EnDASH - A Mobility Adapted Energy Efficient ABR Video Streaming for Cellular Networks

Abhijit Mondal[1], Basabdatta Palit[1], Somesh Khandelia[1], Nibir Pal[1],
Jay Jayatheerthan[2], Krishna Paul[2], Niloy Ganguly[1], Sandip Chakraborty[1]
[1]Indian Institute of Technology, Kharagpur, India
[2]Intel Technology Pvt. Ltd. Bengaluru, India

***Abstract*—User experience of watching videos in smartphones while travelling is often limited by fast battery drainage. Existing client video players use adaptive bitrate (ABR) streaming through Dynamic Adaptive Streaming over HTTP (DASH) to improve user's Quality of Experience (QoE) while ignoring the energy savings aspect, which has been addressed in our work. In this paper, we propose EnDASH - an energy aware wrapper over DASH which minimizes energy consumption without compromising on QoE of users, under mobility. First, we undertake an extensive measurement study using two phones and three service providers to understand the dynamics between energy consumption of smartphones and radio related network parameters. Equipped with this study, the proposed system predicts cellular network throughput from the radio parameters within a finite future time window. The prediction engine captures the effect of associated technology and vertical handovers on throughput, unlike existing works. EnDASH then uses deep reinforcement learning based neural networks to first tune the playback buffer length to the average predicted cellular network throughput and then to select an optimal video chunk bitrate. It achieves a near 30% decrease in the maximum energy consumption than state-of-the-art ABR Pensieve algorithm while performing almost at par in QoE.**

***Index Terms*—4G LTE, Energy Efficiency, ABR Video Streaming, Cellular Networks, Mobility**

I. Introduction

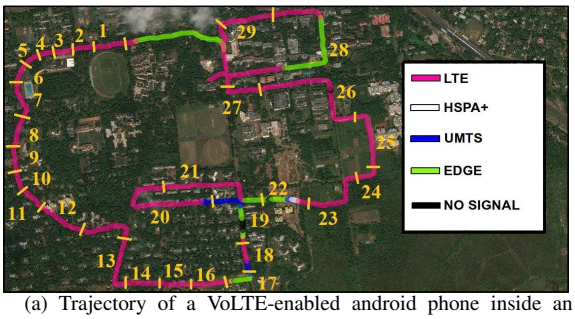
With the pervasive roll out of the 4^{th} Generation (4G) cellular networks, online video streaming in smartphones has become one of the most popular modes of entertainment [1], especially in many developing countries. The availability of ultra cheap data plans, affordable smartphones, and local language based content on YouTube, Netflix, etc., has led to a record increase in the number of mobile video subscribers as well as their engagement time [2]. Subscribers have shown an inclination towards watching streaming videos even while travelling, irrespective of the distances travelled. Provisioning the expected Quality of Experience (QoE) to video users during travelling requires the reception of a stable connection quality at the User Equipments (UEs), which often eludes users in developing nations. This is because in these countries service providers often compromise on the network infrastructure to provide low cost internet [3].

Another non-negligible impact of mobility on video streaming in smartphones is the drainage of battery power. Video streaming itself is a power hungry application [4]. Our experiments show that under mobility, video streaming apps consume even more power (§III). The reason for this can be attributed to the fluctuating connection quality experienced while travelling. This paper proposes to improve the smartphone battery usage through the design of an energy efficient video streaming algorithm that leverages the fluctuating cellular network throughput to choose optimal bitrates while not compromising on the required QoEs.

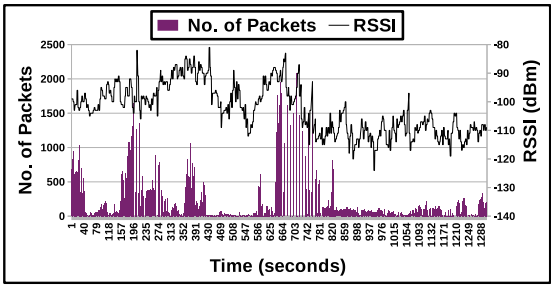
To establish the cellular connectivity scenario, we have carried out an extensive measurement-based study for eleven months over five different cities in India, including urban areas as well as while travelling on highways (§III). We have recorded the signal received by medium budget Moto G5 and Micromax phones, while using the cellular data connection of leading service providers in India, like Airtel, Reliance Jio, and Vodafone. A *key finding* of this experiment is shown in Fig. 1(a), which displays the trajectory of a Moto G5 phone connected to the Airtel network on a single day over the IIT Kharagpur campus area of 8.5 sq.km. It is seen that even within the small area covered, the phone connects to different generations of cellular technologies, each highlighted in different colours. Two inferences can readily be drawn from Fig. 1(a). (i) A service provider does not guarantee a complete 4G connectivity across the entire coverage region. A UE often has to fallback to legacy 3^{rd} Generation (3G) or 2^{nd} Generation (2G) networks. (ii) Even when connected to 4G, the signal strength shows random variations. For example, for the experiments conducted, the RSSI of 4G networks fluctuates between -87dBm to -115dBm. This portrays the typical connectivity scenario in many regions in India. Another important observation from our experiments, shown in Fig. 1(b), is that the volume of data downloaded by video players is not always commensurate with the current network signal strength. Several opportunities to download large chunks of data at high signal strength conditions remain unexploited.

Video streaming applications, including live streaming, predominantly use Dynamic Adaptive Streaming over HTTP (DASH) [5]. In this protocol, the target video is broken into chunks of fixed playback time, and multiple copies of each chunk is stored at different quality levels, i.e., bitrates. The client side video player uses adaptive bitrate (ABR) algorithms [6]–[10] to decide on the bitrate at which the next chunk is to be fetched. Existing DASH algorithms primarily take bitrate decisions based on the estimated network throughput or the playback buffer size while attempting to improve user's QoE. We, on the other hand, hypothesize that it is possible to significantly lower the energy usage without sacrificing QoE

ISBN 978-3-903176-28-7



(a) Trajectory of a VoLTE-enabled android phone inside an academic campus. Associated network standards (4G, HDPA, UMTS, EDGE) highlighted using different colours



(b) Packet trace of a 360p Youtube video download with the temporal variation in the RSSI during the download

Fig. 1. Experimental Observations

if we intelligently utilize the nuances of the cellular network throughput fluctuation during ABR streaming.

To this end, we build an energy efficient video player **EnDASH** as a wrapper over DASH (§IV). EnDASH predicts the average cellular network throughput over a finite future time window to take decisions on the opportune fetching of video chunks by dynamically increasing the playback buffer size (§IV). So, it (i) predicts the the cellular network link throughput from radio related parameters (§IV-A), (ii) then uses the predicted throughput to predict the playback buffer length, and (iii) finally uses the predicted buffer length to choose optimal bitrates for future chunks. Thus, EnDASH consists of three prediction modules, of which the throughput prediction uses random forest learning (step (i)). However, EnDASH works over a smartphone connected to a cellular network which is a randomly varying environment. So, it uses Deep Reinforcement Learning (RL) for the buffer-length prediction mechanism (step (ii)) and bitrate adaptation (step (iii)), which allows EnDASH to exploit the actual performance of previous choices to tune its operation to the current characteristic of the network. Consequently, EnDASH can start without any apriori knowledge and gradually learn through exploration and exploitation. The playback buffer length and bitrate decision engines use Deep Neural Networks to map 'raw' observations to outputs. These two engines operate using $A3C$ [6], a state-of-the-art actor-critic RL algorithm, and run asynchronously with respect to one another. We train individual prediction modules over a large corpus of collected data traces and evaluate EnDASH using an emulation environment.

Evaluation using our emulation platform shows that in comparison to existing ABR algorithms [6]–[8], EnDASH significantly improves the energy savings in smartphones (§V). Energy saved from playing a 2200 second video using EnDASH can be used to gain an additional 1440 seconds of video playback time in comparison to the popular Pensieve algorithm [6]. The energy savings, however, comes at the cost of marginally reduced QoE. One of the most salient features of EnDASH, which sets it apart from existing ABR algorithms [6]–[12] is that its throughput prediction engine captures the impact of not only the received signal strength but also other network related parameters, such as different technologies and vertical handovers. The Mean Absolute Percentage Error (MAPE) of the throughput prediction engine of EnDASH varies from 8% to 13% across different scenarios. The improvement is particularly pronounced in regions having a substantial presence of legacy networks. The improved throughput prediction assists the EnDASH RL engine to accurately capture the playback buffer evolution, which in turn aids the video segment download in an energy-efficient but QoE favourable manner.

II. Background and Related Work

4G LTE smartphones are designed to maintain network connectivity using a Radio Resource Control (RRC) state machine with two states: *CONNECTED* and *IDLE* [13]. With no active transmission, the UE is in the low power *IDLE* state where no radio resource is assigned. Once a packet arrives, the UE jumps to the high power *CONNECTED* state, in which radio resources are assigned and data transmission takes place. To reduce the incumbent delay and energy consumption associated with the state promotion, the UE waits for a duration called tail time in the *CONNECTED* state before returning to the *IDLE* state even after packet transmission is over. To save energy in the tail period, LTE uses Discontinuous Reception (DRX) during which the cellular interface periodically monitors the control channel for incoming packets and then goes to sleep [13]. Evidently, if video is downloaded during poor connection quality, then the smartphone will have a longer *CONNECTED* state dwell time resulting in higher energy consumption. Existing ABR video streaming algorithms, however, primarily focus on improving QoE while paying little attention to energy savings.

Improving QoE: ABR video streaming algorithms either choose buffer occupancy [7], [14] or both buffer occupancy and current chunk or network throughput [8], [11], [15]–[17] to select optimal bitrates for future video chunks. Examples would be BOLA [7] and MPC [8], respectively. Pensieve [6] uses a deep RL algorithm for optimal bitrate selection to maximize over a QoE metric. However, none of these works focus on saving device energy consumption under mobility conditions in 4G LTE networks.

In this work, we aim to improve video user's energy consumption over cellular networks while not compromising on QoE by tuning playback buffer size to network throughput. Hence, the proposed algorithm should use cellular network throughput prediction. Several works focus on bandwidth

prediction for improving the bitrate selection of ABR streaming algorithms [9], [18]–[20]. Some of these works also focus on predicting cellular network throughput [9], [19]–[24]. However, unlike our EnDASH algorithm, none of the works consider the unique situation of co-existence of different technologies and frequent handover from one technology to another for throughput prediction.

Reducing Energy Consumption: Several works in literature investigate energy consumption reduction of mobile phones independently of QoE or ABR streaming algorithms. The BarTendr algorithm in [12] tunes the download sessions in 3G networks to the network conditions for saving energy. However, it quantifies the network condition using received signal strength only while giving no weightage to handovers or associated technologies. GreenTube in [4] proposes to tune cache management to user behaviour and network conditions. A popular method to reduce energy consumption in mobile phones is to optimize the tail energy, which is achieved in [25] by either prefetching or delaying packets.

To tune packet downloads to network conditions so as to save energy requires a detailed energy profiling of the phones and service providers. This can be obtained through detailed measurement studies as in [13]. While [26]–[29] focus on the measurement of power consumption of video traffic over HTTP in 4G networks, the effect of mobility on signal strength in 4G networks is presented briefly. In [13], [30] are presented measurement studies on mobility support in 4G Long Term Evolution (LTE) networks. However, to the best of our knowledge, there is no comprehensive measurement study on video streaming under mobility in cellular-only networks. Furthermore, an inherent assumption in these papers is the uninterrupted availability of 4G signal. In contrast, the present work focuses on optimization of energy consumption and QoE of mobile video users in scenarios where legacy networks are present in addition to 4G, under mobility conditions.

III. PILOT STUDY

To design an energy efficient video streaming algorithm, which tunes playback buffer length with cellular network throughput, we need a detailed comprehension of the complex relationships between radio related parameters, such as signal strength, vertical and horizontal handovers, speed, etc., on one hand, and throughput and energy consumption on the other hand. To develop this understanding, we have carried out an extensive measurement based study as discussed next.

A. Experimental Set-up

1) Hardware Setup: For power consumption and energy profiling, we have selected medium budget VoLTE enabled smartphones - Moto G5 ($149) and Micromax Canvas Infinity ($87). Table I outlines their configuration details. We record the power consumption of the phones using Monsoon Solutions High Voltage Power Monitor (HVPM) [25], [31], [32] (Fig. 2(a)) in both stationary and mobile conditions, when connected to three leading mobile internet service providers, Airtel, Reliance JIO, and Vodafone. The High voltage Power Monitor (HVPM) records the power at a frequency of 5000 Hz. We have collected power consumption data in three different cities (Kolkata, Kharagpur, Guwahati).

TABLE I
DETAILS OF THE MOBILE HANDSETS USED

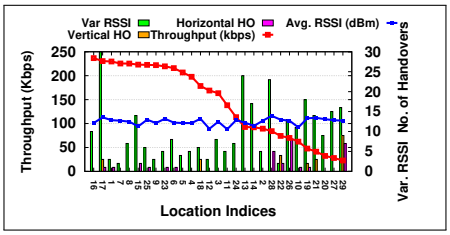
	Moto G5 (Price: US$ 149)	**Micromax Canvas Infinity (Price US$ 87)**
N/W Tech.	GSM/ HSPA/ LTE	GSM/ HSPA/ LTE
N/W Speed	HSPA 42.2/5.76 Mbps, LTE Cat4 150/50 Mbps	HSPA 42.2/5.76 Mbps, LTE Cat4 150/50 Mbps
OS	Android 7.0 (Nougat)	Android 7.1.2 (Nougat)
Chipset	Qualcomm MSM8937 Snapdragon 430 (28 nm)	Qualcomm MSM8917 Snapdragon 425 (28 nm)
CPU	Octa-core 1.4 GHz Cortex-A53	Quad-core 1.4 GHz Cortex-A53
GPU	Adreno 505	Adreno 308

Besides power consumption, we have also collected extensive data on the received throughput of mobile phones in public buses, cars, and while walking across five cities of India (Kharagpur, Kolkata, Guwahati, Bengaluru, Malda). This dataset also includes data collected while travelling on highways. We have used workloads of 6Mb, 100Mb, 1GB file download, as well as of video streaming using Netflix, Hotstar, SonyLiv and Amazon Prime. This has allowed a detailed energy profiling of the smartphones. The entire corpus of collected data traces amounts to more than 50GB and has been collected over a period of eleven months.

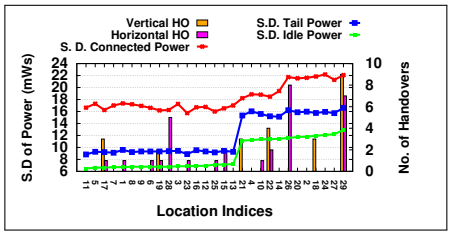
2) Software Setup and Outcome: In this section, we outline the software setup. We have considered two primary workloads- (a) file download, and (b) video streaming.

(a) File Download We have used file downloads to profile the power consumption in different RRC states. To enable the download, we have developed an HTTP client-server program, where the client program runs in the phone and the server is hosted on an Amazon Web Server (AWS). We have collected traffic traces, radio related information, and location and speed information using *tcpdump*, Network Monitor Lite, and GPS Logger applications, respectively. The collected traffic traces have been analyzed using Wireshark. We have followed [25] to derive the RRC state diagram. Accordingly, to measure the *IDLE* state power, we have kept the screen ON, uninstalled all non-default applications, kept all background applications disabled, the WiFi interface turned off, and the mobile network interface switched on but without any active traffic transmission. The corresponding power consumed by the operating system, processor, display, some default background and network-related operations, etc., constitutes the *IDLE* state power. We have continued measuring the power consumption starting from the *IDLE* state, throughout the file download till the trace came back to the *IDLE* power. Once a file download starts, a jump in power consumption is noted. After the download completes, it drops to an intermediate value during the tail time before returning to the *IDLE* state [25]. To derive the RRC states, we have downloaded files of different lengths several times using different phones and service providers in the different cities in stationary condition. The power consumption and dwell time of each RRC state has been found to be different for different phones, service

(a) The Experimental Setup inside a slow moving electric vehicle



(b) Sorted throughput of the twenty-nine stretches of Fig.1(a) and its corresponding variations with RSSI, Vertical and Horizontal Handovers



(c) Variations of Power Consumption with RSSI, and Vertical and Horizontal Handovers over the user's trajectory shown in Fig.1(a)

Fig. 2. Experimental setup and Throughput and Power consumption variations of a user under mobility; Phone: Moto G5, Service Provider: Airtel

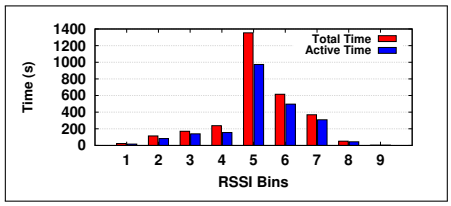


Fig. 3. Total time spent in each RSSI bin and the active time in each bin

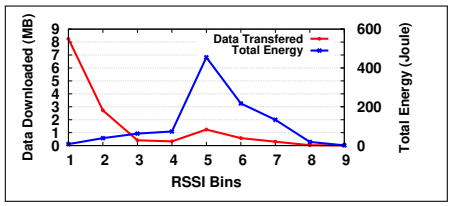


Fig. 4. Amount of data downloaded & Energy Consumption in RSSI Bins

providers as well as cities. With mobility, the average power consumed in each RRC state has remained the same but the variance has increased.

Since the corpus of collected data traces is large, we use one such trace to discuss some interesting observations due to space limitations. We call this a '*typical*' trace which was obtained on a single day when downloading 6MB files to the Moto G5 phone while moving around the IIT Kharagpur campus in a hybrid electric vehicle. The set-up is shown in Fig. 2(a). As Airtel has significantly fluctuating signal quality inside the campus, we chose it as our service provider. For this '*typical*' trace we had downloaded a 6Mb file to our phone several times. During each download session, the vehicle had moved over a different stretch of the trajectory inside the IIT Kharagpur campus. Of all the downloads, the twenty-nine stretches shown in Fig.1(a) could be identified as valid. The sorted throughput over all the twenty-nine stretches of Fig.1(a) is plotted in descending order in Fig.2(b). The x-axis represents the location indices corresponding to the sorted throughput. The mean and variance of the RSSI and also the horizontal and vertical handovers in each of these stretches is shown in Fig.2(b). It is observed that the throughput is affected more negatively by handovers than RSSI fluctuations. For example, the average and the variance of RSSI is nearly the same in location stretches-**29** and **15**. However, stretch **29** has a lower throughput due to handovers. Again, vertical handovers between network technologies are found to have a more negative impact on throughput than horizontal ones, as seen in location stretches **22** and **4**. The effect of handovers on the variation of *CONNECTED*, *TAIL* and *IDLE* state power is shown in Fig.2(c). It is seen that stretches that witness handovers have a higher variance in power consumption than no handover stretches. This is because, during handovers, there is a high amount of control information exchange which leads to the rise in *IDLE* power. Moreover, handovers are associated with lower throughput which increases the *CONNECTED* power. Fluctuating signal quality during handover increases the retransmissions and hence the TAIL state power variations.

Takeaway : 1 : *The wireless network condition is best quantified by throughput which depends significantly on phenomena such as handovers and not on received signal quality alone.*

(b)Video Streaming To understand how throughput fluctuation affects video streaming, we show the packet trace of a YouTube video of length 20 minutes captured when moving along the trajectory of Fig.1(a). Although we learn from the previous section that it is best to quantify the network condition using throughput, in this section we analyze the video download using RSSI only. This is because it is difficult to capture the actual network throughput of a phone while any other application (in this case YouTube) is running.

The captured trace and the RSSI is shown in Fig.1(b). It is seen that the application downloads video chunks even at low RSSIs. To understand this, we have divided the RSSIs on the secondary y-axis of Fig.1(b) into nine bins each of width five, starting from -81 dBm to -126 dBm. The time the UE spends in each of these bins, and the time it remains active is given in Fig.3. It is seen that the percentage of time the UE spends in the best RSSI bin, i.e. *bin*-1 (-85 to -81 dBm) is less than 1%. In comparison, the highest dwell time as well as the highest active time is recorded in *bin*-5 where the RSSI varies between -115 to -111 dBm. If we focus on the energy consumption in these RSSI bins, as given in Fig.4, it is seen that the highest energy is also consumed by the phone in RSSI *bin*-5. Another obvious effect of downloading at low signal strength is the low amount of data downloaded for a longer amount time; with reference to RSSI *bin*-5 in Fig.4.

The rate at which the YouTube playback buffer is filled depends on: a) the bandwidth available, and b) the quality of the video requested by the user. Once the buffer length exceeds a threshold, the download stops and restarts only when the buffer length goes below the threshold. So, if the buffer is

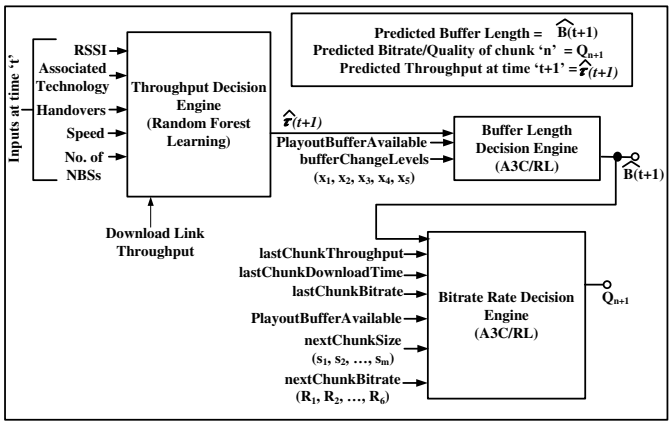


Fig. 5. Composite Representation of the EnDASH model; a cascaded model where the predicted throughput acts as an input network state to the Actor Critic RL based decision engine

full when signal quality improves, then the phone does not download any video packet - a possible explanation for no data download between 430-547 seconds in Fig. 1(b).

Takeaway : 2 : *The current protocol of video download attributes a higher weightage to the playback-buffer length than the user's instantaneous received signal strength or throughput, ensuing a significantly high energy consumption.*

The takeaways of the pilot study provide the design criteria for designing the EnDASH system, discussed next.

IV. EnDASH System

The proposed EnDASH system is an energy efficient client video player, working as a wrapper over DASH. It predicts the cellular network throughput over a finite future time window to take decisions on the opportune fetching of video chunks. EnDASH operates in a time-slotted fashion. So, the entire timeline is divided into discrete time-slots of length 'T'. EnDASH executes the following functions to achieve energy efficient video download: (i) At the beginning of each time slot it uses historical data on radio parameters to predict the cellular network link throughput of the current time slot using a Random Forest Learning (RFL) engine. (ii) It then uses the predicted throughput to predict the optimal playback-buffer length for the current slot that minimizes energy consumption. For this, EnDASH employs the state-of-the-art Actor-Critic (A3C) deep Reinforcement Learning (RL) algorithm. (iii) Before downloading each video chunk within a time slot, EnDASH runs another A3C based deep RL engine to predict the optimal download chunk bitrate. EnDASH is, thus, a cascaded system of three engines: (i) throughput prediction, (ii) buffer length decision, and (iii) bitrate selection as shown in Fig. 5. We next describe these modules.

A. The Throughput Prediction Engine

EnDASH employs a Random Forest Learning (RFL) algorithm for cellular network throughput prediction [9]. At the beginning of time slot 't', it uses the historical information of different radio related parameters in the previous 'x' seconds to predict the average throughput [1] that may be experienced by the user during the time slot (of length 'T'). We represent this as $\mathrm{P_xF_T}$. The input features of the throughput prediction engine are: (i) *RSSI*, (b) *technologies used* - LTE (4G), HSPA+ (3.75G), UMTS (3G), EDGE (2G), (c) *number of vertical and horizontal handovers*, (d) *speed*, (e) *number and technology of neighbouring Base Stations (BSs)*, (f) *download link throughput* - the download rate is measured at the UE in bytes per second. Readings on these features are obtained from the NetMonitor Lite app with a granularity of one second.

Each feature can be represented as a distribution, However, instead of feeding the entire time-series data for each metric, we adopt the summarization technique of [9]. So, we feed only a few key values that best summarize the data and its corresponding distribution. For each of the features we obtain the 25^{th}, 75^{th}, and 90^{th} percentile points, median and mean from its historical data and feed it to the RFL algorithm. We train and test this module using 39662 seconds of collected data on received throughput across five Indian cities.

B. The Buffer Length Decision Engine

The Buffer Length Decision engine also runs at the beginning of each time slot and the playback buffer length is predicted from the predicted network throughput. However, the relationship between the predicted throughput and the buffer length is not straightforward and is not analytically tractable. So, we employ an A3C RL based deep neural network to determine the optimal buffer length.

The components of the RL algorithm corresponding to the buffer decision engine are as follows:

(i) Environment E - video player.

(ii) Input state at timeslot 't' S_t = ($\hat{\tau}_t$, B_{av_t}, $\mathcal{X}$), where $\hat{\tau}_t$ is the average predicted cellular network throughput, B_{av_t} is the current playback buffer capacity available in seconds, $\mathcal{X}$ is the set of possible changes in buffer length. If the current buffer length is B_{t-1}, the next buffer length can be predicted to be $\hat{B}_t = B_t + x$ where $x \in \mathcal{X} := \{-2, -1, 0, +1, +2\}$. $x = 1$ implies an increase in buffer length by a single chunk, each chunk is of eight seconds.

(iii) Action A_t - Decisions on the increase or decrease of buffer length at timeslot t.

(iv) Reward - The reward function is defined as a linear weighted function of energy savings with respect to a baseline ABR algorithm and the QoE score:

$$\Xi_{\mathrm{bufflen}} = w_1 \cdot (|E_{\mathrm{EnDASH}_t} - E_{\mathrm{old}_t}|) + w_2 \cdot QoE \quad (1)$$

The first term in the reward function gives the energy savings with respect to a baseline ABR algorithm. In this work, we have chosen BOLA [7] as the baseline since its energy consumption is the lowest among existing algorithms (excluding EnDASH, §V). E_{old_t} and E_{EnDASH_t} represent the energy consumption of BOLA and of EnDASH at time 't', respectively. The energy consumed while using one particular

[1] Since video rate adaptation algorithms mostly use average throughput, hence for EnDASH, we have predicted the average throughput only [9].

algorithm is obtained as follows: the RRC states are first identified from the download packet capture of a video trace. Next, the dwell time in each RRC state is multiplied by the corresponding power consumption (obtained from the RRC state machine) to get the energy quantities. We calculate the energy savings in this manner because the ground truth on energy consumption cannot be obtained.

Second term in the reward function is the **QoE** metric [8]:

$$\text{QoE} = \sum_{i=1}^{N} q(R_i) - \mu \sum_{i=1}^{N} \delta_i - \sum_{i=1}^{N-1} |q(R_{i+1}) - q(R_i)| \quad (2)$$

The QoE metric is defined for a video with N chunks. R_i is the bitrate of chunk$_i$ and $q(R_i)$ maps the bitrate to a quantity which represents the quality perceived by the user. We have taken $q(R_i) = R_i$ [6]. δ_i is the rebuffering time involved in downloading chunk$_i$ at bitrate R_i. μ represents the degree of penalty associated with δ_i. We have taken $\mu = 4.3$ [6]. The last term represents the playback smoothness. The QoE decreases when there is abrupt variability in throughput between successive chunks. Thus, QoE increases with bitrate, and reduces with rebuffering time and throughput variability.

C. The Bitrate Decision Engine

The predicted playback buffer length is next used for selecting optimal chunk bitrates using a deep RL based algorithm. The components of the RL algorithm are:

(i) Environment *E* - video player.

(ii) Input state before downloading the chunk 'n',

S_n = ($\hat{\mathcal{B}}_t, \tau_c(n-1)$, d_{n-1}, l_{n-1}, B_n, ζ_n, r_n), where $\hat{\mathcal{B}}_t$-predicted playback buffer length for current slot 't', $\tau_c(n-1)$= throughput of last chunk, d_{n-1} = time taken to download last chunk, l_{n-1} = bitrate of last chunk, B_n = available playback buffer length, $\zeta_n \in (s_1, s_2, \cdots s_m)$ = Possible size of next video chunk (m available sizes), $r_{n+1} \in (R_1, R_2, \cdots, R_6)$ = Possible bitrate levels for next video chunk.

(iii) Action A_n - Optimal bitrate decision for the next chunk.

(iv) Reward- QoE score obtained from eqn. (2).

The buffer length and bitrate decision engines run at two different time scales. So, time slot indices for the variables related to the buffer length decision engine and the bitrate decision engine are denoted by 't' and 'n', respectively.

D. Emulation environment for RL training

In the training phase, the RL agent of the buffer and the bitrate decision engine of EnDASH should ideally be trained using real video downloads at actual video streaming clients. However, this demands that entire chunks be downloaded for each training data point. Further, downloads over cellular networks can be quite slow. Hence, to save training time, we train EnDASH and its competing ABR algorithms using an emulation environment that closely mimics a real video client application. The emulator maintains its own representation of a real client's playback buffer. At the beginning of a time slot, the emulator first predicts the average throughput and then the playback-buffer length for the slot. Within the slot, once a chunk is to be downloaded, the emulator first assigns a download time to the chunk based on its bitrate and the internal network throughput, derived from previous chunks. It then depletes the playback-buffer by the chunk's download time to emulate the playback-buffer drainage during an ongoing chunk download. It adds the playback duration of the chunk being currently downloaded to the playback-buffer. The emulator sleeps temporarily once the playback buffer is full. Rebuffering occurs if the playback-buffer is completely drained before the next chunk download. The emulator keeps track of the rebuffering event and rebuffering time. After each slot (chunk download), the emulator prepares the state S_t (S_n) for the RL module of the buffer decision (bitrate decision) engines.

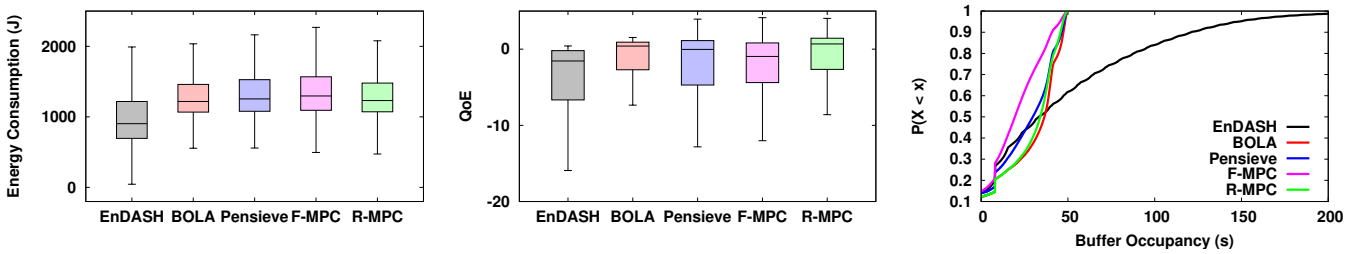
E. The RL Training algorithm

Both the buffer length and bitrate decision engines of EnDASH are trained using state-of-the-art actor-critic method, which involves training two neural networks [6]: an actor network and a critic network. A3C is a policy gradient RL algorithm which estimates the gradient of the total reward from the trajectories of the execution followed by a policy. The action is chosen based on the policy by the actor network while the critic network estimates the advantage of selecting the action by returning a value function. Both networks update their weights in each time step.

Each parameter of the state *S* of both buffer length and bitrate decision engines are passed on to their respective actor and critic network to enable learning of optimal buffer lengths and bitrates by the corresponding RL modules [6]. As in [6], we initiate multiple learning agents to accelerate the training. By default, there are 16 agents. Each agent is designed to experience a different set of input parameters, e.g., network traces. However, the learning agents continuously report their individual tuples of (state, action, reward) to a central model, which aggregates them to generate a single model.

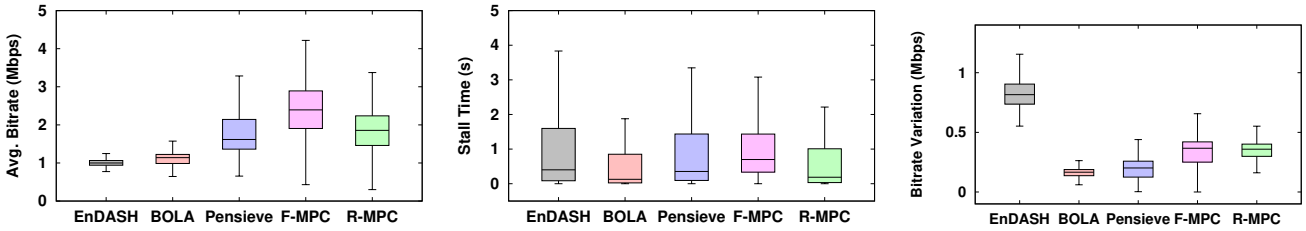
F. Implementation of the RL modules

The *bitrate selection engine* of EnDASH is adopted from the state-of-the-art Pensieve algorithm [6], which also uses A3C to learn optimal bitrates. We have used a pretrained model provided by Pensieve, albeit with different input states. The *buffer length decision engine* has been implemented in Tensorflow [33]. The engine passes five previous values of the buffer length to a 1D convolution layer (CNN) with 128 filters, each of size 4 with stride 1. The remaining inputs, i.e., predicted link throughput and current available buffer length are passed onto another 1D-CNN having the same shape. Results obtained from these layers are subsequently combined in a hidden layer having 128 neurons and uses the softmax function. The same inputs are also used by the critic network, which has the same neural network, but whose final output is a linear neuron. During the training of the algorithm, we set the discount factor to 0.9, i.e., current actions are allowed to be influenced by 100 future steps. The learning rates for the actor and the critic networks are 0.0001 and 0.001, respectively. The entropy factor β has been set to gradually decrease from 1 to 0.1 over iterations.



(a) Energy Consumption (b) QoE score (2) (c) CDF of buffer length

Fig. 6. Performance comparison of EnDASH with baseline ABR streaming algorithms, BOLA [7], Pensieve [6], Fast MPC [8], Robust MPC [8]



(a) Avg. Bitrates in Mbps (b) Stall Time per segment (in seconds) (c) Bitrate Variation

Fig. 7. Comparison of different components of QoE score (average bitrate, stall time, smoothness) of EnDASH with baseline ABR streaming algorithms, BOLA [7], Pensieve [6], Fast MPC [8], Robust MPC [8]

V. Evaluation

In this section we evaluate EnDASH and compare its performance with other popular ABR algorithms, in terms of QoE and energy consumption. We first discuss the methodology that we have adopted.

A. Methodology

To train and test the throughput engine of EnDASH, we have used a corpus of throughput and power consumption traces collected under user mobility, for both file download and video streaming workloads. These include data collected using Moto G5 and Micromax Canvas Infinity phones over Airtel, Reliance JIO, and Vodafone networks, across five cities in India. The entire corpus is of 39662 seconds of data. The real-life throughput traces are formatted to be compatible with the Mahimahi [34] network emulation tool. We have divided the collected data set into 70-30 ratio for training and testing. To train the RL algorithms of the buffer length and bitrate decision engine, we have used another dataset of 57 DASH-ified videos having a total duration of 45 hours. The baseline ABR streaming algorithms that we have used are *BOLA* [7], *Pensieve* [6], *Fast MPC* [8], and *Robust MPC* [8].

B. EnDASH versus Baseline ABR algorithms

Fig. 6 shows the energy consumption, QoE score, and buffer length variation of EnDASH and other baseline ABR algorithms. For evaluating these algorithms, the length of each time slot is set to $T = 30$ seconds, and the length of the historical window is also set to $x = 30$ seconds, i.e., $\mathrm{P}_{30}\mathrm{F}_{30}$. In existing literature, the Pensieve [6] algorithm has been reported to generate optimal chunk bitrates and video quality. Fig. 6(a) shows that EnDASH outperforms Pensieve in terms of energy consumption. However, this energy savings comes at the cost of sacrificing the QoE with respect to Pensieve as seen in Fig. 6(b). Moreover, while the average QoE of EnDASH is comparable with the other algorithms, the inter-quartile range of QoE is significantly high, implying that the corresponding variability is high.

C. QoE Performance Analysis

To understand the QoE performance in detail, we plot the individual components of the QoE metric in Fig.7. We observe that the mean of the average bitrate (Fig. 7(a)) of EnDASH is smaller and its stall time (Fig. 7(b)) is comparable with other algorithms. However, the mean bitrate variation (Fig. 7(c)) is much higher. Simultaneously, stall time displays a high variability. The reason for the reduced average bitrate, higher stall time variability, and higher mean of bitrate variation can be attributed to the tuning of the buffer length to the average throughput instead of the instantaneous throughput. For example, if the average throughput predicted is low, the system is forced to download a video chunk at a low bitrate for the entire timeslot, even though the throughput at multiple instances within the timeslot may be high, resulting in lower bitrates. Although the tuning to instantaneous throughput may improve bitrates, it will be associated with higher overhead. Hence, we focus on tuning to average throughput only. Further, the reduced inter-quartile range of the average bitrate is due to the aggressive fetching of video chunks.

D. Energy Performance Analysis

EnDASH consumes much less energy because the video chunk download, and hence the playback buffer length, is tuned to the average predicted cellular network throughput. As a result, during high throughput conditions, the playback

buffer length will increase, thereby facilitating the download of a higher number of chunks in a slot. This consecutive fetching of chunks reduces the tail energy which eventually manifests in the reduction of overall energy consumption as seen in Fig. 6(a). The resulting trade-off is reflected in the increase in buffer length in comparison with the buffer length of competing ABR algorithms. Fig. 6(c) shows the CDF of playback buffer length of different algorithms. It is seen that while the maximum buffer length of existing algorithms is 50 seconds that of EnDASH can go up to 200 seconds; but this is a rare instance. In nearly 60% of the time the buffer length remains below 50 seconds.

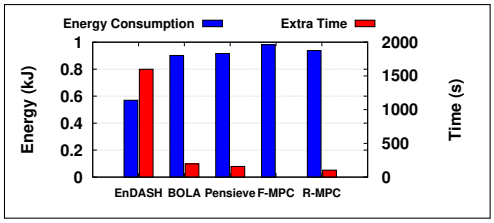


Fig. 8. Energy Consumption and Extra Playtime obtained w.r.t. Fast MPC, which has the highest energy consumption

E. Gain from Energy Savings

In this section, we discuss the gain in video playback time achieved by using EnDASH. FastMPC has the highest energy consumption among all algorithms. Fig.8 shows the gain in video playback time achieved by the algorithms with respect to FastMPC while streaming a 2200 second video. It shows that using the energy saved by streaming the video using EnDASH, one can gain an additional 1403 seconds and 1440 seconds of video playback time in comparison with BOLA and Pensieve, respectively. Thus, one may infer that EnDASH can be used as a potential ABR streaming algorithm for increasing battery backup in smartphones.

F. Feature importance study on throughput prediction engine

The primary objective of the throughput prediction engine is to account for the impact of cellular network technology change, i.e., the switching between 2G, 3G, 4G, etc., on network throughput. Fig.9 shows the feature importance of different input parameters when predicting throughput. We observe that vertical handovers and associated technology (Network Type) have the highest weightage among all parameters, 0.32 and 0.21, respectively. The importance of such features in throughput prediction points out to the absolute necessity of considering the existence of legacy systems when designing algorithms for 4G networks.

G. Importance of Associated Technology

To understand the impact of associated technology and vertical handovers on the throughput prediction error, we have evaluated the MAPE [2] score over different regions. For this,

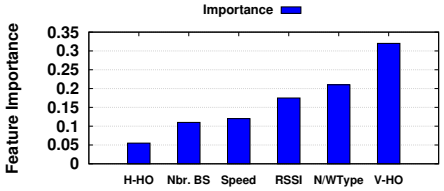


Fig. 9. Feature importance of the input parameters; signal strength, associated technology, and handovers (HOs)) between technology are the three features having the highest contribution in deciding throughput

we have divided the data traces collected in Kharagpur into three categories - (a) crowded Market Place, (b) a residential area with extensive 4G coverage (Residential Area 1), and (c) a residential area with limited 4G coverage (Residential Area 2). The overall MAPE score and the MAPE score for the three different scenarios, each with different combinations of associated network technology and vertical handover, is shown in Fig.10(a). It is observed that the lowest MAPE score is reported when the throughput prediction considers the associated technologies and vertical handovers, especially when 4G coverage is limited. Fig.10(b) shows how the inclusion of both associated technology and vertical handover as features in the throughput prediction improves the EnDASH performance, in terms of energy consumption as well as QoE.

The impact of associated technology can be further understood if we look into the actual throughput traces and its prediction, which Fig.10(c) shows. We have generated Fig.10(c) with $P_{30}F_{30}$. It is observed that if the associated technology and the vertical handover is not considered then there is significant mismatch in low throughput regions where there is a tendency of over estimation of the throughput.

VI. Conclusion

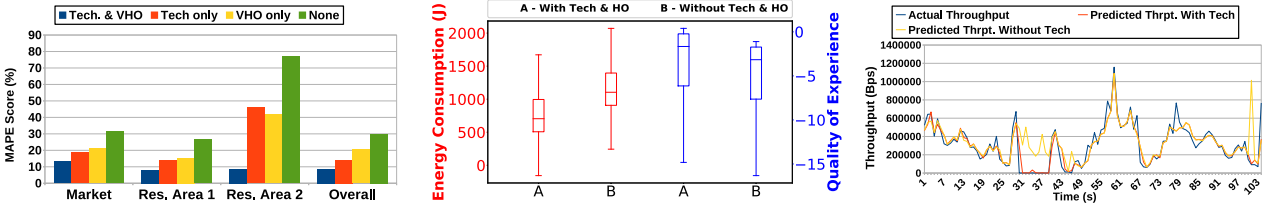
In this paper, we propose EnDASH – an energy aware ABR video streaming algorithm, which minimizes energy consumption while not compromising on QoE of users under mobility. It exploits the high throughput regions in the user's trajectory for aggressive fetching of video chunks thereby reducing energy consumption even in regions with limited 4G coverage and significant presence of legacy networks. To achieve this, it intelligently tunes the playback buffer length with the average predicted throughput and then resorts to optimal bitrate selection for video chunks. As a result, the buffer length increases sometimes; although, the consequent cost escalation is negligible. This is because the cost is incurred due to (a) extra memory usage which is cheap and can be ignored, and (b) aggressive fetching which may lead to some wastage, but such wastage is rare and minimal, therefore, hardly having any cost impact.

EnDASH predicts the cellular network throughput using Random Forest Learning. It tunes the buffer length and selects the optimal chunk bitrates using Reinforcement learning. EnDASH is able to improve the maximum energy consumption by about 30.4% in comparison to the popular Pensieve

[2] MAPE score, which quantifies error in throughput prediction, is:

$$\text{MAPE score} = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{\tau_i - \hat{\tau}_i}{\tau_i}\right| \times 100\% \quad (3)$$

where τ_i and $\hat{\tau}_i$ respectively denote actual and predicted throughput in the i^{th} timeslot.



(a) MAPE score measuring error of throughput prediction in different regions for various combinations of considering associated technology and vertical handover (HO); for $P_{30}F_{30}$

(b) Impact of considering associated technology and vertical handovers (HOs) on performance metrics of EnDASH; for $P_{30}F_{30}$

(c) Predicted vs Actual throughput using the RF algorithm when associated technology and vertical handover (HO) is considered.

Fig. 10. Effect of Associated Technology and Vertical Handovers (HOs) on Throughput Prediction and EnDASH performance for $P_{30}F_{30}$

algorithm although with reduction in QoE. Since EnDASH is a tunable algorithm it can be designed to adapt to a specific requirement, such as energy or QoE – this would be our immediate future work. Additionally, our future work will also involve real-life implementation of EnDASH and investigating the corresponding improvement in energy efficiency.

REFERENCES

[1] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2017-2022," Tech. Rep., 2019.

[2] "How Much Does Mobile Data Cost Around the World?" https://howmuch.net/articles/the-price-of-mobile-internet-worldwide-2019, 2019.

[3] "Still buffering: India low on 4G speed," https://economictimes.indiatimes.com/tech/internet/still-buffering-india-low-on-4g-speed/articleshow/65001463.cms?from=mdr, Jul. 2018.

[4] X. Li, M. Dong, Z. Ma, and F. C. Fernandes, "GreenTube: Power Optimization for Mobile Videostreaming via Dynamic Cache Management," in *Multimedia*. ACM, 2012, p. 279–288.

[5] T. Stockhammer, "Dynamic Adaptive Streaming over HTTP –: Standards and Design Principles," in *ACM MMSys*, 2011, pp. 133–144.

[6] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with Pensieve," in *ACM SIGCOMM*, 2017, pp. 197–210.

[7] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, "BOLA: Near-optimal bitrate adaptation for online videos," in *IEEE INFOCOM*, 2016, pp. 1–9.

[8] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A Control-Theoretic Approach for Dynamic Adaptive Video Streaming over HTTP," in *ACM SIGCOMM*, 2015, pp. 325–338.

[9] D. Raca, A. H. Zahran, C. J. Sreenan, R. K. Sinha, E. Halepovic, R. Jana, V. Gopalakrishnan, B. Bathula, and M. Varvello, "Empowering Video Players in Cellular: Throughput Prediction from Radio Network Measurements," in *ACM MMSys*, 2019, pp. 201–212.

[10] Z. Akhtar, Y. S. Nam, R. Govindan, S. Rao, J. Chen, E. Katz-Bassett, B. Ribeiro, J. Zhan, and H. Zhang, "Oboe: Auto-tuning Video ABR Algorithms to Network Conditions," in *ACM SIGCOMM*, 2018, pp. 44–58.

[11] S. Sengupta, N. Ganguly, S. Chakraborty, and P. De, "HotDASH: Hotspot Aware Adaptive Video Streaming Using Deep Reinforcement Learning," in *IEEE ICNP*, 2018, pp. 165–175.

[12] A. Schulman, V. Navda, R. Ramjee, N. Spring, P. Deshpande, C. Grunewald, K. Jain, and V. N. Padmanabhan, "Bartendr: A Practical Approach to Energy-aware Cellular Data Scheduling," in *ACM MobiCom*, 2010, pp. 85–96.

[13] 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 *ACM Mobisys*, 2012, pp. 225–238.

[14] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A Buffer-based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service," in *ACM SIGCOMM*, 2014, pp. 187–198.

[15] J. Jiang, V. Sekar, and H. Zhang, "Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive," *IEEE/ACM Trans. Netw.*, vol. 22, no. 1, p. 326–340, Feb. 2014.

[16] T. Xu and L. Ma, "Predictive prefetching for MPEG DASH over LTE networks," in *IEEE ICIP*, 2015, pp. 3432–3436.

[17] S. K. Mehr and D. Medhi, ""qoe performance for dash videos in a smart cache environment"," in *IFIP/IEEE IM Symp.*, April 2019, pp. 388–394.

[18] A. Bentaleb, C. Timmerer, A. C. Begen, and R. Zimmermann, "Bandwidth Prediction in Low-latency Chunked Streaming," in *ACM NOSSDAV*, 2019, pp. 7–13.

[19] D. Raca, J. J. Quinlan, A. H. Zahran, and C. J. Sreenan, "Beyond Throughput: A 4G LTE Dataset with Channel and Context Metrics," in *MMSys*, 2018, pp. 460–465.

[20] C. Yue, R. Jin, K. Suh, Y. Qin, B. Wang, and W. Wei, "Linkforecast: Cellular link bandwidth prediction in lte networks," *IEEE Trans. Mobile Computing*, vol. 17, no. 7, pp. 1582–1594, 2018.

[21] D. Raca, A. H. Zahran, C. J. Sreenan, R. K. Sinha, E. Halepovic, R. Jana, and V. Gopalakrishnan, "Back to the Future: Throughput Prediction For Cellular Networks Using Radio KPIs," in *ACM HotWireless*, 2017, pp. 37–41.

[22] D. Raca, A. H. Zahran, C. J. Sreenan, R. K. Sinha, E. Halepovic, R. Jana, V. Gopalakrishnan, B. Bathula, and M. Varvello, "Incorporating Prediction into Adaptive Streaming Algorithms: A QoE Perspective," in *ACM NOSSDAV*, 2018, pp. 49–54.

[23] A. Samba, Y. Busnel, A. Blanc, P. Dooze, and G. Simon, "Instantaneous throughput prediction in cellular networks: Which information is needed?" in *IFIP/IEEE Symposium on INSM*, 2017, pp. 624–627.

[24] A. Ghasemi, "Predictive Modeling of LTE User Throughput Via Crowd-Sourced Mobile Spectrum Data," in *IEEE Int. Symp. DySPAN*, 2018, pp. 1–5.

[25] Y. Yang and G. Cao, "Prefetch-Based Energy Optimization on Smartphones," *IEEE Trans. Wireless Commun.*, vol. 17, no. 1, pp. 693–706, 2018.

[26] 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," *Wirel. Netw.*, vol. 24, no. 8, pp. 3083–3098, 2018.

[27] J. Zhang, G. Fang, M. Guo, and C. Peng, "How video streaming consumes power in 4G LTE networks," in *IEEE WoWMoM*, 2016, pp. 1–3.

[28] 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 *ACM Movid*, 2016, pp. 3:1–3:6.

[29] M. J. Khokhar, T. Ehlinger, and C. Barakat, "From Network Traffic Measurements to QoE for Internet Video," in *IFIP Networking*, May 2019, pp. 1–9.

[30] H. Deng, C. Peng, A. Fida, J. Meng, and Y. C. Hu, "Mobility Support in Cellular Networks: A Measurement Study on Its Configurations and Implications," in *ACM IMC*, 2018, pp. 147–160.

[31] "High Voltage Power Monitor," "https://www.msoon.com/high-voltage-power-monitor".

[32] Y. Geng, W. Hu, Y. Yang, W. Gao, and G. Cao, "Energy-Efficient Computation Offloading in Cellular Networks," in *IEEE ICNP*, 2015, pp. 145–155.

[33] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: A System for Large-scale Machine Learning," in *USENIX OSDI*, 2016, pp. 265–283.

[34] R. Netravali, A. Sivaraman, S. Das, A. Goyal, K. Winstein, J. Mickens, and H. Balakrishnan, "Mahimahi: Accurate Record-and-replay for HTTP," in *USENIX Conf. on UATC*, 2015, pp. 417–429.