization, 2015.
[4] Adreno Video and Display, Qualcomm. https://www.intrinsyc.com/datasheets/qualcomm-adreno-video-and-display-infographic.pdf, 2015.
[5] Per-Title Encode Optimization. Netflix Technology Blog. https://medium.com/netflix-techblog/per-title-encode-optimization-7e99442b62a2, 2015.
[6] Snapdragon 820 - Technology and Traction, Qualcomm. https://www.qualcomm.com/media/documents/files/snapdragon-820-technology-and-traction-presentation-francisco-cheng.pdf, 2015.
[7] Android SensorManger. https://developer.android.com/reference/android/hardware/SensorManager., 2016.
[8] AT&T Application Resource Optimizer (ARO): User Guide. https://developer.att.com/static-assets/documents/aro/release/att-aro-user-guide-5.0.pdf, 2016.
[9] ExoPlayer 2 - Why, what and when? https://medium.com/google-exoplayer/exoplayer-2-x-why-what-and-when-74fd9cb139, 2016.
[10] Xiph.org Video Test Media. https://media.xiph.org/video/derf/, 2016.
[11] FFmpeg Project. https://www.ffmpeg.org/, 2017.
[12] H.264 : Advanced video coding for generic audiovisual services. ITU. https://www.itu.int/rec/T-REC-H.264, 2017.
[13] ISO/IEC 13818-1 Information technology – Generic coding of moving pictures and associated audio information – Part 1: Systems. International Organization for Standardization, 2018.
[14] youtube-dl. https://ytdl-org.github.io/youtube-dl/index.html, 2018.
[15] Monsoon power monitor, Monsoon Solutions Inc. https://www.msoon.com/LabEquipment/PowerMonitor/, 2019.
[16] Snapdragon 430 Mobile Platform, Qualcomm. https://www.qualcomm.com/products/snapdragon-430-mobile-platform, 2019.
[17] Snapdragon 820 Mobile Platform, Qualcomm. https://www.qualcomm.com/products/snapdragon-820-mobile-platform, 2019.
[18] F. Albiero, J. Vehkaperä, M. Katz, and F. Fitzek. Overall performance assessment of energy-aware cooperative techniques exploiting multiple description and scalable video coding schemes. In *Communication Networks and Services Research Conference (CNSR)*. IEEE, 2008.
[19] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. Energy consumption in mobile phones: a measurement study and implications for network applications. In *Proc. of IMC*, 2009.
[20] Brad. 9 Best Phones with Removable Battery In 2019. https://thedroidguy.com/2019/04/9-best-phones-removable-battery-2019-1079207, 2019.
[21] D. Brodowski. CPU frequency and voltage scaling code in the Linux (TM) kernel. 2017.
[22] D. Carraway. Lookbusy – a synthetic load generator. http://www.devin.com/lookbusy/, 2013.
[23] A. Carroll and G. Heiser. An analysis of power consumption in a smartphone. In *USENIX Annual Technical Conference*, 2010.
[24] N. Chang, I. Choi, and H. Shim. DLS: dynamic backlight luminance scaling of liquid crystal display. *IEEE Trans. VLSI Syst.*, 12(8), August 2004.
[25] C. Chen, Y.-C. Lin, A. Kokaram, and S. Benting. Encoding bitrate optimization using playback statistics for HTTP-based adaptive video streaming. https://arxiv.org/abs/1709.08763, 2017.
[26] X. Chen, J. Abhilash, D. Ning, C. H. Yu, G. Maruti, and V. Rath. Smartphone background activities in the wild: Origin, energy drain, and optimization. In *Proc. of ACM MobiCom*, 2015.
[27] X. Chen, N. Ding, A. Jindal, Y. C. Hu, M. Gupta, and R. Vannithamby. Smartphone energy drain in the wild: Analysis and implications. *ACM SIGMETRICS Performance Evaluation Review*, 43(1):151–164, 2015.
[28] L. Cheng, S. Mohapatra, M. E. Zarki, N. Dutt, and N. Venkatasubramanian. Quality-based backlight optimization for video playback on handheld devices. *Advances in Multimedia*, 1, 2007.
[29] J. D. Cock, Z. Li, M. Manohara, and A. Aaron. Complexity-based consistent-quality encoding in the cloud. In *Proc. of IEEE International Conference on Image Processing*, 2016.
[30] M. Covell, M. Arjovsky, Y.-C. Lin, and A. C. Kokaram. Optimizing transcoder quality targets using a neural network with an embedded bitrate model. In *Visual Information Processing and Communication*, 2016.
[31] J. De Cock, A. Mavlankar, A. Moorthy, and A. Aaron. A large-scale video codec comparison of x264, x265 and libvpx for practical VOD applications. In *SPIE, Applications of Digital Image Processing*, 2016.
[32] S. Deng and H. Balakrishnan. Traffic-aware techniques to reduce 3G/LTE wireless energy consumption. In *Proc. of ACM CoNEXT*, 2012.
[33] N. Ding, D. Wagner, X. Chen, A. Pathak, Y. C. Hu, and A. Rice. Characterizing and modeling the impact of wireless signal strength on smartphone battery drain.

ACM SIGMETRICS Performance Evaluation Review, 41(1):29–40, 2013.
[34] F. R. Dogar, P. Steenkiste, and K. Papagiannaki. Catnap: exploiting high bandwidth wireless interfaces to save energy for mobile devices. In *Proc. of ACM MobiSys*, 2010.
[35] M. Dong and L. Zhong. Chameleon: a color-adaptive web browser for mobile OLED displays. In *Proc. of ACM MobiSys*, 2011.
[36] Y. Go, O. C. Kwon, and H. Song. An energy-efficient HTTP adaptive video streaming with networking cost constraint over heterogeneous wireless networks. *IEEE Transactions on Multimedia*, 17(9), September 2015.
[37] M. A. Hoque, M. Siekkinen, and J. K. Nurminen. Using crowd-sourced viewing statistics to save energy in wireless video streaming. In *Proc. of ACM MobiCom*, 2013.
[38] M. A. Hoque, M. Siekkinen, and J. K. Nurminen. Energy efficient multimedia streaming to mobile devices–a survey. *IEEE Communications Surveys & Tutorials*, 16(1):579–597, 2014.
[39] M. A. Hoque, M. Siekkinen, J. K. Nurminen, and M. Aalto. Dissecting mobile video services: An energy consumption perspective. In *Proc. of WoWMoM*, 2013.
[40] M. Hosseini and V. Swaminathan. Adaptive 360 VR video streaming: Divide and conquer! *arXiv preprint arXiv:1609.08729*, 2016.
[41] W. Hu and G. Cao. Energy-aware video streaming on smartphones. In *Proc. of IEEE INFOCOM*, pages 1185–1193. IEEE, 2015.
[42] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. A close examination of performance and power characteristics of 4G LTE networks. In *Proc. of MobiSys*. ACM, 2012.
[43] J. Huang, F. Qian, Z. M. Mao, S. Sen, and O. Spatscheck. Screen-off traffic characterization and optimization in 3G/4G networks. In *Proc. of IMC*, 2012.
[44] N. Jiang, V. Swaminathan, and S. Wei. Power Evaluation of 360 VR Video Streaming on Head Mounted Display Devices. In *Proc. of Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*. ACM, 2017.
[45] J.-S. Leu, M.-C. Yu, C.-Y. Liu, A. P. B. Budiarsa, and V. Utomo. Energy efficient streaming for smartphone by video adaptation and backlight control. *Computer Networks*, 113:111–123, 2017.
[46] C.-Y. Li, C. Peng, S. Lu, and X. Wang. Energy-based rate adaptation for 802.11n. In *Proc. of MobiCom*, 2012.
[47] X. Li, M. Dong, Z. Ma, and F. C. Fernandes. GreenTube: power optimization for mobile videostreaming via dynamic cache management. In *Proc. of ACM Multimedia*, 2012.
[48] Z. Li, A. Anne, K. Ioannis, M. Anush, and M. Megha. Toward a practical perceptual video quality metric. https://medium.com/netflix-techblog/toward-a-practical-perceptual-video-quality-metric-653f208b9652, 2016.
[49] C.-H. Lin, P.-C. Hsiu, and C.-K. Hsieh. Dynamic backlight scaling optimization: A cloud-based energy-saving service for mobile streaming applications. *IEEE Trans. Comput.*, 63, February 2014.
[50] Y.-C. Lin, H. Denman, and A. Kokaram. Multipass encoding for reducing pulsing artifacts in cloud based video transcoding. In *Proc. of IEEE International Conference on Image Processing*, 2015.
[51] J. Liu and L. Zhong. Micro power management of active 802.11 interfaces. In *Proc. of ACM MobiSys*, 2008.
[52] Y. Liu, M. Xiao, M. Zhang, X. Li, M. Dong, Z. Ma, Z. Li, and S. Chen. GoCAD: GPU-assisted online content-adaptive display power saving for mobile devices in internet streaming. In *Proc. of World Wide Web Conference (WWW)*, April 2016.
[53] C. Marshall. Youtube brings us stats for nerds. https://tubularinsights.com/youtube-stats-for-nerds/, 2013.
[54] R. Mittal, A. Kansal, and R. Chandra. Empowering developers to estimate app energy consumption. In *Proc. of MobiCom*, 2012.
[55] S. Mohapatra, R. Cornea, H. Oh, K. Lee, M. Kim, N. Dutt, R. Gupta, A. Nicolau, S. Shukla, and N. Venkatasubramanian. A cross-layer approach for power-performance optimization in distributed mobile systems. In *Proc. of Parallel and Distributed Processing Symposium*. IEEE, 2005.
[56] D. Ochi, Y. Kunita, K. Fujii, A. Kojima, S. Iwaki, and J. Hirose. HMD viewing spherical video streaming system. In *Proc. of the ACM international conference on Multimedia*, 2014.
[57] A. Pathak, Y. C. Hu, and M. Zhang. Where is the energy spent inside my app? Fine grained energy accounting on smartphones with Eprof. In *Proc. of EuroSys*, 2012.
[58] A. Pathak, Y. C. Hu, M. Zhang, P. Bahl, and Y.-M. Wang. Fine-grained power modeling for smartphones using system call tracing. In *Proc. of EuroSys*, pages 153–168. ACM, 2011.
[59] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan. Optimizing 360 video delivery over cellular networks. In *Proc. of the ACM Workshop on All Things Cellular: Operations, Applications and Challenges*, 2016.
[60] F. Qian, Z. Wang, A. Gerber, Z. Mao, S. Sen, and O. Spatscheck. Profiling resource usage for mobile applications: a cross-layer approach. In *Proc. of ACM MobiSys*, 2011.
[61] Y. Qin, S. Hao, K. R. Pattipati, F. Qian, S. Sen, B. Wang, and C. Yue. Quality-aware strategies for optimizing ABR video streaming QoE and reducing data usage. In *Proc. of ACM MMSys*, June 2019.
[62] P. Rondao Alface, J.-F. Macq, and N. Verzijp. Interactive omnidirectional video delivery: A bandwidth-effective approach. *Bell Labs Technical Journal*, 16(4):135–147, 2012.
[63] A. Shye, B. Scholbrock, and G. Memik. Into the Wild: Studying Real User Activity Patterns to Guide Power Optimizations for Mobile Architectures. In *Proc. of Micro*, 2009.
[64] D. Stenberg. Curl. https://curl.haxx.se/, 2019.
[65] B. Varghese, G. Jourjon, K. Thilakarathne, and A. Seneviratne. e-DASH: Modelling An Energy-Aware DASH Player. In *Proc. of WoWMoM*, June 2017.
[66] S. Wei, V. Swaminathan, and M. Xiao. Power efficient mobile video streaming using HTTP/2 server push. In *Proc. of International Workshop on Multimedia Signal Processing (MMSP)*, 2015.
[67] Y. Xiao, R. S. Kalyanaraman, and A. Yla-Jaaski. Energy consumption of mobile YouTube: Quantitative measurement and analysis. In *Proc. of Next Generation Mobile Applications, Services, and Technologies*, 2008.
[68] Z. Yan and C. W. Chen. RnB: rate and brightness adaptation for rate-distortion-energy tradeoff in HTTP adaptive streaming over mobile devices. In *Proc. of ACM MobiCom*, 2016.
[69] J. Zhang, G. Fang, C. Peng, M. Guo, S. Wei, and V. Swaminathan. Profiling energy consumption of DASH video streaming over 4G LTE networks. In *Proc. of International Workshop on Mobile Video*, page 3. ACM, 2016.
[70] J. Zhang, Z.-J. Wang, S. Guo, D. Yang, G. Fang, C. Peng, and M. Guo. Power consumption analysis of video streaming in 4G LTE networks. *Wireless Networks*, May 2017.
[71] J. Zhang, Z.-J. Wang, Z. Quan, J. Yin, Y. Chen, and M. Guo. Optimizing power consumption of mobile devices for video streaming over 4G LTE networks. *Peer-to-Peer Networking and Applications*, 11(5):1101–1114, 2018.
[72] L. Zhang, B. Tiwana, R. P. Dick, Z. Qian, Z. M. Mao, Z. Wang, and L. Yang. Accurate online power estimation and automatic battery behavior based power model generation for smartphones. In *IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, 2010.
[73] Y. Zhang, X. Wang, X. Liu, Y. Liu, L. Zhuang, and F. Zhao. Towards better CPU power management on multicore smartphones. In *Proc. of the Workshop on Power-Aware Computing and Systems*, page 11. ACM, 2013.

A COMPONENT POWER MODEL

CPU power model. We use the following methodology to obtain the coefficients of the model in Eq. (1). For a phone, we create a dataset by setting the frequency of each core to one of the predominant frequencies that is used during video streaming, and varying its utilization from 20% to 100% by running a workload generation tool, Lookbusy [22], in the background. For each setting, we collect data for 10 minutes. The data collected from all the settings is then used to train and validate the model. The 10-fold cross validation error is 1.8% and 1.7% for the LG and Moto phones, respectively, indicating that the model is robust and has no overfitting. After that, we obtain the coefficients of the model for each phone using all the data. The coefficients for the two phones are listed Table 3. The average errors of the models (by comparing model estimation and the measurement from the power monitor) are very low (both 1.3%) for the two phones.

For comparison, we also run 10-fold cross validation using the model in [73]. The cross validation error is 6.8% and 8.1% for the LG and Moto phones, significantly larger than that for our models. Since the CPU model is used to isolate power draw from other components, it is important to use the more accurate model derived using our approach.

Screen power model. For the LG phone, the screen power draw when the screen brightness is set to 30%, 50% or 80% of the full brightness is 293, 440, and 754 mW, respectively (each obtained as the average of five measurements; the variance is small with the the Coefficient of Variation below 0.01). For the Moto phone, the corresponding values are 446, 573, and 858 mW, respectively.

LTE power model. We conduct a set of experiments in a commercial LTE network. In each experiment, we use `cURL` [64] to download

Table 3: CPU power model.

		CPU frequency (Mhz)	P_Δ (mW)	P_B (mW)
LG	Small cores	653	167	38
		730	179	56
	Large cores	653	201	38
		1037	291	46
Moto	Small cores	768	26	34
	Large cores	960	49	34

data from a large file stored at a remote well-provisioned server using TCP (since TCP is the commonly used transport protocol in ABR streaming). The downloading follows a periodic on-off pattern, on for 2 minutes and then off for 2 minutes. Only the CPU and the LTE interface card are active during each experiment (the screen is off). The LTE power draw is obtained by subtracting the CPU power (derived using our model) from the total power measured by the power monitor. To investigate the impact of signal strength on the LTE power draw, we conduct experiments at three locations with significantly different signal strength, the RSRP (Reference Signals Received Power) values being in a wide range from -108 to -86 dBm. The measurements at each location contain four hours of data collection. We observe that the LTE power draw increases linearly with the downlink throughput, and is a function of RSRP, with higher RSRP leading to lower power draw. The power draw of the Idle state is approximately zero. Correlating the power and the throughput, we confirm that the Tail time is approximately 11 seconds, which is consistent with the results in [27, 42].

Our proposed model considers both the Active power and Tail power; transition power draw among different RRC states is not included since the transition latency is very short [42]. The device is considered as being in the *Active state* if there is data transfer within the past second, and transits to the *Idle state* when there is no traffic for 11 seconds. Let $P_{LTE,A}$ and $P_{LTE,T}$ denote respectively the Active power and Tail power, respectively. We set $P_{LTE,T}$ as a constant, obtained directly from the measurements. Based on the earlier observations, we can model $P_{LTE,A}$ as a function of both downlink throughput and RSRP. Specifically, we can divide the value of RSRP into multiple ranges, and model $P_{LTE,A}$ as a linear function of the downlink throughput for each range. This approach, however, has two drawbacks: (i) it is difficult to decide the ranges of the RSRP values, and (ii) the model will be very complex. For simplicity, we propose the following model that does not take RSRP into account, and only considers the downlink throughput: $P_{LTE,A} = \alpha_d S_d + \beta$, where S_d is downlink throughput, and α_d and β are coefficients that can be obtained using linear regression from the training data. As we shall show below, this model provides good accuracy even though it does not consider RSRP. We also explored the model in [42], which considers both downlink and uplink throughput and found that it is less accurate than our model. This might be because the traffic in the uplink is sparse (primarily ACKs corresponding to the data packets); including the uplink throughput leads to more noises in the model, and hence larger errors.

Table 4: LTE power model.

	α_d (mW/Mbps)	β (mW)	$P_{LTE,T}$ (mW)
LG V20	29.2	1050.2	110.4
Moto G5	23.1	429.4	53.2

Table 5: WiFi power model.

	LG V20		Moto G5	
	α_d (mW/Mbps)	β (mW)	α_d (mW/Mbps)	β (mW)
<15 Mbps	35.3	770.9	23.1	261.8
(15, 30] Mbps	21.0	971.2	10.2	410.2
(30, 50] Mbps	15.1	1091.2	4.2	592.2
>50 Mbps	1.7	1682.0	0.7	759.1

The coefficients of the proposed LTE power model are obtained using the measurements collected at the three locations with varying signal strengths. The 10-fold cross validation error is 6.4% and 7.1% for the LG and Moto phones, respectively. We then obtain the coefficients of the models for the two phones, as shown in Table 4. Comparing the estimations from the model and the ground-truth measurements, the average error is 5.2% and 6.6% for the two phones, respectively.

WiFi power model. We conduct a set of experiments over a WiFi network with varying throughput and signal strength settings. The phone is close or far away from the AP (leading to higher or lower signal strength). The downlink throughput is not limited or limited to a value by running `tc` on the server (where the bandwidth limit is varied from 2 to 130 Mbps). Each experiment involves periodic on-off data downloading, as that used for obtaining the LTE power model. In each setting, we repeat the on-off downloading procedure for 10 minutes. Only the WiFi interface card and CPU are active during the experiments. The WiFi power is obtained as the total power measured by the power monitor minus the CPU power (derived from our model). We observe a piecewise linear relationship between the WiFi power and downlink throughput. The signal strength does impact power draw: for the same downlink throughput, higher signal strength tends to lead to lower power draw (figure omitted).

Since Tail time is very short (only hundreds of milliseconds), we do not consider it in the power model. Specifically, we consider each 100 ms interval. If there is traffic, we say the WiFi interface is in *Active state*; otherwise, we say it is in *Idle state*. As the LTE power model, we do not take signal strength into account for simplicity. Based on the piecewise linear observation earlier, we divide the downlink throughput into ranges, and for each range, adopt a linear throughput-based model as $P_{WiFi} = \alpha_d S_d + \beta$, where S_d is the downlink throughput, and α_d and β can be obtained using linear regression. The 10-fold cross validation error for the four cases is 0.2-1.8% for the LG phone and 0.3-1.6% for the Moto phone. The coefficients of the models (Table 5) are learned using the entire dataset. The average error of the model is 1.2% for the LG phone and 1.1% for the Moto phone.