Journal of Network and Systems Management (2021) 29:29
https://doi.org/10.1007/s10922-021-09595-8

A New Application-Aware No-Reference Quality Assessment Method for IP Voice Services

Péter Orosz[1] · Tamás Tóthfalusi[1]

Received: 14 June 2018 / Revised: 17 February 2021 / Accepted: 24 February 2021 /
Published online: 16 March 2021

Abstract

The increasing number of Voice over LTE deployments and IP-based voice services raise the demand for their user-centric service quality monitoring. This domain's leading challenge is measuring user experience quality reliably without performing subjective assessments or applying the standard full-reference objective models. While the former is time- and resource-consuming and primarily executed ad-hoc, the latter depends upon a reference source and processes the voice payload that may offend user privacy. This paper presents a packet-level measurement method (introducing a novel metric set) to objectively assess network and service quality online. It is accomplished without inspecting the voice payload and needing the reference voice sample. The proposal has three contributions: (i) our method focuses on the timeliness of the media traffic. It introduces new performance metrics that describe and measure the service's time-domain behavior from the voice application viewpoint. (ii) Based on the proposed metrics, we also present a no-reference Quality of Experience (QoE) estimation model. (iii) Additionally, we propose a new method to identify the pace of the speech (slow or dynamic) as long as voice activity detection (VAD) is present between the endpoints. This identification supports the introduced quality model to estimate the perceived quality with higher accuracy. The performance of the proposed model is validated against a full-reference voice quality estimation model called AQuA, using real VoIP traffic (originated in assorted voice samples) in controlled transmission scenarios.

Keywords Voice service · Performance metrics · VoLTE · VoIP · QoE · Voice call quality · PESQ · POLQA

✉ Péter Orosz
orosz@tmit.bme.hu

✉ Tamás Tóthfalusi
tothfalusi@tmit.bme.hu

[1] Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics, Budapest 1117, Hungary

1 Introduction

Regarding the latest telecommunication standards [1, 2], the IP-based voice services became the dominators of global voice communication in the last decade. The new technologies and architectures, i.e., Voice over LTE (VoLTE), Voice over IP (VoIP), offer improved service quality and high-definition real-time voice services. This technological evolution raised users' expectations against interactive voice services. Transmitting voice data real-time end-to-end over an IP-based network path, which is the current operational model of the mentioned services in the near past, defines strict criteria. The requirement against the management tasks includes the control of end-to-end delay, delay variation, and packet reordering. From the telecommunication operator and service management point of view, the main task is to inspect the quality of the provided voice service and acquire live feedback about the state of the IP-based real-time voice transmission. The most crucial supervision task is the analysis of the listening quality. Operators have multiple options to monitor whether it complies with the raised expectation against user experience.

The service provider can monitor the signaling plane or the voice traffic and measure the classic packet-level QoS parameters (e.g., delay, delay variation, packet loss, and reordering). While this method reveals primary transmission impairments, it does not give a result that is based on the viewpoint of the voice application. The analysis of the classic packet-level metrics does not consider the effect of the application-level packetized media and jitter buffer properties that directly influence the listening quality.

To analyze the degradation of the waveform, the provider can post-process the impaired voice sample using reference-based objective quality standards (e.g., PESQ [3], POLQA [4]). The primary disadvantages of these models are (a) offline operation (i.e., post-processing), (b) requirement for the original voice sample (as a reference), and as a fundamental privacy issue, (c) analyzing of user data, i.e., decoding the whole voice conversation for quality inspection.

The commonly used network protocol for delivering the audio samples within the previously mentioned ecosystems is the Real-Time Transport Protocol (RTP) [5]. It travels over IP/UDP protocols to carry the media flow end-to-end. RTP applies a unique timestamping mechanism combined with continually increasing sequence numbers to convey the time-sensitive media samples over packet-switched systems. The main features of the protocol are the ability to detect packet losses and recover orderliness and timing at the receiver side. The latter feature is based on a special timestamp format derived from the voice codec's internal sampling frequency.

As a shared property, the IETF RFC recommendations [6–12] separate the delay, reorder, and loss concept to measure the traditional packet-level QoS metrics. As we will show in this paper, in real-time media transmission, it may have several benefits to handle these parameters side by side.

This paper aims to present VoicePerf (Voice Service Performance Measurement Metrics), a new listening quality inspection model for real-time RTP flows. VoicePerf considers the application-level media buffering properties and inspects the playout timing properties of the arrived packets. It operates without processing the voice payload (i.e., the private media data) and the need for the corresponding reference voice material. Instead of analyzing media data directly at the endpoint's playout buffer, we model its operation and estimate its state. This approach enables to detach the evaluation from the user equipment and implement it at any point of the IP voice network.

Based on the new metric-set, we propose a derivative objective quality model that estimates the voice session's perceived quality on the Mean Opinion Score (MOS) scale. The novelty of the proposed QoE estimation model is threefold: (i) it introduces new performance metrics that enable measuring and describing the time-domain behavior of the service from the viewpoint of the voice application. (ii) Based on the proposed metrics, we also introduce a no-reference Quality of Experience (QoE) estimation model. (iii) Additionally, we propose a new method to detect the pace of the speech (dynamic or slow) when voice activity detection (VAD) is present between the endpoints. This identification supports the introduced quality model to estimate the perceived quality with higher accuracy.

The remainder of the paper is organized as follows. Section 2 presents the related works in the field of voice quality analysis. Section 3 introduces our new method focusing on the base metrics, and Sect. 4 describes the derived QoE estimation model. In contrast, Section 5 presents 480 test cases and validation results to demonstrate the effectiveness of our model. Finally, Section 6 concludes the paper.

2 Related Work

Examining the previous publications and standards, the E-model [13] is a widespread no-reference voice quality estimation method. It is an ITU-T standard for subjective quality estimation, registered as Recommendation G.107. The E-model applies several input parameters, which are derived from previously known network impairment probability (e.g., packet loss probability) or waveform-based parameters (e.g., talker echo, loudness, and noise). The standard results in an R-value, which could be translated into the MOS [14] scale.

Several papers analyzed the efficiency of the E-model to inspect the user perception and found that it is not accurate enough for estimating subjective quality [15]. It underestimates the audio quality in a longer transmission delay scenario [16] and does not consider the consecutive media frame loss [17].

Raake et al. [16] proposed a new parameter set for the E-model and enhanced the original algorithm with two new inputs: random loss and burst ratio of the packet loss. The presented work also defined two viewpoints for packet loss: macroscopic loss (speech quality changes over time) and microscopic loss (the effect of packet loss at the decoder side). The loss prediction method has been adopted in the E-model standard.

Takahashi et al. [15] also proposed a G.107 standard-based work for quality inspection. The authors found a correlation between network delay and speech distortion. They also examine the problem of terminals' loss rate. Regarding the implementation of the jitter buffer, it can differ in each terminal type, which results in different packet loss rates. Thus, a dedicated calibration data file (describing the characteristics of the jitter buffer) should be used to support the model [18].

Takahashi et al. [19] examined further the previous problem space [15] and analyzed the relationship between delay effects and speech distortion when both coincide. The authors defined an E-model-based method to derive a subjective MOS from the R-value. The proposed quality estimation model uses the following input parameters: one-way-delay, echo-path loudness rating, noise level, and equipment impairment factor. It considers the applied audio codec and the packet loss rate as known in advance.

The second group of the related works applies a reference waveform or a full-reference standard (e.g., PESQ [3], POLQA [4]) to analyze the degraded voice material.

Jung and Manzano [17] analyzed the impact of consecutive packet loss. The authors also found that the E-model does not consider the burst packet loss. According to this observation, the publication introduces burst-related metrics (duration and density metrics for burst and gap events). Their experimental results showed that the defined loss parameters correlate with the result of the PESQ standard. The authors defined multiple loss-range categories for MOS estimation. They found that under 7% of packet loss, there is no benefit, in terms of accuracy, including the burst in the estimation formula.

Ouyang et al. [20] implemented an Android App for VoLTE service quality inspection. The proposed method analyzes the wireless plane of the communication path and applies the POLQA standard within an initial calibration step. The new model includes two phases: a training phase and a testing phase. In the training phase, the mobile phone is directly connected to an external box to calculate MOS indices based on reference waveforms. In the testing phase, the client operates in the background and periodically transmits network metrics to a database server for further examination.

Zou et al. [21] proposed a machine learning algorithm (random forest-based training and assessment) for voice quality estimation. The algorithm uses nine input parameters during the quality estimation phase (e.g., UDP length, bitrate, and average packet loss). The authors analyzed 2400 degraded voice samples and compared the PESQ scores as a reference. The statistical results showed that the correlation between the proposed method and PESQ is higher than between PESQ and the E-model.

Conway [22] published a passive assessment model that combines the offline algorithm of the PESQ standard [3] and an audio sample replacement technique to exchange the original user payload with a test audio source. Using this payload injection method, the author avoided any privacy issues raised by decoding user's conversation. However, the presented model does not reflect the behavior of the application's jitter buffer. Therefore, the real data loss present at the input of the voice decoder is not incorporated in the result of the quality assessment.

Han and Muntean [23] proposed HCQ, a hybrid model for call quality assessment. The authors combined the advantages of the reference-based (e.g., PESQ, POLQA) and no-reference-based (e.g., ITU-T G.107) standards and mixed them into one method. HCQ operates with two MOS values, which are calculated in the final processing phase. The first value is the result of the online model, and an offline reference-based model estimates the other one. The offline MOS estimation is based on the PESQ standard, where the method records a few seconds of the voice and inspects its quality. The disadvantage of this model is that the recorded audio samples are sent through the network for further analysis, which is not a viable scenario in many cases. Regarding the offline calculation phase, the authors recommend a reduced reference waveform-based method.

Sun and Ifeachor [24] introduced a new passive non-intrusive model for developing accurate solutions to predict voice quality for IP-based voice services. Based on the methodology, the authors defined regression models for predicting conversational voice quality for four widespread audio codecs: G.729, G.723.1, AMR, and iLBC. Using real VoIP traces, the authors showed that the prediction accuracy of the generated models is close to the combined ITU PESQ/E-model method.

Lin et al. [25] proposed a new parametrical neural network-based model to analyze the audio quality in VoIP services. The publication complements the methodology presented in [24]. The major impact of their work is the statistics-based packet loss assessment. The presented method is more efficient computationally than [24] since it does not require Markov model mapping.

Let us note that both [24] and [25] omits the impact of packet reordering and packet losses derived from the overrun of the playout buffer.

Majed et al. [26] examined the network delay in VoLTE systems as a model input parameter. The authors found that 3GPP metrics (e.g., IPDV) do not represent the behavior of jitter-buffers in real VoLTE services. They defined a set of improvements to achieve a higher correlation.

Abareghi et al. [27] published an improved ITU-T standard, namely P.563 [28]. This standard realizes a non-intrusive model for voice quality evaluation, but it is not appropriate to examine IP-based conversation (e.g., VoIP calls). The authors presented a new distortion class for proper network condition detection.

Broom [29] found that the system characteristics should be utilized as an impairment factor for MOS estimation. The proposed work concludes that the packet loss and jitter values are insufficient for estimating voice quality. The author introduced a calibration method to measure the characteristics of the VoIP equipment. Using this device- specific parameter, a higher correlation rate is achievable.

Luksa et al. [30] represented that the audio codec (G.711, GSM, Speex, iLBC 20, iLBC 30, and G.729) has no effect on the intelligibility of voice performance and does not influence the objective voice quality.

Orosz and Tóthfalusi proposed a metric-set and a preliminary objective model to measure voice service quality online [31]. The number of voice scenarios used for building the model and perform the validation was 34. Based on these early results, our current paper introduces an enhanced metric set and a new, more generic estimation model that is created by involving 480 voice samples (scenarios) and validated using the k-fold cross-validation method.

In contrast to previously published works, our proposed method distinguishes the packet loss events, defines three loss-categories, and applies a time windowing technique to model the endpoint's playout buffer behavior. Additionally, it is enhanced with a new method to detect the pace of the speech (dynamic or slow) when voice activity detection (VAD) is present between the endpoints. This identification enables weight packet loss differently for dynamic and slow speech segments. This auxiliary method assists the primary model to estimate the perceptive listening quality with higher correlation.

3 The Proposed Quality Model

3.1 Voice Codec and Transport Protocol Background

To maintain low end-to-end delay, RTP operates with a small-sized application-level receiver buffer, namely a playout or (de-) jitter buffer, to smooth out delay variation and perform reordering when needed. The recovery processes (jitter elimination and packet reordering) are limited by the effective size of this buffer. Typically, real-time voice codecs release 10–30 ms voice frames at their outputs. 20 ms frame size is the most common choice for VoLTE and VoIP codecs such as Adaptive Multi-Rate Wideband (AMR-WB) [32], Opus [33], and Speex [34]. Accordingly, the average extent of the jitter buffer is usually in the range of 60–140 ms, which provides temporal storage for a few consecutive media frames only.

The proposal is optimized for passive network measurement at an appropriate aggregation point of the network (typically, in the IP Multimedia Subsystem (IMS) network core where RTP packets are unencrypted). At this high traffic density point of the network, RTP voice conversations can be detected and measured concurrently with a dedicated monitoring device. In contrast, monitoring RTP traffic at the receiver endpoint may require additional computing resources. From the viewpoint of a voice service provider, it is more common and reasonable to perform service monitoring in the IMS core network, where the voice sessions also appear and RTP header fields can be accessed. Our method invokes only two protocol metadata, i.e., the sequence number to measure packet orderliness and the packet timestamp to analyze timeliness.

To examine the behavior of real-time media flows from the application's perspective, we have to overview the operation of the RTP protocol. RTP supports media delivery by recovering order and time information at the receiver end. These protocol features enable scheduled decoding of media samples delivered through the network. Each RTP header contains a sequence number and a timestamp value. While the sequence number is incremented by one for each packet and used to detect loss or reordering, the timestamp carries codec-related temporal information and is used to schedule the arrived packets in time. For voice services, RTP timestamp is incremented in every packet by the number of audio samples it carries. When a packet contains 20 ms of audio sampled at 8 kHz, the timestamp is incremented by 160 in each RTP packet (see Fig. 1), for example. Let us note that the initial value of the sequence number is random within the 16-bit range.

Let S_i be the sequence number of the i^{th} arrived packet, and let P_i be the packetizing period of the used codec.

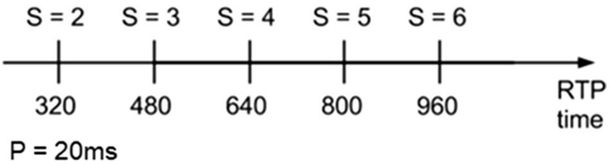

Fig. 1 Timestamp increment in RTP packets

3.2 Introducing the Packet-Level VoicePerf Metrics

Based on the media recovering mechanism above, VoicePerf applies a reference timeline, on which the current time is incremented by the delta arrival time between two consecutive packets. The delta time is calculated from the local timestamp of the arrived packets. The reference timeline is used to analyze the arrived packet's relevance in time.

Let t_i be the arrival time and let d_i be the delta time of the *ith* RTP packet from the previously arrived packet. Let *Ct* be the current time that represents a time relative to the beginning of the measurement. When a media frame arrives, the d_i value is calculated and added to the current time *Ct* (Fig. 2).

To inspect the arrived packet based on its expected arrival interval, VoicePerf calculates time windows for the classification and the metric calculation. Our proposed model defines six reference points in time to handle five time windows. The reference points and thus the time windows are re-calculated for each arrived packet in real-time and fitted on the reference timeline to determine the performance metrics (see Fig. 3). The width of a time window represents the duration of the voice sample within a media frame (20 ms, typically).

Let Rp_{ij} be the *jth* reference point of the *ith* arrived packet, where $1 \leq j \leq 6$. This model considers the P_i parameter as available information in each calculation period. When the RTP stream involves multiple payload types (i.e., multiple codecs and sampling period combinations), P_i should be properly synchronized to the calculation cycles. The sampling rate could be pre-defined if it is previously known from the codec properties. Otherwise, it could be indirectly measured during the initial phase of the algorithm. (The initial phase will be discussed later in this section). Let us note that the resolution of P_i should correspond with the resolution of *Ct*. (In Sect. 4, we applied P_i with microsecond resolution.)

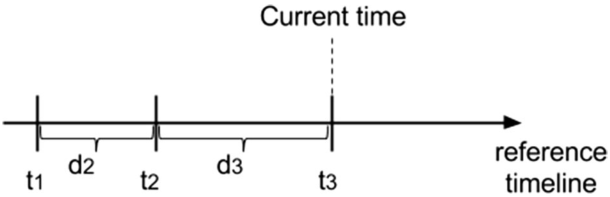

Fig. 2 Reference time incremented by delta arrival times

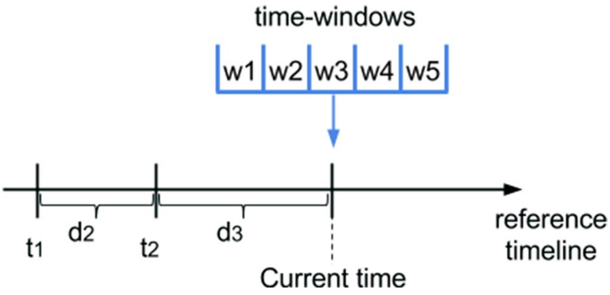

Fig. 3 Time window fitting on the reference timeline

Let T be a threshold value that defines the overall interval of the time windows. Its value is determined by the playout buffering scheme of the voice application. T is practically an integer value, and $T \geq 2$. The T threshold makes the method scalable for any playout buffer size. $T = 2$ case assumes a jitter buffer of 100 ms. According to the recommendation of the International Telecommunication Union (ITU), the one-way delay between the communication user equipment should be kept under 150 ms (including propagation delay as well as buffering delay). Most of the common voice software applications have a default jitter buffer of around 80–120 ms.

Using the previously defined T parameter, Rp_{ik} reference points are calculated by formulae (1–6), respectively:

$$Rp_{i1} = \left(S_i \times P_i\right) - ((T + 0.5) \times P_i) \tag{1}$$

$$Rp_{i2} = \left(S_i \times P_i\right) - \left(1.5 \times P_i\right) \tag{2}$$

$$Rp_{i3} = \left(S_i \times P_i\right) - \left(0.5 \times P_i\right) \tag{3}$$

$$Rp_{i4} = \left(S_i \times P_i\right) + \left(0.5 \times P_i\right) \tag{4}$$

$$Rp_{i5} = \left(S_i \times P_i\right) + \left(1.5 \times P_i\right) \tag{5}$$

$$Rp_{i6} = \left(S_i \times P_i\right) + ((T + 0.5) \times P_i) \tag{6}$$

Rp_{ik} reference points define five time windows (w_1, w_2, w_3, w_4 *and* w_5) based on the following intervals:

$$Rp_{i1} \leq w_1 < Rp_{i2}$$

$$Rp_{i2} \leq w_2 < Rp_{i3}$$

$$Rp_{i3} \le w_3 < Rp_{i4}$$

$$Rp_{i4} \le w_4 < Rp_{i5}$$

$$Rp_{i5} \le w_5 \le Rp_{i6}$$

The width of the three middle time windows (w_2, w_3, *and* w_4) is fixed and determined by the sample length within a voice frame. The two outer windows (w_1, w_5) specify adjustable intervals that could be increased by the T threshold value.

Rp_{ik} reference points are updated for each arrived packet, and time windows are slid accordingly. Then they are fitted on the reference timeline to perform classification and to calculate the performance metrics. Figure 4 presents an example calculation: *T* = *2 (100 ms jitter buffer)*, *S* = *5*, *P* = *20* and *Ct* = *104*.

VoicePerf takes the arrived sequence number S_i and the packetizing period P_i to determine the relevant time windows w_1–w_5 and the expected arrival interval for the current voice frame (see Formulae 1–6). The classification categories are defined via the corresponding time windows. Classifying a packet to a pre-defined category is a 1:1 association, where categories correspond to performance metrics (Fig. 5).

Our method has an initialization phase followed by an assessment phase. During the initial phase, the reference timeline and the packetizing period should be appropriately adjusted. A reference timer is typically a common local clock source (e.g., TSC, HPET), which is used to timestamp the arrived packets. However, the reference timeline is initialized by the $S_0 \times P_0$ product. If reordering or loss occurs in the init phase, the reference timeline must be re-initialized by the next in-order packet. The method maintains the initialization phase until the packetizing period is fixed and no out-of-order packet arrives for at least a $(4 + 2T)P_i$ period. This criterion

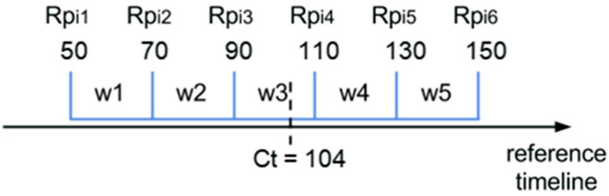

Fig. 4 Time windows in [ms] in the case of *S* = *5*, *P* = *20*, *Ct* = *104*, *T* = *2*

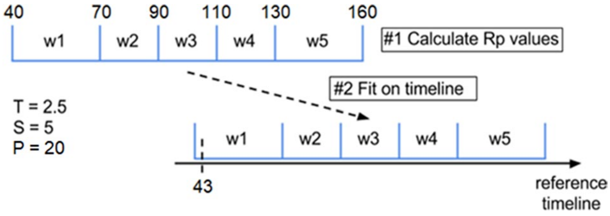

Fig. 5 Time windows in the case of *S* = *5*, *P* = *20*, *Ct* = *43*, *T* = *2.5*

grants that the values of the $\mathrm{RP}_{\mathrm{ik}}$ reference points will be higher than the initial value of the timeline. As soon as the above conditions are met, the method steps into the assessment phase. The assessment phase relies on the previously introduced time windows, applying the categories and the corresponding performance metrics.

VoicePerf defines four categories for the arrived packets, which are differentiated by the time windows. Furthermore, it specifies an additional category for packets that have not arrived and are therefore considered lost (namely, not-arrived-loss). Derivative performance metrics are the *number of packets in* $<category_k>$*/call* or the *percent of packets in* $<category_k/call>$.

- Early arrived loss (EAL)
- Late arrived loss (LAL)
- Time window offset
- Intra-window delay variation (IwDV)
- Not-arrived loss (NAL)

In the rest of the paper, we apply the abbreviated format for indexing the parameters.

3.3 Loss Category Types

Since voice codecs and thus RTP operate with time frames, the decoding algorithm always has to process a voice frame sequence. When a packet comes earlier than expected, the media application stores its payload in the playout buffer. When a packet with an early arrival faces a filled buffer, the application may discard its payload (i.e., the voice frame) despite the packet arrived at the endpoint. This is because its playout time is far from the current time window. VoicePerf recognizes the mentioned behavior and defines the early-arrived loss metric to represent the media frames that arrived too early. Using the time window fitting, the packet is counted as early arrived loss when $Ct<Rp_{i1}$.

Similarly, the method also differentiates the late arrived loss event. When a voice frame arrives too late from its expected interval, the application may handle the missing time information with different techniques because the received audio information is irrelevant in time. VoicePerf counts the *ith* packet as a late arrived loss when $Ct>Rp_{i6}$.

When Ct of the *ith* arrived packet is greater than or equal to Rp_{i1} or less than or equal to Rp_{i6}, the packet is categorized as time window offset. The reason is that the application may use the carried voice frame and places it in the playout buffer. VoicePerf also counts each packet within a measurement period with the appropriate time window counter. The variance of arrivals within the time windows can be an early indication of a transmission timing problem.

3.4 Measuring the Window-Based Timing Offset

Based on the result of the continuous window fitting, a time window offset is calculated for each arrived packet. This offset represents the deviation of the currently

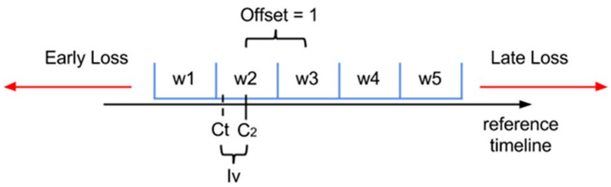

Fig. 6 VoicePerf metrics, based on the time windows

assigned time window from the expected (optimal) w_3 window. For example, when an arrived packet is assigned to w_2, its offset is 1. The offset is a signed integer value and could reveal network timing or routing misbehavior.

Since VoicePerf operates with timestamps and the C_n center point of each time window is given, the $|Ct - C_n|$ formula properly defines the intra-window delay variation (IwDV) metric. Let us note here that we advisedly avoid using the term *jitter* in this context. IwDV_n (where n is the index of the time window) is a more precise measurement of timing error (typically in microsecond resolution) within time window w_n. Both the offset and the intra-window delay variation are calculated in each measurement period.

3.5 Representation of the Categories

Figure 6 summarizes the four metrics defined by the windows: early-arrived loss, late-arrived loss, time window offset, and intra-window variation.

It should be noted that the not-arrived-loss metric is also determined in the measurement phase, but it is calculated only from the sequence number and the T threshold.

Based on the time windows, Ct exactly defines the relationship between the packet and the playout buffer usage. Generating a histogram from the metric counters represents graphical feedback about the network conditions besides buffer usage from the application's perspective. Figures 7, 8 and 9 show synthesized examples.

Determining sub-windows means that the pre-defined time windows are split into sub-intervals, which enables to represent low-level timing properties of the voice call. The time window histogram represents a rough overview, but we get a higher resolution view of arrivals by applying sub-windows (Fig. 8). The reason is that the aggregated counters often veil slight variations. Since loss categories cover large intervals and w_1, w_5 exterior windows are defined by the TPi threshold; the sub-windows should be used only for w_3 or the middle three (w_2, w_3, w_4) windows.

Figure 9 represents an example of good service quality since the arrived packets are mainly grouped to the w_3 window.

In a rare but realistic scenario when a network device introduces an additional fixed delay (e.g., due to hardware reconfiguration), the categorization of the incoming voice packets may drive to a constant offset. Our method detects a continuous

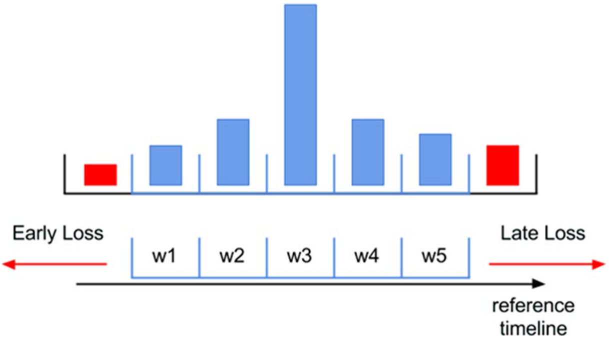

Fig. 7 Live chart about the packet classification

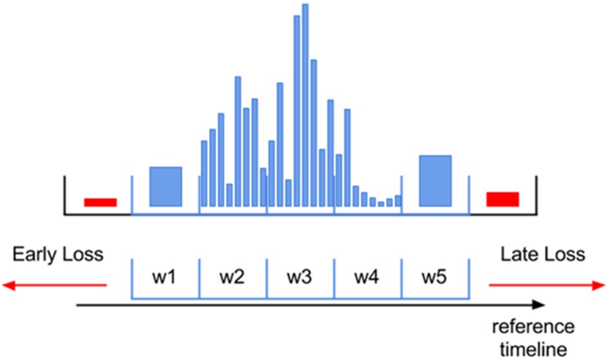

Fig. 8 Applying sub-windows on time windows

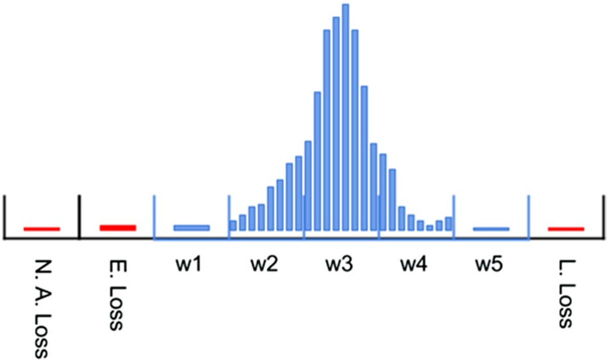

Fig. 9 Drawing histogram based on the sub-windows and categories, in case the of an optimal service operation

shifting based on the time window counters, and a reset event is effectuated by stepping back to the initial phase.

4 Estimating Perceptual Quality Using the Proposed Metrics

In the next phase, we aimed to define a quality model derived from the VoicePerf metrics, which can estimate Quality of Experience on the common P.800 MOS-scale (see Sect. 4.4 for a detailed introduction). This quality index represents the overall performance of the call session. It is important to note that by calculating QoE, the primary aim is to provide an indication of how a network path complies with the requirement of delivering real-time voice traffic end-to-end. Typically, a low-level degradation of the transmission, in terms of timing, can be eliminated by the receiver side jitter buffer, and therefore the perceived service quality may be unaffected. Nevertheless, such low-level impairments can be forerunners for a negative trend within the network path (i.e., evolving congestion) that will degrade user experience in the very near future. Meanwhile, there are various transmission errors the jitter buffer cannot cope with and therefore result in voice quality degradation at the application-level.

During our preliminary work, we found a correlation between VoicePerf metrics and service QoE. To define appropriate functions for the QoE estimation model, we had to profile the loss categories by applying the Audio Quality Analyzer software (AQuA) [35] from Sevana.

There are several test methodologies to measure perceptual quality degradation in a voice material [36]. One of the well-known algorithms is the Perceptual Evaluation of Speech Quality (PESQ), which is standardized as ITU-T P.862 recommendation [3]. PESQ applies a full-reference (FR) model to estimate the quality degradation and calculate a Mean Opinion Score (MOS) [37] value. FR means that the algorithm is based on a reference speech signal, which is compared to the degraded waveform.

A more recent ITU-T standard is the Perceptual Objective Listening Quality Analysis (POLQA) model [4], which applies new methods for wideband and super-wideband voice signals.

The AQuA software is an industrial product that includes and utilizes the features of the previously defined models, but it developed further to handle some of their weaknesses.

The AQuA software is an easy-to-use voice quality tester. It also implements an FR model, which provides quality analysis between audio files. The software calculates an estimated P.800 MOS value, a classic indication of perception, and a quality percentage value, both derived from the waveform. The latter, i.e., the quality percentage calculation procedure, is based on the analysis of spectrum vibration, energy distortion, and other waveform-specific parameters. AQuA uses a different perceptual model than PESQ to reveal more information about the loss of voice quality. This is because there are cases of degradation that PESQ fails to detect [38].

To give a comparative summary, Sevana collected the features of the well-known speech quality calculation methods, i.e., PESQ, POLQA, and AQuA [39].

4.1 Speech Pace: Slow and Dynamic Speech Types

We shortly discuss the voice activity detection (VAD) method commonly used in IP-based voice services to identify the slow and dynamic speech categories. VAD is an essential tool to reduce the voice bit rate in VoLTE and VoIP communication. It recognizes the silent periods of the conversation and marks the beginning and the end of the suppressed intervals accordingly. While each voice codec implements its own silence suppression algorithm, they have the following common properties. During a silence interval, the sender does not transmit voice frames; meanwhile, the decoder at the receiver-end generates comfort noise for the listener. This is a typical scenario when one of the participants just listens without saying anything. The recognition of these suppressed periods supports the quality model to obtain a higher accuracy estimation of the perceived quality.

Therefore, the fundamental task is the identification of the speech type without processing the voice data. Accordingly, we propose a real-time method that relies on the relationship between variation of voice packet rate, voice/silence ratio and speech type. The packet rate is constant (50 packet per second, typically) as long as the voice coder is not in silence state; we can detect the ratio of voice and silence states as per time unit. Since the codec significantly reduces the packet rate in suppression mode, we propose a method to identify the speech type based on the effective packet rate (corrected by the measured loss) versus the maximum packet rate (i.e., the constant 50 pps packet rate of the voice mode). We show real-life speech examples in Figs. 10 and 11.

According to the analysis of 12 selected voice samples (Table 1), the average effective packet rate is 0.883×50 pps for the dynamic speeches and 0.703 for the slow ones. Subsequently, we set up a 0.8 threshold to automatically differentiate the two defined types. The selection criteria of the voice samples were the n number of (at least t second long) silence blocks with parameters set to $n=10$ and $t=1$. Using

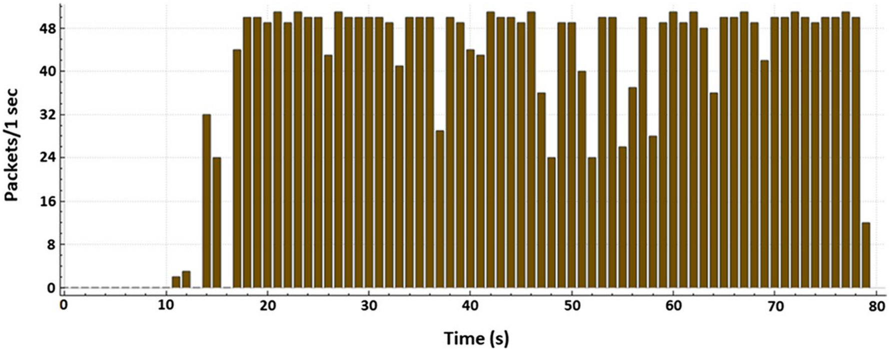

Fig. 10 Packet rate pattern for dynamic speech

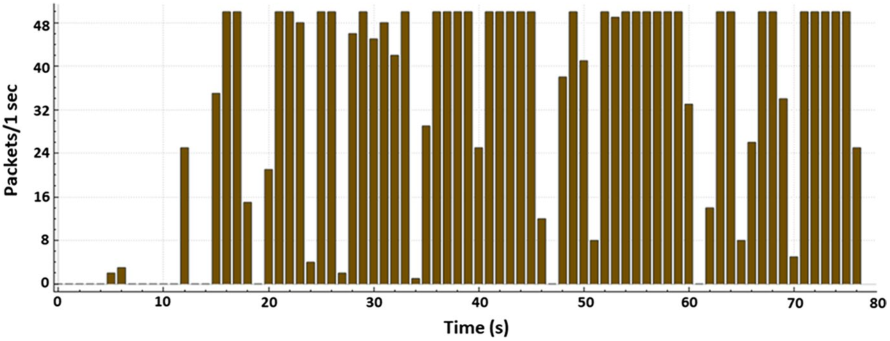

Fig. 11 Packet rate pattern for slow speech

Table 1 Speech type versus effective packet rate (pps)

Speech type	#1	#2	#3	Average
Male—dynamic	0.877	0.872	0.904	0.88433
Male—slow	0.726	0.767	0.653	0.71533
Female—dynamic	0.883	0.869	0.893	0.88167
Female—slow	0.722	0.619	0.732	0.691

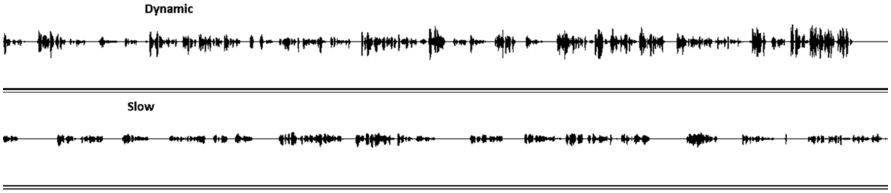

Fig. 12 Examples: audio waveforms of dynamic and slow voice samples classified by our method with parameter n and t set to 10 and 1, respectively

these criteria, we have analyzed 40 various voice samples, all 1 min long. Figure 12 shows examples of both slow and dynamic speech samples.

4.2 Measurement Infrastructure and Scenarios

To determine weights for the loss categories based on the AQuA results, we generated 480 audio files with various impairment parameters (described later in this subsection).

We used two × 86–64 workstations with Ubuntu (v16.04 LTS) operating system as caller and callee (see Fig. 13) and the NetEM kernel module [40] in both directions between them to emulate network impairments. We applied Ekiga (v4.0.1) [41] to perform the VoIP calls and arecord [42] to redirect and record the audio files. We generated 480 test cases (160 for each loss type) derived from slow

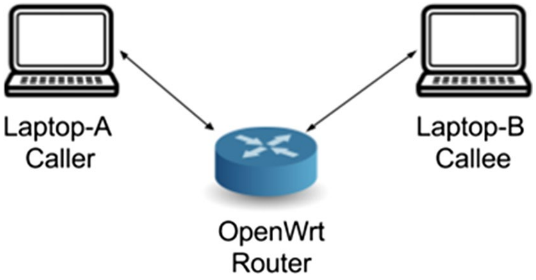

Fig. 13 Laboratory setup for audio test file generation

and dynamic reference speeches. As voice references, we chose four independent samples from audiobooks: a slow (calm) male speech, a slow female speech, a dynamic male speech, and a dynamic female speech. We replayed the audio files and streamed them into the VoIP call. In the Ekiga client, we applied AMR-WB as the default codec type and set up a jitter buffer of 100 ms. Since the input jitter buffer eliminates the effect of a reordered packet with a relatively low sequence offset and grants an ordered playout, a reordered packet with a lower delay variation than the buffer capacity does not result in voice quality degradation. However, a more extensive sequence offset drives to receiver-side packet elimination, i.e., data loss. We checked this statement with replayed audio files.

Since the voice payload is not processed during the evaluation, the number of reference samples should represent packet-level variations only in terms of packet per second rate introduced by the voice activity detection algorithm. Since the presented method is independent of the payload content, a highly variable pps rate is the only factor that must be assured within each sample. For each test case generation, we chose one of the four reference speeches and messed up the network properties. To induce early loss, not arrived loss, and late loss events, we activated and deactivated the following NetEm kernel module parameters (network delay, reordering, and loss) during the call:

- *loss i%* (where i = 0…40, incremented by 1) to induce NAL,
- *delay 200 ms reorder j%* (where j = 100…60, decremented by 1) to induce LAL,
- *delay 200 ms reorder i%* (where i = 0…40, incremented by 1) to induce EAL.

In contrast, to measure reordering and packet loss independently, we analyzed both of them side by side to provide a higher accuracy estimation of voice quality degradation.

Using the previously presented AQuA analyzer, we calculated the MOS value for each output voice file and measured its packet-level loss metrics according to the proposed VoicePerf method.

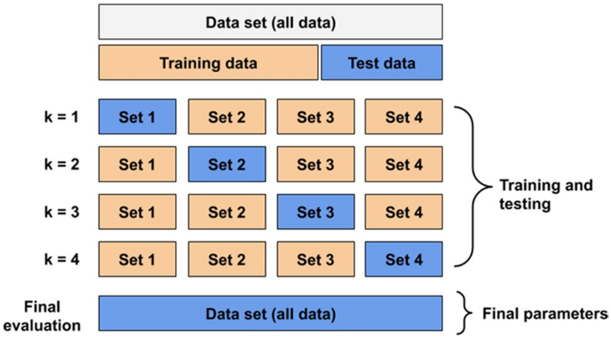

Fig. 14 Steps of k-fold cross-validation, k = 4

4.3 Methodology for Estimating the Perceptive Quality

In the next phase, we applied the k-fold cross-validation model [43] to correlate the waveform-based MOS values with our packet loss properties (not-arrived loss, late loss, and early loss categories).

The k-fold cross-validation method is designed to predict and test on the same data set and estimate the accuracy of the examined model. The data set is split into fix-sized subsets of training and test cases (see Fig. 14), based on the predefined *k* value.

The *k* value defines the number of loops (more precisely, the number of training and test rounds) and divides the data set into equal parts.

The k-fold cross-validation technique could be effectively used to find the weight parameters for our loss categories.

Each training phase results in a formula to estimate the MOS value from the measured loss categories, and the test phases show the correlation with the expected AQuA MOS values.

We applied $k = 6$ to split our 480 voice samples into training and testing data sets. As a first step, we split the voice samples into two independent data sets: 240 samples for dynamic speeches and 240 samples for slow speeches. To better correlate with the AQuA MOS results, we further split the 240 slow speech samples into two subcategories: mid-slow (120 samples) and slow (120 samples). We evaluated the original 240 slow samples without subcategories during the first measurement steps, and we get 0.56 as an average delta MOS value. As an optimization step, we examined the audio samples and created further two subcategories based on the length of the silence. To differentiate the mid-slow and slow categories, we set the *t* time value to 1 for mid-slow and to 1.5 for slow (*t* is previously defined in subsect. 4.1). The n value was unchanged (n = 10).

To create a MOS estimation formula from the loss categories, we applied regression analysis using linear estimation in IBM SPSS [44]. We performed several regression analyses (e.g., linear, non-linear, and curve-estimation models, supported by SPSS), but linear regression resulted in the highest correlation with the AQuA MOS values.

As a training data set for the dynamic speech MOS estimation, we applied 200 training samples and 40 test samples for each round. For the mid-slow and slow categories, we used 100 training cases and 20 test cases.

Tables 2, 3 and 4 summarize the coefficients of the regression lines. *B1* is the weight for the not-arrived loss rate, *B2* is for the early-arrived loss rate, and *B3* is for the late-arrived loss rate.

Tables 5, 6 and 7 summarize the correlation of each round with the AQuA MOS values.

Table 2 The coefficients of the linear-regression (dynamic speech-based rounds)

Round (k)	Constant	B1	B2	B3
1	3.943	− 4.162	− 2.304	− 3.947
2	3.941	− 4.177	− 2.273	− 3.976
3	3.941	− 4.149	− 2.360	− 3.913
4	3.921	− 4.017	− 2.082	− 3.948
5	3.931	− 4.074	− 2.261	− 3.892
6	3.938	− 4.200	− 2.320	− 3.917

Table 3 The coefficients of the linear-regression (mid-slow speech-based rounds)

Round (k)	Constant	B1	B2	B3
1	3.816	− 3.816	− 2.490	− 3.391
2	3.895	− 5.315	− 2.505	− 3.905
3	3.896	− 5.677	− 2.416	− 4.015
4	3.903	− 5.080	− 2.867	− 4.025
5	3.847	− 5.363	− 2.493	− 3.656
6	3.910	− 5.220	− 2.602	− 4.040

Table 4 The coefficients of the linear-regression (slow speech-based rounds)

Round (k)	Constant	B1	B2	B3
1	4.444	− 1.288	− 1.408	− 1.188
2	4.542	− 1.661	− 1.686	− 1.627
3	4.491	− 1.375	− 1.644	− 1.381
4	4.522	− 1.492	− 1.666	− 1.572
5	4.520	− 1.540	− 1.659	− 1.535
6	4.509	− 1.451	− 1.496	− 1.446

Table 5 Summary of the average delta (dynamic speech-based rounds)

Round (k)	Average delta MOS
1	0.124
2	0.159
3	0.148
4	0.167
5	0.139
6	0.115

Table 6 Summary of the average delta (mid-slow speech-based rounds)

Round (k)	Average delta MOS
1	0.330
2	0.291
3	0.364
4	0.320
5	0.258
6	0.787

Table 7 Summary of the average delta (slow speech-based rounds)

Round (k)	Average delta MOS
1	0.241
2	0.266
3	0.204
4	0.247
5	0.199
6	0.246

4.4 MOS Estimation

To estimate the MOS value from the VoicePerf loss values, the loss categories (i.e., early loss, late loss, not-arrived loss) have to be weighted by the linear regression coefficients.

Let P_c be the rate of the given category counter versus the total RTP packets sent, where c means the category (NAL, EAL, and LAL), and $0 \leq P_c \leq 1$.

Using the measured loss values as explanatory variables, the MOS values for the different speech types can be determined by unique estimation functions. Based on the outcomes of the regression analysis, we defined a (predictor) function (1) for the dynamic speech samples (*g*(P_{NAL}, P_{EAL}, P_{LAL})), and two others for the slow speech samples (h_1(P_{NAL}, P_{EAL}, P_{LAL}), h_2(P_{NAL}, P_{EAL}, P_{LAL})).

The formulae' coefficients suggest that the punishment of the early-, late- and not-arrived loss should not be equal. In case of an early-arrived loss event, buffer resources can be available for a portion of the early arrived packet burst; thus, they can be stored within the jitter buffer, and the rest are dropped. While in case of a late-arrived loss event, all packets within the burst are dropped.

Let us note that the regression analysis results in less than − 5.0 weight values in some rounds. This is because of the AQuA's measurement range and output values, which is used as a reference.

The estimated QoE value is calculated by (7–10). Let *f*(P_{NAL}, P_{EAL}, P_{LAL}, *st*) be the function used in (10), where *st* is the speech type that returns the appropriate formula (*g*(P_{NAL}, P_{EAL}, P_{LAL}), *h1*(P_{NAL}, P_{EAL}, P_{LAL}), *or h2*(P_{NAL}, P_{EAL}, P_{LAL}) respectively) based on the speech type and the corresponding loss categories. (For the pseudo-code, see Algorithm I)

$$g\left(P_{NAL},P_{EAL},P_{LAL}\right) = 3.936 + P_{NAL} \times (-4.13) + P_{EAL} \times (-2.267) + P_{LAL} \times (-3.933) \tag{7}$$

$$h1\left(P_{NAL},P_{EAL},P_{LAL}\right) = 3.878 + P_{NAL} \times (-5.256) + P_{EAL} \times (-2.573) + P_{LAL} \times (-3.837) \tag{8}$$

$$h2\left(P_{NAL},P_{EAL},P_{LAL}\right) = 4.504 + P_{NAL} \times (-1.466) + P_{EAL} \times (-1.593) + P_{LAL} \times (-1.453) \tag{9}$$

$$VoicePerfMOS = f\left(P_{NAL},P_{EAL},P_{LAL},st\right) \tag{10}$$

Algorithm I Pseudo-code of the estimator algorithm

```
function calcVperfMOS(P, T, speechType)

  Initialize EAL, LAL, NAL, CT and numberOfRTPPackets
  with 0

  while (process all RTP packets)
  do
     if numberOfRTPPackets < (4 + (2 * T)) then
        Initialize CT with (P * packetSeqNumb)
        Wait for in-order packets, save prevPacketTs
     end if

     numberOfRTPPackets = numberOfRTPPackets + 1
     deltaT = packetTs - prevPacketTs
     prevPacketTs = packetTs
     CT = CT + deltaT
     nextExpectedSeqNumb = packetSeqNumb + 1
     rp1 = packetSeqNumb * P - ((T + 0.5) * P)
     rp6 = packetSeqNumb * P + ((T + 0.5) * P)

     if CT < rp1 then
        EAL = EAL + 1
    else if CT > rp6 then
        LAL = LAL + 1
    end if

    Also check for NAL based on the previously arrived
    sequence numbers

  end while

  pN = NAL / numberOfRTPPackets
  pE = EAL / numberOfRTPPackets
  pL = LAL / numberOfRTPPackets

  return f(pN, pE, pL, speechType)
end function

function f(pN, pE, pL, speechType)

 if speechType = dynamic then
   return 3.936 + pN*(-4.13) + pE*(-2.267) + pL*(-3.933)
 else if speechType = slow1 then
   return 3.878 + pN*(-5.256) + pE*(-2.573) + pL*(-3.837)
 else if speechType = slow2 then
   return 4.504 + pN*(-1.466) + pE*(-1.593) + pL*(-1.453)
 end if

end function
```

5 Validation Results

To validate our voice quality assessment model (7–10) in a laboratory environment, we implemented Algorithm I in C++, and applied it to the 480 sample files.

Table 8 The coefficients of the linear-regression (final evaluation round)

Speech type	Constant	B1	B2	B3
Dynamic	3.936	– 4.130	– 2.267	– 3.933
Mid-slow	3.878	– 5.256	– 2.573	– 3.837
Slow	4.504	– 1.466	– 1.593	– 1.453

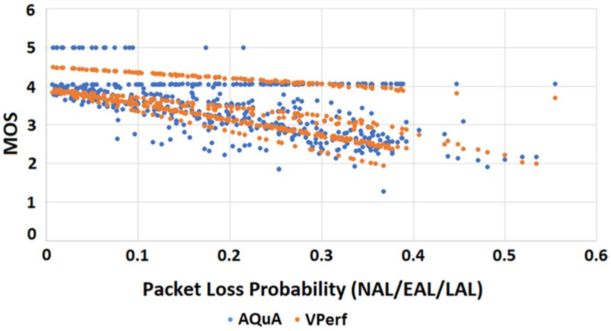

Fig. 15 Correlation between AQuA and VoicePerf MOS (480 test cases)

The final evaluation rounds of the k-fold cross-validation result in a final formula for each speech type, containing the final weights for the loss categories (summarized by Table 8). We applied these weights in our quality assessment model (7–10).

We applied the AQuA [35] analyzer tool to calculate the MOS values for each speech sample. To determine the accuracy of our VoicePerf quality assessment model, we calculated the window-specific counters and the different loss category counters per speech sample. We applied the VoicePerf MOS formulae (7–10).

Figure 15 represents the correlation between AQuA MOS and VoicePerf MOS values. The validation results reveal that the absolute error range is between 0.003 and 1.2 MOS, and the average delta MOS is 0.256.

To conclude the observations, the difference value between AQuA and VoicePerf MOS in 91% of test cases is less or equal to 10% (0.5 MOS), in 87% of the cases, it is less or equal to 8% (0.4 MOS), and in 64% of the cases is less or equal to 4% (0.2 MOS).

The results show that the average delta percent (Table 9), which comes from the difference between AQuA [35] MOS and VoicePerf model-based MOS values, is

Table 9 Numerical summary of MOS assessment validation

Speech type	Average delta MOS	Number of measurements
Dynamic	0.14	240
Slow-1	0.39	120
Slow-2	0.23	120
	0.25	480

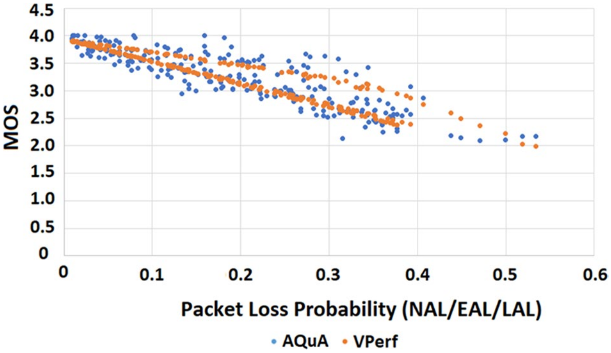

Fig. 16 Validation results for 240 test cases: correlation between AQuA and VoicePerf MOS values with dynamic speech type

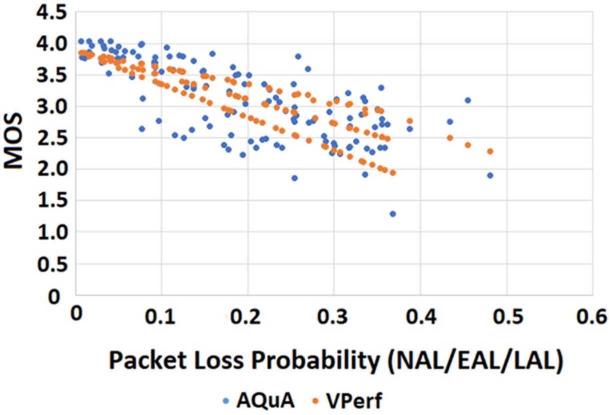

Fig. 17 Validation results for 120 test cases: correlation between AQuA and VoicePerf MOS values with mid-slow speech type

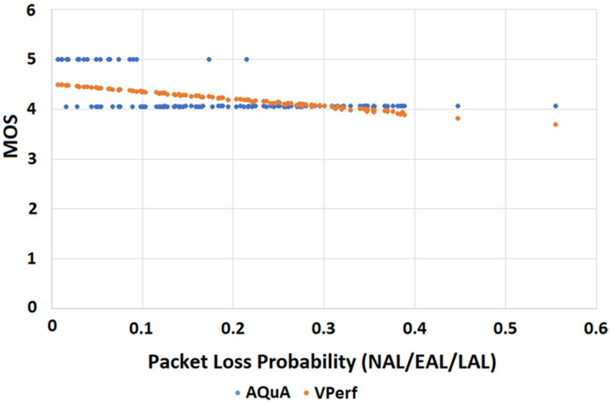

Fig. 18 Validation results for 120 test cases: correlation between AQuA and VoicePerf MOS values with slow speech type

2.8% in the case of dynamic speech (Fig. 16), 7.8% in the case of mid-slow speech (Fig. 17), and 4.7% in the case of slow speech (Fig. 18). The mid-slow and slow speech test cases frequently contain 1–3 s long silence blocks.

In 4% of the validation cases, the large early-arrived loss values (> 30%) have triggered the reset function of our proposed method with a relatively high frequency. The reset event can be triggered by a pre-defined *c* number of consecutive early- or late-arrived packets, and *c* was set to 8 during the validation process (see Sect. 3.5). Accordingly, each reset introduced a short-term offset in the packet classification, i.e., *c* in-time arrived packets were classified as early-arrived and increased the loss counter of that test case. However, this additional loss is significantly lower in magnitude (and can be compensated) than the improvement in the accuracy of our MOS estimation model. Without the reset function (re-sync), the number of packets classified as early-loss would be significantly larger, degrading the estimation accuracy in the mentioned cases.

We have examined these samples without the reset function and got a worse correlation with the Aqua MOS values.

We note here that besides AMR-WB, we have verified our model with Speex codec as well, and we got similar results and accuracy. Table 9 shows the summary of the validation results. For reference, we made the validation data set available online [45].

6 Conclusion

In the last decade, IP-based communication ecosystems (i.e., VoIP, VoLTE, and VoWiFi) dominate the telecommunication market. This technological progression necessitates tight control over several transmission parameters to offer reliable and high-quality voice services. It raised the demand for real-time service quality monitoring of voice applications with a focus on the user's perception.

This paper proposed an objective quality model called VoicePerf, contributing three novelties in assessing the quality of IP voice services. (i) A new classification-based metric system for RTP-based voice traffic (early-arrived loss, late-arrived loss, not-arrived loss, time window offset, and intra-window variation). (ii) We introduced a quality estimation model that calculates a MOS value for the call session derived from the measured packet-level metrics. (iii) Furthermore, we proposed a new method to identify the speech pace (dynamic or slow) based on the voice activity detection (VAD) induced traffic properties. This identification supports the proposed model to estimate the service quality with higher accuracy.

Our approach to monitoring service quality relies on the emulation of media frame availability inside the endpoint's playout buffer that is a crucial element of the voice transmission. The key differentiator of our method is the categorization of the packet loss events. It defines three loss types, and applies a time windowing technique to model the endpoint's playout buffer. The main benefit of the integrated reorder-loss calculation is the capability to assign a status for each arrived RTP packet from the voice decoder's perspective. Based on these statistics, we get a multi-layer overview of the transmission that can be applied for estimating the perceptive quality. From the implementation viewpoint, we modeled the properties of the receiver-end playout buffer in our independent measurement system.

We validated our proposal against a full-reference objective model called AQuA with 480 voice scenarios (samples) using k-fold cross-validation. We found a high correlation between VoicePerf MOS and AQuA MOS in all scenarios. Results revealed that our model estimates the perceptive quality with an average of 5.12% error ratio.

Acknowledgements We would like to thank Sevana Inc. for the academic license of the AQuA voice quality analyzer software.

Funding Open access funding provided by Budapest University of Technology and Economics.

References

1. "IP Multimedia Subsystem (IMS)", 3GPP TS 23.228, Online, available: http://www.3gpp.org/DynaReport/23228.htm
2. "System architecture for the 5G System (5GCS)", 3GPP TS 23.501, Online, available: http://www.3gpp.org/DynaReport/23501.htm
3. "Perceptual Evaluation of Speech Quality (PESQ)", Online, available: http://www.itu.int/rec/T-REC-P.862. Accessed 31 July 2020
4. "Perceptual Objective Listening Quality Assessment", Online, available: https://www.itu.int/rec/T-REC-P.863. Accessed 31 July 2020
5. Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V.: "RTP: a transport protocol for real-time applications", IETF RFC 3550, Online, available: https://tools.ietf.org/html/rfc3550. Accessed 31 July 2020
6. Morton, A., Ciavattone, L., Ramachandran, G., Shalunov, S., Perser, J.: "Packet reordering metrics", IETF RFC 4737, Online, available: https://tools.ietf.org/html/rfc4737. Accessed 31 July 2020
7. Jayasumana, A., Piratla, N., Banka, T., Bare, A., Whitner, R.: "Improved packet reordering metrics", IETF RFC 5236, Online, available: https://tools.ietf.org/html/rfc5236. Accessed 31 July 2020
8. Morton, A., Claise, B.: "Packet delay variation applicability statement", IETF RFC 5481, Online, available: https://tools.ietf.org/html/rfc5481. Accessed 31 July 2020
9. Demichelis, C., Chimento, P.: "IP packet delay variation metric for IP performance metrics (IPPM)", IETF RFC 3393, Online, available: https://tools.ietf.org/html/rfc3393. Accessed 31 July 2020
10. IETF IPPM Reordering draft, Online, available: https://tools.ietf.org/html/draft-ietf-ippm-reordering-13. Accessed 31 July 2020
11. Raisanen, V., Grotefeld, G., Morton, A.: "Network performace measurement with periodic streams", IETF RFC 3432, Online, available: https://tools.ietf.org/html/rfc3432. Accessed 31 July 2020

12. Morton, A., Ramachandran, G., Maguluri, G.: “Reporting IP network performance metrics: different points of view”, IETF RFC 6703, Online, available: https://tools.ietf.org/html/rfc6703. Accessed 31 July 2020
13. "The E-Model, a computational model for use in transmission planning". ITU-T Rec. G.107 (2005)
14. Ding, L., Goubran, R. A.: Assessment of effects of packet loss on speech quality in VoIP. In: The 2nd IEEE International Haptic, Audio and Visual Environments and Their Application (2003)
15. Takahashi, A., Yoshino, H., Kitawaki, N.: Perceptual QoS Assessment technologies for VoIP. IEEE Commun. Mag. **42**, 28–34 (2004)
16. Raake, A.: Short- and long-term packet loss behavior: towards speech quality prediction for arbitrary loss distributions. IEEE Trans. Audio Speech Lang. Process. **14**, 1957–1968 (2006)
17. Jung, Y., Manzano, C.: Burst packet loss and enhanced packet loss-based quality model for mobile voice-over Internet protocol applications. IET Commun. **8**, 41–49 (2014)
18. Broom, S., Hollier, M.: "Speech quality measurement tools for dynamic network management", MESAQIN (2003)
19. Takahashi, A., Kurashima, A., Yoshino, H.: Objective assessment methodology for estimating conversational quality in VoIP. IEEE Trans. Audio Speech Lang. Process. **14**, 1984–1993 (2006)
20. Ouyang, Y., Yan, T., Wang, G.: CrowdMi: scalable and diagnosable mobile voice quality assessment through wireless analytics. IEEE Internet Things J. **2**, 287–294 (2015)
21. Zou, W., Yang, F., Li, X.: A packet-layer quality assessment system for VoIP using random forest. In: IEEE International Conference on Computer and Information Technology (2014)
22. Conway, A. E.: A passive method for monitoring voice-over-IP call quality with ITU-T objective speech quality measurement methods. In: 2002 IEEE International Conference on Communications. Conference Proceedings. ICC 2002 (Cat. No.02CH37333), vol. 4, pp. 2583–2586. (2002)
23. Han, Y., Muntean, G.: Hybrid real-time quality assessment model for voice over IP. In: IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (2015)
24. Sun, L., Ifeachor, E.C.: Voice quality prediction models and their application in VoIP networks. IEEE Trans. Multimed. **8**(4), 809–820 (2006)
25. Lin, J. C. w., Fournier-Viger, P.: Employing the neural networks to parametrically assess the quality of a voice call. In: 2016 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS), pp. 1–5. Montreal, QC (2016)
26. Majed, N., Ragot, S., Lagrange, X., Blanc, A.: Delay and quality metrics in voice over LTE (VoLTE) networks: an end-terminal perspective. In: International Conference on Computing, Networking and Communications (2017)
27. Abareghi, M., Homayounpour, M. M., Dehghan, M., Davoodi, A.: Improved ITU-P.563 non-intrusive speech quality assessment method for covering VOIP conditions. In: 10th International Conference on Advanced Communication Technology (2008)
28. "Single-ended method for objective speech quality assessment in narrow-band telephony applications", ITU-T Rec. P.563, May 2004
29. Broom, S.R.: VoIP quality assessment: taking account of the edge-device. IEEE Trans. Audio Speech Lang. Process. **14**, 1977–1983 (2006)
30. Luksa, D., Faajt, S., Krhen, M.” Sound quality assessment in VOIP environment. In: 37th International Convention on Information and Communication Technology (2014)
31. Orosz, P., Tothfalusi, T.: “VoicePerf: A Quality Estimation Approach for No-reference IP Voice Traffic”, IEEE/IFIP Network Operations and Management Symposium (NOMS). Budapest, Hungary (2020)
32. “Wideband Coding of Speech at Around 16 kbit/s using Adaptive Multi-Rate Wideband (AMR-WB)”, Online, available: https://www.itu.int/rec/T-REC-G.722.2/en. Accessed 31 July 2020
33. JM. Valin, K. Vos, T. Terriberry, “Definition of the Opus Audio Codec”, IETF RFC 6716, Online, available: https://tools.ietf.org/html/rfc6716. Accessed 31 July 2020
34. “Speex: A Free Codec for Free Speech”, Online, available: https://www.speex.org/. Accessed 31 July 2020
35. Sevana, “AQuA—Audio Quality Analyzer”, Online, available: https://sevana.biz/products-aqua/. Accessed 31 July 2020
36. Jelassi, S., Rubino, G., Melvin, H., Youssef, H., Pujolle, G.: Quality of experience of VoIP service: a survey of assessment approaches and open issues. In: IEEE Communications Surveys & Tutorials, vol. 14, no. 2, pp. 491–513, Second Quarter 2012. https://doi.org/10.1109/SURV.2011.120811.00063

37. “Mean Opinion Score (MOS) terminology”, Online, available: https://www.itu.int/rec/T-REC-P.800.1-200303-S/en. Accessed 31 July 2020
38. Microtronix SYSTEMS LTD, “Perceptual Evaluation of Speech Quality (PESQ)”, Online, available: https://www.microtronix.ca/pesq.html. Accessed 31 July 2020
39. Sevana, “PESQ, POLQA, AQUA, Feature Comparison Table”, Online, available: https://www.slideshare.net/sevana/sevana-aqua-endtoend-drive-testing-technology-125301405/. Accessed 31 July 2020
40. Linux kernel, Online, available: http://www.kernel.org. Accessed 31 July 2020
41. Ekiga software, Online, available: http://www.ekiga.org/. Accessed 31 July 2020
42. Arecord, Online, available: https://alsa.opensrc.org/Arecord. Accessed 31 July 2020
43. Breiman, L., Spector, P.: Submodel selection and evaluation in regression. The X-Random Case. Int. Stat. **60**(3), 291–319 (1992)
44. IBM Analytics, “IBM SPSS Software”, Online, available: https://www.ibm.com/analytics/spss-statistics-software/. Accessed 31 July 2020
45. Orosz, P., Tóthfalusi, T.: VoicePerf validation data set, Online, available: https://drive.google.com/file/d/1bniS-tMTNNaxY0s6BERrbBa7OJG2m6nG/view?usp=sharing

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Péter Orosz is an associate professor and the head of Smart Communications Laboratry at the Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics (BME) Hungary. He received his Computer Science master degree in the field of software engineering (2003) and Ph.D. in infocommunication systems (2010) at the University of Debrecen, Hungary. Previously he has been working for the University of Debrecen. His research interest covers communication networks, network and service management, QoS-QoE managed networks, online QoE prediction for media services, and hardware acceleration of network functions.

Tamás Tóthfalusi is currently a Ph.D. candidate at the Budapest University of Technology and Economics (BME). He received the B.Sc. degree in Infocommunication Networks as part of the degree programme in Engineering Information Technology at the University of Debrecen from 2006 to 2010, and the M.Sc. degree in Hardware Programming at the University of Debrecen from 2010 to 2012. His research interests include hardware- (FPGA) based acceleration of network processes, network management and measurement, VoLTE signaling and neural networks. He is a member of the SmartCom Laboratory at BME. He has been involved in industrial research and development projects in these topics.