Deep learning-based no-reference video streaming QoE estimation using WebRTC statistics

Keisuke Sakakibara
Graduate School of Information Science and Engineering
The university of Electro-Communications
Chofu, Tokyo, Japan
Email: k.sakakibara@net.lab.uec.ac.jp

Satoshi Ohzahata
Graduate School of Information Science and Engineering
The university of Electro-Communications
Chofu, Tokyo, Japan
Email: ohzahata@uec.ac.jp

Ryo Yamamoto
Graduate School of Information Science and Engineering
The university of Electro-Communications
Chofu, Tokyo, Japan
Email: ryo-yamamoto@uec.ac.jp

***Abstract*—Web Real-Time Communication (WebRTC) enables low-latency communication and is commonly used for real-time video streaming. To ensure acceptable Quality of Experience (QoE) for video streaming services, a real-time QoE monitoring model is required. While Full-Reference (FR) models have a high correlation with subjective evaluations, they are not suitable for real-time monitoring. In contrast, No-reference (NR) models exhibit real-time performance but lower correlation. In this paper, we propose a deep learning-based NR model that achieves the same level of QoE estimation accuracy as the FR model. For the training data, we used the QoE computed from Video Multimethod Assessment Fusion (VMAF), an FR model known to be highly correlated with subjective ratings. The proposed model is trained to estimate QoE based on WebRTC statistics. Our experiments show that the proposed NR model outperforms the existing model in terms of Root Mean Squared Error (RMSE) and Pearson Correlation Coefficient (PCC).**

***Index Terms*—QoE, VQA, WebRTC, Deep learning**

I. Introduction

In recent years, video streaming services have become widespread on the internet. HTTP Live Streaming (HLS) and Dynamic Adaptive Streaming Over HTTP (MPEG-DASH) are often used for on-demand video streaming services but have a latency of several seconds because these streaming services rely on reliable transport of TCP. WebRTC is often utilized as a video streaming service because of its low latency, which is achieved by using UDP as the transport protocol. This allows real-time communication with a latency of a few milliseconds to less than one second. Thus, it is suitable for many applications that require real-time performance, such as live streaming and robotic remote control [1]. In the case of live streaming services, resource management is an important issue because of the large number of users involved. However, in addition to resource management, it is also important to maximize user Quality of Experience (QoE); therefore, continuous real-time QoE monitoring is required.

Monitoring user scores is crucial for assessing Quality of Experience (QoE), as it is a subjective metric that depends on user evaluation. However, obtaining consistent feedback from users regarding QoE can be quite expensive because it requires specific environments and human resources. ITU-T/R recommendations are specified for various services that are necessary for subjective assessment. Since service providers aim for objective assessments to estimate QoE and ensure optimal QoE for streaming services, various objective QoE estimation models have been studied for different applications and architectures [2]. For example, ITU-T P.1203 [3] is a standardized model for this purpose. However, most of these models focus on TCP-based streaming services such as HLS, DASH, and HTTP adaptive streaming (HAS). Although many studies have been conducted on QoE estimation for WebRTC, most of them assume videoconferencing and do not consider high-quality video streaming. Among the objective QoE estimation models, Full-Reference (FR) models correlate highly with subjective evaluations, whereas No-Reference (NR) models have real-time performance but lower correlation because they cannot access the full information of the video [4]. However, since live streaming services require Real-time QoE monitoring, an NR model is required. WebRTC statistics have been shown to be effective as non-referenced metrics for QoE estimation [5].

In this paper, we propose a no-reference model that can estimate QoE in real-time from WebRTC statistics using a neural network trained with VMAF [6] as the ground truth, which is well correlated with subjective evaluation as an objective QoE evaluation model. Evaluation experiments showed that the proposed NR model outperformed the Root Mean Squared Error (RMSE) and Pearson Correlation Coefficient (PCC) values of the existing model. Contributions of this study are as follows.

- We created a dataset for WebRTC video streaming that includes WebRTC statistics and MOS of VMAF.
- We show that the estimation accuracy is improved when the resolution and frame rate of the reference video are

provided as inputs to the model.

- The proposed method uses a simple Deep Neural Network (DNN) and achieves 0.97 in PCC with 6.59K Floating-Point Operations (FLOPs).

The remainder of this paper is organized as follows. Section II describes the related studies and techniques. Section III describes the proposed model, including the dataset and QoE estimation model. The accuracy of the proposed model is evaluated in Section IV. Finally, Section V provides a summary and discusses future work.

II. Related works

A. Video Quality Assessment for QoE

Video quality assessment can be divided into two categories, depending on the subject being evaluated. The International Telecommunication Union (ITU) has standardized various quality assessment models for subjective and objective in ITU-T/R recommendations. Each model is described in relation to a real-time streaming service using WebRTC in the following.

1) Subjective Video Quality Assessment for QoE: Subjective assessment is performed by watching a video and evaluating the quality of the user's experience. Several evaluation models have been standardized, depending on the type of media, such as TV videos and multimedia applications, and specifications for experimental environments and evaluation videos have been defined [7]. However, subjective assessment requires a lot of time and human resources and is costly because of the need to implement a specified environment. In addition, in cases such as real-time streaming services, where QoE must be constantly monitored, users must constantly provide feedback on QoE. Therefore, it is difficult to implement such a system because it requires constant operation by the user while watching the video.

2) Objective Video Quality Assessment for QoE: Objective assessment is a method for evaluating the Quality of Experience (QoE) with metrics measured as Quality of Service (QoS) metrics in a system. Unlike subjective evaluations that rely on user scores, objective assessments use information from videos (e.g., reference and distorted video, resolution, and frame rate) and networks (e.g., jitter, packet loss rate, and bit rate) to make quality estimations. Additionally, these metrics to estimate QoE are collected in real-time compared to subjective assessments. Therefore, objective assessments play an important role in the evaluation of QoE in resource management of streaming services. The three models are as follows.

- Full-Reference (FR) model: FR models estimate QoE by comparing the reference videos in the streamer and distorted videos in the viewer. Thus, the FR model has the highest estimation accuracy among the three models because the FR model uses all the video information. However, it is not suitable for real-time QoE estimation because the FR model needs not only to obtain the reference videos before distortion but also large computational resources are required for video comparison.
- Reduced-Reference (RR) model: In contrast to FR models, RR models do not use the reference videos as is, but use the features extracted from the reference videos and distorted videos for QoE estimation. This model is also unsuitable for real-time QoE estimation because RR requires features related to the original video.
- No-Reference (NR) model: NR models do not use any knowledge of the original video but only the information available on the viewer side to estimate QoE. Therefore, it is suitable for real-time streaming services that use QoE for the streaming controls because NR models estimate QoE based on information easily obtained in real-time, such as network information mentioned above.

Quality of Experience (QoE) is evaluated using the Mean Opinion Score (MOS) value, which is derived from the Absolute Category Rating (ACR) method specified in ITU-T recommendation P.910 [8]. The MOS value is rated on a 5-point scale, from 1 (bad) to 5 (excellent), based on subjective evaluation. Subjective assessments using objective metrics include clustering approach [9], [10] and regression approach [5], [11].

B. QoE in WebRTC

WebRTC is a promising technology for low-latency streaming and has been evaluated in terms of QoE. A study investigating quality assessment using FR models in WebRTC [12] reported various studies on QoE in WebRTC, and many studies have analyzed various Quality of Service (QoS) parameters such as throughput, jitter, packet loss, and bitrate. According to the NR model based on multiple regression using WebRTC statistics [5], four metrics of WebRTC statistics are used to estimate QoE. However, this study only focused on videoconferencing and did not consider high-quality video streaming with high resolution and frame rate.

C. VMAF

Video Multimethod Assessment Fusion (VMAF) [6] is an FR model developed by Netflix for media streaming services. It is a machine learning-based model that scores each frame ranging from 0 (lowest quality) to 100 (highest quality) by regression estimation using Support Vector Regression (SVR) with multiple image quality metrics. It is known to correlate better with subjective assessments than existing FR models, such as PSNR [13], SSIM [14], and MS-SSIM [15]. In a study on the mapping between VMAF scores and MOS values [11], an optimized mapping model is proposed for 15, 30, and 60 fps, which are commonly used in delivery services. The performance of this model in terms of estimation accuracy has been demonstrated to be superior to the results obtained using the video quality assessment model specified by ITU-T Recommendation P.1203 for TCP-based video streaming services [3].

D. QoE estimation model using machine learning

Regardless of FR or NR models, many models that use machine learning and deep learning have been proposed [2].

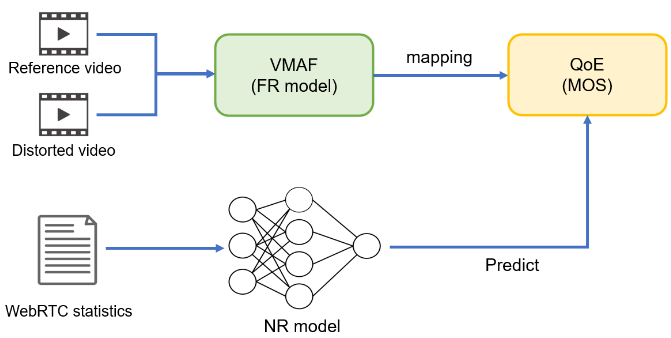


Fig. 1: Overview of the proposed model.

One of these, the NR model [16], which is based on a DNN and uses 89 network parameters as inputs, outperforms existing FR models. However, some of the input parameters, such as stalls and overall streaming score, are not obtained in real-time, and there is potential for further research on real-time evaluations.

III. Proposed Model

The proposed model utilizes deep learning with WebRTC statistics as input along with QoE measured through MOS values derived from VMAF, which has a high correlation with subjective assessments. The MOS values are used as training data to estimate the QoE objectively in real-time and with high accuracy. As shown in Fig. 1, the upper part of the figure displays the process of calculating the VMAF score, which is an FR model that uses two recorded videos, a reference video, and a distorted video. The videos must have the same resolution and number of frames, but if there are missing frames or freezes owing to network factors, the distorted video on the viewer side may be affected. In this study, a text box with a frame number corresponding to each frame of the reference video is drawn in advance, and the video is processed using the system described in [12] to match the number of frames between the original and distorted videos. The computed VMAF score, which ranges from 0 to 100, is converted to a MOS value ranging from 1 to 5 using a model in reference [11]. The lower part of Fig. 1 shows the QoE estimation model, which is built by training a neural network that takes WebRTC statistics as input and estimates the QoE per second generated in the process shown in the upper part.

A. Environment for data collection

An overview of the dataset collection system is shown in Fig. 2. The system uses the QoE measurement system of Elastest [17], an open-source platform for end-to-end testing of various WebRTC applications used in the study [12]. A videoconferencing demo application, AppRTC [18] is used to perform P2P communication via WebRTC.

First, after starting AppRTC on the Docker container, launch two Google Chrome browsers on the local machine and connect them to the AppRTC server to exchange Session Description Protocol (SDP) and Interactive Connectivity Establishment (ICE), which are necessary for establishing WebRTC

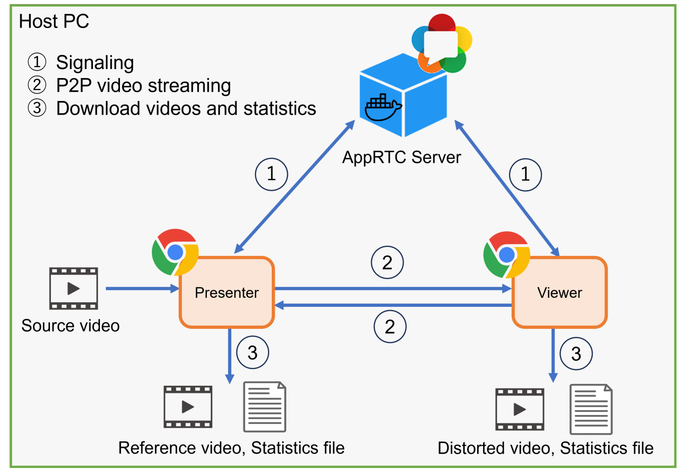


Fig. 2: Data collection system.

TABLE I: Configuration for data collection.

Video	Animation [19]
Codec	VP8
Resolution (16:9)	720p, 1080p
Frame rate (fps)	15, 30, 60
Duration (s)	10
Packet loss rate (%)	0, 15, 30, 45
Jitter (ms)	0, 25, 50, 75
Bandwidth (Mbps)	6, 5, 4, 3, 2, 1, 0.75, 0.5, 0.25

P2P communication (①). SDP contains information about available parameters, such as codecs, bitrate, IP addresses, and port numbers. By exchanging the SDP, browsers set the content for P2P communication. The ICE contains information on the communication route candidates between the two browsers and is registered with each browser to establish communication. Once the browsers are connected, P2P video streaming begins (②). At this time, the port number of the viewer's browser receiving the stream is extracted, and "tcconfig" is used to apply the various network conditions listed in Table I. "tcconfig" is a Python wrapper for the "tc command" and can easily control various network parameters such as bandwidth, latency, and packet loss.

In this system, the browser functions as a presenter on the sender side, and the other browser functions as a viewer on the receiver side. The presenter encodes the source video and sends it to the viewer, which utilizes Google Chrome's Chrome Option to stream the video file as a camera image. While streaming is in progress, the MediaRecorder API is used to record the streams of both the presenter and the viewer, and WebRTC statistics are collected every second using WebRTC's getStats API (③). Once the stream is completed, the videoconference is terminated, and the recordings and statistics are downloaded. The dataset is created by processing the recordings and WebRTC statistics collected during this process. Note that if bandwidth limitation or packet loss occurs during streaming, Google Congestion Control (GCC) dynamically changes the bitrate, resolution, and frame rate.

B. Dataset

1) Data processing: The previous section involved collecting recordings to determine the VMAF scores. Frame-by-frame models used for estimating QoE face the challenge of handling missing frames and distorted videos. The ElasTest system, which calculates QoE, includes video preprocessing to address these issues and calculates the VMAF score by aligning the resolution and number of frames. The VMAF score is calculated for each frame and then converted to a scale from 1 to 5 MOS values, as the VMAF score is expressed as a value from 0 to 100. The following equation from the study [11] is used to convert the VMAF score to MOS values:

$$MOSfromVMAF = a_i + b_i VMAF + c_i VMAF^2 \quad (1)$$

Each coefficient is optimized for 15, 30, and 60 fps.

Next, preprocessing of the WebRTC statistics is required. The WebRTC statistics collected in the previous section are JSON-format data containing various types of statistics. The WebRTC statistics used in this paper are 44 of the type "inbound-rtp." These statistics include statistics regarding the receiving stream, such as the jitter, packet loss, resolution, and frame rate. Note that this statistic can be obtained from the viewer. First, statistics such as ID and SSRC, which are considered unrelated to the estimation of QoE, are removed. Then, the accumulated statistics are recalculated into one-second values.

2) Configuration for data collection: The parameters utilized for data collection are presented in Table I. The source video employed is an animation that is segmented into 10-second clips. The network conditions are based on those used in previous studies [11], [12]. Three network parameters are changed to control the packet loss rate, jitter, and bandwidth. The jitter is set with a base delay of 100 ms, and the bandwidth condition is established by selecting five or six levels for each resolution and frame rate. Data are collected separately for each condition and a combination of jitter and bandwidth limitations. A total of five streaming sessions are conducted for each condition, resulting in 8,100 seconds of collected data in total, with 50 seconds of data recorded for each condition.

C. QoE estimation model

We propose a QoE estimation model that utilizes a DNN with the same architecture as the NR model [16], which achieves the same level of accuracy as the FR models. The hidden layer of the DNN comprises three layers, with 12, 48, and 48 nodes, respectively, and the model outputs a single estimated QoE value at the output layer. The number of trainable parameters is 3337 and the activation function is a Rectified Linear Unit (ReLU) function, with the loss function being the Mean Absolute Error (MAE). The WebRTC statistics provided as inputs to the model have completely different scales for each parameter, requiring pre-processing before input into the model. Specifically, the jitter and bytes received per second have a scale difference of approximately 10^8, and preprocessing is necessary to standardize the data, scaling the mean to zero and the variance to one.

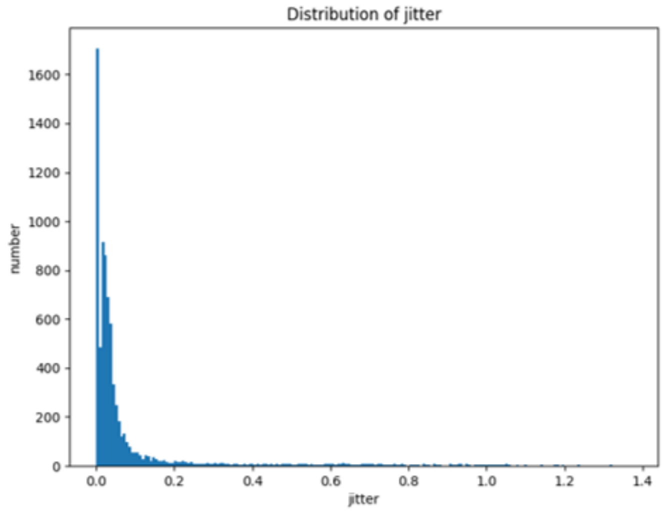


Fig. 3: The distribution of jitter values.

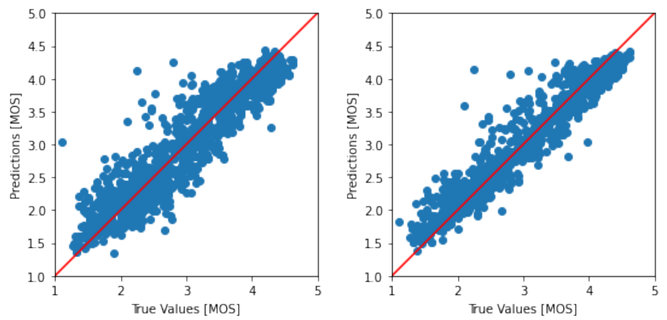
In addition, Spearman's rank correlation coefficients and p-values are calculated between the MOS values and each parameter to identify inputs that were considered uncorrelated. Only parameters for which the p-value is less than 0.05 are selected as inputs to the model.

When creating training data for neural networks, outliers must be handled carefully. Although the interquartile range (IQR) can be used to remove outliers, it may result in the deletion of more than half of the dataset if one outlier is present. Therefore, outliers are not removed in this study, and detailed examinations are required to determine whether they are true outliers.

For example, consider the histogram of the jitter in the dataset shown in Fig. 3. It can be seen from the figure that most of the jitter values in the dataset are concentrated around $0.0 - 0.1$. However, the data are also widely distributed in the range above 0.1. Therefore, if IQR is used, a large number of jitter values are eliminated as outliers, and whether jitter values above 0.1 are appropriate as outliers depends on the network and device. However, the jitter value is related to many network conditions such as bandwidth and congestion and their combination can also cause a high jitter value. According to the research on the application of WebRTC to in-vehicle communication [20], it shows that jitter values greater than 0.1 seconds may well be observed. After preprocessing, the DNN model is trained using 24 different WebRTC statistics. The dataset is split at a ratio of 8:2 and is used for training and testing. The initial learning rate is set to 0.001, and 100 epochs are trained with a batch size of 32.

IV. Results

To evaluate the estimation accuracy of the trained models on the test data, the proposed model is evaluated using the Root Mean Squared Error (RMSE) and Pearson Correlation Coefficient (PCC), which are commonly used in regression

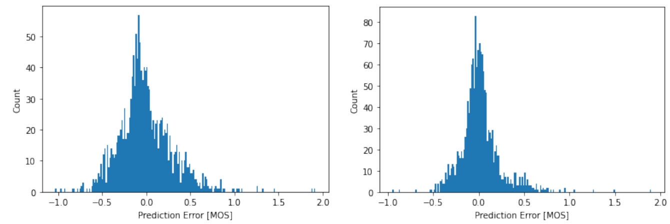


(a) 24 parameters of "inbound-rtp" as input. **(b)** 26 parameters of "inbound-rtp" and "media-source."

Fig. 4: Estimation scatter plot of each model.

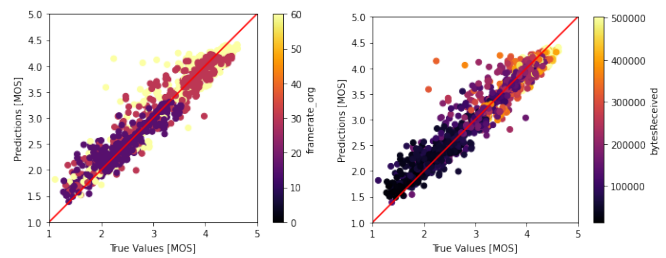
TABLE II: RMSE and PCC by model.

Parameters	RMSE	PCC
24	0.2926	0.9443
26	**0.2198**	**0.9702**



(a) 24 parameters of "inbound-rtp" as input. **(b)** 26 parameters of "inbound-rtp" and "media-source."

Fig. 5: The distribution of estimation error



(a) Frame rate of the reference video **(b)** Bytes received per second

Fig. 6: Estimation scatter plot colored by statistics

models to evaluate the estimation accuracy of the trained models on the test data. Two models were trained. One model was trained with the WebRTC statistics included in the "inbound-rtp" described in Section III as input, and the other model was trained with the resolution and frame rate information of the reference video included in the "media-source" type statistics. Note that this statistic can be obtained from the presenter.

A. Accuracy of proposed model

Figure 4 shows the estimation scatterplot of the model trained from the WebRTC statistics contained in "inbound-rtp." The vertical axis represents the estimations and the horizontal axis represents the correct QoE score. (a) shows the estimation of the model with 24 parameters, and (b) shows the estimation of the model with the resolution and frame rate added to the input before they are changed by GCC. The RMSE and PCC values of each model are listed in Table II.

The results from Fig. 4 show that the model that only uses "inbound-rtp" statistics has a slightly skewed distribution from the center line, with several data points that are significantly different from the estimations. Eleven of these data points had errors of ± 1 or higher. However, the model that added the resolution and frame rate of the source video to the input exhibited less error variation, with only four estimation errors of ± 1 or higher. Additionally, the latter model achieved higher accuracy for both RMSE and PCC, as shown in Table II. This is because even the same resolution and frame rate values can have different meanings, depending on whether they are the result of degradation or the original values. Adding the original media information to the input is considered to have improved the accuracy because the model can learn the differences in dynamic changes. Additionally, the accuracy of the NR model is better than the RMSE value (0.8686) of the same NR model achieved in the previous study [16]. Although a direct comparison is not possible owing to the difference in the dataset and data volume, the proposed model enables direct estimation of QoE from WebRTC statistics, even for UDP-based WebRTC streaming.

B. Analysis of the model

The histograms in Fig. 5 depict the distribution of the estimation error for each model. The one in (a) shows that the error distribution is slightly shifted to the left, with an error center of 0.0, indicating that the model slightly underestimates QoE. In (b), the error distribution is more evenly centered around 0.0, suggesting that this model has a smaller error than (a) and does not overestimate or underestimate the QoE. This indicates that incorporating information about the original video in the "media-source" can improve accuracy and produce an unbiased estimation model by allowing comparisons that consider changes in the stream due to statistics, such as FR and RR models, even though it is an NR model.

Next, we examined the test data with large estimation errors based on a model that incorporated the original media information to enhance the accuracy. As shown in Fig. 4(b), a scatter plot is colored according to the parameter values. Fig. 4(a) displays the original frame rate of each test data, while Fig. 4(b) shows the data colored by the number of bytes received. From Fig. 4(a), it can be seen that most of the large errors occur at 60 fps, whereas from Fig. 4(b), it can be seen that the data are collected at a bit rate of approximately $300,000 - 400,000$ Bytes/s, that is, approximately $2.4 - 3.2$ Mbps. This indicates that at least 4 Mbps is required for delivery at 60 fps over 720p, suggesting that a large error occurs when quality degrades due to network constraints during high-quality streaming. In particular, packet loss is not combined with other network conditions during data collection, and streams that include more than 15 % of packet losses are approximately 10 % of the dataset. Moreover, the fact that only a small amount of packet loss occurs when jitter occurs suggests that the effects of packet loss may not have been fully learned. In addition, because each plot in Fig. 4, 6 shows the estimated QoE for one second, a few values have a significantly high estimation error and it is possible that some parameters contain outliers that cause large errors. Therefore, it can be affected that the estimation can be improved by collecting more data and performing appropriate outlier processing.

The computational complexity is evaluated using FLOPs and calculated by keras-flops [21]. The FLOPs of the proposed model is 6.59K. This indicates that the computational complexity of the proposed model is lower than that of FR models that use large neural networks [22], [23].

V. Conclusion

We created a dataset that includes the QoE of the FR model and WebRTC statistics for real-time QoE estimations during WebRTC streaming under various conditions. Using this dataset, we trained DNN models and found that providing not only "inbound-rtp" statistics but also source media information that changes dynamically based on network conditions improves model accuracy. Our results show that the deep learning-based NR model with WebRTC statistics is effective in estimating QoE for WebRTC streaming with higher RMSE and PCC values than those in existing studies. The proposed model exhibits low computational complexity, making it suitable for real-time monitoring of large-scale video streaming systems. However, because the dataset only includes animation currently, it is necessary to add different types of videos in the training to enhance the generality of the proposed model. Additionally, it is important to optimize parameters such as the number of layers and nodes in the DNN and compare its practicality and accuracy with those of existing machine learning models in terms of processing latency, computational cost, and other factors.

References

[1] A. Tiberkak, A. Hentout, and A. Belkhir, "Webrtc-based mosr remote control of mobile manipulators," *International Journal of Intelligent Robotics and Applications*, vol. 7, pp. 1–17, 04 2023.

[2] G. Kougioumtzidis, V. Poulkov, Z. D. Zaharis, and P. I. Lazaridis, "A survey on multimedia services qoe assessment and machine learning-based prediction," *IEEE Access*, vol. 10, pp. 19 507–19 538, 2022.

[3] "Parametric bitstream-based quality assessment of progressive download and adaptive audio-visual streaming services over reliable transport," ITU-T Recommendation P.1203, 10 2017.

[4] N. Barman and M. G. Martini, "Qoe modeling for http adaptive video streaming–a survey and open challenges," *IEEE Access*, vol. 7, pp. 30 831–30 859, 2019.

[5] J. B. Husić, A. Alić, S. Baraković, and M. Mrkaja, "Qoe prediction of webrtc video calls using google chrome statistics," in *2021 20th International Symposium INFOTEH-JAHORINA (INFOTEH)*, 2021, pp. 1–6.

[6] Netflix. (2016, June) Toward a practical perceptual video quality metric. [Online]. Available: https://netflixtechblog.com/toward-a-practical-perceptual-video-quality-metric-653f208b9652

[7] J. Okamoto and T. Hayashi, "Latest trends in image media quality assessment technologies," *IEICE Fundamentals Review*, vol. 6, no. 4, pp. 276–284, 2013.

[8] "Subjective video quality assessment methods for multimedia applications," ITU-T Recommendation P.910, 9 1999.

[9] I. Orsolic, D. Pevec, M. Suznjevic, and L. Skorin-Kapov, "A machine learning approach to classifying youtube qoe based on encrypted network traffic," *Multimedia Tools and Applications*, vol. 76, p. 22267–22301, 11 2017.

[10] I. Orsolic, D. Pevec, M. Suznjevic, and L. Skorin-Kapov, "Youtube qoe estimation based on the analysis of encrypted network traffic using machine learning," in *2016 IEEE Globecom Workshops (GC Wkshps)*, 2016, pp. 1–6.

[11] K. Yamagishi, N. Egi, N. Yoshimura, and P. Lebreton, "Derivation procedure of coefficients of metadata-based model for adaptive bitrate streaming services," *IEICE Transactions on Communications*, vol. E104.B, pp. 725–737, 1 2021.

[12] B. García, F. Gortázar, M. Gallego, and A. Hines, "Assessment of qoe for video and audio in webrtc applications using full-reference models," *Electronics*, vol. 9, no. 3, p. 462(article number), 2020. [Online]. Available: https://www.mdpi.com/2079-9292/9/3/462

[13] Q. Huynh-Thu and M. Ghanbari, "Scope of validity of psnr in image/video quality assessment," *Electronics Letters*, vol. 44, pp. 800 – 801, 02 2008.

[14] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[15] Z. Wang, E. Simoncelli, and A. Bovik, "Multiscale structural similarity for image quality assessment," in *The Thirty-Seventh Asilomar Conference on Signals, Systems Computers, 2003*, vol. 2, 2003, pp. 1398–1402 Vol.2.

[16] X. Tao, Y. Duan, M. Xu, Z. Meng, and J. Lu, "Learning qoe of mobile video transmission with deep neural network: A data-driven approach," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1337–1348, 2019.

[17] Elastest home. [Online]. Available: https://elastest.io/

[18] Apprtc demo code. [Online]. Available: https://github.com/webrtc/apprtc

[19] T. Roosendaal, "Big buck bunny," in *ACM SIGGRAPH ASIA 2008 Computer Animation Festival*, ser. SIGGRAPH Asia '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 62. [Online]. Available: https://doi.org/10.1145/1504271.1504321

[20] N. Kaneko, T. Ito, H. Katsuda, T. Watanabe, H. Abe, and R. Onishi, "Applying and evaluating multipath redundant communication technology for webrtc-based video streaming," July 2022. [Online]. Available: https://www.ipsj.or.jp/dp/contents/publication/51/TR0303-03.html

[21] "keras-flops," https://github.com/tokusumi/keras-flops.

[22] X. Liu, G. Chuai, X. Wang, Z. Xu, and W. Gao, "Qoe assessment model based on continuous deep learning for video in wireless networks," *IEEE Transactions on Mobile Computing*, vol. 22, no. 6, pp. 3619–3633, 2023.

[23] T. N. Duc, C. T. Minh, T. P. Xuan, and E. Kamioka, "Convolutional neural networks for continuous qoe prediction in video streaming services," *IEEE Access*, vol. 8, pp. 116 268–116 278, 2020.