CORA Cork Open Research Archive
Cartlann Taighde Oscailte Chorcaí

Title	Incorporating prediction into adaptive streaming algorithms: a QoE perspective
Authors	Raca, Darijo;Zahran, Ahmed H.;Sreenan, Cormac J.;Sinha, Rakesh K.;Halepovic, Emir;Jana, Rittwik;Gopalakrishnan, Vijay;Bathula, Balagangadhar;Varvello, Matteo
Publication date	2018-06
Original Citation	Raca, D., Zahran, A. H., Sreenan, C. J., Sinha, R. K., Halepovic, E., Jana, R., Gopalakrishnan, V., Bathula, B. and Varvello, M. (2018) 'Incorporating Prediction into Adaptive Streaming Algorithms: A QoE Perspective', NOSSDAV '18 Proceedings of the 28th ACM SIGMM Workshop on Network and Operating Systems Support for Digital Audio and Video, Amsterdam, Netherlands, 12-15 June, 3210457: ACM, 49-54. doi: 10.1145/3210445.3210457
Type of publication	Conference item
Link to publisher's version	https://dl.acm.org/citation.cfm?doid=3210445.3210457 - 10.1145/3210445.3210457
Rights	© 2018 Association for Computing Machinery. This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record was published in NOSSDAV '18 Proceedings of the 28th ACM SIGMM Workshop on Network and Operating Systems Support for Digital Audio and Video, http://dx.doi.org/10.1145/3210445.3210457
Download date	2026-11-02 06:17:02
Item downloaded from	https://hdl.handle.net/10468/6409

UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Incorporating Prediction into Adaptive Streaming Algorithms: A QoE Perspective

Darijo Raca, Ahmed H. Zahran, Cormac J. Sreenan
University College Cork
{d.raca,a.zahran,cjs}@cs.ucc.ie

Rakesh K. Sinha, Emir Halepovic, Rittwik Jana, Vijay Gopalakrishnan, Balagangadhar Bathula, Matteo Varvello
AT&T Labs – Research
{sinha,emir,rjana,gvijay,varvello,balab}@research.att.com

ABSTRACT

Streaming over the wireless channel is challenging due to rapid fluctuations in available throughput. Encouraged by recent advances in cellular throughput prediction based on radio link metrics, we examine the impact on Quality of Experience (QoE) when using prediction within existing algorithms based on the DASH standard. By design, DASH algorithms estimate available throughput at the application level from chunk rates and then apply some averaging function. We investigate alternatives for modifying these algorithms, by providing the algorithms direct predictions in place of estimates or feeding predictions in place of measurement samples. In addition, we explore different prediction horizons going from one to three chunk durations. Furthermore, we induce different levels of error to ideal prediction values to analyse deterioration in user QoE as a function of average error.

We find that by applying accurate prediction to three algorithms, user QoE can improve up to 55% depending on the algorithm in use. Furthermore having longer horizon positively affects QoE metrics. Accurate predictions have the most significant impact on stall performance by completely eliminating them. Prediction also improves switching behaviour significantly and longer prediction horizons enable a client to promptly reduce quality and avoid stalls when the throughput drops for a relatively long time that can deplete the buffer. For all algorithms, a 3-chunk horizon strikes the best balance between different QoE metrics and, as a result, achieving highest user QoE. While error-induced predictions significantly lower user QoE in certain situations, on average, they provide 15% improvement over DASH algorithms without any prediction.

CCS CONCEPTS

• **Information systems** → **Multimedia streaming**; • **Networks** → Public Internet; **Wireless access networks**; **Network measurement**;

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
NOSSDAV'18, June 12–15, 2018, Amsterdam, Netherlands
© 2018 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

KEYWORDS

HAS, 4G, LTE, Mobility, throughput prediction, DASH, adaptive video streaming

ACM Reference Format:
Darijo Raca, Ahmed H. Zahran, Cormac J. Sreenan and Rakesh K. Sinha, Emir Halepovic, Rittwik Jana, Vijay Gopalakrishnan, Balagangadhar Bathula, Matteo Varvello. 2018. Incorporating Prediction into Adaptive Streaming Algorithms: A QoE Perspective. In *NOSSDAV'18: 28th ACM SIGMM Workshop on Network and Operating Systems Support for Digital Audio and Video, June 12–15, 2018, Amsterdam, Netherlands.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION & BACKGROUND

Traffic analytics indicate a continuous increase of video consumption in mobile networks. By 2020, 75% of all mobile traffic will have a video element[1]. Streaming video content over mobile is challenged by frequent throughput fluctuations caused by rapid changes in channel conditions and network load. Hence, video streaming users are more prone to experiencing problems, such as poor quality and stalls.

Most popular video streaming providers (Youtube, Netflix, Amazon Prime, Hulu) use Dynamic Adaptive Streaming over HTTP (DASH) [12] to deliver video content. In DASH systems, or more generally HTTP adaptive streaming (HAS), video content is split into multiple chunks, with chunk duration varying from 2 to 20 seconds. Every chunk is encoded in different bitrates going from low to high quality. This splitting allows the video player to dynamically adapt the video quality to the network conditions, with the goal to maximize the user experience.

In the literature [4, 11], many video quality adaptation algorithms exist for HAS. These algorithms usually consider combinations of network and/or application states in their quality selection decision. By its nature, quality adaptation algorithms have a modular design. The *bandwidth estimation module* captures the network state, while the *application monitoring module* captures the video player state by monitoring playback buffer and streamed video quality. Finally, the *bitrate adaptation module* combines information from the previous modules to decide the quality of the chunks to be requested. The reader is referred to [4, 11] and the references therein for a detailed view of the adaptation strategies.

[1] https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.pdf

The main goal of HAS algorithms is to maximize overall user experience (QoE). To achieve this goal, algorithms focus on maximizing video quality while minimizing quality switches and rebuffering events. Finding a right balance between these conflicting requirements is challenging in cellular networks. HAS algorithms use different throughput estimation techniques (arithmetic, harmonic, exponential moving average, median) as a step towards quality selection. This estimate is usually integrated with advanced decision mechanisms to select the quality of the next chunk. Even with these sophisticated mechanisms in place, inaccurate bandwidth estimations can lead to incorrect incorrect quality selections, potentially leading to video stalls, poor video quality, or frequent quality switches.

In Section 3, we show, that state of the art bandwidth estimators exhibit noticeable errors due to inherent throughput fluctuations in mobile networks. We further show that their performance is significantly lower than what can be attained by leveraging radio KPIs and network-related information as done in [10, 14]. Additionally, [16] and [6] propose prediction-based HAS algorithms and demonstrate that they can improve the streaming performance in comparison to typical HAS algorithms. However, neither of these (or other) studies considered fusing prediction to existing adaptation algorithms. We believe this step is natural due to modular design of HAS algorithms. This modularity enables integrating throughput prediction in adaptation algorithms to reap the benefits of novel accurate throughput estimation techniques.

In this paper, we explore integrating throughput prediction in three HAS adaptation algorithms and quantify its impact on overall user QoE. More specifically, this study focuses on addressing the following research questions:

(1) *How the predicted throughput should be integrated in the adaptation algorithm? Should it be used as a throughput sample or replace the estimate?*
(2) *How the predicted throughput horizon may influence QoE?*
(3) *How inaccurate prediction may impact the streaming performance and user QoE?*

We investigate these questions in an experimental testbed that uses real 4G cellular traces and real video content. In our experiments, we use two video QoE models developed for HAS [5, 13] to evaluate QoE performance. Our results indicate that assisting adaptation algorithms with accurate predictions can improve user QoE by up to 55% for all tested algorithms. We additionally found that extending prediction horizon helps in eliminating stall events and improving overall switching behavior. Furthermore, inaccurate throughput prediction reduces the QoE benefit but users would still enjoy on average a 15% QoE improvement in the presence of 30% error in the predicted throughput.

2 EXPERIMENTAL TESTBED & METHODOLOGY

In this section, we describe our experimental testbed (Section 2.1), followed by a summary of HAS algorithms under test. Trace classification and selection is described in Section 2.3. For performance evaluation, we examine different QoE metrics and models, which we describe in Section 2.4. Finally, Section 2.5 describes few ways to feed prediction to the HAS player.

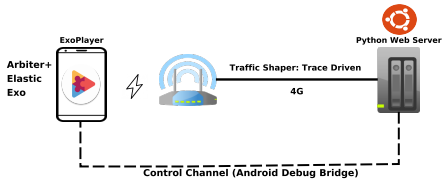


Figure 1: Testbed Architecture

2.1 Experimental Testbed

Figure 1 depicts our testbed architecture which consists of a mobile device (Nexus 6 running Android OS version 7.1.1), a wireless access point (WAP), and a server (PC running Ubuntu 16.04 equipped with 16GB of RAM and Intel i7 CPU). The mobile device streams video content from the server through WAP. The server also acts as a *traffic shaper* by inserting bandwidth profiles from 4G traces between itself and the WAP. This is achieved using Linux *traffic control* (tc[2]). Finally, *Android Debug Bridge* (ADB) is used to feed throughput prediction values to the mobile device.

The mobile device runs a full-fledged video player we have built using ExoPlayer,[3] a media player platform for Android developed by Google. ExoPlayer supports the DASH standard via a stand-alone library, and also provides a (default) adaptive streaming algorithm. In addition to ExoPlayer's default adaptation algorithm, we have also implemented Elastic [1] and Arbiter+ [15].

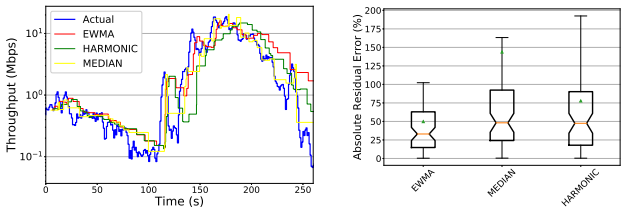
For video sources, we use publicly available dataset [8]. Videos are split in 4-second chunks and encoded with ten representative rates (kbps): 231, 369, 553, 744, 1044, 1748, 2349, 3006, 3856, 4310. For bandwidth profiles, we use dataset with over 150 4G traces collected from an operational cellular network [9]. The collected bandwidth profiles are a mixture of different mobility patterns: static, pedestrian, bus, car and train.

2.2 Video Streaming Algorithms

Many HAS algorithms can be found in the literature [11]. For our evaluation, we select three algorithms: Arbiter+, Elastic and Exo (ExoPlayer's default adaptive streaming algorithm). Selected algorithms use information from both bandwidth estimators, as well as from buffer occupancy when deciding on the rate of the next chunk.

As a bandwidth estimator, Arbiter+ uses the exponential moving average (EWMA) of the last ten chunk download rates. Alongside EWMA Arbiter+ employs two additional rate scale factors, to track variation in throughput samples and buffer occupancy. Unlike Arbiter+, Elastic uses the harmonic average of the last five chunk download rates. However, in addition to the harmonic estimator, Elastic tracks buffer levels and uses a control-theory approach to decide chunk quality. By its nature, harmonic mean is a conservative estimate. Finally, Exo calculates the median throughput from the last N chunk download rates, where N is derived such that the square root of that last N chunk sizes is smaller than $2000 bytes^{\frac{1}{2}}$. Intuitively, the median is (more) resistant to outliers and also represents a slightly less conservative estimate compared to the harmonic mean. Also, Exo uses safety factor by taking only 75% of bandwidth estimate. In addition, it tracks buffer occupancy as well, triggering

[2]https://wiki.debian.org/TrafficControl
[3]https://github.com/google/ExoPlayer



(a) Time-series of instantaneous throughput **(b) Boxplot of residual error for different bandwidth estimators**

Figure 2: Performance analysis of traditional bandwidth estimators

chunk replacement function (a function that re-downloads same chunks with higher quality) if buffer level gets close to saturation.

For buffer length, we use recommended values for each algorithm (60 seconds for Arbiter+ and 30 seconds for Elastic and Exo). Initial delay is set to two chunks. After stall event, playing is resumed after one chunk is downloaded.

To further confirm our above assumptions, Fig 2a shows time-series of instantaneous throughput (blue line) for a highly-variable trace (see Section 2.3). In addition to actual throughput, we additionally plot throughput estimates from the bandwidth estimator used by each algorithm, *i.e.,* Arbiter+, Exo, and Elastic (red, green, and yellow line, respectively).

EWMA shows the highest average throughput among all bandwidth estimators. Compared to the actual throughput, EWMA tends to overestimate throughput (58% of the times) while harmonic and median show neutral bias, and overall lower values compared to EWMA. Trend-wise, EWMA follows actual throughput closely with resistance to small variations due to smoothing effect. Instead, harmonic reacts to throughput changes slower than both EWMA and median.

To further showcase the limitation of current throughput estimation mechanisms, we analyze the absolute value of residual error between throughput estimates and the actual download rate of a chunk. Figure 2b shows boxplots of the residual error for EWMA, median, and harmonic, measured while experimenting with a randomly selected 4G trace. Figure 2b shows that EWMA achieves, overall, the lowest residual error while median and harmonic perform comparable. Nevertheless, EWMA's average residual error is still 50% (green dot) which is way higher than what obtained in recent works [10, 14], e.g., 10% in [14]. Such low residual error is achieved by *predicting* future throughput, e.g., by using machine learning models coupled with network and device data, rather than using historical throughput information. Hence, we are interested in exploiting the impact of novel, accurate throughput prediction methods on the streaming performance.

2.3 Trace Classification

Based on Mangla et. al [6] result, we consider the standard deviation of throughput within a trace as discriminant of (potentially) *good* or *bad* input traces. The rationale is that we hypothesize that prediction will be mostly effective in presence of traces with high throughput standard deviation (in particular when the standard deviation is higher than highest representation rate resulting in bandwidth fluctuations spanning across all available rates). From dataset, we select highly variable traces with a standard deviation in the range [4.2, 6.3] Mbps, and low variable traces with a standard deviation in the range of [0.6, 1.2] Mbps. We further filter out traces with very high average throughput (6 Mbps), *i.e.,* average throughput larger than the highest video quality (4.3 Mbps). The rationale here is to avoid testing scenarios where all algorithms converge to the highest quality level regardless of throughput variation (as fluctuations will get averaged out by throughput estimator). Finally, for every trace, we repeat experiment ten times to get statistically significant results. Out of 130 traces we ended up with 26 traces satisfying the latter constraint. As expected, the majority of high-variable traces are collected in the highly mobile environment (car), while low-variable traces where collected while devices have been static or moving with low velocity (pedestrian).

2.4 Video QoE Models

To evaluate the performance of HAS algorithms, we analyze standardized QoE metrics, such as average representation rate, switching behavior (e.g., instability), stall frequency and duration. However, in order to compare algorithms performance these metrics cannot be studied independently. For example, an algorithm achieving highest average throughput but frequent stalls is inferior to a more cautious algorithm with no stalls, as it provides better end-user experience.

We thus resort to using two video QoE models developed for HAS [5, 13]. These models blend individual QoE metrics to compute a score aiming at representing user QoE. Both models are derived from subjective testing of users grading video clips with various induced impairments. The first model (*Yao QoE*) was derived from data collected in a lab environment [5] and it is limited to five minutes. Such limitation does not apply to the second model which relies on data crowdsourced from users watching videos posted on a website [13] (*Clay QoE*). The score derived from these QoE models can be summed up by the following equation:

$$QoE_{score} = \nu \times QoE_{max} - (\kappa_{TQ} \times I_{TQ} + \kappa_{VQ} \times I_{VQ}) \quad (1)$$

$$+\Upsilon(I_{TQ}, I_{VQ}) \quad (2)$$

Where I_{TQ}, and I_{VQ}, represent temporal and visual quality impairment factors, respectively. Similarly, κ_{TQ} and κ_{VQ} represent their respective weights. Temporal quality impairments refer to degradation due to initial delay and stall events (stall number and stall duration). Analogously, visual quality impairments take into account average rate and switching behavior. QoE_{max} indicates the maximum value (score) of QoE or growth factor depending on QoE model. Similar to impairment weights, ν is weight for the QoE_{max} score. Finally, $\Upsilon(I_{TQ}, I_{VQ})$ represents a cross-effect function of impairment factors occurring simultaneously. When multiple impairments happen, their cumulative subjective effect is not simply the sum of each impairment separately [13]. Function Υ compensates for this effect.

The Yao QoE score starts at 100, and it is reduced by its impairments, compensated by Υ function of the stall, switching and initial delay impairments. The Clay QoE initial score is based on average rate, deducted by impairments capturing stall and switching deterioration. Clay QoE doesn't include cross-effect compensation function Υ.

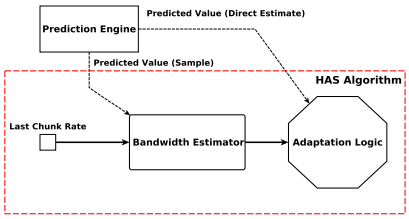


Figure 3: Two approaches in feeding prediction to HAS algorithm

We analyse both QoE models across different traces allowing us to make the following observations: both models perceive stall impairments similarly with high correlation (0.9); models calculate switching impairment differently (Clay uses standard deviation between rates, while Yao relies on difference in Video Quality Metric [7] (VQM) between chunks); unlike Clay, Yao uses cross-effect compensation function which limits negative impact of multiple impairments.

Because of the aforementioned observations, we decided to use an average of two models to represent an overall QoE score. For the type of average, we select geometric mean as the model scores are on a different scale. Table 1 summarize QoE metrics and their notation.

Table 1: QoE Metrics Notation

Metric	Summary
r_{avg}	Average representation rate
i_{avg}	Average instability as defined in [2]
s_{num}	Average number of stalls
s_{dur}	Average stall duration
QoE	Geometric mean of Clay and Yao QoE

2.5 HAS Algorithm Modification

The predicted throughput can be integrated into the adaptation logic in two different ways. First, the predicted throughput may replace the entire throughput *estimation* in the algorithm (*E-type*). Alternatively, the predicted throughput may be used to replace the estimated throughput *samples* (*S-type*). Fig. 3 shows both *E-type* and *S-type* approaches. The prediction engine is responsible for obtaining throughput prediction. This prediction can come from the network [3] or from the device itself (by leveraging radio KPIs and machine learning techniques [10, 14]). For each of these approaches, we further investigate the impact of the throughput horizon on the performance.

3 EVALUATION

3.1 Accurate Predictions

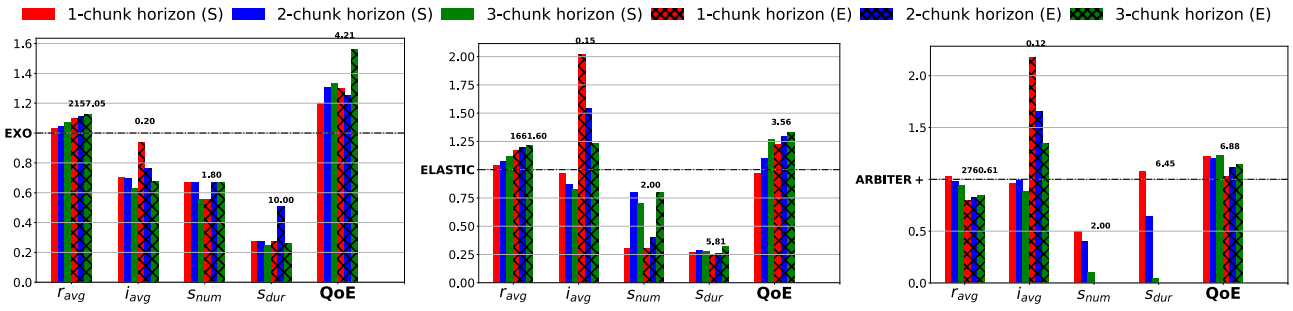
In the following, we explore the impact of integrating error-free throughput predictions with different horizons on video QoE. We consider three prediction horizons as multiples of chunk duration, *i.e.,* 4, 8, and 12 seconds. In the following plots, prediction horizons are associated with *red*, *blue*, and *green* color, respectively. As a notation mark, we use *S* to denote usage of prediction as a sample (*S-type*) and *E* for prediction as an estimate (*E-type*). We further visually separate sample and estimate results using a diagonal pattern. Finally, note that each metric is normalized with respect to the original algorithm performance (referred to as *no-prediction* setup), whose actual metric values are offered in the figure.

3.1.1 Traces with High Variability. We start by focusing on results obtained when considering *traces with high variability*, *i.e.,* traces characterized by a high value of throughput standard deviation. Figure 4 shows that integrating prediction noticeably improves the QoE metrics of all adaptation algorithms. Specifically, prediction enables all algorithms to reduce/eliminate stalls. Additionally, prediction enables the algorithms to reduce the switching instability. In particular, average instability can be reduced by 12%-37%. Furthermore, improving the accuracy of bandwidth estimation enables the algorithm to enhance their selected chunk quality. For example, integrating prediction fixes throughput underestimation with Elastic and Exo leading to a higher chunk quality. On the other hand, integrating the prediction with Arbiter results in a lower average representation rate. All these improvements add up thus boosting the overall user QoE by 23%-55%.

Increasing horizon duration has a positive impact on stall performance and switching behavior. Extending the horizon results in averaging over a longer period and thus reducing variability between subsequent prediction values. This leads to improved switching performance. Longer horizon enables a client to promptly reduce quality and avoid stalls when the throughput drops for a relatively long time that can deplete the buffer. For all algorithms, 3-chunk horizon shows the highest gain.

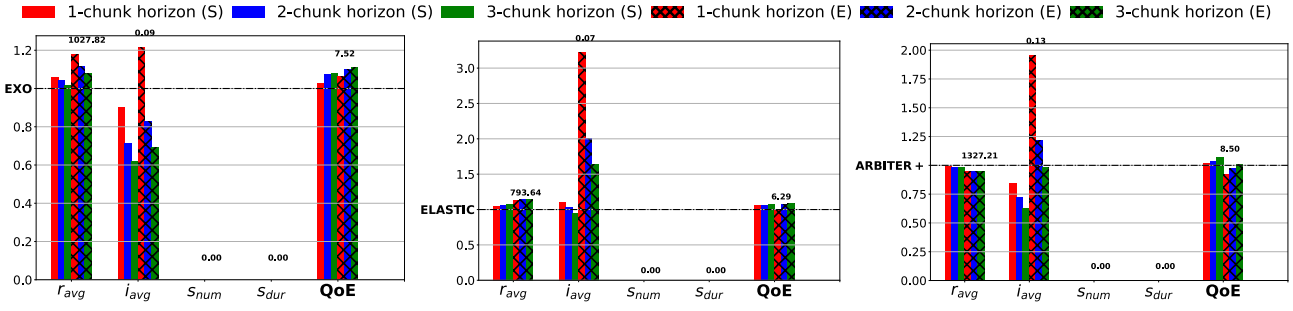
Our results illustrates that the favored prediction integration approach varies among different HAS algorithms. Our results shows that Arbiter favors *S-type* integration while Elastic and Exo favor *E-type* integration. This is attributed to the nature of EWMA that features a memory element in the throughput estimation. This feature introduces temporal correlation to subsequent estimates leading to improving both stability and quality rate. By passing the throughput as a direct estimate, these benefits are invalidated leading to a degraded streaming performance. On the contrary, conservative estimators (e.g., harmonic mean and median) tend to suppress temporal improvement in network conditions. Note that the median would overlook high throughput samples until they dominate and harmonic mean would deem them outliers. By supplying the throughput directly to the adaptation logic, the adaptation logics would have a better vision of the underlying network changes. However, such benefit become more noticeable with larger prediction horizons.

3.1.2 Traces with Low Variability. Next we focus on results obtained when considering *traces with low variability*, *i.e.,* traces characterised by a low value of throughput standard deviation. Figure 5 shows that HAS algorithms achieve stall-free session due to relatively low variation in available throughput. HAS algorithms can counter small variance in bandwidth throughput, as averaging smooth out variations. However, even with the absence of stalls, integrating prediction helps in improving QoE metrics across different algorithms. In particular, prediction improves average instability by 5%-38%. Average representation rate shows similar trend for HAS algorithms as for the high-variable case. Accurate bandwidth estimates corrects throughput underestimation with Elastic and Exo, while reducing overestimation for Arbiter+. As a result, overall user QoE improves by 7%-11%.



(a) Exo (b) Elastic (c) Arbiter+

Figure 4: Performance evaluation of QoE metrics for high-variable bandwidth traces across different HAS algorithms



(a) Exo (b) Elastic (c) Arbiter+

Figure 5: Performance evaluation of QoE metrics for low-variable bandwidth traces across different HAS algorithms

Similar to the previous scenario, increasing horizon improves switching stability. Overall, 3-chunk horizon gives the best performance across all algorithms.

Our findings confirm observation from the previous section, with Arbiter+ algorithm showing the highest boost with *S-type* prediction, while Elastic and Exo prefer *E-type* prediction.

3.2 Inaccurate Predictions

In practice, obtaining ideal prediction is unattainable. To evaluate the impact of prediction errors, we induce errors in our throughput values for different horizons. Similar analysis has been carried out in [6], where authors induce errors to their prediction values. However, their approach is not applicable to existing HAS algorithms as they assumed having multiple disjoint prediction values for a future horizon, with error increasing as the horizon increases. On the other hand, we only consider having one prediction value for next x seconds. We model the actual predicted throughput sample $R^E_{Hk_i}$ as:

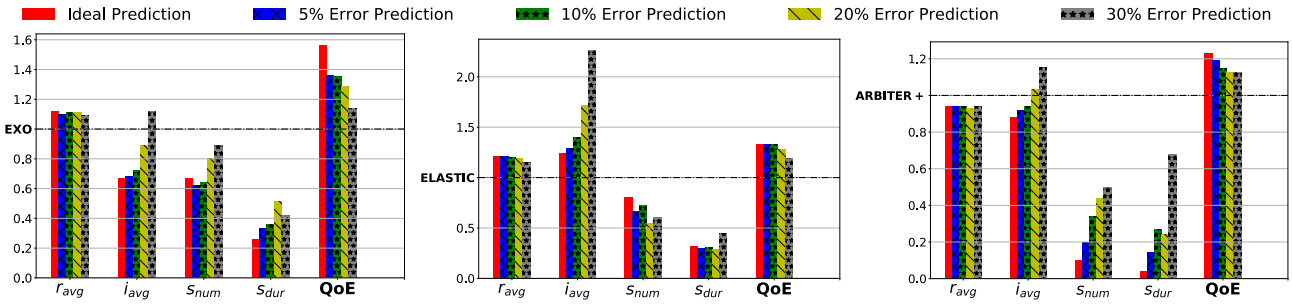
$$R^E_{Hk_i} = R_{Hk_i} + R_{Hk_i} \times N(0, \sigma^2) \tag{3}$$

where R_{Hk_i} is sample i of ideal average throughput over next k seconds (horizon k), $N(0, \sigma^2)$ represents a Gaussian (normal) distribution with zero mean and σ^2 variance.

Let's define absolute value of residual error (ARE) as a difference between actual predicted value and error-induced values. We use different values of σ^2 (5, 10, 20, 30) that induce an ARE of (5%, 10%, 20%, 30%), respectively.

We analyse the impact of adding errors to our prediction values. Due to space constraints, we conduct our experiments for the traces with high variability only. We use 3-chunk horizon as it shows the highest improvement across all HAS algorithms. For Arbiter+ we incorporate *S-type* prediction, while direct estimate is used for Elastic and Exo.

Fig. 6 shows performance across different QoE metrics. Similar to previous sections, we show normalized values against performance of each algorithm without prediction as the reference point (black dotted line). We use red color to mark ideal prediction, while predictions with 5%, 10%, 20%, and 30% are coloured with blue, green, yellow and gray, respectively. There are no significant differences in average rate quality across different error levels for all HAS algorithms. In particular, even with the 30% included prediction errors difference to (ideal) rate quality is less than 2% for Arbiter+ and Exo, and less than 6% for Elastic. On the other hand, switching behavior and stall performance worsen with the error increase. On average, average instability increases by 60%. This result in higher instability than in *no-prediction* case when the error reaches 30%. Intuitively, overall user QoE drops with the increase in error levels. However, 5% prediction error lowers QoE by 20% and 4% compared to the ideal prediction for Exo and Arbiter+, respectively, while the loss in QoE is negligible with Elastic. Overall, this limits prediction impact. This result is intuitive, as Elastic and Arbiter+ employ additional functions limiting bandwidth estimation effects (e.g., Elastic uses more cautious approach by insisting on minimizing stall events). On the other hand, Exo relies heavily on rate estimation as the additional module that monitors buffer occupancy is only activated



(a) Exo (b) Elastic (c) Arbiter+

Figure 6: Performance evaluation of QoE metrics for high-variable bandwidth traces across different HAS algorithms using error-induced predictions

for chunk replacement function. As a consequence, error-induced prediction doesn't cause high QoE drop. Still, applying prediction with the significantly high errors (30%) provides overall higher QoE than in *no-prediction* case. The average increase in QoE across all algorithms with the 30% prediction error is 15%.

4 CONCLUSIONS & FUTURE WORK

In this paper, we investigate how to integrate throughput prediction with state-of-the-art HTTP adaptive streaming (HAS) algorithms and quantify its impact on overall user QoE. We explore different ways prediction can be delivered to the player's bandwidth estimator, either as a direct estimate or a sample. Furthermore, we look at prediction horizons beyond one chunk duration and examine how different levels of error-induced predictions negatively impact user experience.

We find that, regardless of the algorithm in use, user QoE improves by a significant 23% in presence of accurate throughput prediction. Furthermore, highest QoE is observed in presence of longer throughput prediction horizons. Most notably, accurate prediction eliminates stall events in an environment with highly fluctuating throughput. While error-induced predictions lower significantly the user QoE in some instances, it still provides a clear 15% gap on average, compared to HAS algorithms with no prediction.

Our results are very encouraging and motivate future exploration. First, we plan to extend our analysis beyond the current 4 second chunk duration. Second, using a 4K video dataset will allow us to extend our experiments to 4G traces with high throughput. Finally, in our evaluation we use pre-calculated values for prediction horizon. The next challenge is moving from "offline" analysis to real-time prediction in the wild. Influenced by [10, 14] we plan to leverage the Android API to obtain necessary channel information and combine it with a Machine Learning library (e.g., TensorFlow) to provide accurate predictions to HAS algorithms in real time.

ACKNOWLEDGEMENTS

The authors acknowledge the support of Science Foundation Ireland (SFI) under Research Grant 13/IA/1892.

REFERENCES

[1] L. De Cicco, V. Caldaralo, V. Palmisano, and S. Mascolo. 2013. ELASTIC: A Client-Side Controller for Dynamic Adaptive Streaming over HTTP (DASH). In *2013 20th International Packet Video Workshop*. 1–8. https://doi.org/10.1109/PV.2013.6691442

[2] J. Jiang, V. Sekar, and H. Zhang. 2014. Improving Fairness, Efficiency, and Stability in HTTP-Based Adaptive Video Streaming With Festive. *IEEE/ACM Transactions on Networking* 22, 1 (Feb 2014), 326–340.

[3] Jan Willem Kleinrouweler, Britta Meixner, and Pablo Cesar. 2017. Improving Video Quality in Crowded Networks Using a DANE. In *Proceedings of the 27th Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'17)*. ACM, New York, NY, USA, 73–78. https://doi.org/10.1145/3083165.3083167

[4] J. Kua, G. Armitage, and P. Branch. 2017. A Survey of Rate Adaptation Techniques for Dynamic Adaptive Streaming Over HTTP. *IEEE Communications Surveys Tutorials* 19, 3 (thirdquarter 2017), 1842–1866.

[5] Y. Liu, S. Dey, F. Ulupinar, M. Luby, and Y. Mao. 2015. Deriving and Validating User Experience Model for DASH Video Streaming. *IEEE Transactions on Broadcasting* 61, 4 (Dec 2015), 651–665.

[6] T. Mangla, N. Theera-Ampornpunt, M. Ammar, E. Zegura, and S. Bagchi. 2016. Video Through a Crystal Ball: Effect of Bandwidth Prediction Quality on Adaptive Streaming in Mobile Environments. In *Proceedings of the 8th International Workshop on Mobile Video (MoVid '16)*. ACM, New York, NY, USA, Article 1, 6 pages.

[7] M. H. Pinson and S. Wolf. 2004. A new standardized method for objectively measuring video quality. *IEEE Transactions on Broadcasting* 50, 3 (Sept 2004), 312–322. https://doi.org/10.1109/TBC.2004.834028

[8] J. J. Quinlan, A. H. Zahran, and C. J. Sreenan. 2016. Datasets for AVC (H.264) and HEVC (H.265) Evaluation of Dynamic Adaptive Streaming over HTTP (DASH). In *Proceedings of the 7th International Conference on Multimedia Systems (MMSys '16)*. ACM, New York, NY, USA, Article 51, 6 pages.

[9] D. Raca, J. J. Quinlan, A. H. Zahran, and C. J. Sreenan. 2018. Beyond Throughput: a 4G LTE Dataset with Channel and Context Metrics. In *Proceedings of the 9th International Conference on Multimedia Systems (MMSys '18)*. ACM, New York, NY, USA.

[10] D. Raca, A. H. Zahran, C. J. Sreenan, R. K. Sinha, E. Halepovic, R. Jana, and V. Gopalakrishnan. 2017. Back to the Future: Throughput Prediction For Cellular Networks Using Radio KPIs. In *Proceedings of the 4th ACM Workshop on Hot Topics in Wireless (HotWireless '17)*. ACM, New York, NY, USA, 37–41.

[11] Y. Sani, A. Mauthe, and C. Edwards. 2017. Adaptive Bitrate Selection: A Survey. *IEEE Communications Surveys Tutorials* 19, 4 (Fourthquarter 2017), 2985–3014.

[12] Thomas Stockhammer. 2011. Dynamic Adaptive Streaming over HTTP –: Standards and Design Principles. In *Proceedings of the Second Annual ACM Conference on Multimedia Systems (MMSys '11)*. ACM, New York, NY, USA, 133–144. https://doi.org/10.1145/1943552.1943572

[13] J. De Vriendt, D. De Vleeschauwer, and D. Robinson. 2013. Model for estimating QoE of video delivered using HTTP adaptive streaming. In *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*. 1288–1293.

[14] C. Yue, R. Jin, K. Suh, Y. Qin, B. Wang, and W. Wei. 2017. LinkForecast: Cellular Link Bandwidth Prediction in LTE Networks. *IEEE Transactions on Mobile Computing* PP, 99 (2017), 1–1.

[15] A. H. Zahran, D. Raca, and C. Sreenan. 2018. ARBITER+: Adaptive Rate-Based InTElligent HTTP StReaming Algorithm for Mobile Networks. *IEEE Transactions on Mobile Computing* (2018), 1–1. https://doi.org/10.1109/TMC.2018.2825384

[16] X. K. Zou, J. Erman, V. Gopalakrishnan, E. Halepovic, R. Jana, X. Jin, J. Rexford, and R. K. Sinha. 2015. Can Accurate Predictions Improve Video Streaming in Cellular Networks?. In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications (HotMobile '15)*. ACM, New York, NY, USA, 57–62.