Journal of Engineering and Technology

ISSN 2180-3811 eISSN 2289-814X https://jet.utem.edu.my/jet/index

AN INTELLIGENT SYSTEM FOR PREDICTING VIDEO STREAMING QUALITY OF SERVICE

O. E. Ojo*[1], M. K. Kareem[2], I. K. Ogundoyin[3], M. O. Abolarinwa[4], O. O. Daramola[2] and A. M. Joshua[2]

[1] Department of Information Technology, Osun State University, Osogbo, Nigeria.

[2] Department of Computer Science, Federal University of Agriculture Abeokuta, Nigeria.

[3] Department of Computer Science, Osun State University, Osogbo, Nigeria.

[4] Department of Cyber Security, Osun State University, Osogbo, Nigeria.

corresponding: oluwafolake.ojo@uniosun.edu.ng

Article history:
Received Date:
15 April 2024
Revised Date:
17 June 2024
Accepted Date:
6 August 2024

Keywords:
Machine Learning,
Quality of Service,
Video Streaming

Abstract— A primary concern associated with live streaming systems has been the significant delay in waiting time. Typically, a new user will abandon a server before receiving service due to the waiting time, startup delay, or other types of delays exceeding their tolerance. To address these issues, the optimal approach is to possess the ability to precisely forecast certain factors. This study presents a very efficient prediction framework for assessing the quality of video streaming. The proposed

system aims to precisely forecast the duration of play (total length of a particular movie), overall download speed, complete end-to-end delay, and beginning buffering latency using the extreme gradient boosting (XGBoost) machine learning algorithm. The results of the experiment show that the accuracy reached for playing time, end-to-end delay, buffering latency, and download rate is 0.985, 0.731, 0.937, and 0.927, respectively. This indicates that the system is effective in addressing the issue of extended waiting times during live broadcasting. Thus, the framework also offers a way to solve the live-stream delay to improve users' quality of experience. In conclusion, these findings are pivotal to the video streaming industry market to aid the service providers in designing their systems optimally, assigning the resources efficiently and increasing user retention.

I. Introduction

The continuous transfer of audio or video files from the sender to the recipient is called multimedia streaming. Streaming is a method of delivering data over the Internet that does not require the end user to download the complete file [1]. Multimedia streaming is excellent for broadcasting sports, gaming, and TV. Livestreaming is the recording and distribution of web videos in real-time. No preloading of media is required for users to watch or listen to content that is streamed in real-time. One of the main gripes with live streaming systems has been prolonged wait times. Due to excessive wait times, lengthy startup times, and other types of delays that exceed what the user can tolerate, most

new users quit the server before being served [2, 3]. Viewers would be more willing to participate if there was less lag time in live broadcasting. Users waste a large amount of time between when they make a request for a video or piece of music and when it begins to play in real time. Fixing the delay issue will boost the number of live-streaming system users [4]. Due to the ever-increasing demand for video streaming, streaming is utilized to watch videos without downloading. The delay in video streaming cannot be kept constant and varies depending to the throughput. As a result, the quality of the streaming video may suffer. The delay, which is also reliant on the network over which the video is delivered, is beyond the control of the content providers. The user's perception of video quality is vitally important. Because the delay may influence video quality, the user's tolerance for delay is evaluated [5]. Using a machine learning approach, the purpose of this research is to improve the user experience during video live streaming by attempting to forecast playing length, buffering latency, total download time, and end-to-end delay in video live streaming.

II. Related Work

This section examines existing strategies for enhancing video service quality with minimal delay or waiting time. For example, the researchers in [6] created Hypertext Transfer Protocol (HTTP) -adaptable streaming utilizing supervised machine learning (SMASH). SMASH is a client-side predictive model learner for video quality adaptation. SMASH was developed by combining real-time, continuous video from nine prominent ABRs in a variety of settings. Transfer rate, cushioning level, and encoding rate were among the dataset parameters. The researchers developed an AI model to select the appropriate bitrate for the next video fragment to download based on carefully specified criteria and streaming yield from the transformation logic of nine ABR calculations across

multiple streaming scenarios (around 1,000,000 records). SMASH beats cutting-edge players in terms of QoE and consistency across streaming settings. In [7], the tile-based viewport adaptive streaming of 360-degree video efficacy was studied. Tile-based approaches enhance the average viewport's peak signal-to-noise ratio (V-PSNR) under low delay settings, according to experiments. Several approaches demonstrate different trade-offs between average viewport quality and variation. Most tile-based approaches' performance quickly decreases as segment time and buffer size rise for information with no focus. The simple non-tiled technique works well even with lengthy delay settings like HTTP Adaptive Streaming. YouQ, a method for classifying end users' QoE when watching YouTube videos, was created in [8]. The system monitors and analyzes traffic traces and application-level quality metrics. Based on computed traffic features per video session, collected data was used to create machine learning models for QoE classification. To evaluate the YouQ system and methods, 1060 YouTube videos streamed across 39 bandwidth situations were collected and tested on several classification algorithms. An in-depth study was carried out in [9] to investigate the problems that can arise with streaming algorithms, quality of experience, network protocols, delays, and other related areas, as well as potential remedies. In addition to this, it focuses on the many difficulties that can arise with the quality of experience influencing elements in an unstable network environment. In [10], a deep neural network-based methodology is proposed for determining the optimal video bit rates to maximize a user's QoE. Many input characteristics, including user perceptions of video quality, buffering times, and the general flow of the video session, are linearly combined to get the QoE. The solution has been shown to enhance QoE by an average of 8.84% over the state-of-the-art ML-based technique

Pensieve, as shown by experimental findings.

For forecasting the overall quality of HTTP Adaptive Streaming sessions, a novel advanced machine learning technique known as the LSTM network was used [11]. The suggested technique considered quality, stalling durations, content qualities, and padding. The experiment's findings show that the LSTM network can forecast the general quality of HTTP Adaptive Streaming sessions. Additionally, for evaluating overall quality, the temporal links between segments in sessions are crucial. Furthermore, it was found that segment-MOS is the best metric for displaying the segment quality characteristic.

Another way statistical models can further develop the streaming experience is by expecting what a client will observe so that (portions of) it tends to be stored on the gadget before the client squeezes play, permitting the video to begin quicker and additionally with better quality. For instance, one can exploit the pattern of a client watching a particular series by following the scene. One can also structure this as supervised learning by joining different pieces of their review history with current client connections and other relevant information. To support varying network and device capabilities, movies and shows are frequently encoded at different video quality levels. Adaptive streaming algorithms determine which video quality is delivered throughout playback based on network and device conditions [12].

A Deep Q-Learning technique for DASH Video Streaming was proposed by [13]. This application is suitable for learning methods due to the complexity and variety of the video content, as well as the mobile and wireless channels. To enhance DASH's user experience, the D-DASH framework integrates deep learning and reinforcement learning methods. Different learning designs, such as feed-forward and recurrent deep neural networks, as well as sophisticated techniques, are suggested and assessed.

Innovative algorithms, both heuristic and learning-based, are thoroughly evaluated against D-DASH designs to assess performance metrics like picture quality throughout video segments. The numerical findings, which were obtained using both actual and simulated channel traces, show D-superiority DASHs in almost every quality parameter considered. The D-DASH framework not only results in a much greater QoE, but it also converges to the rate-selection strategy more quickly than the other learning algorithms taken into consideration in the study. This reduces the training period, making D-DASH a strong option for client-side runtime learning. The study makes use of deep neural networks to develop excellent video adaptation techniques while correctly and densely capturing the environment's experience [13].

Furthermore, Bandwidth Prediction Schemes for Bitrate Level Determination in SDN-enabled Adaptive Streaming was also proposed by the researchers in [14]. The HTTP Adaptive Streaming (HAS) was used to transport the large bulk of Internet video traffic. Recent studies indicate that the effectiveness of pure client-driven HAS adaptability is uncertain. Clients alter quality in response to local feedback, therefore it is often subpar. For video, a network-assisted streaming architecture (BBGDASH) with constrained bitrate recommendations was presented, achieving client flexibility and quality control while doing so. Aside from the fact that BBGDASH is an efficient method of video compression. When it comes to distribution, it works best in a wireless network environment [14, 15]. Additionally, the researchers introduced BBGDASH+, an intelligent streaming architecture for HTTP adaptive video streaming that takes advantage of SDN to enable real-time throughput forecasts. Error-Based Bounding (EBB) and Confidence-Based Bounding (CBB) were proposed to harness the power of time series prediction to establish the proper

bitrate boundaries of the intended bitrate in a wireless environment. The accuracy of the projected boundaries for the suggested technique was initially investigated to see how configurational factors (such as prediction horizon and evaluation sample rate) influenced them. When compared to only client-based HAS applications, the results demonstrate that the recommended algorithms (EBB and CBB) can greatly enhance end-user QoE [14].

The authors in [16] designed a tiling mechanism for optimal prediction of virtual reality (VR) videos to be supplied to the clients in line with the bandwidth constrained networks for 360-degree VR video streaming. The algorithm anticipates and brings in the tiles that are expected to be ordered by the user; thereby decreasing latency and enhancing the quality of experience. Performance evaluation using simulation revealed that compared to the prior techniques suggested, the algorithm was more efficient yielding better tile prediction accuracy and less stalling. The study also provided evidence to support tile selection for predicting the most suitable tile for streaming a VR video in bandwidth constrained environments. The authors in [17] developed an intelligent Machine Learning based scheduling solution for live omnidirectional (360°) video streaming based on Reinforcement Learning by implementing Continuous Actor-Critic Learning Automata (CACLA). This solution optimally solves problems related to traffic routing, adapts to dynamic changes in the network, and optimizes traffic classes, which is proven by higher efficiency in simulations than contemporary scheduling algorithms. The authors in [18] studied the scalability of video streaming, an investigation was carried out on waiting-time prediction and two prediction methods were proposed. The scheme was assessed using merging strategies of streams along with operating policies. It was noted that the study demonstrated the feasibility of

achieving accurate waiting-time estimates where the model can be integrated with a more robust version of cost-based scheduling. This rendered Waiting-time prediction valuable for performance enhancement of video streams, it was evident that waiting-time prediction could significantly enhance the performance of video streaming. Another work focused on the problem of QoE sustainability in video streaming applications, especially in dynamic bandwidth conditions [19]. The solution entails development of a client-side video streaming application that delivers video based on predicted bandwidth. It employs the moving average prediction algorithm and includes various protocols and libraries such as RTMP, HTTP, SIGAR, VLC media player. Freeze time was used as a new metric to measure the performance of the scheme. The obtained experimental outcomes revealed that freeze time was reduced by more than 60% and the PSNR was improved by at least 2 dB compared to static video stream delivery. Additionally, the authors in [20] designed a low-latency prediction-based adaptation technique for HTTP-based live streaming (LOLYPOP). To achieve low latency, the LOLYPOP employs TCP throughput predictions on multiple time scales and a relative prediction error distribution. The algorithm also aims to achieve the highest possible quality of experience by maximizing video quality while reducing the number of skipped segments and quality transitions. The authors compared the performance of LOLYPOP to the benchmark FESTIVE algorithm, with transport latency capped at three seconds, following which the authors noted improvements in the average video quality as well as better flexibility in adapting to the profiles of the users or requirements of the service provider. In [21], the authors optimized the observation window duration for tile prediction and computation /transmission durations to maximize the QoE. A closed-form optimal solution was

developed by decomposing the problem into two sub-problems, identifying resource-limited and prediction-limited regions. Simulation results using existing predictors and a real dataset validated the approach, demonstrating improved QoE through joint optimization.

Futhermore, the researchers in [22] carried out investigation on low-latency live video streaming which is a crucial aspect of user QoE. The researchers develop practical live streaming algorithms within the iterative Linear Quadratic Regulator (iLQR) based Model Predictive Control and Deep Reinforcement Learning frameworks, namely MPC-Live and DRL-Live, to maximize user live streaming QoE by adapting the video bitrate while maintaining low end-to-end video latency in dynamic network environment. Experimental results using real network traces demonstrate that these algorithms achieve near-optimal performance within the 2-5 second latency range, enhancing QoE for live video streaming.

The authors in [23] presented an analytical mode of end-to-end video quality prediction and control in IP networks and the relationship between QoS parameters and perceived video quality at the receiver end. Using this model, packet loss is explored in terms of its effects on decoded video and a correlation is made between PDR and actual video quality. Also, it quantifies the rate-distortion properties of an encoder by predicting the average quality achievable per channel bandwidth. The authors provided experimental evidence to confirm the viability of the proposed model in the estimation of video quality. The findings of this work provided the basis for further research in the domain of video quality modelling and prediction. Lastly, the researchers in [24] proposed HotDASH, a system that utilises an opportunistic approach of prefetching specific temporal video segments commonly requested by users known as hotspots. HotDASH uses a cascaded reinforcement learning model implemented by the

actor-critic algorithm known as A3C and artificial neural networks to optimise the quantity of prefetch and bitrate. Thus, only trained on various bandwidth conditions, HotDASH improves QoE rates (by 16. 2%) and average bitrate (by 14. 31%) compared to baseline algorithms. The works presented here demonstrate the applicability of reinforcement learning in improving video streaming and guide our study of adaptive bitrate streaming.

From the reviewed literature, the current state of existing methods reveals that research on utilising machine learning methods for predictive modelling and optimisation in quality video streaming is limited. However, despite numerous works focusing on various solutions, there is a need to design a robust prediction model for qualitative actual prediction of video streaming quality attributes. This gap is the main objective of this work, which proposes a predictive framework based on a machine learning algorithm (precisely the XGBoost algorithm) as the main method. This study is centred on deploying XGBoost for predicting the playback duration, download speed, end-to-end delay and buffering latency of a video streaming service. The proposed approach will create an appropriate model for the video streaming service to improve users' Quality of Experience.

III. Methodology

The proposed method uses wait time to estimate real-time video transmission. To do so, the acquired and collected datasets will be divided into different sections based on attributes and individual contents, and the divided data sets will then be trained using XGBoost. The boosting technique is used in XGBoost, a tree-based model. XGBoost is a user-friendly approach that emphasizes quick learning and classification through parallel processing. This method has lately acquired popularity, and the Data Competition has determined that it is the most accurate and useful solution for structured datasets. After that, the proper models are used to predict the wait time.

eXtreme Gradient Boosting is the name of the xgboost model. For comparing expected and observed values (sample or population values) by a model or estimator, the root-mean-square deviation (RMSD) or root-mean-square error (RMSE) is a frequently used metric.

A. XGBoost Features

The XGBoost framework is utilized in this research based on its embedded features to handle various predictive tasks, including regression, classification, ranking, and user-defined prediction, among others. XGBoost has been made very flexible and highly resistant due to its superior gradient boosting technique and strong regularisation strategies, making the model accurate and efficient. Besides, its ability to perform equally well on different platforms, thus making it easy to deploy in different environments is another advantage for this research. Hence, the study harnesses the strengths of XGBoost, which guarantees accuracy. The prediction architecture as shown Figure 1 is divided into three segments: data acquisition, model training and the model evaluation. The workflow of the procedures for predicting playing length, total download rate, and end-to-end delay is represented in Figure 2 and the Usecase diagram for the proposed system is presented in Figure 3. The procedure for classification of the video dataset is represented in Table 1.

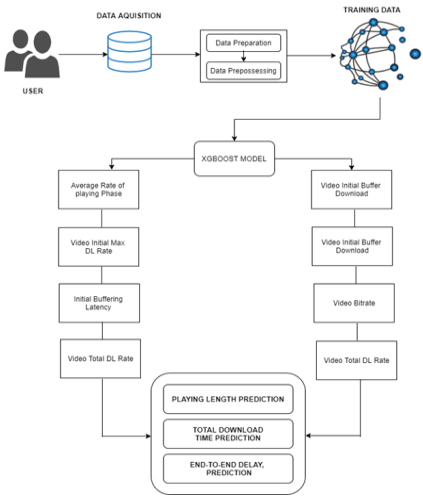

Figure 1: Prediction Architecture

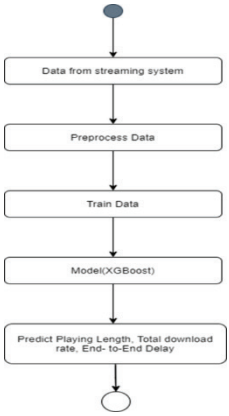

Figure 2: Workflow of the procedures

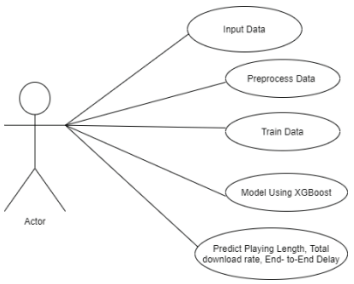

Figure 3: Usecase diagram

Table 1: Algorithm 1 for XGBoost Model

Input: Video dataset (V) Output: Learned Ensemble model (LE)
1. Start 2. Structured live streaming video dataset: {V_1, V_2, · · ·, V_n} 3. Unstructured dataset: {T1, T2, · · ·, Tn} 4. Length for the used period of characteristic A: tA 5. Length for the used period of characteristic B: tB 6. Output: 7. Ensemble XGBoostModel with learned parameters 8. Initialize the training instance set Mv = 0 9. if Program Type is characteristic A then 10. for all available video time intervals t($1 \leq t \leq tA$) do 11. Put (Vt) into MvA 12. end for 13. else 14. for all available video time intervals t($1 \leq t \leq tB$) do 15. Pt = Concat(Vt, Tt) 16. Put (Pt) into MvB 17. end for 18. end if 19. Initialize all the learnable video parameters (Vp) 20. Repeat 21. Minimize the objective function within MvA by Boosting 22. Minimize the objective function within MvB as NN 23. until the Convergence criterion met 24. Return LE 25. Stop

B. Data Acquisition

The data used in this study was obtained from Kaggle video streaming network historical logs [25]. The set of contents is then divided into two types: those that have successive previous works and those that do not. Since the datasets are divided into two categories, and the contents are then classified as popular contents, an interface is created between the two models, allowing the result to be presented based on classification likelihood. The popular likelihood is expected. More than 70% of Cn(=Cnew) is represented by this dataset.

C. Model Training

This project was initially implemented on Kaggle, it was later tested on a laptop with 8 GB RAM, 500GB, Intel Core i5. To replicate the results in this work a minimum of 1GB RAM,

Intel Atom® processor or Intel® Core™ i3 processor, Operating systems: Windows* 7 or later, macOS, and Linux will be good to go. Python as a programming language was used to implement the methodology. All the libraries are imported using Python's import keyword.

Figure 4 illustrates the data import process, wherein the dataset's location is specified as an argument in the pd.read_csv() function, thereby converting the dataset into a pandas data frame. Subsequently, the column "Total time of Playing phase (kbps)" is removed via the drop() function, and the resulting dataset is assigned to the variable "dataset".

```
data = pd.read_csv('/kaggle/input/videodownspeed/SpeedVideoDataforModeling.csv')

y = data[[' Video Total DL Rate(kbps)', 'E2E RTT(ms)', 'Playing Length(ms)', 'Initial B
uffering Latency(ms)']]

data_DL = data.drop([' Video Total DL Rate(kbps)', 'E2E RTT(ms)', 'Playing Length(ms)',
'Initial Buffering Latency(ms)'], axis = 1)
```

Figure 4: Data import process

Furthermore, the sample of the dataset for predicting the playing time and total download rate are displayed in Figures 5 and 6 respectively. Similarly, the sample of the dataset for predicting end-to-end delay and initial buffering latency are presented in Figures 7 and 8.

	Video Initial Max DL Rate(kbps)	E2E RTT(ms)	Average Rate of Playing phase (kbps)	Initial Buffering Latency(ms)	Stalling Ratio	VMOS	Video Total DL Rate(kbps)	Stalling Length	Stalling times	video bitrate	Video Initial buffer download(byte)	SQuality	SLoading	Sstalling	Play Length(s
0	49450	54	3719	1108	0.0	3.96	3795	0	0	2934	1643944	4.33	4.04	5.0	30
1	50517	52	5902	1095	0.0	3.96	5859	0	0	2903	1555356	4.33	4.05	5.0	30
2	47908	47	5806	1051	0.0	3.98	5857	0	0	2903	1535444	4.33	4.08	5.0	30
3	56457	55	5978	1099	0.0	3.96	5989	0	0	2903	1534092	4.33	4.05	5.0	30
4	56690	54	5931	1133	0.0	3.95	5932	0	0	2934	1582192	4.33	4.02	5.0	30
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
89261	27218	44	5527	1134	0.0	3.95	5605	0	0	2934	1582192	4.33	4.02	5.0	30
89262	25879	28	6108	938	0.0	4.01	6159	0	0	2934	1556892	4.33	4.16	5.0	30
89263	46312	42	5385	870	0.0	4.03	5681	0	0	2934	1542107	4.33	4.21	5.0	30
89264	36507	24	5833	773	0.0	4.06	6097	0	0	2903	1572292	4.33	4.29	5.0	30
89265	17570	93	4142	1917	0.0	3.72	4209	0	0	2903	1560593	4.33	3.51	5.0	30

89266 rows × 15 columns

Figure 5: Sample of the dataset for predicting playing time

Video nitial Max DL ate(kbps)	E2E RTT(ms)	Average Rate of Playing phase (kbps)	Initial Buffering Latency(ms)	Stalling Ratio	VMOS	Playing Length(ms)	Stalling Length	Stalling times	video bitrate	Video Initial buffer download(byte)	SQuality	SLoading	Sstalling	Video Total DL Rate(kbps)
49450	54	3719	1108	0.0	3.96	30013	0	0	2934	1645944	4.33	4.04	5.0	3795
50517	52	5902	1095	0.0	3.96	30006	0	0	2903	1555356	4.33	4.05	5.0	5859
47908	47	5806	1051	0.0	3.98	30006	0	0	2903	1535444	4.33	4.08	5.0	5857
56457	55	5978	1099	0.0	3.96	30003	0	0	2903	1534092	4.33	4.05	5.0	5989
56690	54	5931	1133	0.0	3.95	30005	0	0	2934	1582192	4.33	4.02	5.0	5932
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
27218	44	5527	1134	0.0	3.95	30017	0	0	2934	1582192	4.33	4.02	5.0	5605
25879	28	6108	938	0.0	4.01	30005	0	0	2934	1556892	4.33	4.16	5.0	6159
46312	42	5385	870	0.0	4.03	30012	0	0	2934	1542107	4.33	4.21	5.0	5681
36507	24	5833	773	0.0	4.06	30006	0	0	2903	1572292	4.33	4.29	5.0	6097
17570	93	4142	1917	0.0	3.72	30020	0	0	2903	1560593	4.33	3.51	5.0	4209

ws × 15 columns

Figure 6: Sample of the dataset for predicting of Total Download Rate

Video initial Max DL late(kbps)	Playing Length(ms)	Average Rate of Playing phase (kbps)	Initial Buffering Latency(ms)	Stalling Ratio	VMOS	Video Total DL Rate(kbps)	Stalling Length	Stalling times	video bitrate	Video Initial buffer download(byte)	SQuality	SLoading	Sstalling	E2E RTT(ms)
49450	30013	3719	1108	0.0	3.96	3795	0	0	2934	1645944	4.33	4.04	5.0	54
50517	30006	5902	1095	0.0	3.96	5859	0	0	2903	1555356	4.33	4.05	5.0	52
47968	30006	5806	1051	0.0	3.96	5857	0	0	2903	1535444	4.33	4.08	5.0	47
56457	30003	5978	1099	0.0	3.96	5989	0	0	2903	1534092	4.33	4.05	5.0	55
56690	30005	5931	1133	0.0	3.95	5932	0	0	2934	1582192	4.33	4.02	5.0	54
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
27218	30017	5527	1134	0.0	3.95	5605	0	0	2934	1582192	4.33	4.02	5.0	44
25879	30005	6108	938	0.0	4.01	6159	0	0	2934	1556892	4.33	4.16	5.0	28
46312	30012	5385	870	0.0	4.03	5681	0	0	2934	1542107	4.33	4.21	5.0	42
36507	30006	5833	773	0.0	4.06	6097	0	0	2903	1572292	4.33	4.29	5.0	24
17570	30020	4142	1917	0.0	3.72	4209	0	0	2903	1560593	4.33	3.51	5.0	93

rs × 15 columns

Figure 7: Sample of dataset for predicting End-to-End Delay

Video initial Max DL late(kbps)	E2E RTT(ms)	Average Rate of Playing phase (kbps)	Playing Length(ms)	Stalling Ratio	VMOS	Video Total DL Rate(kbps)	Stalling Length	Stalling times	video bitrate	Video Initial buffer download(byte)	SQuality	SLoading	Sstalling	Initial Buffering Latency(ms)
49450	54	3719	30013	0.0	3.96	3795	0	0	2934	1645944	4.33	4.04	5.0	1108
50517	52	5902	30006	0.0	3.96	5859	0	0	2903	1555356	4.33	4.05	5.0	1095
47968	47	5806	30006	0.0	3.96	5857	0	0	2903	1535444	4.33	4.08	5.0	1051
56457	55	5978	30003	0.0	3.96	5989	0	0	2903	1534092	4.33	4.05	5.0	1099
56690	54	5931	30005	0.0	3.95	5932	0	0	2934	1582192	4.33	4.02	5.0	1133
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
27218	44	5527	30017	0.0	3.95	5605	0	0	2934	1582192	4.33	4.02	5.0	1134
25879	28	6108	30005	0.0	4.01	6159	0	0	2934	1556892	4.33	4.16	5.0	938
46312	42	5385	30012	0.0	4.03	5681	0	0	2934	1542107	4.33	4.21	5.0	870
36507	24	5833	30006	0.0	4.06	6097	0	0	2903	1572292	4.33	4.29	5.0	773
17570	93	4142	30020	0.0	3.72	4209	0	0	2903	1560593	4.33	3.51	5.0	1917

rs × 15 columns

Figure 8: Sample of the dataset for predicting Initial Buffering Latency

For effectiveness, each dataset in this study is segmented into two variables X and Y; X consists of the independent variables and Y contains a dependent variable representing the total delay time to be predicted. Additionally, 70% of the dataset from each category is used for training while 30% is used for testing. Feature importance derived from the XGBoost algorithm is used to select the most robust features for the model creation. The most relevant features for playing length prediction are shown in Figure 9 while Figures 10, 11 and 12 illustrate the features affecting total download rate, buffering latency and end to end delay, respectively.

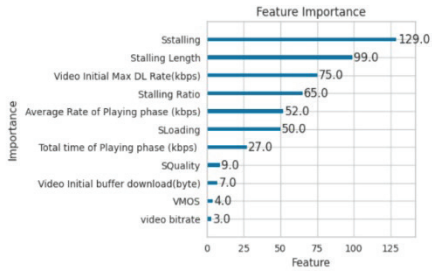

Figure 9: Features of playing length

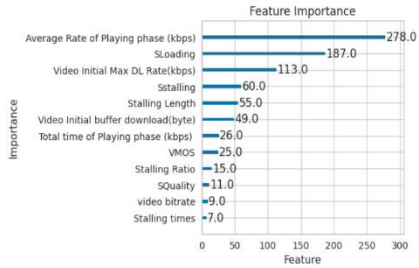

Figure 10: Features of total download rate

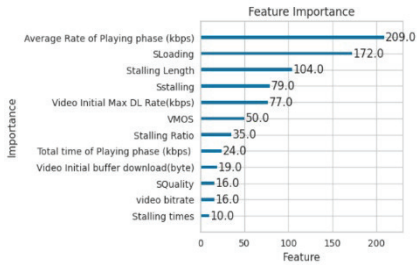

Figure 11: Features of buffering latency

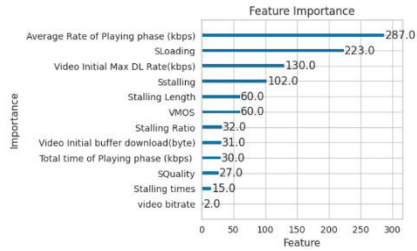

Figure 12: Features of End-to-End Delay

IV. Results

This section reports the empirical outcome of the proposed model in establishing various performance indicators. In particular, the accuracy of the model in terms of estimating playing length is demonstrated in Figure 13, achieving a remarkable 98.5% accuracy rate. The line of best fit closely aligns with the 45-degree line which means that the predictions are very accurate. In addition, the high predictive power of the model can be seen in Figure 14, whereby the R2 value is 0. 927 for the total download rate, which corresponds to an accuracy of 92.7%.

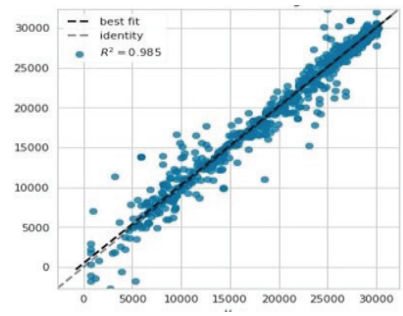

Figure 13: Play Length Results

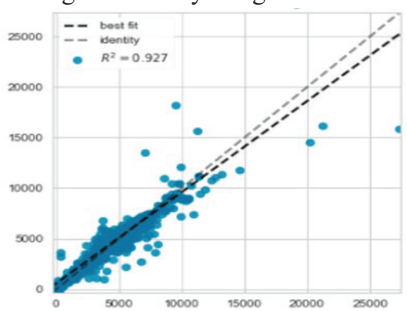

Figure 14: Results for Total Download Rate

Likewise, Figure 15 shows the R2 value of 0. 731 for End-to-end delay, meaning that its accuracy is 73.1%. Moreover, Figure 16 shows a high R2 value

of 0. 937 for playing length, corresponding to an accuracy of 93.7%. Taken collectively, these results speak to the capacity of the model to account for a large proportion of the variability in the sample, ranging from approximately 73.1% to 93. 7% uttermost cases, according to the performance criterion.

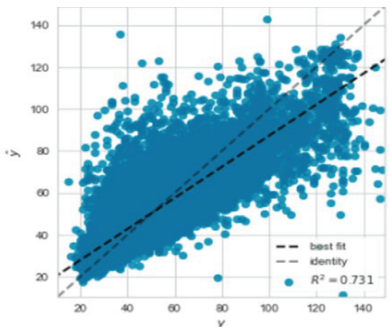

Figure 15: End-to-End Delay Results

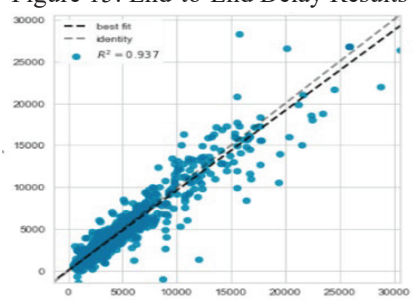

Figure 16: Buffering Latency Results

The Root Mean Squared Error (RMSE) was employed as a metric to evaluate the discrepancy between predicted and observed values, providing a quantitative assessment of the model's performance. Alternatively referred to as the Root-Mean-Square Deviation (RMSD), this metric is calculated using the formula given in Equation 1.

$$\mathrm{RMSE} = \frac{\sum(y_i - \hat{y}_i)^2}{N} \quad (1)$$

where, i = variable i, N = Number of non-missing data points, y = actual observations time series, $\hat{y}_i$ = estimated time series.

The results of the RMSE generated for the model are presented in Figure 17, which shows the RMSE in different scenarios. From Figure 17, the RMSE scores were obtained for playing length, Total Download Rate, End-to-End Delay and Buffering Latency Respectively.

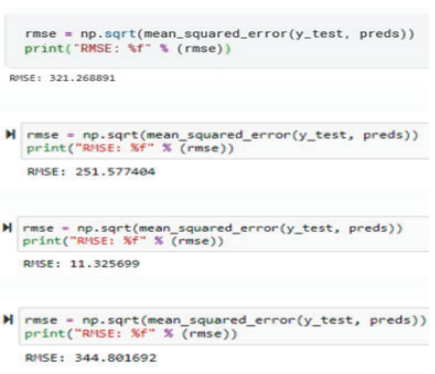

Figure 17: RMSE Results

V. Conclusion

In conclusion, this study presents an effective mechanism to forecast video streaming quality using a machine learning model. Through empirical investigation, it is established that the proposed scheme can predict with the lowest accuracy at 73.1% and the highest accuracy at 93. 7% using key performance indicators of live streaming videos such as download rate, initial buffering, and playing length. The accuracy results validate that the scheme is robust and suitable for forecasting real-time scenarios. The study further revealed the prospect of the XGBoost model to predict media streaming quality of service and end-user quality of experience. The high effectiveness of the proposed model indicates its applicability for widespread implementation in the sphere of streaming services and, therefore, the general improvement of the quality of services provided to users. Further investigation can include testing the proposed scheme with datasets from different video streaming platforms, evaluating with more parameter metrics such as recall, F1 scores and many more, and comparing this model with other machine learning algorithms.

VI. References

[1] Ereira, R., & Pereira, E. G. (2016). "Video streaming: Overview and challenges in the internet of things", *Pervasive Computing: Next Generation Platforms for Intelligent Data Collection*, 417-444.

[2] Krishnan, S. S., & Sitaraman, R. K. (2012), "Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs" in *Proceedings of the 2012 Internet Measurement Conference*, pp. 211-224.

[3] Mandelbaum, A., & Zeltyn, S., "Data-stories about (im) patient customers in tele-queues*", Queueing Systems*, vol 75, no 2, pp 115-146, 2013.

[4] Chen, Y., Li, Q., Zhang, A., Zou, L., Jiang, Y., Xu, Z., & Yuan, Z., "Higher quality live streaming under lower uplink bandwidth: an approach of super-resolution-based video coding", in *Proceedings of the 31st ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, pp. 74-81, 2021.

[5] Li, B., Wang, Z., Liu, J., & Zhu, W, "Two decades of internet video

streaming: A retrospective view. ACM transactions on multimedia computing", Communications, *And Applications* (TOMM), vol 9(1s), pp 1-20, 2013.

[6] Sani, Y., Raca, D., Quinlan, J. J., & Sreenan, C. J., "SMASH: A supervised machine learning approach to adaptive video streaming over HTTP" in *12th International Conference on Quality of Multimedia Experience (Qomex), IEEE,* pp. 1-6, 2016.

[7] Nguyen, D. V., Tran, H. T., & Thang, T. C, "An evaluation of tile selection methods for viewport-adaptive streaming of 360-degree video", *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM),* vol 16(1), pp 1-24.

[8] Orsolic, I., Pevec, D., Suznjevic, M., & Skorin-Kapov, L., "A machine learning approach to classifying YouTube QoE based on encrypted network traffic", *Multimedia tools and applications*, vol 76, pp 22267-22301, 2017.

[9] Abdullah, M. T. A., Lloret, J., Cánovas Solbes, A., & García-García, L., "Survey of transportation of adaptive multimedia streaming service in internet", *Network Protocols and Algorithms*, vol 9(1-2), pp 85-125, 2017.

[10] Lekharu, A., Moulii, K. Y., Sur, A., & Sarkar, A, "Deep learning-based prediction model for adaptive video streaming" in *International Conference on Communication Systems & Networks (COMSNETS), IEEE*, pp. 152-159, 202.

[11] Tran, H. T., Nguyen, D. V., Ngoc, N. P., & Thang, T. C. "Overall quality prediction for HTTP adaptive streaming using LSTM network", *IEEE Transactions on Circuits and Systems for Video Technology,* vol 31(8), pp 3212-3226, 2020.

[12] Peña-Ancavil, E., Estevez, C., Sanhueza, A., & Orchard, M., "Adaptive Scalable Video Streaming (ASViS): An Advanced ABR Transmission Protocol for Optimal Video Quality", *Electronics*, vol 12(21), pp 4542, 2023.

[13] Gadaleta, M., Chiariotti, F., Rossi, M., & Zanella, A., "D-DASH: A deep Q-learning framework for DASH video streaming*", IEEE Transactions on Cognitive Communications and Networking,* vol 3(4), pp 703-718, 2017.

[14] Al-Issa, A. E., Bentaleb, A., Barakabitze, A. A., Zinner, T., & Ghita, B. (2019), "Bandwidth prediction schemes for defining bitrate levels in SDN-enabled adaptive streaming", in *15th International Conference on Network and Service Management (CNSM), IEEE,* pp. 1-7, 2019.

[15] Al-Issa, A. E., Bentaleb, A., Zinner, T., Mkwawa, I. H., & Ghita, B., "BBGDASH: A Max-

Min Bounded Bitrate Guidance for SDN Enabled Adaptive Video Streaming" in *22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN), IEEE,* pp. 307-314, IEEE, 2019.

[16] Nguyen, T. C., & Yun, J. H., "Predictive tile selection for 360-degree VR video streaming in bandwidth-limited networks", *IEEE Communications Letter*s, vol 22(9), pp 1858-1861, 2018.

[17] Comșa, I. S., Muntean, G. M., & Trestian, R., "An innovative machine learning based scheduling solution for improving live UHD video streaming quality in highly dynamic network environments", *IEEE Transactions on Broadcasting*, vol 67(1), pp 212-224, 2020.

[18] Sarhan, N. J., Alsmirat, M. A., & Al-Hadrusi, M., "Waiting-time prediction in scalable on-demand video streaming", *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM),* vol 6(2), pp 1-25, 2010.

[19] Fowdur, T. P., & Narrainen, L., "Enhanced video streaming using dynamic quality control with bandwidth prediction" in *International Conference on Computer as a Tool (EUROCON),* pp. 1-6, 2015.

[20] Miller, K., Al-Tamimi, A. K., & Wolisz, A., "QoE-based low-delay live streaming using throughput predictions", *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol 13(1), pp 1-24, 2016.

[21] Wei, X., Yang, C., & Han, S., "Prediction, communication, and computing duration optimization for VR video streaming", *IEEE Transactions on Communications,* vol 69(3), pp 1947-1959, 2020.

[22] Sun, L., Zong, T., Wang, S., Liu, Y., & Wang, Y., "Towards optimal low-latency live video streaming", *IEEE / ACM transactions on networking*, vol 29(5), pp 2327-2338, 2021.

[23] He, Z., & Chen, C. W. (2002, August), "End-to-end video quality analysis and modeling for video streaming over IP network", in *Proceedings of IEEE International Conference on Multimedia and Expo*, vol. 1, pp. 853-856), 2002.

[24] Sengupta, S., Ganguly, N., Chakraborty, S., & De, P., "HotDASH: Hotspot aware adaptive video streaming using deep reinforcement learning", in *IEEE 26th International Conference on Network Protocols (ICNP)*, pp. 165-175, 2018.

[25] https://www.kaggle.com/datasets/brightdavid/videodownspeed