Received 25 January 2021; revised 22 June 2021; accepted 7 July 2021. Date of publication 26 July 2021; date of current version 22 September 2021. The review of this article was arranged by Associate Editor Dr. Karen Livescu.

Digital Object Identifier 10.1109/OJSP.2021.3099065

Forecasting Video QoE With Deep Learning From Multivariate Time-Series

HOSSEIN EBRAHIMI DINAKI[1] (Student Member, IEEE), SHERVIN SHIRMOHAMMADI[1] (Fellow, IEEE), EMIL JANULEWICZ[2], AND DAVID CÔTÉ[2]

[1] DISCOVER Lab, School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, ON K1N 6N5, Canada
[2] Blue-Planet Analytics, Ciena Corporation, Ottawa, ON K2K 0L1, Canada

CORRESPONDING AUTHOR: HOSSEIN EBRAHIMI DINAKI (e-mail: hebra056@uottawa.ca).

This work was supported in part by Mitacs Accelerate Cluster under Grant IT12571 and in part by Ciena Corporation.

ABSTRACT The end users' satisfactory Quality of Experience (QoE) is a fundamental criterion for networked video service providers such as video-on-demand providers (Netflix, YouTube, etc.), cloud gaming providers (Google Stadia, PlayStation Now, etc.) and videoconferencing providers (Zoom, Microsoft Teams, etc.). To know the QoE, providers today typically predict it from the Quality of Service (QoS) parameters or the client-side's actual QoE metrics measured at the current time-step. But the former does not precisely reflect the users' experience, and the latter has a delay between QoE measurements at the client-side and the user's current experience. Mitigating this delay can provide a noticeable improvement in the delivery system's performance. For example, accurate forecasting of QoE for the near future allows the service management system to take a proactive approach and fix delivery issues before they become a noticeable problem at the end user, or at least reduce overall QoE degradation. QoE forecasting can also be used in rate adaptation in DASH or resource allocation in wireless networks. In this paper, we propose a method to prognosticate QoE metrics. Using data collected from an industry video streaming testbed for three different classes, we define a multivariate time series forecasting problem. We then model a hybrid state-of-the-art deep learning method, BiLSTM-CNN, to forecast the QoE metrics in advance. Evaluation of our proposed method compared to four other well-known ML models of Support Vector Regression (SVR), Multi-Layer Perceptron (MLP), Long short-term memory (LSTM), and Bidirectional LSTM (BiLSTM) demonstrates the superior performance of our proposed method.

INDEX TERMS Artificial intelligence, recurrent neural networks, forecasting, machine learning algorithms, multimedia communication, quality management, streaming media, time series analysis.

I. INTRODUCTION

Video streaming services have noticeably grown in recent years. According to Cisco's Visual Network Index (VNI) report, video will occupy 82% of all internet traffic by 2022 [1]. Services such as Video On Demand from Netflix, YouTube, Amazon Prime, etc., or live video services from Twitch, YouTube Gaming, etc. will lead to a global video streaming market of \$102 billion by 2023 [2]. To sustain this growth, providers must ensure that users have a satisfactory experience with their services, because poor experience is the key reason for customer turnover [3]. This means that providers should monitor the QoE and take corrective actions if it degrades.

Historically, QoS such as packet loss, delay, and jitter were considered service satisfaction metrics. However, since QoS does not reflect a user's quality perception, QoE is now used instead. QoE describes the degree of enjoyment or irritation of the end-user while using a video service [4]. A significant amount of research has been conducted to understand, measure, and model QoE in different video services. This knowledge can help service and network providers deliver high-quality and cost-effective video services while efficiently managing network operations [5]. At its simplest, a video streaming chain includes the client side, server side, and the network between these two sides. To manage QoE in such a chain, there are two approaches: proactive and reactive.

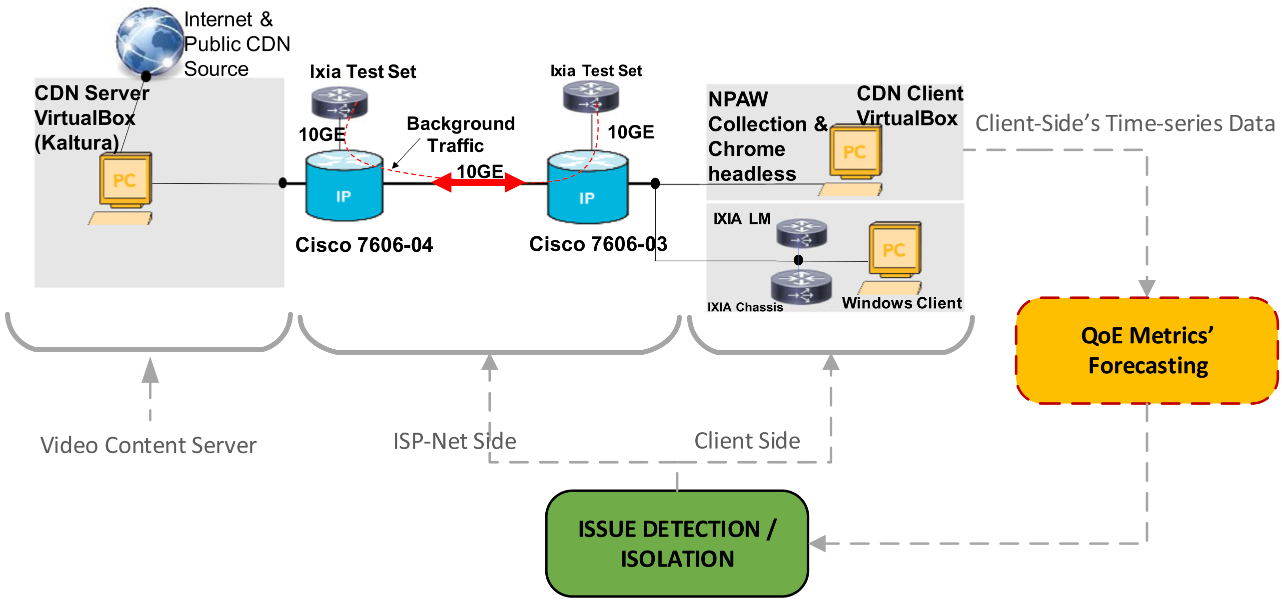

FIGURE 1. Data collection testbed and a sample use case of the proposed forecasting mechanism: issue detection/isolation framework.

In proactive approaches [6], the system uses QoS parameters extracted from network monitoring, and tries to estimate the QoE from these parameters. Based on the estimated QoE, the system anticipates the users' satisfaction or dissatisfaction. This method usually employs an approximation function or a trained ML model to predict the QoE score. Then, if QoE is less than satisfactory, a resource allocation or fault diagnosis system will decide about a possible action to resolve the issue. But this approach uses QoS parameters, which do not necessarily reflect the QoE of end users. It also requires full end-to-end access over the delivery path in order to measure the QoS parameters. But such access is not given in many cases, such as multi-Autonomous (AS) networks where multiple entities own different sections of the path.

In reactive approaches [7], the management system monitors the QoE metrics from the client-side. Theses QoE metrics are accessible to the provider because they are measured directly at the client side and do not require access to the delivery network. Then, the management system decides what action to take depending on the measured QoE. For instance, if QoE is degrading, a fault diagnosis system can try to detect and localize the issue leading to the degradation [8]. While this approach uses the actual QoE metrics which are a more precise reflection of the user's experience, it has an inherent shortcoming: it's always late. When QoE degradation is detected, it means that the user has already experienced that degradation on his/her screen. It would therefore be an important contribution if we could reduce or ideally eliminate this lateness, giving the provider enough time to localize and fix the fault before the user notices any degradation, or at least reduce the latter's duration.

In this study, we introduce QoE forecasting to achieve the above goal. Our method allows the provider to become more proactive in fault mitigation or resource allocation by using forecasted QoE values. We propose a deep learning sequence model that forecasts QoE metrics at the next time step and before occurring on the client's screen. The use cases of our proposed approach are video streaming applications that can benefit from QoE forecasting, such as fault diagnosis in packet video networks [8], anomaly detection in CDN and wireless networks [7][6], and resource scheduling in wireless networks for adaptive DASH delivery [9].

Our system is shown in Fig. 1. We collected a dataset from the testbed, shown in the figure's upper left part, by streaming a vast number of videos from the CDN server to the CDN client. During the streaming, we collected the quality metrics in three different scenarios: without congestion, congestion in the client network, and congestion in the ISP network. The collected data is a multivariate time series that includes QoE and QoS metrics from the client and ISP sides. The details of the data collection are presented in section III. The figure also shows a sample use case for a fault diagnosis system in simple multi-AS networks.

To forecast QoE, we designed and implemented a BiLSTM-CNN model to solve our multivariate time series forecasting problem. The method is a combination of Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN). We chose this combination as the RNN family has been shown to perform well for addressing time-series problems [10] and CNN is potent for spatial feature extraction [11]. Specifically, we use BiLSTM as an efficient RNN method which extracts the features related to the temporal dependency of the QoE metrics, and CNN which increases the forecasting accuracy by extracting local dependency. To improve the proposed model's performance, we tried the *tanh* and scaled exponential linear units (SELU) [12] activation functions, and we chose *SELU* since it outperforms *tanh* among the three LSTM based models. To evaluate our proposed BiLSTM-CNN model, we compared it with four other ML algorithms: LSTM, BiLSTM, SVM and MLP. The results show a lower forecasting error by our proposed method. To the best of our knowledge, no other work has attempted to forecast video QoE for future time steps.

The rest of the paper is organized as follows: Section II describes related works. Section III evaluates the collected data. Section IV describes the prognostics problem and the proposed method, while section V gives a detailed description of the obtained results. Finally, in Section VI, we present the conclusions and future works.

II. RELATED WORKS

Although many existing solutions estimate or predict the *current* video QoE using QoS parameters and/or QoE metrics, we are not aware of any work that forecasts the QoE for the *next* time step. As explained earlier, such forecasting can be very useful for operators to take preemptive actions and fix issues, reducing or eliminating periods of low QoE for the user. One example is [13], which presented a proactive mechanism for network fault forecasting using QoE to find the root cause in an IPTV network. Authors collected data from the Telenor IPTV delivery network, where the Set-top box (STB), WiFi, and Links traffic (i.e., two aggregate switches) were the vantage points. In each time-step, they presented the overall QoE for each user as “Unavailable”, “Major”, “Minor”, or “OK”. They then summarized the QoE to a binary class (1 for ‘OK’ and 0 otherwise). They used the QoE collected from the STB as ground truth and the network features from WiFi and switches as input features fed into their ML models. Using this dataset, they estimated whether the network will be faulty or not in the next time step. Another example is the mobile network anomaly detection approach presented in [6], which deploys an ML method to predict QoE metrics using network metrics. The author in [7] also presents a reactive framework, consisting of Q-WATCH and Q-RANK, that identifies anomalous systems in a CDN, utilizing QoE as ground-truth and detecting the failures. It monitors freezing time, and the bitrate drops at the client-side, and calculates QoE using an approximation function. Finally, to detect and isolate network impairments in video networks, we proposed a reactive ML system in [8] which uses only the QoE metrics measured at the client’s video player. We use the same experimental setup and dataset as in [8] for this work too, although the work and the contribution of this paper; i.e., forecasting the video QoE, are totally different form [8].

As mentioned, some works predict QoE at current time. An example is [14], which proposed a proactive service quality assessment process called Q-Score to estimate a single QoE metric using the network performance metrics. An ML-based QoE prediction approach is introduced in [15] that can be applied in a proactive approach. It predicts rebuffering events from QoS parameters using a Bayesian Network; hence, it needs to monitor network parameters. Another QoE prediction method is presented in [16] and uses client-side metrics such as bitrate, frame rate, and user information. To increase accuracy, it adds a preprocessing block to extract appropriate features and feeds them into the prediction model. Yet another example is [17] which uses measured video metrics and users’ survey data to estimate QoE. Data is collected experimentally including objective metrics such as total stall duration and the number of stalls for each video session. At the end of the video session, the survey data filled by the users were added to the dataset. Three ML models were then trained to classify QoE into five levels. Finally, a continuous QoE prediction method using a neural network is presented in [18]. The prediction model’s inputs include rebuffering information, bandwidth-induced fluctuations, and QoE memory of the previous events. An extension of this approach is presented in [19] by the same authors, using a high-performance ensemble model and multiple datasets.

We can see that predicting the video QoE at the current time is a common approach in most proactive and almost all reactive methods. However, as aforementioned, an administrator does not usually have access to the whole end-to-end path. So, the reactive approach would be more common, although as explained above the reactive approach suffers from lateness. To counter that lateness, we introduce a semi-proactive approach by forecasting the video QoE for the next time step. Using this prognosticating approach, the management mechanism has more insight into the QoE’s next state and more time to take appropriate action.

III. DATA COLLECTION AND ANALYSIS

To build an appropriate dataset of video metrics, we employed the testbed presented in the upper left corner of Fig. 1 at Ciena Inc. In the testbed, the path includes two Cisco7606 routers, one Kaltura [20] video server, and a client. A combination of 20 SD and HD videos were uploaded to the video server, and multiple videos were streamed through the network to the client. While videos were streaming, we collected QoS and QoE metrics on the network and the client side, respectively.

The professional tools we used to collect data included IXIA [21], Nice People At Work (NPAW) [22], Route Optimization and Analysis Explorer Suite (ROA) [23] and Chrome-headless. IXIA generates background traffic through the link between Cisco routers. During this background traffic, the path experiences packet loss that varies between 0 to 20%. NPAW, a professional real-time media evaluation suite, is used to collect the client-side QoE metrics. ROA suite monitors the network path and the status of the generated anomaly, and collects four QoS metrics: latency, httprtt, jitter, and packet loss. To avoid popping up multiple players on the screen, we used the browser automation framework Selenium and wrote a python script to automatically run multiple videos on the client in a headless mode without popping the player up.

To automatically collect data, we wrote a data collection program in python. When the videos play on the client, this program simultaneously extracts and records both the QoS and QoE. The QoE metrics come from NPAW, and QoS metrics come from ROA. In every time window (Δt), we save the QoS and the QoE metrics in a CSV format, while several

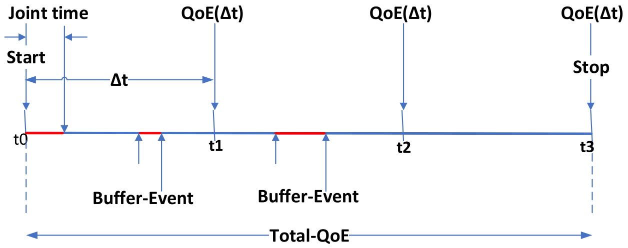

FIGURE 2. Steps of the QoE's metrics collection [8].

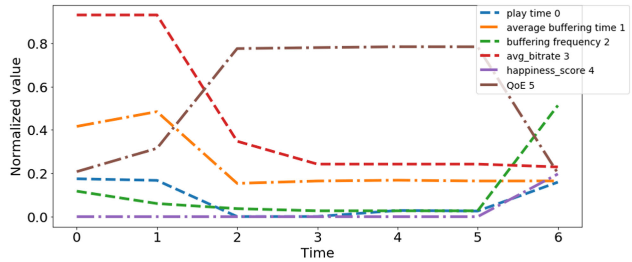

FIGURE 3. Normalized Value of 6 features at 7 time-steps (0 to 6) during a single video playback [8].

videos are passing through the network, both with and without congestion.

A. QOE MEASUREMENT

QoE's evaluation is a complicated and expensive process due to its subjective nature and dependency on the end user's perception and expectation. The widely-used subjective QoE measurement approach; i.e., mean opinion score (MOS) [24] [25], is not practical for real-time and live measurements. Therefore, approaches such as the Video Quality Metric (VQM) [26] are proposed to estimate the QoE objectively. In this study, we used the NPAW suite to measure real-time QoE and other related quality parameters at the client-side. The NPAW itself measures the QoE only at the end of the video playback, which is not useful for real-time QoE forecasting. Hence, we implemented a real-time method to measure the QoE value in time steps of Δt starting from the beginning of a video's playback. Fig. 2 shows the process for calculating the QoE during playback.

We calculated QoE using the NPAW events of JOINT-TIME, BUFFER, and STOP. These events are in JSON format, and we parse them to extract the metrics that directly affect the end user's QoE. The Playback duration, join time buffering, buffering length, buffering frequency, and average bitrate are the metrics used for our QoE calculation. At every Δt and at the stop time, the QoE value is calculated based on the following formulas [27]:

$$Q_1 = \exp\left(-\sum_{i=1}^{k} \frac{N_i * L_i * W_i}{T_i}\right) \tag{1}$$

Where:

- N_i is the number of buffering events in time window i;
- L_i is the average buffering lengths in time window i;
- W_i is a weight factor
- T_i is time period in seconds of each time window i;
- k is the number of time windows of a video;

Q_1 gives us a value between 0 and 1.

To calculate video quality from the given average bitrate, we used the following equation [28]:

$$Q_2 = v_1 - \frac{v_1}{\left(\frac{B_{rv}}{v_2}\right)^{v_3}} \tag{2}$$

Where Q_2 represents the maximum video quality at each video bitrate (BrV) and its value is between 0 to 4. Coefficients v_1 , v_2, and v_3 are dependent on the codec type, video format, key frame interval and video display size. The QoE value is as follows [8]:

$$Q = 1 + C_1 Q_2 Q_1 \tag{3}$$

Where C_1 is a scaling coefficient. When the bitrate's value was not available in the real-time data, we measured the time window QoE values using equation 4 [8]:

$$Q_{total} = \begin{cases} 1 + C_1 Q_2 Q_1 & \text{If bitrate is available} \\ C_2 Q_1 & \text{If bitrate is not available} \end{cases} \tag{4}$$

Where C_2 is a scaling coefficient.

Traditionally in ITU standards, QoE was calculated out of 5. But in the NPAW tool that we used it is calculated out of 10. So, in our case, $C_1 = 10$ and $C_2 = 2.25$ fit with our measurements.

B. DATASET

We collected data in three classes: Congestion_Free, Net_Congestion, and Client_Issue. To create the Congestion_Free scenario, we streamed between 20 to 40 videos simultaneously. Due to the 10 Gbps link, this number of simultaneous videos did not cause congestion. To create the Net_Congestion scenario, we generated background traffic using IXIA and streamed hundreds of videos simultaneously, overloading the 10 Gbps link and causing congestion. To create Client_Issue, we limited the bandwidth at the client network and streamed 20 to 40 videos simultaneously, creating congestion. In total, we streamed 2355 videos and recorded more than 17000 samples for the three classes. From each real-time event at the client-side, we extracted six metrics: playback duration, buffering length, buffering frequency, average bitrate, happiness score (given by NPAW) and the calculated Q_{total}. The dataset has three labels: no issue, ISP issue, client issue. Fig. 3 shows a sample of the six features collected in 7 Δt time-steps during single video playback. Each video has a unique token. As shown, by decreasing the average buffering length at time-step 2 and increasing the buffering frequency at time-step 6, the user's QoE increases and reduces, respectively. Since NPAW measures the happiness score only at the stop time of playback, its recorded value is zero for the first six time-steps in this example.

Our QoE metrics include playtime, average buffering length, buffering frequency, average bitrate, and happiness score, collected with the NPAW tool. The total QoE value is calculated using the model presented in section III-A. It should be noted that any QoE model, such as ITU-T P.1203 or others, can also be used in our architecture if their required input metrics are available and the results have a high correlation with the labels.

IV. THE PROPOSED FORECASTING MODEL

Although in general QoE depends on more recent experiences, the *recency* is not enough if it is affected by the *primacy* effect, for example, if multiple buffering events occurred in the past [29]. The primacy effect is the human tendency to recall the events at the beginning of an events' series [30]. In other words, the early samples of the QoE have beneficial information for forecasting the QoE in the next time step. Therefore, our proposed algorithm should extract appropriate temporal features from the input sequences. Also, the QoE metrics have a nonlinear relation with each other because, for example, the user's experience always suffers from a buffering event, but the strength of this negative impact depends on the video bitrate level [31]. In modelling the forecasting process, these valuable dependencies should be taken into account. Therefore, due to the nature of our multivariate time series, we proposed a hybrid deep learning method called BiLSTM-CNN.

Over the past years, CNN has shown its strong capability to represent complex data, especially in image feature extraction [32]. LSTM has also become a proven solution for short and long-term dependencies identification with excellent results in Natural Language Processing, image captioning [33] and question answering [34]. The combination of CNN and LSTM has shown to help a system extract spatial and temporal features of the data efficiently [12][33]. BiLSTM has further improved LSTM in forecasting accuracy for time series data [10]. Therefore, we employed BiLSTM in our hybrid model to capture the temporal dependencies of the QoE metrics. To increase the model's performance and extract local features that summarize the QoE metrics' nonlinear dependencies, we employed a CNN layer after the BiLSTM layers in the proposed model. The BiLSTM layer accepts QoE metrics as input and extracts temporal features in order to feed into the CNN Layer for spatial feature extraction.

The input data includes the happiness score and Q_{total} that directly show the quality level perceived by the user, and the average rebuffering time, number of rebuffering, and average bitrate that impact the QoE. The video playback duration has been utilized as another input information since it is a useful metadata that shapes the final QoE score. We used the first six time-steps as input and the seventh time-step would be the expected result from the forecasting model.

A. BILSTM

The usual RNN models such as LSTM have a causal structure where the hidden activation propagates information only

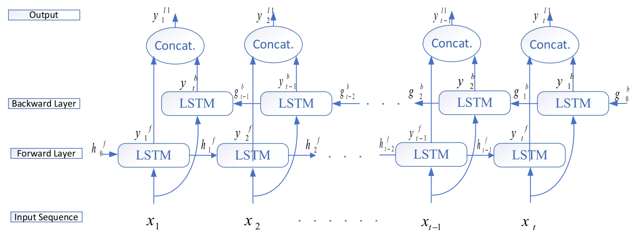

FIGURE 4. **BiLSTM; Architecture of one hidden layer.**

in the forward direction through time. This approach may lose some useful information. However, BiLSTM provides learning in both forward and backward directions and allows the model to compute a representation that depends on both past and future [32]. By feeding the input sequence into the LSTM units, BiLSTM can precisely capture the underlying context and achieve higher performance than unidirectional LSTM [10]. Fig. 4 describes a single hidden BiLSTM's layer, where h is the hidden activation for forward propagation, and g is the hidden activation for backward propagation. At each point t, in order to map input sequence x to output sequence y, the model combines the forward (from left to right) and backward (from right to left) propagation outputs. The output representation depends on the past and the future and can capture the complex temporal dependencies of our time series QoE metrics.

The next section describes the structure of the proposed hybrid neural network model.

B. CNN

CNN uses the input data's spatial structure and connects the input patches to the single units in the subsequent hidden layer [32]. These connections are formed by merely sliding the patch window across the input data. This patchy operation is a convolution that extracts local features of the data by applying a set of weights for the filter. By changing the filter's weights, we can change what the filter is activating and looking for. Different filters can activate different feature maps.

A pooling layer is used to downsample the features extracted from each convolution layer and reduce the feature map size. It allows the network to deal with multiple scales of the input data. We employed the Max-pooling technique, which selects the most active feature of the feature patch chosen by the sliding window. Max-pooling helps make the system robust and the representation approximately invariant to the inputs' small deviations.

C. BILSTM-CNN

In this section, we introduce the proposed model designed to leverage advantages of BiLSTM for temporal feature extraction and CNN for capturing the complex dependencies among QoE metrics.

Let x^t be a vector of QoE metrics at time step t, and y^{t+1} and $\hat{y}^{t+1}$ be the actual and forecasted QoE feature vector at time instant $t + 1$, respectively. In order to forecast the next

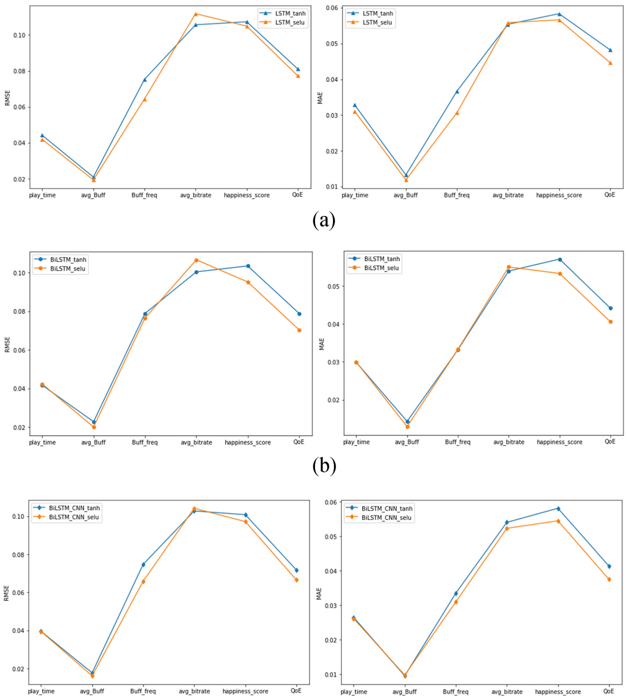

FIGURE 5. Performance comparison between *SELU* and *tanh* activation functions for LSTM, BiLSTM and BiLSTM-CNN Forecasting models in terms of RMSE and MAE. (a) LSTM, (b) BiLSTM, and (c) BiLSTM-CNN.

QoE vector at time instant t + 1, we have to take into account the following function:

$$\hat{y}^{t+1} = f\left(x^1, \ldots, x^{t-2}, x^{t-1}, x^t\right) \tag{5}$$

Where $x^1, \ldots, x^{t-2}, x^{t-1}, x^t$ is the QoE feature vector at each time step from the beginning of the video playback until time instant t. The nonlinear function $f(\cdot)$ is approximated by BiLSTM-CNN. The proposed model feeds the normalized input features into the BiLSTM layers to reduce temporal variations in input QoE metrics. Then, the last BiLSTM layer's output is fed into a CNN layer to reduce spatial variation. The CNN layer's output passes through the Max-pooling layer, then fed into a fully connected layer, transforming the features into a divisible space that makes the output more straightforward for final decision/forecast. Fig. 6 shows the architecture of our proposed model. The input is the metrics vectors of the first six time-steps and the model's output is the forecasted metrics vectors of the seventh time step. The metrics vector includes d = 6 features presented in Fig. 3; i.e., playtime, buffering-length, buffering-frequency, average-bitrate, happiness-score and $\mathrm{QoE}_{\mathrm{total}}$ which are collected in each time-step for each video.

1) ACTIVATION FUNCTION

Applying nonlinear activation function allows the network to deal with nonlinear data and introduce complexity into the learning process [32]. Therefore, the network can solve more complex tasks. The default activation function for LSTM models is *tanh*. We used the scaled exponential linear units (SELU) activation function, which induces self-normalizing properties [12] and converges to zero mean and unit variance when propagated through multiple layers. This convergence property makes training very robust by avoiding vanishing and exploding gradient problems. It outperforms other regularization methods such as weight normalization and batch normalization. The following formulation shows the behavior of the SELU activation function [12]:

$$SELU\,(x) = \lambda \begin{cases} x & \text{if } x > 0 \\ \alpha \exp(x) - \alpha & \text{if } x \leq 0 \end{cases} \tag{6}$$

Where $\alpha = 1.67733$ and $\lambda = 1.0507$, the same values as proposed in [12].

To evaluate the performance of SELU compared to tanh, we trained the LSTM, BiLSTM, and BiLSTM-CNN forecasting models with both activation functions. Table 1 represents the specifications of the test models.

TABLE 1. Specifications of the Activation Function's Test models.

	LSTM	BiLSTM	BiLSTM-CNN
Batch Size	128	128	128
Training Epoch	400	400	400
Learning rate	0.001	0.001	0.001
Hidden Units	5	5	4 BiLSTM 1 CNN

The performance comparison has been made for all six QoE features. Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) are the performance criteria in this evaluation. The comparison results for all six forecasted features are presented in Fig. 5. As depicted in this figure, for most QoE metrics, the SELU activation function converges to the higher performance.

2) ARCHITECTURE

For our proposed model, by testing the different number of layers, we experimentally determined that more than four BiLSTM layers and one hidden CNN layer do not show noticeable progress in the performance. We utilized drop out after the fully connected layer, which led to a very positive effect on minimizing the overfitting between the training and validation data. The final BiLSTM-CNN model was implemented with four BiLSTM layers, one 1D-Convolutional layer, a Maxpooling layer, a fully connected layer and an output dens layer. We kept the drop-out value to 0.5 after the fully connected layer. We set the batch size for training to 128, and the adopted optimizer was Nesterov-Adam or Nadam [35]. We applied SELU for the model's activation function, which is very effective by inducing self-normalizing properties.

V. EVALUATION AND RESULTS

To evaluate the proposed model's accuracy, we trained the model with the dataset described in Section III, divided into three parts: 80% for training, 10% for validation and 10% for testing. The hybrid BiLSTM-CNN model is compared with

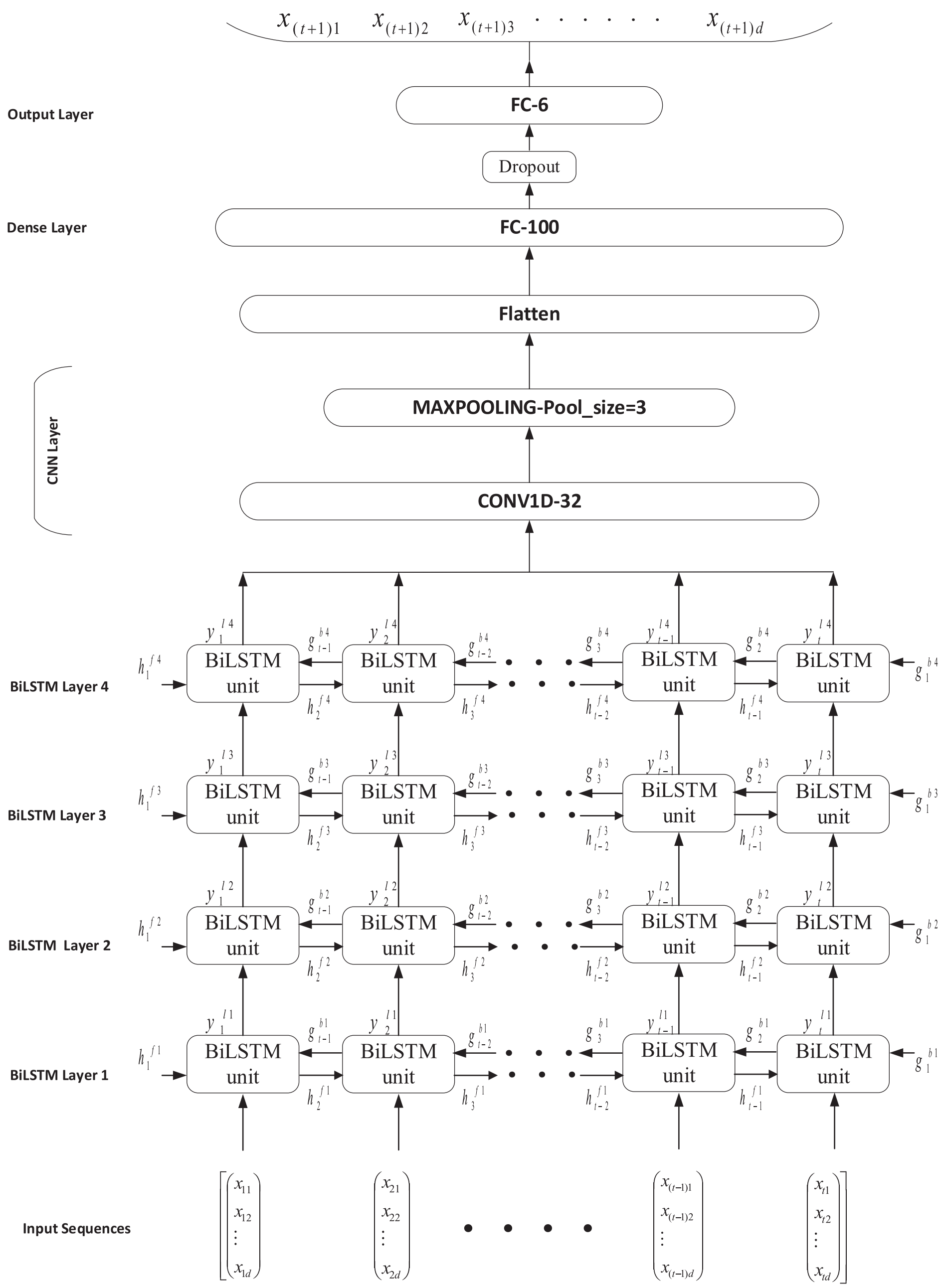

FIGURE 6. BiLSTM-CNN Architecture.

four anchor models: LSTM and BiLSTM as the state-of-the-art time-series' forecasting model, MLP which is the most basic neural network model, and the SVR as the most classic time series forecasting model. The tuned MLP has 4 hidden layers, with batch normalization layer, drop out equal to 0.2, relu activation function and adam optimizer. We have implemented the SVR algorithm using the Scikit-Learn machine learning library with the rbf kernel and scale gamma. The implementations of BiLSTM-CNN, MLP, LSTM and BiLSTM are based on the Keras library with the TensorFlow backend. As a standard practice in ML to achieve good accuracy while overcoming overfitting, we used greedy search and a wide variety of algorithms with different configurations to tune the hyperparameters. We then used the best configuration for each algorithm in our performance evaluations.

To evaluate the proposed methods' performance, we utilized MAE, a linear measure for error scoring, and also RMSE, a quadratic scoring criterion that shows how the residuals (forecasted error) spread out. MAE gives equal weight to the errors; however, RMSE penalizes the variance. In other

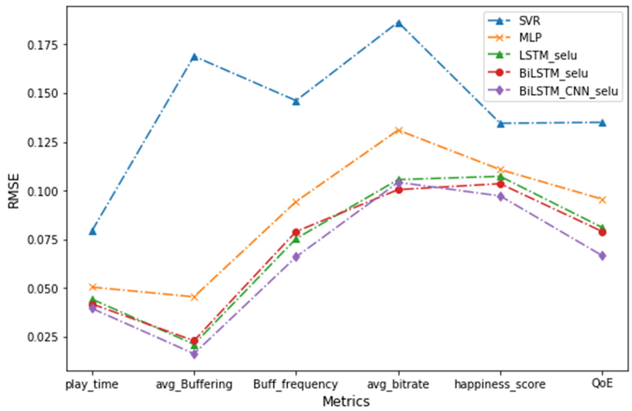

FIGURE 7. Comparison of five forecasting ML Models in terms of RMSE for each metric.

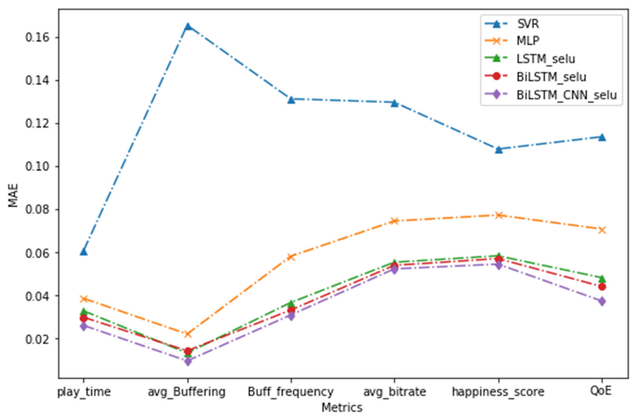

FIGURE 8. Comparison of five forecasting ML Models in terms of MAE for each metrics.

words, errors with larger absolute value will be given higher weight by RMSE. These measures' formulation is as follows:

$$RMSE = \sqrt{\frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}} (\hat{y}_i - y_i)^2} \qquad (7)$$

$$MAE\,(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} |\hat{y}_i - y_i| \qquad (8)$$

Where, y_i is the i^{th} instance or observation and $\hat{y}_i$ is its associated forecasted value. The smaller the RMSE and the MAE, the smaller the error.

For all five models, we compared the forecasting of all six input quality metrics in terms of MAE and RMSE.

Figs. 7 and 8 show the proposed BiLSTM-CNN model's evaluation results compared to the four anchor models. As shown, the worse results belong to the SVR followed by MLP. The rest of the three models have a close competition. BiLSTM, compared to LSTM, has better results for five metrics in terms of MAE and four metrics in terms of RMSE As shown for all six features, the best performance belongs to BiLSTM-CNN. The average bitrate and the happiness score have the worst scores as expected due to unavailability in some samples. Each utilized module in the proposed model helped us improve our forecasting system's performance. Accurate forecasting is vital for the next steps of video service management. Therefore, every single percentage of classification accuracy is important and might lead to noticeable business return due to providing better video QoE. To visualize the forecasted features' accuracy, we plotted the actual and the predicted QoE for all five models in Fig. 9. As we expected, from a to e,

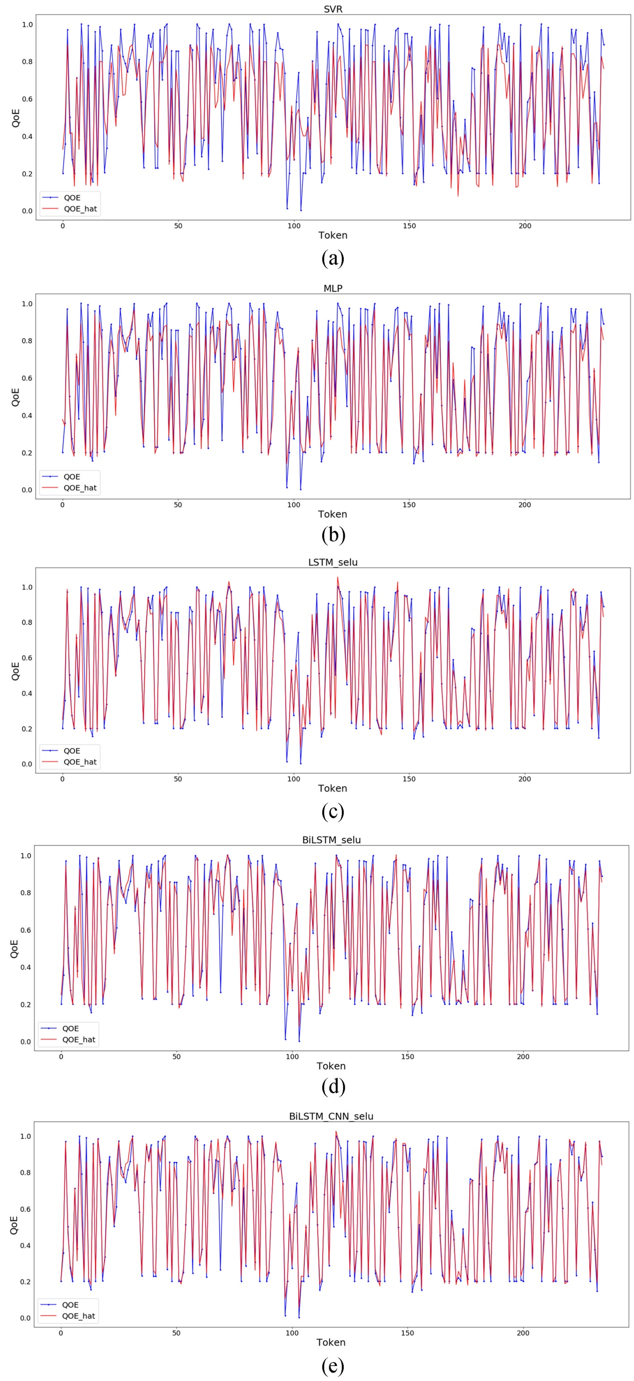

FIGURE 9. Forecasted vs actual QoE for the five ML Models. Each streamed video has a unique token.

we can see the least to most correlated forecast, respectively, with our method performing the best.

The proposed prognosticating approach gives the video service provider enough time to localize and mitigate the fault before the user notices any degradation, or at least reduces the latter's duration. In addition, the proposed forecasting process can be used in other video streaming applications, such as anomaly detection in CDN and wireless networks and resource scheduling in wireless networks for adaptive DASH delivery.

Our approach only needs the current QoE collected from the client side; thus, it is easily accessible to the video service provider and requires no access to the end-to-end path. Using the current QoE collected from the client side, the whole processing will be done in the provider's servers.

VI. CONCLUSION

It is worth noting that the existing systems estimate or measure the "current" video QoE using QoS parameters and/or QoE metrics, and we are not aware of any work that forecasts the QoE for the "near future. We proposed a ML based multivariate time series method to forecast video QoE before its effect shows at the client's screen. This approach gives more time to the management system to avoid QoE degradation and reduce customers' dissatisfaction. Evaluation results showed that our proposed BiLSTM-CNN method attains higher performance than other prominent ML methods. For future work, we plan to forecast the QoE at time steps higher than t + 1, using for example multi-time-steps forecasting. The challenge there is loss of accuracy as we forecast further and further into the future.

REFERENCES

[1] "Cisco visual networking index: Forecast and trends, 2017–2022," *Cisco*, Feb. 27, 2019. Accessed: Feb. 24, 2021. [Online]. Available: www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html

[2] "Video streaming market size 2021 industry statistics, emerging technologies, business challenges, segmentation, explosive factors of revenue expansion and strategies 2023—MarketWatch." Accessed: Jan. 22, 2021. [Online]. Available: https://www.marketwatch .com/press-release/video-streaming-market-size-2021-industry-statistics-emerging-technologies-business-challenges-segmentation-explosive-factors-of-revenue-expansion-and-strategies-2023-2021-01-13?siteid=bigcharts&dist=bigcharts& tesla=y

[3] H. Nam, K. H. Kim, and H. Schulzrinne, "QoE matters more than QoS: Why people stop watching cat videos," in *Proc. IEEE INFOCOM*, San Francisco, CA, USA, Jul. 2016, pp. 1–9.

[4] Y. Chen, K. Wu, and Q. Zhang, "From QoS to QoE: A tutorial on video quality assessment," *IEEE Commun. Surv. Tut.*, vol. 17, no. 2, pp. 1126–1165, Apr. 2015.

[5] L. Skorin-Kapov, M. Varela, T. Hoßfeld, and K.-T. Chen, "A survey of emerging concepts and challenges for QoE management of multimedia services," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 14, no. 2s, pp. 1–29, May 2018.

[6] C. V. Murudkar and R. D. Gitlin, "QoE-driven anomaly detection in self-organizing mobile networks using machine learning," in *Proc. Wireless Telecommun. Symp.*, Apr. 2019, pp. 1–5.

[7] C. Wang, "QoE based management and control for large-scale VoD system in the cloud—Chapter 5," Ph.D. dissertation, Carnegie Mellon Univ., Pittsburgh, PA, USA, 2017.

[8] H. E. Dinaki, S. Shirmohammadi, E. Janulewicz, and D. Cote, "Deep learning based fault localization in video networks using only client-side QoE," *IEEE Trans. Artif. Intell.*, vol. 1, no. 2, pp. 130–138, Oct. 2020.

[9] F. Rodrigues, I. Sousa, M. P. Queluz, and A. Rodrigues, "QoE-Aware scheduling algorithm for adaptive HTTP video delivery in wireless networks," *Wireless Commun. Mobile Comput.*, vol. 2018, 2018, Art. no. 9736360.

[10] S. Siami-Namini, N. Tavakoli, and A. S. Namin, "The performance of LSTM and BiLSTM in forecasting time series," in *Proc. 2019 IEEE Int. Conf. Big Data, Big Data 2019*, 2019, pp. 3285–3292.

[11] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *ICASSP, IEEE Int. Conf. Acoust., Speech Signal Process. Proc.*, 2015, pp. 4580–4584.

[12] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 972–981.

[13] J. Ahmed, A. D. N. Junior, C. Kilinc, D. Pan, J. R. I. Riu, and J. Gustafsson, "Using blackbox ML techniques to diagnose QoE problems for an IPTV service," in *Proc. IEEE/IFIP Netw. Operations Manage. Symp.*, 2020, pp. 1–8.

[14] H. H. Song *et al.*, "Q-score: Proactive service quality assessment in a large IPTV system," in *Proc. ACM SIGCOMM Internet Meas. Conf., IMC*, 2011, pp. 195–208.

[15] V. Vasilev, J. Leguay, S. Paris, L. Maggi, and M. Debbah, "Predicting QoE factors with machine learning," in *Proc. IEEE Int. Conf. Commun.*, 2018, pp. 1–6.

[16] R. Shalala, R. Dubin, O. Hadar, and A. Dvir, "Video QoE prediction based on user profile," in *Proc. 2018 Int. Conf. Comput., Netw. Commun., ICNC 2018*, 2018, pp. 588–592.

[17] U. Bulkan and T. Dagiuklas, "Predicting quality of experience for online video service provisioning," *Multimedia Tools Appl.*, vol. 78, no. 13, pp. 18787–18811, Jul. 2019.

[18] C. G. Bampis, Z. Li, and A. C. Bovik, "Continuous prediction of streaming video QoE using dynamic networks," *IEEE Signal Process. Lett.*, vol. 24, no. 7, pp. 1083–1087, Jul. 2017.

[19] C. G. Bampis, Z. Li, I. Katsavounidis, and A. C. Bovik, "Recurrent and dynamic models for predicting streaming video quality of experience," *IEEE Trans. Image Process.*, vol. 27, no. 7, pp. 3316–3331, Jul. 2018.

[20] "Video streaming and video selivery platform | Kaltura," Accessed: Aug. 21, 2021. [Online]. Available: https://corp.kaltura.com/products / video-delivery-video-streaming-platform/

[21] "Ixia makes networks stronger," Accessed: Feb. 24, 2021. [Online]. Available: https://www.ixiacom.com/

[22] "Nice people at work (NPAW)," Accessed: Feb. 24, 2021. [Online]. Available: https://nicepeopleatwork.com/

[23] "Route optimization and analysis explorer suite," Accessed: Dec. 26, 2020. [Online]. Available: https://www.blueplanet.com/resources/ route-optimization-and-analysis-explorer-suite-ds.html

[24] "BT.500: Methodologies for the subjective assessment of the quality of television images," ITU, Accessed: Feb. 24, 2021. [Online]. Available: www.itu.int/rec/R-REC-BT.500-14-201910-I/en

[25] "P.917: Subjective test methodology for assessing impact of initial loading delay on quality of experience," Accessed: Aug. 21, 2021. [Online]. Available: https://www.itu.int/rec/T-REC-P.917/en

[26] M. H. Pinson and S. Wolf, "A new standardized method for objectively measuring video quality," *IEEE Trans. Broadcast.*, vol. 50, no. 3, pp. 312–322, Sep. 2004.

[27] D. Z. Rodriguez, J. Abrahao, D. Begazo, R. L. Rosa, and G. Bressan, "Quality metric to assess video streaming service over TCP considering temporal location of pauses," *IEEE Trans. Consum. Electron.*, vol. 58, no. 3, pp. 985–992, 2012.

[28] "G.1070: Opinion model for video-telephony applications," Accessed: Aug. 21, 2021. [Online]. Available: https://www.itu.int/rec/T-REC-G. 1070-201806-I/en

[29] C. G. Bampis, Z. Li, A. K. Moorthy, I. Katsavounidis, A. Aaron, and A. C. Bovik, "Study of temporal effects on subjective video quality of experience," *IEEE Trans. Image Process.*, vol. 26, no. 11, pp. 5217–5231, Nov. 2017.

[30] A. J. Greene, C. Prepscius, and W. B. Levy, "Primacy versus recency in a quantitative model: Activity is the critical distinction," *Learn. Memory*, vol. 7, no. 1, pp. 48–57, Jan. 2000.

[31] M. S. Anwar, J. Wang, A. Ullah, W. Khan, S. Ahmad, and Z. Fei, "Measuring quality of experience for 360-degree videos in virtual reality," *Sci. China Inf. Sci.*, vol. 63, no. 10, pp. 1–15, Oct. 2020.

[32] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[33] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *2015 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Apr. 2015, pp. 3156–3164.

[34] D. Wang and E. Nyberg, "A long short-term memory model for answer sentence selection in question answering," in *ACL-IJCNLP 2015 – 53rd Annu. Meeting*, 2015, pp. 707–712.

[35] T. Dozat, "Incorporating Nesterov Momentum into Adam," in *Proc.Workshop Track (ICLR)*, 2016, pp. 1–4.

HOSSEIN EBRAHIMI DINAKI (Student Member, IEEE) received the Ph.D. degree in electrical and computer engineering from the Department of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, ON, Canada, in 2021. His research interests include applied AI on QoE-aware resource management, network diagnosis, and cloud gaming. He has expertise in evolutionary algorithms, deep artificial neural networks, and data science. He is currently working on productionizing machine learning (MLOps) for NLP applications. Also, he has professional experience in frequency and network planning for Digital Video Broadcasting. During his Ph.D., he was awarded the University of Ottawa Excellence Scholarship, QEII Scholarship, Admission Scholarship and Mitacs Accelerate internship at Ciena.

SHERVIN SHIRMOHAMMADI (Fellow, IEEE) received the Ph.D. degree in electrical engineering from the University of Ottawa, Ottawa, ON, Canada, in 2000. He is currently a Professor with the School of Electrical Engineering and Computer Science, University of Ottawa. He is the Director of the DISCOVER Lab, doing research in measurement methods and Applied AI for multimedia and networking systems. The results of his research, funded by more than $26 million from public and private sectors, include 400 publications, three best paper awards, more than 70 researchers trained at the postdoctoral, Ph.D., and master's levels, 30 patents and technology transfers to the private sector, and a number of awards. He is the Editor-in-Chief of the IEEE Transactions on Instrumentation and Measurement, and was also an Associate Editor-in-Chief of the *IEEE Instrumentation and Measurement Magazine* in 2014 and 2015, and is currently on its editorial board. He has been an IEEE Instrumentation and Measurement Society AdCom Member since 2014, served as the Vice President of its Membership Development Committee from 2014 to 2017, and was a Member of the IEEE I2MTC Board of Directors from 2014 to 2016. Dr. Shirmohammadi is an IEEE Fellow for contributions to multimedia systems and network measurements, winner of the 2019 George S. Glinski Award for Excellence in Research, a Senior Member of the ACM, a University of Ottawa Gold Medalist, and a licensed Professional Engineer in Ontario.

EMIL JANULEWICZ received the Master of Applied Science degree in electrical engineering from Carleton University, Ottawa, ON, Canada, in 2010 and completed Graduate level courses in computer science from McGill University, Montreal, QC, Canada, in 2015. His research interests include reinforcement learning, information theory, and advanced graphical model techniques. From 2010 to 2016, he was a Service Professional for Ericsson Canada Inc., helping introduce the latest LTE core networks into the field all over the U.S. Since then, he has been with Ciena and is currently a Network Architecture Advisor to the Office of the CTO. He was the recipient of the Ontario Graduate Scholarship and the Ontario Graduate Scholarship in Science and Technology. His work with Carleton University was awarded the University Medal for Outstanding Graduate Work at the Masters level.

DAVID CÔTÉ received the Ph.D. degree in physics from Université de Montréal, Montreal, QC, Canada, in 2007. He is currently the Chief Data Scientist with Blue Planet Division, Ciena and Co-Founder of Blue Planet Analytics Program, Ciena. His group has developed and productized several machine learning applications recognized by industry innovation awards and commercial success. He also participates in several research projects with industrial and academic partners and holds numerous patents and publications in the field. Prior to joining Ciena, he was a Particle Physicist for 14 years with Stanford Linear Accelerator Center (USA) and CERN (Switzerland), notably, where he gained substantial hands-on experience with big data engineering, data science, and world-class research.