Machine Learning-Based No-Reference QoE Estimation for Scalable Real-time Video Streaming

Yuki Kendo
The University of Electro-Communications
Chofu, Tokyo, Japan
Email: y.kendo@net.lab.uec.ac.jp

Ryunosuke Sato
The University of Electro-Communications
Chofu, Tokyo, Japan
Email: r.sato@net.lab.uec.ac.jp

Satoshi Ohzahata
The University of Electro-Communications
Chofu, Tokyo, Japan
Email: ohzahata@uec.ac.jp

Ryo Yamamoto
The University of Electro-Communications
Chofu, Tokyo, Japan
Email: ryo-yamamoto@uec.ac.jp

***Abstract*—WebRTC SFU is a scalable, low-latency video delivery technology that transmits video streams. In real-time streaming services, implementing real-time QoE estimation methods is essential to maintain a certain level of user-perceived quality. In this paper, we propose a machine learning-based method for estimating no-reference QoE in WebRTC SFU, addressing the challenge of handling diverse video content types and scalability. To adapt to content diversity, we collect video data by considering motion scores and integrate different content types into a combined dataset. In the evaluation, the model trained on the combined dataset demonstrated better performance in estimating QoE for each content type than the models individually trained for each content type. For scalability, CatBoost is capable of performing approximately 10.22 million inferences per second, and the proposed method shows real-time performance for large-scale content distribution.**

***Index Terms*—QoE, Video Quality Assessment, WebRTC, Machine Learning**

I. Introduction

In recent years, real-time video streaming services have become increasingly popular. Web Real-Time Communication (WebRTC) [1], [2] is widely adopted in applications such as interactive video conferencing. To support large-scale real-time video distribution scenarios, a client-server architecture known as the WebRTC Selective Forwarding Unit (SFU) [3] has been used. In a WebRTC SFU system, since the SFU server forwards the received media streams from the sender to the receivers without re-encoding, this architecture enhances the scalability of WebRTC-based real-time video distribution systems. However, in the case of congestion, the available bandwidth becomes lower than the video bitrate, and frame losses or playback freezes occur on the receiver side. To address this issue, Simulcast has been proposed [4]. In Simulcast, the sender transmits multiple versions of the encoded video stream at different quality levels to the SFU, which then selects the most appropriate stream for each receiver based on the estimated available bandwidth. This approach enables adaptive real-time video distribution in dynamic network conditions.

As the scale of users in real-time video streaming services increases, efficient resource management while maintaining the Quality of Experience (QoE) of users becomes a critical challenge. QoE is a metric that reflects users' subjective perception of media quality based on their viewing experience [5]. Ideally, QoE should be measured using real-time user feedback; however, continuous manual feedback is impractical. Furthermore, subjective evaluation requires dedicated environments and human resources, resulting in high costs. Therefore, in the context of streaming services, QoE is estimated using objective indicators that the service provider can monitor and measure automatically [6].

In [7], a machine learning-based no-reference (NR) QoE estimation method for a reliable transport protocol is proposed, which estimates the Mean Opinion Score (MOS) using network statistics. Although this model employs ITU-T P.1203 values as the target variable, since P.1203 assumes a reliable transport protocol, the model is inherently unsuitable for real-time communications. [8] proposed an NR QoE estimation method for WebRTC peer-to-peer (P2P) communication, where VMAF was employed as the target and QoE was estimated using receiver-side network statistics. However, because the model was developed using data collected only in a P2P environment, it is not suitable for large-scale real-time distribution scenarios such as WebRTC SFU. In addition, these previous works did not consider differences in video properties, such as motion intensity, which biases the model performance toward specific video characteristics.

To address these issues, this paper proposes a no-reference QoE estimation model for real-time video streaming that considers different content types using machine learning. The proposed method also uses WebRTC receiver-side statistics as input features, and QoE is estimated as the output. To create a dataset with diverse content in a motion score, we collect data from three types of video content in a WebRTC SFU environment. The diversity of the motion characteristics is quantitatively validated using the VMAFmotion metric [9], which captures the degree of motion intensity within the video sequences. By selecting content with low, medium, and high VMAFmotion values, we constructed datasets that represent a

wide range of motion dynamics. We then trained the model and evaluated its accuracy using these data sets.

The key contributions of this study are as follows:

- We constructed a WebRTC QoE dataset under an SFU environment using three different types of video content, capturing both WebRTC statistics and subjective quality metrics based on the Full Reference (FR) model Video Call MoS (VCM) [10].
- We established a QoE evaluation framework for WebRTC SFU-based streaming, where the highest-quality original video (3000 kbps) was used as the reference. Transmitted videos with resolution switching were resized to the reference resolution using FFmpeg for a fair comparison.
- We evaluated that a model trained on a dataset composed of different content types outperforms models trained on single-content datasets, indicating better generalization across diverse video content.
- From a scalability perspective, we showed that CatBoost can estimate QoE at approximately 10.22 million inferences per second, while XGBoost achieves around 2.38 million inferences per second, highlighting their applicability to real-time scenarios.

II. Related Work

Video quality assessment methods are classified into two categories: subjective and objective. Subjective assessment refers to methods in which users watch videos and evaluate their perceived quality. Although evaluation procedures and experimental environments have been standardized [11], such methods require significant time and cost, making them difficult to apply in scenarios requiring continuous QoE monitoring, such as real-time streaming services. In contrast, objective assessment estimates video quality using information related to the video or network without requiring user evaluation. Objective assessment plays an essential role in the management of real-time streaming services.

ITU-T Recommendation J.143 [12] outlines three types of objective assessment methods for QoE. Full Reference (FR) and Reduced Reference (RR) methods estimate video quality using either the original video or its extracted features in comparison with the received video. The Video Multimethod Assessment Fusion (VMAF) [13], developed by Netflix, is a full-reference (FR) quality assessment model that employs machine learning to fuse multiple objective quality metrics. To improve its performance, the VCM [10] extends VMAF by integrating temporal features, enabling a more accurate estimation of the Mean Opinion Score (MOS) values. Experiments have shown that the VCM outperforms the VMAF in QoE prediction. However, because both methods involve complex image analysis, their processing times are substantial, making them unsuitable for real-time applications.

On the other hand, No Reference (NR) methods estimate quality based on information available on the receiver side, such as network statistics. Reference [7] proposed a machine learning-based NR QoE estimation method that predicts MOS values using only network information, with ITU-T P.1203 values serving as the reference. The evaluation results show that the method achieved $R^2 = 0.968$ with Random Forest, enabling inference of 4,000 samples per second, and $R^2 = 0.65$ with Linear Regression, capable of inferring 1.1 million samples per second. However, the study does not discuss how the video content is selected for the dataset, and it is unclear whether the model can handle the variability in network statistics caused by content characteristics such as motion intensity. In [8], a deep-learning-based NR model for WebRTC P2P is proposed. In this model, MOS values derived from VMAF using a mapping function [14] are used as the target variable, and WebRTC statistical information is used as input to estimate QoE. This method outperforms existing NR models in terms of estimation accuracy. However, the use of a P2P-based environment for data collection limits its applicability to WebRTC SFU-based streaming systems.

III. Proposed Method

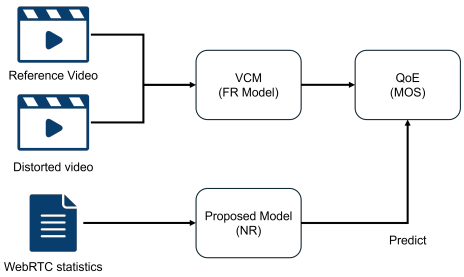

Fig. 1. Proposed Method.

A. Overview

In this section, we propose a QoE estimation method for real-time video streaming, building upon the prior work presented in [8], focusing on supporting diverse video content using machine learning. The proposed machine learning-based NR model takes WebRTC statistical metrics as input and uses the MOS values computed from the FR model VCM as training labels to consider video freezing or skipping. An overview of the proposed method is shown in Figure 1. In the upper part of the diagram, we calculate the QoE scores using VCM by processing recorded video files from both the sender and receiver sides because VCM requires both the original (reference) and degraded videos for comparison.

In the lower part of the diagram, the 1-second interval MOS values obtained from the upper process are used as the target variable, while the corresponding WebRTC statistical data serve as input features for training the NR QoE estimation model.

The dataset is collected in a video delivery environment using WebRTC SFU, where various network conditions are emulated using the 'tc' command [15] to ensure data diversity. After data collection, preprocessing is performed.

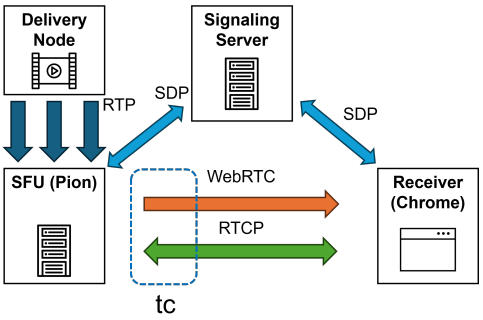

Fig. 2. Data Collection Environment.

B. Data Collection Environment

Figure 2 illustrates an overview of the dataset-collection system. The data collection environment is implemented using Pion [16], an open-source WebRTC stack written in Go. The connection between the SFU and the receiver client is established using WebRTC signaling procedure via the Session Description Protocol (SDP). The SFU receives media streams in the form of Real-time Transport Protocol (RTP) packets from the sender and forwards them to the receiver [17]. Additionally, both the sender and receiver exchange RTP Control Protocol (RTCP) packets for control and feedback purposes, as is standard in WebRTC. In the system, the quality adaptation of Simulcast is performed using the Google Congestion Control (GCC) congestion control algorithm. The receiver client uses Google Chrome as the WebRTC endpoint. On the sender side, the 'tc' command is used to emulate various network conditions by applying bandwidth limitations, packet loss, and delay parameters to the streaming traffic. These network conditions can lead to playback impairments, such as video freezes or frame skips, which are critical factors affecting the perceived QoE. During streaming for data collection, the incoming video stream of the receiver is recorded using the MediaRecorder API, and WebRTC statistical data are simultaneously collected at one-second intervals via the 'getStats' API. The dataset is constructed by combining the recorded video files for FR QoE estimation with the corresponding WebRTC statistics and 'tc' network settings for NR QoE model training/estimation.

C. Data Collection Conditions

1) Network Settings for Data Collections: Table I summarizes the network setting parameters used during dataset collection. The network conditions are configured using the 'tc' command in Linux [15]. Bandwidth limitations are implemented using the token bucket filter (TBF) mechanism provided by 'tc.' Additionally, the maximum queuing delay is set to 100 ms to emulate the buffer constraints under network congestion.

2) Streaming Videos Settings for Data Collections: Table II summarizes the video bit rates and resolutions used in the data collection environment. For the experiments, we employ three videos with 30 fps as content sources: an animation video [18], a conference keynote recording [19], and a sports broadcast [20]. To enable a frame-by-frame comparison during the QoE calculation using VCM, we embed QR codes containing the frame number in each video frame to identify the corresponding frames between the target variable and distorted videos.

TABLE I
NETWORK PARAMETER SETTINGS.

Communication Parameter	Value
Video Content	Animation [18], Conference [19], Sports [20]
Bandwidth Limit (kbps)	250–4000 (increments of 50)
Packet Loss Rate (%)	0–2 (increments of 0.1)
Network Delay (ms)	10, 50, 100

TABLE II
VIDEO BITRATE AND RESOLUTION OF STREAMING CONTENT.

Video Bitrate (kbps)	Resolution
3000	1920×1080
1500	1280×720
1000	854×480
500	640×360
250	426×240

To integrate data from content with different motion levels, we analyzed the VMAF motion scores [9] as an indicator of motion intensity. Table III presents the average and standard deviation of the VMAF motion scores for each content type at a bitrate of 3000 kbps. A higher VMAF motion score indicates content with an overall higher motion, while a larger standard deviation implies greater variation in motion intensity within the video. This metric helps identify different characteristics of the content type in terms of motion dynamics, which affect the performance of QoE estimation.

TABLE III
MEAN AND STANDARD DEVIATION OF VMAF MOTION SCORES.

Content Type	Mean	Standard Deviation
Animation	5.2755	4.6107
Conference	3.6092	0.5972
Sports	11.0189	7.5963

3) Preprocessing: We used 41 WebRTC statistics related to the receiver-side media stream, all of which are of the "inbound-rtp" type and obtained by the WebRTC environment. Among these, we excluded 12 fields, such as identifiers (e.g., ID, SSRC), which are considered unrelated to QoE estimation. In the preprocessing, the feature standardization is applied to ensure a fair comparison among models sensitive to feature scaling. We then processed the QoE values calculated using the VCM. Since the VCM outputs QoE scores on a per-frame basis and the video is encoded at 30 frames per second, we computed the average QoE score over every 30 frames to obtain per-second QoE values. QoE is typically assessed by comparing a degraded video with its original counterpart at the same resolution. However, in WebRTC SFU-based streaming, the video resolution is adaptively switched according to the network conditions, making direct comparison difficult. In this study, the original video with the highest resolution and bitrate (3000 kbps) is used as a reference. The transmitted

TABLE IV
EXPERIMENTAL ENVIRONMENT.

Component	Specification
Operating System	Ubuntu 24.04.2 LTS
CPU	Intel Core i9-10980XE @ 3.00GHz (30 cores)
GPU	NVIDIA GeForce RTX 3080
Memory	160 GB
Python Version	3.8.16
PyCaret Version	3.0.0
psutil Version	5.9.4
Memray Version	1.15.0

TABLE V
NUMBER OF DATA SAMPLES IN EACH DATASET FOR EXPERIMENT 3.

Dataset Name	Number of Samples
All_train	774,884
All_test	332,094
Animation_test	159,618
Conference_test	158,609
Sports_test	156,194

video with resolution switching was resized to the reference resolution using FFmpeg before the QoE evaluation. After this preprocessing, we train the machine learning model using 29 types of WebRTC statistical features and the QoE values calculated by VCM.

IV. EXPERIMENTAL SETTINGS

To assess the performance and content diversity robustness, the proposed method is evaluated for prediction accuracy and scalability. We created QoE estimation models using PyCaret [21], an AutoML library for Python that provides 26 different machine learning algorithms. From these algorithms, we select the top five models that achieve the highest prediction accuracy in the 10-fold cross-validation. These models are subjected to hyperparameter tuning and final testing. The dataset is split into training and test sets in a 70:30 ratio. Five machine learning models are selected for further evaluation: Random Forest Regressor (rf), Extra Trees Regressor (et), Extreme Gradient Boosting (xb), CatBoost Regressor (cb), Light Gradient Boosting Machine (lg). The experimental environment used for all evaluations is summarized in Table IV.

A. Accuracy Evaluation

We conducted three experiments as follows:

Experiment 1 evaluates the QoE estimation models trained by each content type. For this purpose, each model is trained on a dataset of each content type and evaluated using the same content type data as the training data.

Experiment 2 evaluates the generality of the models trained in Experiment 1. We evaluate each content-specific model to predict the QoE for datasets of other content types.

Experiment 3 investigates constructing a generalized model that applies to every content type. First, we extract 70 % of the training data from each content type and merge them into a single dataset. The remaining 30 % of each content type is used as individual test sets (Animation_test, Conference_test, and Sports_test). From the combined dataset, 70 % is used for training (All_train), and the remaining 30 % is used as a test (All_test). The data sizes of each dataset used in Experiment 3 are summarized in Table V. Here, All_train denotes the combined dataset used for model training, and All_test, Animation_test, Conference_test, and Sports_test are used for model evaluation.

The following metrics are used as indicators of the prediction accuracy: Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Coefficient of Determination (R^2). Smaller MAE, MSE, and RMSE values indicate better predictive performance. For R^2, values closer to 1 represent higher accuracy.

B. Scalability Evaluation

To assess the feasibility of deploying the proposed method in large-scale real-time prediction systems, we evaluate the estimation performance of the five machine learning models trained on the combined dataset in Experiment 3. We measure the inference time, memory usage, and CPU utilization per core. For each model, the number of input samples is fixed at 1,000,000, and the evaluation is repeated 100 times to ensure statistical reliability. In the evaluation, only the inference stage is measured; other operations, such as data preprocessing or model loading, are excluded from the measurements.

V. EXPERIMENTAL RESULTS

A. Accuracy Evaluation

1) Experiment 1: Accuracy Evaluation for Each Video Content Model: Table VI presents the performance evaluation results of the models for each video content. The superior performance of the bagging-based models (et, rf) can be attributed to their robustness against noise and outliers, as they independently train multiple decision trees and average the predictions. In contrast, boosting-based models (xb, cb, lg) iteratively learn from errors, which can make them more sensitive to outliers, possibly resulting in slightly lower accuracies. When comparing the model performance across different content types within the same algorithm, the prediction accuracy ranks highest for Conference, followed by Sports and Animation.

TABLE VI
PERFORMANCE EVALUATION OF EACH MODEL ON EACH CONTENT.

Content	Model	MAE	MSE	RMSE	R^2
Animation	et	0.1388	0.1220	0.3493	0.8667
Conference	et	0.0568	0.0537	0.2317	0.9233
Sports	et	0.1263	0.0900	0.3000	0.9005
Animation	rf	0.1387	0.1223	0.3497	0.8664
Conference	rf	0.0582	0.0555	0.2355	0.9207
Sports	rf	0.1277	0.0915	0.3025	0.8989
Animation	cb	0.1964	0.1342	0.3663	0.8553
Conference	cb	0.0646	0.0529	0.2301	0.9243
Sports	cb	0.1707	0.1003	0.3167	0.8891
Animation	xb	0.1864	0.1324	0.3639	0.8553
Conference	xb	0.0661	0.0559	0.2364	0.9201
Sports	xb	0.1653	0.0990	0.3167	0.8891
Animation	lg	0.2018	0.1377	0.3711	0.8495
Conference	lg	0.0662	0.0547	0.2338	0.9219
Sports	lg	0.1735	0.1018	0.3191	0.8875

TABLE VII
EVALUATION OF MODELS TRAINED ON ANIMATION DATA.

Test Content	Model	MAE	MSE	RMSE	R^2
Conference	et	0.2762	0.1557	0.3946	0.7759
Sports	et	0.3206	0.2146	0.4632	0.7628
Conference	rf	0.2803	0.1677	0.4095	0.7587
Sports	rf	0.3406	0.2359	0.4857	0.7392
Conference	cb	0.2683	0.1401	0.3743	0.7984
Sports	cb	0.3469	0.2321	0.4818	0.7434
Conference	xb	0.2726	0.1459	0.3820	0.7899
Sports	xb	0.3322	0.2278	0.4773	0.7482
Conference	lg	0.3026	0.1586	0.3982	0.7718
Sports	lg	0.3168	0.2126	0.4611	0.7650

TABLE VIII
EVALUATION OF MODELS TRAINED ON CONFERENCE DATA.

Test Content	Model	MAE	MSE	RMSE	R^2
Animation	et	0.5322	0.4515	0.6719	0.5066
Sports	et	0.7907	0.7949	0.8916	0.1212
Animation	rf	0.5366	0.4584	0.6771	0.4990
Sports	rf	0.7998	0.8098	0.8999	0.1047
Animation	cb	0.5309	0.4634	0.6807	0.4936
Sports	cb	0.8259	0.8559	0.9251	0.0538
Animation	xb	0.5265	0.4604	0.6807	0.4936
Sports	xb	0.8091	0.8253	0.9084	0.0876
Animation	lg	0.5187	0.4487	0.6698	0.5096
Sports	lg	0.7989	0.8108	0.9004	0.1036

TABLE IX
EVALUATION OF MODELS TRAINED ON SPORTS DATA.

Test Content	Model	MAE	MSE	RMSE	R^2
Animation	et	0.4049	0.3058	0.5530	0.6658
Conference	et	0.7569	0.6962	0.8344	-0.0022
Animation	rf	0.4432	0.3544	0.5953	0.6126
Conference	rf	1.0420	1.2347	1.1112	-0.7773
Animation	cb	0.3786	0.2826	0.5316	0.6911
Conference	cb	0.6160	0.5367	0.7326	0.2275
Animation	xb	0.3920	0.2950	0.5431	0.6776
Conference	xb	0.6437	0.5704	0.7553	0.1789
Animation	lg	0.3919	0.2859	0.5347	0.6875
Conference	lg	0.6732	0.5805	0.7619	0.1644

2) Experiment 2: Cross-Prediction Using Model Trained on Other Content Dataset: Tables VII, VIII, and IX show the prediction performance of the models trained in Experiment 1 when applied to different content type datasets. Overall, the prediction accuracy decreased when models are used to predict content different from what they are trained on, indicating limited generality for cross-content evaluation. One possible explanation is the difference in temporal dynamics between Conference and Sports content, as reflected by their VMAF motion scores. As shown in Table III, the average VMAF motion score for Sports is approximately 11.02 with a standard deviation of approximately 7.60, whereas for Conference, the average is approximately 3.60 with a standard deviation of approximately 0.60. This difference in both the mean and variability indicates a clear contrast in motion intensity between the two content types, which may have reduced the model's ability to generalize to other types of content datasets. In contrast, for the Animation evaluation dataset, models trained on Sports data show better accuracy than those trained on Conference data. This suggests that the motion characteristics of sports content are more similar to those of animation than to those of conferences, allowing models trained on sports data to better capture the dynamics of animation content.

TABLE X
MODEL PERFORMANCE ON ALL_TEST DATASET USING THE COMBINED (ALL) TRAINING DATA.

Model	MAE	MSE	RMSE	R^2
et	0.1217	0.0948	0.3078	0.9009
rf	0.1214	0.0952	0.3085	0.9004
cb	0.1775	0.1120	0.3346	0.8829
xb	0.1567	0.1022	0.3197	0.8931
lg	0.1674	0.1088	0.3298	0.8862

3) Experiment 3: Accuracy Evaluation with a Model Combined Dataset: Tables X and XI present the results of the accuracy evaluation when the training data consisted of a combined all three content types.

We first examine the results when the test part is the combined dataset (All_test), as shown in Table X. Compared to the results of the models trained on individual content types in Experiment 1, the models trained on the All dataset have a slightly lower accuracy than those trained specifically on Conference or Sports content, but better than those trained solely on Animation content. Next, we analyze the prediction performance of each content test dataset using the model trained on the combined dataset, as shown in Table XI. Compared with the results of Experiment 2, the overall accuracy improved across all content types, suggesting that the combined model trained on diverse content is better at generalizing and adapting to different video characteristics than the models trained on a single content type.

As shown in Table III, the VMAF motion scores for Conference, Animation, and Sports exhibit increasing average motion levels of approximately 3.61, 5.28, and 11.02, respectively, with corresponding standard deviations of approximately 0.60, 4.61, and 7.60. This indicates that the combined dataset incorporates both low-motion and high-motion content, as well as content with varying degrees of motion fluctuation. Such diversity likely helped the combined model to better capture the relationship between motion intensity and changes in the WebRTC statistics, thereby improving the prediction accuracy.

B. Scalability Evaluation

The results of the scalability performance evaluation are shown in Table XII.

a) Inference Time: CatBoost achieved the shortest inference time of 97.78 ms for 1,000,000 input samples, whereas the Extra Trees Regressor (ET) required 1516.06 ms, which is approximately 11 times slower than CatBoost. CatBoost is capable of performing approximately 10.22 million inferences per second.

b) Memory Usage: XGBoost used only 4.36 MB at 1,000,000 input samples, whereas both ET and RF consumed approximately 1.77 GB, indicating a significant difference in memory efficiency between the three models.

TABLE XI
MODEL PERFORMANCE ON EACH CONTENT'S TEST DATA USING THE COMBINED (ALL) TRAINING DATA.

Content	Model	MAE	MSE	RMSE	R^2
Animation	et	0.1488	0.1256	0.3545	0.8626
Conference	et	0.0669	0.0569	0.2386	0.9180
Sports	et	0.1383	0.0966	0.3108	0.8934
Animation	rf	0.1491	0.1271	0.3566	0.8609
Conference	rf	0.0678	0.0582	0.2413	0.9162
Sports	rf	0.1380	0.0983	0.3134	0.8916
Animation	cb	0.2298	0.1525	0.3905	0.8332
Conference	cb	0.0920	0.0620	0.2490	0.9107
Sports	cb	0.2047	0.1207	0.3474	0.8668
Animation	xb	0.2049	0.1391	0.3729	0.8479
Conference	xb	0.0771	0.0571	0.2389	0.9178
Sports	xb	0.1837	0.1101	0.3318	0.8785
Animation	lg	0.2153	0.1465	0.3828	0.8397
Conference	lg	0.0940	0.0610	0.2469	0.9123
Sports	lg	0.1961	0.1169	0.3418	0.8710

c) CPU Utilization: CatBoost used 59.81 % CPU, while XGBoost reached 92.79 %, making it the most CPU-intensive model among those evaluated.

TABLE XII
SCALABILITY EVALUATION RESULTS.

Model	Data Size	Time (ms)	Memory (MB)	CPU (%)
catboost	1,000,000	97.78	8.12	59.81
et	1,000,000	1516.06	1765.38	84.75
lightgbm	1,000,000	496.84	1196.03	78.94
rf	1,000,000	1309.96	1765.38	82.88
xgboost	1,000,000	420.32	4.36	92.79

VI. CONCLUSION

In this paper, we propose a machine learning construction method to estimate the Quality of Experience (QoE) in a scalable manner using WebRTC statistical metrics. To build a machine learning model, we collected datasets comprising three different types of video content in a Simulcast of WebRTC SFU environment. The evaluation of the constructed models revealed that incorporating content with diverse levels and patterns of motion into the training dataset significantly enhanced the model's overall performance and generality. In addition, the scalability evaluation revealed that the CatBoost model achieved the best real-time performance and scalability among the five models, with an inference time of approximately 100 ms, memory usage of approximately 8 MB, and CPU utilization of approximately 60 % for 1,000,000 input samples. To improve the accuracy and deployability of QoE estimation models, our future work should focus on expanding the diversity and balance of video content, as well as applying feature selection and outlier handling to reduce noise and enhance efficiency.

REFERENCES

[1] W3C Web Real-Time Communications Working Group, "WebRTC: Real-Time Communication in Browsers," W3C Recommendation, World Wide Web Consortium, Mar. 2025.

[2] H. T. Alvestrand, "Overview: Real-Time Protocols for Browser-Based Applications." RFC 8825, Jan. 2021.

[3] B. Grozev, L. Marinov, V. Singh, and E. Ivov, "Last N: relevance-based selectivity for forwarding video in multimedia conferences," in *Proceedings of ACM*, NOSSDAV '15, pp. 19–24, 2015.

[4] B. Grozev, G. Politis, E. Ivov, T. Noel, and V. Singh, "Experimental evaluation of simulcast for webrtc," *IEEE Communications Standards Magazine*, vol. 1, no. 2, pp. 52–59, 2017.

[5] 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. 19507–19538, 2022.

[6] B. Argin, M. Demir, E. D. Salik, A. G. Önalan, H. Batum, and E. G. Soyak, "Advancing webrtc qoe assessment with machine learning in real-world wi-fi scenarios," in *IEEE MeditCom 2024*, pp. 263–268, 2024.

[7] P. H. S. Panahi, A. Hossein Jalilvand, and A. Diyanat, "An efficient network-based qoe assessment framework for multimedia networks using a machine learning approach," *IEEE Open Journal of the Communications Society*, vol. 6, pp. 1653–1669, 2025.

[8] K. Sakakibara, S. Ohzahata, and R. Yamamoto, "Deep learning-based no-reference video streaming qoe estimation using webrtc statistics," in *2024 International conference on artificial intelligence in information and communication (ICAIIC)*, pp. 365–370, IEEE, 2024.

[9] FFmpeg Developers, "vmafmotion filter — ffmpeg documentation." Web page, 2025. Accessed: 1 Aug. 2025. Available: https://ffmpeg.org/ffmpeg-all.html#vmafmotion.

[10] G. Mittag, B. Naderi, V. Gopal, and R. Cutler, "Lstm-based video quality prediction accounting for temporal distortions in videoconferencing calls," in *ICASSP 2023*, pp. 1–5, 2023.

[11] International Telecommunication Union, ITU-T, "Subjective video quality assessment methods for multimedia applications," Recommendation P.910, ITU, Oct. 2023.

[12] International Telecommunication Union, *User requirements for objective perceptual video quality measurements in digital cable television. CCITT Recommendations; electronic version*. Recommendations, Geneva: ITU, 2000.

[13] Z. Li, A. Aaron, and M. Manohara, "Toward a practical perceptual video quality metric." Accessed: Aug. 1, 2025.

[14] 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. 104, no. 7, pp. 725–737, 2021.

[15] "tc-tbf." Accessed: Aug. 1, 2025.

[16] "Pion," 2023. Accessed: Aug. 1, 2025.

[17] "Ffmpeg." Accessed: Aug. 1, 2025.

[18] T. Roosendaal, "Big buck bunny," p. 62, Association for Computing Machinery, 2008.

[19] Blender Official, "Keynote — blender conference 2024." YouTube video, Oct. 2024. Available at: https://youtu.be/VZ5022VaMmA?feature=shared.

[20] Soccer Aid for UNICEF, "Soccer aid for unicef 2024 — official match highlights." YouTube video, June 2024. Available at: https://www.youtube.com/watch?v=rEJz00Jutps.

[21] M. Ali, *PyCaret: An open source, low-code machine learning library in Python*, April 2020. PyCaret version 3.0.0.