An Open Software for Bitstream-based Quality Prediction in Adaptive Video Streaming

Huyen T. T. Tran
The University of Aizu
Aizuwakamatsu, Japan
tranhuyen1191@gmail.com

Duc Nguyen
The University of Aizu
Aizuwakamatsu, Japan
nvduc712@gmail.com

Truong Cong Thang
The University of Aizu
Aizuwakamatsu, Japan
thang@u-aizu.ac.jp

ABSTRACT

HTTP Adaptive Streaming (HAS) has become a popular solution for multimedia delivery nowadays. However, because of throughput fluctuations, video quality may be dramatically varying. Also, stalling events may occur during a streaming session, causing negative impacts on user experience. Therefore, a main challenge in HAS is how to evaluate the overall quality of a session taking into account the impacts of quality variations and stalling events. In this paper, we present an open software, called *BiQPS*, using a Long-Short Term Memory (LSTM) network to predict the overall quality of HAS sessions. The prediction is based on bitstream-level parameters, so it can be directly applied in practice. Through experiment results, it is found that *BiQPS* outperforms four existing models. Our software has been made available to the public at https://github.com/TranHuyen1191/BiQPS.

CCS CONCEPTS

• **Information systems** → **Multimedia streaming**;

KEYWORDS

Overall Quality Prediction, HTTP Adaptive Streaming, Quality Prediction Software

ACM Reference Format:
Huyen T. T. Tran, Duc Nguyen, and Truong Cong Thang. 2020. An Open Software for Bitstream-based Quality Prediction in Adaptive Video Streaming. In *Proceedings of ACM Multimedia Systems Conference (MMSys'20).* ACM, New York, NY, USA, Article 4, 6 pages. https://doi.org/10.475/123_4

1 INTRODUCTION

Watching online videos (both Live and On-Demand) has become one of the most popular user activities on the Internet. According to [4], the average online video viewing hours have increased 68% globally in 2019. As of 2019, streaming video accounts for over 60% of all traffic on the Internet [15].

HTTP Adaptive Streaming (HAS) has become the standard solution for multimedia streaming over the Internet nowadays [16]. In HAS, a video is divided into short segments, each is encoded into multiple versions corresponding to different quality levels. Based on an estimated throughput, segments with suitable versions are delivered to a user device. Because of throughput fluctuations, the segments' versions are often changing, resulting in quality variations during a streaming session. In addition, stalling events may occur when segments do not reach the user device before their playback deadlines. It is well-known that the two factors of quality variations and stalling events cause negative impacts on user experience [18]. Therefore, it is crucial to evaluate the overall quality of a streaming session taking into account the impacts of both the factors.

In the literature, there have been many models proposed for predicting the overall quality of HAS sessions [2, 6, 7, 9, 21, 22]. Most previous models are built based on segment quality values which are calculated by quality metrics such as VQM [9], STRRED [2, 6], and MOS [7, 21, 22]. However, obtaining segment quality values is non-trivial, demanding high storage, computation, and human resources. For example, original videos are required for computing the VQM and STRRED values of versions of individual segments. To obtain MOS values, it is commonly necessary to conduct subjective tests which are time-consuming and labour-intensive [3].

So far, only a few bitstream-based models have been proposed [11, 17]. The inputs of these models can be directly extracted from streamed videos with low complexity and does not require accesses to original videos. Yet, as will be shown later, the performances of these models are unstable and not very effective for multiple datasets.

In this paper, we present a bitstream-based quality prediction software, called *BiQPS*, for HAS. The software takes as inputs four bitstream-level parameters of individual segments in a session, namely quantization parameter (QP), bitrate (BR), resolution (RS), and frame-rate (FR). The initial delay is considered a special case of stalling. The prediction takes advantage of an advanced Long-Short Term Memory (LSTM) network with bidirectional and attention mechanisms. To evaluate the software, we use three different datasets containing sessions of different lengths. Experiment results show that our software is effective in predicting overall quality for both short and long sessions. In addition, it is found that this software consistently outperforms existing bitstream-based models including the standardized ITU-T P1203 model.

So far, little effort has been made to provide high performance and open-source software for overall quality prediction in HAS. Our software have been made public to facilitate research in quality evaluation for HAS. The software is a cross-platform Python implementation. By using our software, researchers can quickly verify the effectiveness of new adaptation algorithms. For practitioners, our software provides a new tool for monitoring/evaluating streaming services with high accuracy.

Permission to make digital or hard copies of part or all 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 third-party components of this work must be honored. For all other uses, contact the owner/author(s).
MMSys'20, June 2020, Istanbul, Turkey
© 2020 Copyright held by the owner/author(s).
ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4

The rest of this paper is organized as follows. Section 2 highlights existing overall quality models. The architecture of the software is described in Section 3. In Section 4, we present an evaluation of the software and existing models. Finally, Section 5 concludes the paper.

2 RELATED WORK

In the literature, a lot of overall quality models have been proposed for HAS [7, 9, 14, 21, 22]. However, most models are limited to short sessions with durations of 1–3 minutes [7, 9, 21]. Also, these models mainly focus on the impact of quality variations which is generally modeled by some statistics of segment quality values such as average [21, 22], standard deviation [21], and minimum [7]. As mentioned above, it is difficult to obtain segment quality values in practice. For the impact of stalling events, previous models commonly use some statistics such as the number of stalling events [9] and the sum of stalling durations [9, 14].

To the best of our knowledge, there are only three bitstream-based models proposed in the literature [11, 14, 17]. In particular, the model in [17] uses four inputs, namely the average of quantization parameter values over all macro blocks of all video frames, the average and maximum of stalling durations, and the number of stalling events. To predict the overall quality, the inputs are fed into a random neural network. The model is evaluated using a dataset containing 118 very short sessions with a length of 16 seconds.

In [14], a bitstream-based model is also proposed, in which a session is divided into three temporal intervals. For each interval, the impact of quality variations is modeled by frequencies of quality switching types, each is defined based on specific resolutions and frame-rates. Regarding the impact of stalling events, the model uses the average duration and total number of events. To combine the intervals, each is simply assigned a weight to represent its contribution to the overall quality. Experiment result shows that the model can well predict for sessions with lengths from 1 to 4.5 minutes. However, it should be noted that this model is specific to a very limited set of switching types. Therefore, it can not be applied for general cases in practice.

In the latest stage of ITU-T Rec. P.1203 standardization, a software (called *P.1203*) is presented for predicting the overall quality of both short and long sessions (i.e., from 1 to 5 minutes) [11]. In this software, each segment is attributed by a quality value which is obtained from the corresponding bitstream-level parameters such as bitrate, QP, and resolution. To model the impact of quality variations, a session is divided into three intervals. The average values of segment quality values in individual intervals are used as inputs, along with various statistics calculated over the whole session, such as the total number of quality direction changes and the difference between the maximum and minimum quality levels. For the impact of stalling events, the software uses the durations and positions of stalling events. To predict the overall quality value, the inputs are aggregated using statistical functions and a decision tree. As will be shown later, the performance of this software is unstable and not very effective over multiple datasets.

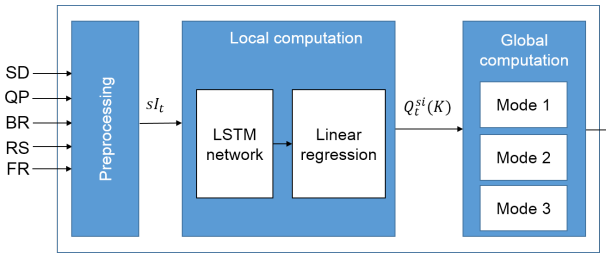


Figure 1: General architecture

3 SOFTWARE

In this section, we first present the general architecture of the software. Then, the main components are described in detail.

Fig. 1 shows the general architecture of the software with three main components, namely preprocessing, local computation and global computation. Given a session, the preprocessing component is responsible to divide the session into short intervals, each is attributed by a quality value $Q_t^{si}(K)$ calculated by the local computation. Through the global computation, the values $Q_t^{si}(K)$ of all the intervals are aggregated into an overall quality value Q^o.

3.1 Preprocessing component

In our software, each video segment n is represented by five parameters, namely stalling duration SD_n, quantization parameter QP_n, bitrate BR_n, resolution RS_n, and frame-rate FR_n. In particular, SD_n is the amount of time that the user has to wait since the end of previous segment $n-1$ until the start of segment n. If segment n arrives at the client before the playback of segment $n-1$ finishes (called the playback deadline), SD_n is set to 0. Otherwise, a stalling event occurs and SD_n is a positive value. Note that, for the first segment, SD_0 is the initial delay to fill up a video buffer before the video starts to play. QP_n is calculated as the average of quantization parameter values over all macro-blocks of all frames in segment n. BR_n, RS_n, and FR_n are respectively the bitrate, resolution, and frame-rate of segment n. Obviously, these parameters are very basic and can be easily extracted from a video bitstream.

Assume that there exists a stalling event with the length of 0.5 seconds just before starting the playback of segment n. In addition, segment n has an average QP of 26, a bitrate of 3500 kbps, a resolution of 1280×720, and a frame-rate of 24fps. The vector consisting of the parameters of segment n is

$$\mathbf{x}_n = \begin{bmatrix} SD_n \\ QP_n \\ BR_n \\ RS_n \\ FR_n \end{bmatrix} = \begin{bmatrix} 0.5 \\ 26 \\ 3500 \\ 921600 \\ 24 \end{bmatrix}. \quad (1)$$

Let denote N the number of segments in the session. The input matrix composed of the vectors of all the segments is

$$\mathbf{I} = \left[\mathbf{x}_1, ..., \mathbf{x}_n, ..., \mathbf{x}_N\right]. \quad (2)$$

In the preprocessing component, the matrix $\mathbf{I}$ is divided into overlapped sub-matrices $\{\mathbf{sI}_t | 1 \le t \le N-K+1\}$ corresponding to

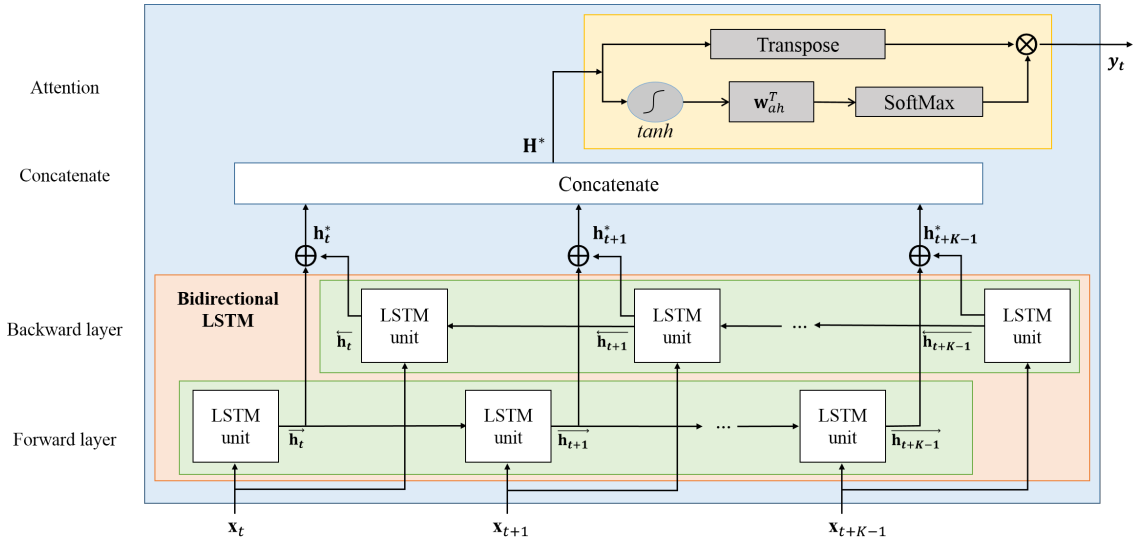


Figure 2: Unrolled LSTM network with bidirectional and attention mechanisms

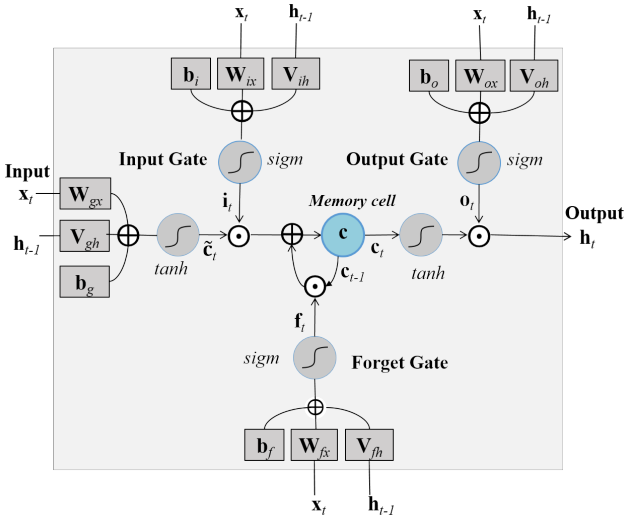


Figure 3: LSTM unit architecture

short intervals t of K segments.

$$\mathbf{sI}_t = \left[\mathbf{x}_t, \mathbf{x}_{t+1}, ..., \mathbf{x}_{t+K-1}\right]. \tag{3}$$

These sub-matrices $\mathbf{sI}_t$ are then inputted into the local computation in order to compute the corresponding interval quality values $Q_t^{si}(K)$.

3.2 Local computation

In the local computation, each input matrix $\mathbf{sI}_t$ is mapped to an output value $Q_t^{si}(K)$. Here, $\mathbf{sI}_t$ is first inputted into an advanced LSTM network with bidirectional and attention mechanisms as shown in Fig. 2. This network can exploit temporal relationships between inputs (i.e., segment parameters) to quantify the impacts of quality variations and stalling events.

In particular, each vector $\mathbf{x}_t$ in $\mathbf{sI}_t$ is fed into two LSTM units, one in the forward layer and another in the backward layer, to calculate the corresponding hidden states. The architecture of the LSTM unit is shown in Fig. 3. Let denote $\overrightarrow{\mathbf{h}_t}$ the hidden state of the forward layer and $\overleftarrow{\mathbf{h}_t}$ the hidden state of the backward layer. $\overrightarrow{\mathbf{h}_t}$ and $\overleftarrow{\mathbf{h}_t}$ are calculated by the following equations.

$$\overrightarrow{\tilde{\mathbf{c}}_t} = tanh(\overrightarrow{\mathbf{W}_{gx}}\mathbf{x}_t + \overrightarrow{\mathbf{V}_{gh}}\overrightarrow{\mathbf{h}_{t-1}} + \overrightarrow{\mathbf{b}_g}), \tag{4}$$

$$\overrightarrow{\mathbf{i}_t} = sigm(\overrightarrow{\mathbf{W}_{ix}}\mathbf{x}_t + \overrightarrow{\mathbf{V}_{ih}}\overrightarrow{\mathbf{h}_{t-1}} + \overrightarrow{\mathbf{b}_i}), \tag{5}$$

$$\overrightarrow{\mathbf{f}_t} = sigm(\overrightarrow{\mathbf{W}_{fx}}\mathbf{x}_t + \overrightarrow{\mathbf{V}_{fh}}\overrightarrow{\mathbf{h}_{t-1}} + \overrightarrow{\mathbf{b}_f}), \tag{6}$$

$$\overrightarrow{\mathbf{o}_t} = sigm(\overrightarrow{\mathbf{W}_{ox}}\mathbf{x}_t + \overrightarrow{\mathbf{V}_{oh}}\overrightarrow{\mathbf{h}_{t-1}} + \overrightarrow{\mathbf{b}_o}), \tag{7}$$

$$\overrightarrow{\mathbf{c}_t} = \overrightarrow{\mathbf{f}_t} \odot \overrightarrow{\mathbf{c}_{t-1}} + \overrightarrow{\mathbf{i}_t} \odot \overrightarrow{\tilde{\mathbf{c}}_t}, \tag{8}$$

$$\overrightarrow{\mathbf{h}_t} = \overrightarrow{\mathbf{o}_t} \odot tanh(\overrightarrow{\mathbf{c}_t}), \tag{9}$$

$$\overleftarrow{\tilde{\mathbf{c}}_t} = tanh(\overleftarrow{\mathbf{W}_{gx}}\mathbf{x}_t + \overleftarrow{\mathbf{V}_{gh}}\overleftarrow{\mathbf{h}_{t-1}} + \overleftarrow{\mathbf{b}_g}), \tag{10}$$

$$\overleftarrow{\mathbf{i}_t} = sigm(\overleftarrow{\mathbf{W}_{ix}}\mathbf{x}_t + \overleftarrow{\mathbf{V}_{ih}}\overleftarrow{\mathbf{h}_{t-1}} + \overleftarrow{\mathbf{b}_i}), \tag{11}$$

$$\overleftarrow{\mathbf{f}_t} = sigm(\overleftarrow{\mathbf{W}_{fx}}\mathbf{x}_t + \overleftarrow{\mathbf{V}_{fh}}\overleftarrow{\mathbf{h}_{t-1}} + \overleftarrow{\mathbf{b}_f}), \tag{12}$$

$$\overleftarrow{\mathbf{o}_t} = sigm(\overleftarrow{\mathbf{W}_{ox}}\mathbf{x}_t + \overleftarrow{\mathbf{V}_{oh}}\overleftarrow{\mathbf{h}_{t-1}} + \overleftarrow{\mathbf{b}_o}), \tag{13}$$

$$\overleftarrow{\mathbf{c}_t} = \overleftarrow{\mathbf{f}_t} \odot \overleftarrow{\mathbf{c}_{t-1}} + \overleftarrow{\mathbf{i}_t} \odot \overleftarrow{\tilde{\mathbf{c}}_t}, \tag{14}$$

$$\overleftarrow{\mathbf{h}_t} = \overleftarrow{\mathbf{o}_t} \odot tanh(\overleftarrow{\mathbf{c}_t}), \tag{15}$$

where the parameters of $\overrightarrow{\mathbf{W}}$, $\overrightarrow{\mathbf{V}}$, $\overrightarrow{\mathbf{b}}$, $\overleftarrow{\mathbf{W}}$, $\overleftarrow{\mathbf{V}}$, and $\overleftarrow{\mathbf{b}}$ are learned in the training process.

Then, the hidden states $\overrightarrow{\mathbf{h}_t}$ and $\overleftarrow{\mathbf{h}_t}$ are aggregated using element-wise addition as follows.

$$\mathbf{h}_t^* = \overrightarrow{\mathbf{h}_t} \oplus \overleftarrow{\mathbf{h}_t}. \tag{16}$$

Let $\mathbf{H}^* = [\mathbf{h}_t^*, \mathbf{h}_{t+1}^*, ..., \mathbf{h}_{t+K-1}^*]$ be a matrix consisting of the output vectors $\mathbf{h}_t^*$. The weights of the hidden states are calculated by the following equation.

$$\alpha = softmax(\mathbf{w}_{ah}^T tanh(\mathbf{H}^*)), \tag{17}$$

Table 1: Performance of Models in Predicting Overall Quality

Model			VL (1 minute)				VL04 (1 minute)				VL13 (4 minutes)			
			Performance		*Coefficient*		*Performance*		*Coefficient*		*Performance*		*Coefficient*	
			PCC	*RMSE*	*Slope*	*Intercept*	*PCC*	*RMSE*	*Slope*	*Intercept*	*PCC*	*RMSE*	*Slope*	*Intercept*
Vriendt's			0.714	0.670	0.704	0.336	0.711	0.628	0.805	0.502	0.804	0.617	1.101	-0.443
Yin's			0.599	0.766	0.00003	3.229	0.702	0.635	0.00003	3.352	0.625	0.810	0.00002	3.606
Singh's			0.787	0.597	1.019	0.371	0.350	0.835	0.481	1.860	0.061	1.034	0.133	2.825
P.1203			0.922	0.369	1.119	-1.203	0.888	0.411	1.041	0.075	0.931	0.381	1.180	-0.506
BiQPS	*Mode 1*	$K = 10$	0.892	0.433	1.084	-1.531	0.895	0.398	0.791	0.451	0.888	0.478	0.993	-0.593
		$K = 20$	0.902	0.413	0.898	-0.478	0.903	0.384	0.677	1.062	0.907	0.436	0.926	-0.058
		$K = 30$	0.900	0.417	0.819	0.016	0.896	0.397	0.644	1.233	0.910	0.429	0.897	0.226
		$K = 40$	0.906	0.404	0.810	0.230	0.896	0.396	0.634	1.328	0.906	0.438	0.881	0.417
		$K = 50$	0.931	0.350	0.863	0.204	0.901	0.387	0.653	1.338	0.902	0.448	0.871	0.567
		$K = 60$	0.946	0.310	0.906	0.165	0.906	0.377	0.687	1.268	0.896	0.461	0.869	0.676
	Mode 2	$K = 10$	0.738	0.645	0.803	-0.562	0.851	0.469	0.597	1.018	0.774	0.656	0.676	0.523
		$K = 20$	0.802	0.571	0.700	0.179	0.880	0.423	0.583	1.278	0.765	0.667	0.629	0.959
		$K = 30$	0.851	0.502	0.717	0.393	0.871	0.439	0.573	1.405	0.868	0.514	0.754	0.849
		$K = 40$	0.892	0.432	0.738	0.504	0.887	0.413	0.607	1.415	0.879	0.495	0.806	0.811
		$K = 50$	0.917	0.381	0.822	0.347	0.900	0.389	0.650	1.347	0.881	0.491	0.822	0.790
		$K = 60$	0.944	0.315	0.901	0.176	0.906	0.377	0.687	1.268	0.880	0.493	0.814	0.889
	Mode 3	$K_1 = 60, K_2 = 50$	**0.949**	**0.301**	0.868	0.283	**0.910**	**0.371**	0.689	1.269	**0.941**	**0.350**	0.953	0.280

where $\mathbf{w}_{ah}$ is a parameter to be learned.

The output of the LSTM network is given by

$$\mathbf{y_t} = \mathbf{H}^* \alpha^T. \tag{18}$$

Finally, the quality value $Q_t^{si}(K)$ of interval t is calculated through a linear regression as described in the following equation.

$$Q_t^{si}(K) = \mathbf{w}_r \mathbf{y_t} + b_r, \tag{19}$$

where $\mathbf{w}_r$ and b_r are parameters to be learned.

To obtain the parameters in this component, we use two datasets, one from [19] and another from ITU-T Rec. P.1203 standardization (called TR04) [13]. The training sets in the datasets are used to learn the parameters, and the test set to qualify the performance. The loss function is calculated as the root mean square error between the predicted quality values and the corresponding subjective quality values. To minimize this function, we use a stochastic gradient descent method based on an Adam optimization algorithm [8]. The parameters of the Adam algorithm are set as follows: $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1e-08$. The learning rate is set to 0.01. The parameters corresponding to the highest performance for the test set are recored and used as default values in the software.

3.3 Global computation

In this component, we implement three modes corresponding to three temporal pooling ways to combine the values $\{Q_t^{si}(K) | 1 \leq t \leq N - K + 1\}$ into an overall quality Q^o. In the following, each mode will be described in detail.

3.3.1 Mode 1. In the first mode, Q^o is calculated using the mean pooling which is given by the following equation.

$$Q^o = \frac{\sum_{t=1}^{N-K+1} Q_t^{si}(K)}{N - K + 1}. \tag{20}$$

In some previous studies, the mean pooling is found to be very effective for aggregating instantaneous quality values into an overall quality value [1, 6]. However, so far, there exits no study using this way to combine interval quality values.

Since the interval length K may affect the performance of the pooling way, we investigate six different interval lengths from 10 to 60 segments. The performance of the software for each length will be reported and discussed in the next section.

3.3.2 Mode 2. For the second mode, we use the median pooling. In particular, Q^o is calculated by

$$Q^o = \text{median}_{t=1}^{N-K+1} Q_t^{si}(K). \tag{21}$$

Similar to *Mode 1*, the performance of *Mode 2* will be investigated using different interval lengths from 10 to 60 segments.

3.3.3 Mode 3. With respect to the third mode, we use a weighted sum of four statistics of $Q_t^{si}(K)$, namely average, minimum, maximum, and last, to compute Q^o as described in the following equation.

$$\begin{aligned} Q^o =& w_1 \times \frac{\sum_{t=1}^{N-K_1+1} Q_t^{si}(K_1)}{N - K_1 + 1} + w_2 \times \min_{t=1}^{N-K_2+1} Q_t^{si}(K_2) \\ &+ w_3 \times \max_{t=1}^{N-K_2+1} Q_t^{si}(K_2) + w_4 \times Q_{N-K_2+1}^{si}(K_2). \end{aligned} \tag{22}$$

This pooling way has been proposed in [20]. By using a subjective dataset, the optimal weights $\{w_i | i = [1, 2, 3, 4]\}$ and interval lengths corresponding to individual statistics were empirically derived by ANOVA analysis as described in [20]. Specifically, the values of the weights are as follows: $w_1 = 0.426, w_2 = 0.28, w_3 = 0.014$, and $w_4 = 0.28$. To calculate the average, the interval length is $K_1 = 60$ segments. For the other statistics, the interval length is $K_2 = 50$ segments. In the software, the optimal weights and lengths are used as default settings. It should be noted that the quality model in [20] is not based on bitstream-level parameters and LSTM network.

4 EVALUATION

In this section, we evaluate the performance of our software in comparison to four reference models, namely *Vriendt's* [21], *Yin's* [22], *Singh's* [17], and *P.1203* [13]. Among the models, *Vriendt's* and *Yin's* models use statistics of segment quality values in MOS to predict the overall quality. Meanwhile, *Singh's* model and the *P.1203* model are based on bitstream-level parameters. All the models are tested over three test datasets, namely *VL* [19], *VL04* [13], and *VL13* [13]. Note that the datasets are not used for building any models/softwares here. The *VL* and *VL04* datasets respectively consist of 161 and 60 sessions of approximately 60 seconds. The *VL13* dataset is composed of fifteen 4-minute long sessions.

Similar to [5, 23], we implement *Vriendt's* and *Yin's* models with settings stated in the corresponding publications [21, 22]. In *Singh's* model, the training process of the random neural network is similar to that used to train the local computation in our software as mentioned in Subsection 3.2. For the *P.1203* model, we use an implementation of the standard that is free to use for research purposes [10, 13][1].

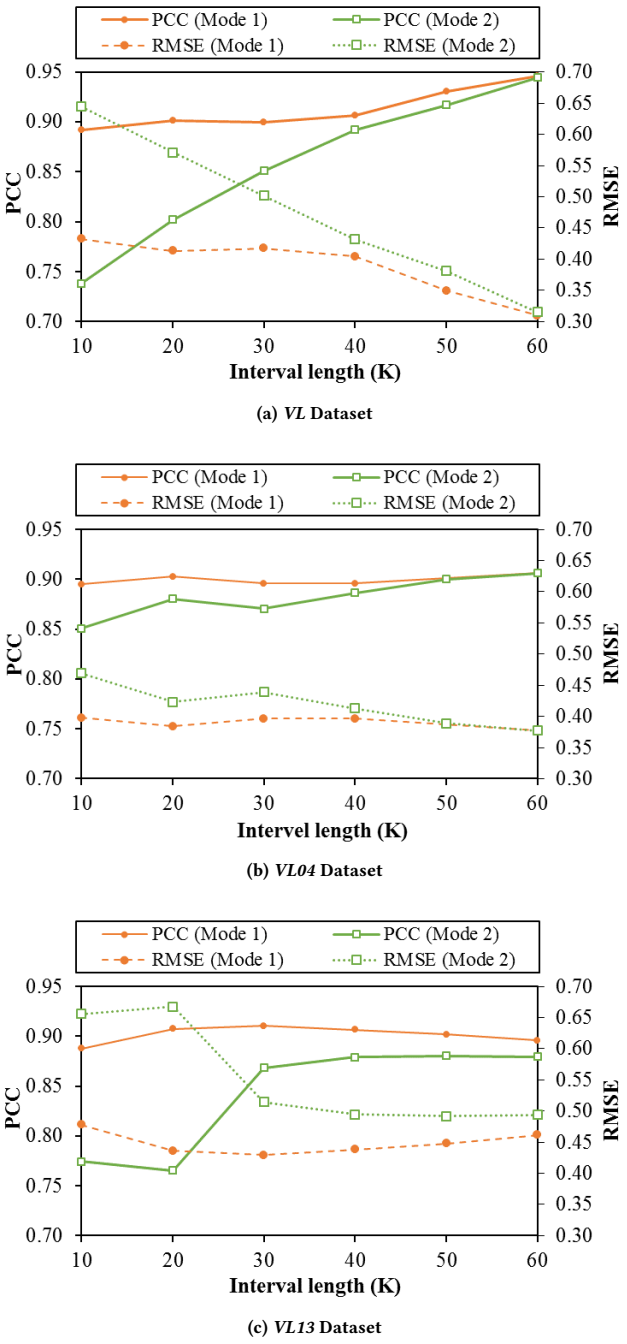
Following ITU-T P.1401 recommendation [12], a first order linear regression between predicted quality values and subjective quality values is performed for each model to compensate for possible variances between subjective tests of different datasets. The adjustment coefficients of slope and intercept of each model are reported for each dataset in Table 1. To evaluate the performance of each model, we use two metrics, namely Pearson Correlation Coefficient (PCC) and Root-Mean-Squared Error (RMSE). In particular, the PCC and RMSE are respectively used to measure the linear relationship and difference between quality values predicted from a model and subjective quality values in a dataset. Note that a higher PCC and a lower RMSE mean better prediction performance.

Table 1 shows the obtained results of the models for each dataset. It can be seen that the *BiQPS* software at *Mode 3* achieves the highest performance for all the datasets. In particular, the PCC and RMSE values are respectively 0.949 and 0.301 for the *VL* dataset, 0.910 and 0.371 for the *VL04* dataset, and 0.941 and 0.350 for the *VL13* dataset. This means that our software can accurately predict the overall quality of both short and long sessions.

Regarding *Vriendt's* and *Yin's* models, their performances are consistently low for all the datasets (i.e., PCC≤0.804 and RMSE≥0.617). This result means that the statistics used in these models such as average and standard deviation may be not effective to aggregate segment quality values into an overall quality in HAS.

Among the bitstream-based models, *Singh's* model has the worst performance. In particular, its PCC and RMSE values are respectively 0.787 and 0.597 for the *VL* dataset, 0.350 and 0.835 for the *VL04* dataset, and 0.061 and 1.034 for the *VL13* dataset. Such a performance is unacceptable since the PCC values are lower than 0.5 for two among the three datasets. The reason may be that this model is originally built for very short sessions of 16 seconds, resulting in a very low performance when applied for much longer sessions. In addition, the use of average QP over all macro blocks of all frames in a whole session may be not an effective pooling way because it can not reflect quality variations in the session.

[1]https://github.com/itu-p1203/itu-p1203/



(a) *VL* Dataset

(b) *VL04* Dataset

(c) *VL13* Dataset

Figure 4: Performance of our software using *Mode 1* and *Mode 2* with different interval lengths K

With regard to the *P.1203* model, its performance is very high for the *VL* and *VL13* datasets (i.e., PCC≥0.922 and RMSE≤0.381). However, for the *VL04* dataset, the obtained performance is a little low (i.e., PCC = 0.888 and RMSE = 0.411). This result shows that, although the *P.1203* model is designed for various session durations

(i.e., 1–5 minutes), its performance is not consistent across different datasets.

For more details of our software, Fig 4 shows the performance of *Mode 1* and *Mode 2* using different interval lengths K for the three datasets. It can be seen that, for the *VL* and *VL04* datasets, the performance generally increases for both the modes when intervals become longer. In addition, the highest performance is reached at $K = 60$ where the interval is almost the whole session. In particular, the PCC and RMSE values are respectively 0.946 and 0.310 for *Mode 1* and 0.944 and 0.315 for *Mode 2* in the *VL* dataset. For the *VL04* dataset, the performance is PCC = 0.906 and RMSE = 0.377 for both modes.

In general, for the *VL13* dataset, the performance of both modes first increases when the interval length increases. After reaching the saturation point, the performance decreases when the interval length keeps going up. In particular, the highest performance is achieved at $K = 30$ for *Mode 1* (i.e., PCC = 0.910 and RMSE = 0.429) and at $K = 50$ for *Mode 2* (i.e., PCC = 0.881 and RMSE = 0.491).

With respect to all the datasets, the performance of *Mode 1* is always higher or equal to that of *Mode 2*. In addition, when $K = 20$ or $K = 50$, *Mode 1* obtains a quite high performance for all the datasets (i.e., PCC$\geq$0.901 and RMSE$\leq$0.448). Especially, for the *VL* and *VL04* datasets, the performance of *Mode 1* at $K = 50$ is significantly higher than those of the reference models. In particular, the PCC and RMSE values are respectively 0.931 and 0.350 for the *VL* dataset and 0.901 and 0.387 for the *VL04* dataset. Therefore, *Mode 1* at $K = 50$ can also be used to predict the overall quality of short sessions.

It can be seen from the above that, among the considered pooling ways, a weighted sum of statistics as in *Mode 3* is the best way for combining interval quality values. This helps to achieve high performance for both short and long sessions. As *Mode 1* and *Mode 2* do not take into account the factors such as recency and worst impairment, their performances are somewhat lower for long sessions.

5 CONCLUSION

In this study, we have presented an open software for predicting the overall quality of HAS sessions. Thanks to using bitstream-level parameters as inputs, the software is very simple and easy to use in practice. By using the three datasets, we shows that the software achieves a very high performance for both short and long sessions. In addition, it outperforms the four reference models/softwares. To obtain the high prediction performance, our software should be set at *Mode 3* or *Mode 1* (with the interval length of $K = 20$ or $K = 50$ segments). This software can help researchers and practitioners to monitor and evaluate adaptation strategies in HTTP Adaptive Streaming. In future work, we plan to integrate our software into popular HTTP Adaptive Streaming test-bed such as dash.js.

REFERENCES

[1] C. G. Bampis, Z. Li, I. Katsavounidis, and A. C. Bovik. 2018. Recurrent and Dynamic Models for Predicting Streaming Video Quality of Experience. *IEEE Transactions on Image Processing* 27, 7 (Jul. 2018), 3316–3331. https://doi.org/10.1109/TIP.2018.2815842

[2] C. G. Bampis, Z. Li, I. Katsavounidis, TY Huang, C. Ekanadham, and A. C. Bovik. 2018. Towards Perceptually Optimized End-to-end Adaptive Video Streaming. *submitted to IEEE Transactions on Image Processing* (2018). arXiv:eess.IV/1808.03898 https://arxiv.org/abs/1808.03898

[3] S. Chikkerur, V. Sundaram, M. Reisslein, and L. J. Karam. 2011. Objective Video Quality Assessment Methods: A Classification, Review, and Performance Comparison. *IEEE Transactions on Broadcasting* 57, 2 (June 2011), 165–182. https://doi.org/10.1109/TBC.2011.2104671

[4] Conviva. 2019. Conviva state of streaming. (2019). https://www.conviva.com/state-of-streaming/, accessed 2020-01-15.

[5] Z. Duanmu, A. Rehman, and Z. Wang. 2018. A Quality-of-Experience Database for Adaptive Video Streaming. *IEEE Transactions on Broadcasting* 64, 2 (Jun. 2018), 474–487. https://doi.org/10.1109/TBC.2018.2822870

[6] N. Eswara, S. Ashique, A. Panchbhai, S. Chakraborty, H. P. Sethuram, K. Kuchi, A. Kumar, and S. S. Channappayya. 2019. Streaming Video QoE Modeling and Prediction: A Long Short-Term Memory Approach. *IEEE Transactions on Circuits and Systems for Video Technology* (2019), 1–1.

[7] Zhili Guo, Yao Wang, and Xiaoqing Zhu. 2015. Assessing the visual effect of non-periodic temporal variation of quantization stepsize in compressed video. In *2015 IEEE International Conference on Image Processing (ICIP)*. Quebec City, Canada, 3121–3125.

[8] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. (2014). arXiv:1412.6980 http://arxiv.org/abs/1412.6980

[9] Yao Liu, Sujit Dey, Fatih Ulupinar, Michael Luby, and Yinian Mao. 2015. Deriving and validating user experience model for DASH video streaming. *IEEE Transactions on Broadcasting* 61, 4 (2015), 651–665.

[10] Alexander Raake, Marie-Neige Garcia, Werner Robitza, Peter List, Steve Göring, and Bernhard Feiten. 2017. A bitstream-based, scalable video-quality model for HTTP adaptive streaming: ITU-T P.1203.1. In *Ninth International Conference on Quality of Multimedia Experience (QoMEX)*. Erfurt, Germany, 1–6.

[11] Recommendation ITU-T P.1203.3. 2017. Parametric bitstream-based quality assessment of progressive download and adaptive audiovisual streaming services over reliable transport-Quality integration module. *International Telecommunication Union* (2017).

[12] Recommendation ITU-T P.1401. 2012. Methods, metrics and procedures for statistical evaluation, qualification and comparison of objective quality prediction models . *International Telecommunication Union* (2012).

[13] Werner Robitza, Steve Göring, Alexander Raake, David Lindegren, Gunnar Heikkilä, Jörgen Gustafsson, Peter List, Bernhard Feiten, Ulf Wüstenhagen, Marie-Neige Garcia, Kazuhisa Yamagishi, and Simon Broom. 2018. HTTP Adaptive Streaming QoE Estimation with ITU-T Rec. P.1203 - Open Databases and Software. In *Proceedings of the 9th ACM Multimedia Systems Conference*. Amsterdam, Netherlands, 466–471. https://doi.org/10.1145/3204949.3208124

[14] Demóstenes Zegarra Rodríguez, Renata Lopes Rosa, Eduardo Costa Alfaia, Julia Issy Abrahão, and Graça Bressan. 2016. Video quality metric for streaming service using DASH standard. *IEEE Transactions on Broadcasting* 62, 3 (2016), 628–639.

[15] Sandvine. 2019. The Global Internet Phenomena Report. (Sept. 2019). https://www.sandvine.com/phenomena, accessed 2020-01-11.

[16] Michael Seufert, Sebastian Egger, Martin Slanina, Thomas Zinner, Tobias Hoßfeld, and Phuoc Tran-Gia. 2015. A survey on quality of experience of HTTP adaptive streaming. *IEEE Communications Surveys & Tutorials* 17, 1 (2015), 469–492.

[17] Kamal Deep Singh, Yassine Hadjadj-Aoul, and Gerardo Rubino. 2012. Quality of experience estimation for adaptive HTTP/TCP video streaming using H. 264/AVC. In *2012 IEEE Consumer Communications and Networking Conference (CCNC)*. Las Vegas, USA, 127–131.

[18] Samira Tavakoli, Sebastian Egger, Michael Seufert, Raimund Schatz, Kjell Brunnström, and Narciso García. 2016. Perceptual quality of HTTP adaptive streaming strategies: Cross-experimental analysis of multi-laboratory and crowdsourced subjective studies. *IEEE Journal on Selected Areas in Communications* 34, 8 (2016), 2141–2153.

[19] H. T. T. Tran, D. V. Nguyen, D. D. Nguyen, N. P. Ngoc, and T. C. Thang. 2019. An LSTM-based Approach for Overall Quality. In *IEEE Conference on Computer Communications Conference (INFOCOM 2019)*. Paris.

[20] H. T. T. Tran, D. V. Nguyen, D. D. Nguyen, N. P. Ngoc, and T. C. Thang. 2020. Cumulative Quality Modeling for HTTP Adaptive Streaming. In *submitted ACM Transactions on Multimedia Computing Communications and Applications*. Available on: https://arxiv.org/abs/1909.02772.

[21] 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)*. Ghent, Belgium, 1288–1293.

[22] Xiaoqi Yin, Abhishek Jindal, Vyas Sekar, and Bruno Sinopoli. 2015. A control-theoretic approach for dynamic adaptive video streaming over HTTP. *ACM SIGCOMM Computer Communications Review* 45, 4 (2015), 325–338.

[23] Z. Duanmu, K. Ma, and Z. Wang. 2017. Quality-of-Experience of Adaptive Video Streaming: Exploring the Space of Adaptations. In *Proceedings of the 25th ACM international conference on Multimedia*. Mountain View, USA, 1752–1760.