Performance evaluation of real-time speech through a packet network: a Random Neural Networks based approach

(*Performance Evaluation*, 57(2):141–162, 2004)

Samir Mohamed, Gerardo Rubino *, Martín Varela

INRIA/IRISA, Campus de Beaulieu, 35042 Rennes, France.

Abstract

This paper addresses the problem of quantitatively evaluating the quality of a speech stream transported over the Internet *as perceived by the end user*. We propose an approach being able to perform this task automatically and, if necessary, in real time. Our method is based on using G-networks (open networks of queues with positive and negative customers) as neural networks (in this case, they are called Random Neural Networks) to learn, in some sense, how humans react vis-a-vis a speech signal that has been distorted by encoding and transmission impairments. This can be used for control purposes, for pricing applications, etc.

Our method allows us to study the impact of several source and network parameters on the quality, which appears to be new (previous work analyzes the effect of one or two selected parameters only). In this paper we use our technique to study the impact on performance of several basic source and network parameters on a non–interactive speech flow, namely *loss rate*, *loss distribution*, *codec*, *forward error correction*, and *packetization interval*, all at the same time. This is important because speech/audio quality is affected by several parameters whose combined effect is neither well identified nor understood.

Key words: Packet audio, Random Neural Networks, G–Networks, Speech transmission performance, Speech quality assessment, network loss models.

1 Introduction

There is nowadays an increasing number of applications running over the Internet and involving real–time transmission of speech and audio streams. Among these applications we have Voice over IP [14], IP telephony [38,2], audio streaming, teleconferencing, etc. One of the major concerns of these applications is to maximize the QoS for a given network state. Traditionally, this is done by keeping some of the network parameters (e.g., packet loss rate and delay variation) within certain limits and little attention has been paid to the quality as perceived by the end-users of the applications. However, the current Internet infrastructure provides basically a *Best–effort* service, with no provisions for QoS. Therefore, in order to deliver the best possible quality, a better understanding of the combined effects of all of the parameters *a priori* involved is a must.

Assume that we are requested to analyze the capacity of a network system used to transport an audio flow without degrading its quality below an acceptable level, using some accurate model of the system. The usual scenario is to analyze, say, the loss rate (because we know it is critical to the final quality of the audio stream), and to try and determine when it is suffieciently small as to permit an acceptable QoS level. If the model is, for example, an $M/M/1/N$ queue with load $\varrho \neq 1$, the analysis could return that in steady-state, if $N \geq \lceil \ln(\varepsilon/[\varepsilon\varrho + 1 - \varrho])/\ln \varrho \rceil$ then the loss probability is less than ε. Perhaps we would also analyze the mean delay, or the jitter. This paper shows that the quality, *as perceived by the final user*, depends in a complex way on many parameters (six in our study), not just on loss probabilities or on the bit rate of the source, etc. We also exhibit a method being able to map such a set of parameters onto a numerical value close to the one a human observer would give to the stream in a *subjective test* [45]. Interestingly, the object used for this purpose is an open network of Markovian queues, but not as a model of the communication service: it is used in a completely different way, as a *learning* tool.

Generally speaking, the analysis of speech/audio quality can be done using either *objective* tests or *subjective* ones. Objective tests are usually explicit functions of measurable parameters related to the encoder or to the net-

⋆ This research was partly funded by the IST project FABRIC (Federated Applications Based on Real-time Interacting Components) # IST-2001-37167, and by PAIR (Associated Team – INRIA / Facultad de Ingeniería, Universidad de la República Oriental del Uruguay)

* Corresponding author: Tel. +33 299 847 296, Fax. +33 299 842 529.
Email addresses: Samir.Mohamed@irisa.fr (Samir Mohamed), Gerardo.Rubino@irisa.fr (Gerardo Rubino), Martin.Varela@irisa.fr (Martín Varela).

work [5,58,44]. Subjective tests are based on evaluations made by human subjects under well defined and controlled conditions [45]. Obviously, the reference is the end-user's perception, which is directly captured by subjective tests. Concerning available objective tests, it is well known that they do not always correlate very well with human perception [36,54]. More details are given in the following paragraphs.

The most commonly used objective measures are Signal-to-Noise Ratio (SNR), Segmental SNR (SNRseg), Perceptual Speech Quality Measure (PSQM) [5], Measuring Normalizing Blocks (MNB) [58], ITU E-model [44], Enhanced Modified Bark Spectral Distortion (EMBSD) [61], Perceptual Analysis Measurement System (PAMS) [54] and PSQM+ [4]. Except for the ITU E-model, all these metrics propose different ways to compare the received sample *with the original one*, something we want to avoid (for instance, in order to use the automatic quantitative evaluation of quality for control purposes, or for real–time monitoring applications).

Subjective quality assessment methods measure the overall perceived audio quality, and as implied by their name, they are carried out by human subjects. The most commonly used one for audio quality evaluation is the Mean Opinion Score (MOS) [45], recommended by the ITU. It consists of having a set of subjects listen to degraded audio samples in order to rate their quality, according to a predefined quality scale (see the first part of Section 3 and Section 5 for more details). That is, human subjects are trained to *build* a mapping between a set of processed audio samples and the quality scale. Although MOS studies have served as the basis for analyzing many aspects of signal processing, they present several limitations: a) very stringent environments are required; b) the process can not be automated (by definition); c) they are very costly and time consuming, which makes them unsuitable to be frequently repeated.

The approach presented in this paper extends and improves the method proposed in [51] for evaluating real–time speech quality. The main idea in [51] is to use Artificial Neural Networks (ANN) to learn the way human subjects evaluate the quality using measures of both network and encoding distortions as inputs. The methodology is general and any parameter can *a priori* be taken into account (possibly at the cost of a larger subjective quality database, see Section 3 [1]). The technique presents advantages over other objective evaluation methods, since it does not need to access the original signal, and it's not computationally intensive, which allows for use in real–time applications or embedded devices. Besides, it offers a good correlation in terms of MOS with subjective measures (by construction). The improvement over our original

[1] Fortunately, this has to be done only once provided that the database has sufficient information representing the effect of all the parameters considered.

Neural Network approach is obtained by using a different type of tool called Random Neural Networks (RNN), which are in fact a particular kind of open queuing networks, called G–Networks in the performance evaluation area. This recently invented model [20,22,21] appears to capture accurately and robustly the function mapping the various parameters involved with the quality metric. RNN have been used in many different domains such as image and video compression [27,13,12], error–correcting codes [1], land mine detection [32], video quality assessment [52], where they proved themselves better than the ANN for this kind of application, and video quality enhancement [11,28]. A survey of RNN applications is given in [3]. There are several other applications of RNN in the area of imaging and video, such as [29,19,33,24,25,26]. Using RNN for real–time speech evaluation, we didn't find the common over–training problems associated with ANN [48], which hinder the NN's ability to extrapolate. This is particularly useful in our context, since the NN's extrapolation ability allows for a smaller number of subjective tests needed to train the network (see Section 3), and thus it lowers the cost of the method.

Our main contribution in this paper is the use of this RNN–based quality assessment mechanism in order to study and analyze the impact of certain quality-affecting parameters such as the codec used, the redundancy, the network loss rate (LR), the mean loss burst size (MBS) and the packetization interval (PI) on real–time speech quality. The effects of some of these parameters on speech quality have been the subject of previous research [40,16]. However, those studies normally consider only one or two parameters at once, thus neglecting the effects of their interaction as a whole. The approach we present allows us to better understand the influence of all parameters considered at the same time (cf. Section 6.2 for an example of such an analysis).

Previous studies of the performance of a speech stream either concentrated on the effect of network parameters without paying attention to encoding parameters, or the contrary (cf. Section 2 for some annotated references). Papers that consider network parameters and use subjective tests for the evaluation restrict the study to only one or two of the most important ones, due to the high costs associated with subjective tests. Concerning objective approaches, there is no previously published objective quality test that can take into account the direct influence of the whole set of parameters, considered simultaneously, on the perceived quality.

The organization of the rest of the paper is as follows: Section 2 presents related works. In Section 3 we present an overview of the methodology used. The network aspects considered are discussed in Section 4. Section 5 describes the experiments we carried out. The results we obtained are presented in Section 6. In Section 7 we describe possible applications of our approach, and some conclusions of our work are provided in Section 8.

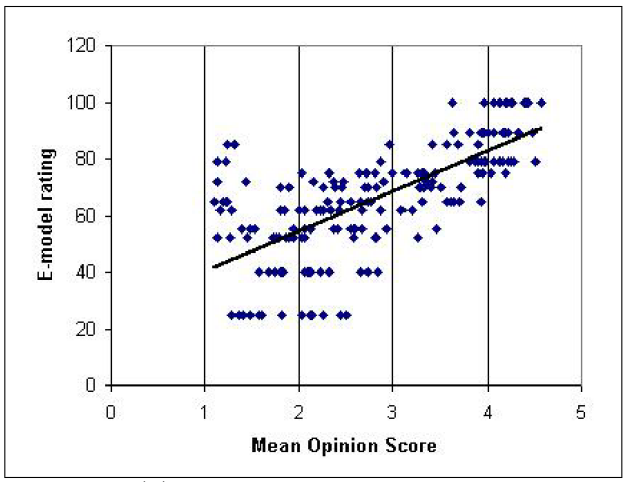
2 Related Works

In [40], the effect of both LR and PI on speech quality is presented. The study is based on an automatic speech recognition system which is in turn based on hidden Markov models instead of the usual subjective quality tests. A similar ANN–based study is presented in [56], using data obtained from the objective measure PESQ. It should be noted that [53] showed that PESQ is not accurate for evaluating voice quality when considering network parameters. LR and jitter effects are studied using subjective data for several codecs in [16]. The effect of packet loss on several speech codecs is evaluated in [9]. The authors concluded that there are two main effects of packet loss on speech quality. First, there are missing of fragments of the speech. Second, the adaptation logic is disrupted after the lost segment (in the decoding part). There are other studies [46] where the main goal is to differentiate the performance of speech codecs.

Pure delay effects on speech quality in telecommunications are evoked in [47]. That work is focused mainly on conversation, and the authors argue that round-trip delays in the range of 500 ms are annoying. In [59] the authors study the variation of speech quality for different error concealment techniques (silence, waveform, or LPC repair). Additionally, they evaluate the subjective speech quality against loss and they compare the result for redundancy and no redundancy cases. The impact of cell delay variation on speech quality in ATM networks is studied in [49].

From the available objective speech quality measures in the literature, only the ITU E-model does not need the access to the original signal to compute the quality. Therefore, it is the only available measure which is computationally simple [36] and can be used in real–time applications. However, the E-model was not designed as a quality assessment tool (as clearly stated in the reccomendation defining it), but as a network planning tool.The E-model provides the means to take several quality–affecting parameters into account when designing a telephony network, most of which are more related to the field of signal processing than to that of computer networks. For example, impairments due to packet losses have only been explicitly included into the E-Model in the latest revision of the reccomendation (dated from March 2003), and then only a uniform loss distribution is considered, which does not provide an accurate model of packet loss on a big network like the Internet. Therefore, the assessments obtained by means of the E-Model are to be considered in the appropriate context, and it is not surprising that they are not as close to MOS tests as one would want [36]. Fig. 1(a), which is taken from [36], shows a scatter plot for the quality evaluation given by this model with respect to MOS (cf. Fig. 4(b) for our model). Ideally, there should be a one-to-one relation between the MOS values and the E-model's output. However, as seen in

Fig. 1(a), there are strong variations. We observe that for a single MOS value, the E-model's output has important variations. For example, for MOS=2.5, there are output values ranging from 20 to 80, while the entire range of output is 0–100. In [18,10] the authors argue that the E-model is not yet suitable to correctly consider the impairments caused by the transmission over a best–effort network and that more research is needed to adapt it to this function.



(a) Performance of E-model

Figure 1. E-model results against the MOS subjective values on a set of speech samples distorted by both encoding and network impairments (source: [36]).

To the best of our knowledge there is currently no objective quality measure in the literature that can evaluate speech signals distorted by both encoding and network impairments, in rea–time and with good performance.

3 Overview of the Method

In this Section, we describe the overall steps to be followed to build a tool in order to automatically assess, and in real–time if necessary, the quality of speech transmitted over packet networks. The automatic evaluation is performed by a suitable trained Random Neural Network (RNN).

We start by choosing a set of *a priori* quality-affecting parameters corresponding to the considered applications and to the network supporting the transmission. Examples are the bit rate of the source, the parameters corresponding to the redundancy used, the loss rate in the network, etc. Then, for each parameter π we must select a minimal value, $\pi_{\min}$, a maximal one, $\pi_{\max}$, and a set of representative, or significant values in the interval $[\pi_{\min}, \pi_{\max}]$. For example, we will probably choose the percentage loss rate as one of the key network parameters. If its value is expected to vary from 0 to 10%, and

if the typical observed values are between 0 and 5%, then we may use 0, 1, 2, 5, and 10% as selected loss rate values.

Let us denote by $\{\pi_1, \cdots, \pi_P\}$ the set of selected parameters. Let us then denote by $\{p_{i1}, \cdots, p_{iH_i}\}$ the set of possible values chosen for parameter π_i (here, $\pi_{\min} = p_{i1}$ and $\pi_{\max} = p_{iH_i}$). We call *configuration* of the set of quality-affecting parameters, any set of possible values for each one. The total number of possible configurations (that is, the number $\prod_{i=1}^{P} H_i$) is usually large. We must then select a part of this large cardinality set of configurations, which will be used as (part of) the input data for the NN in the learning phase.

In order to generate a speech database composed of samples corresponding to different configurations of the selected parameters, a simulation environment or a testbed must be implemented. This is used to send media samples from the source to the destination and to control the underlying packet network. Every configuration in the defined input data must be mapped into the system composed of the network, the source and the receiver. Working with IP networks and speech streams, the source controls for instance the bit rate, the packetization interval and the encoding algorithm; the routers' behavior contribute to the loss rate and the loss distribution, together with the traffic conditions in the network. The destination stores the transmitted audio sample and collects the corresponding values of the parameters (that is, the configuration). Then, by running the testbed or by using simulations, we produce and store a set of distorted samples along with the associated configurations.

More formally, we select an audio sequence σ, a set of S configurations denoted by $\{\gamma_1, \cdots, \gamma_S\}$ where $\gamma_s = (v_{s1}, \cdots, v_{sP})$, v_{sp} being the value of parameter π_p in configuration γ_s. From sequence σ, we build a set $\mathcal{T} = \{\sigma_1, \cdots, \sigma_S\}$ of speech samples that have encountered varied conditions when transmitted and that constitute the "training part" of the database. That is, sequence σ_s corresponds to the result of sending σ through the source-network system where the selected P parameters have the values of configuration γ_s.

After building these samples, a subjective quality test must be carried out. There are several subjective quality evaluation methods in the recommendations of ITU-T. In general, a group of human subjects is invited to evaluate the quality of the samples (i.e. every subject gives each sample a score from a predefined quality scale $[M_{\min}, M_{\max}]$). The subjects should not establish any relation between the samples and the corresponding parameters' values. After performing a screening and statistical analysis in order to remove the grading of the individuals suspected to give unreliable results [43], the average of the scores given by the remaining subjects to each sequence is computed; the average corresponding to sequence σ_s is denoted here by μ_s.

In the third and last step, a suitable RNN architecture and a training algorithm

must be selected. The samples database is divided into two parts: one of which is used to train the RNN and the other one to test its accuracy (in the *validation* phase of the process). Once trained and validated, the RNN will then be able to evaluate the quality measure for any given values of the parameters. Formally, the trained RNN is seen as a function f having P real variables and with values in $[M_{\min}, M_{\max}]$, such that

(i) for any sample $\sigma_s \in \mathcal{T}$, $f(v_{1s}, \cdots, v_{Ps}) \approx \mu_s$,
(ii) and such that for *any other* vector of parameter values $(v_1, \cdots, v_P)$, the value of $f(v_1, \cdots, v_P)$ is close to the MOS that would be associated in a subjective evaluation to any media sample for which the selected parameters had those specific values $v_1, \cdots, v_P$.

Once the three steps described above are completed successfully, we implement the final tool, which is composed of two modules: the first one collects the values of the selected quality-affecting parameters (based on the network state, the codec parameters, etc.). These values are fed into the second one, which is the trained RNN that will take the given values of the quality-affecting parameters and correspondingly computes the MOS quality score.

3.1 On the Random Neural Network model

Let us briefly describe the mathematical structure of the RNN model [34,30,31,20]. An RNN is an open Markovian queuing network with positive and negative customers. These models are also called G-networks [23,20,21]. The RNN we use here is a particular case of a G-network. Each neuron behaves as a $./M/1$ queue processing units in, say, FIFO order (even if this is not actually essential). The server rate at neuron i is denoted by r_i and, after leaving neuron (queue) i, a customer leaves the network with probability d_i, goes to queue j as a positive customer with probability p^+_{ij} and as a negative customer with probability p^-_{ij}. Customers arrive from outside at neuron i as positive ones, according to a Poisson process with rate λ^+_i. In our RNN, no negative customers arrive from the environment. If a customer goes from i to j as a negative one, when it arrives at queue j it kills the last customer at the queue (if any), and it kills itself in all cases. Transfers between queues are, as usual with queuing network models, instantaneous. The previously described dynamics thus means that at any point in time, there are only positive customers in the queues. It has then been shown [23,21] that when the associated vector Markov process $\vec{N_t} = (N^1_t, \cdots, N^M_t)$, where N^i_t is the number of customers in queue i at time t and M is the number of nodes, is stable, its distribution is

of the product-form type: in equilibrium, that is, in the stationary case,

$$\Pr(\vec{N_t} = (k_1, \cdots, k_M)) = \prod_{i=1}^{M} (1 - \varrho_i)\varrho_i^{k_i}.$$

Factors ϱ_i are, as usual, loads in the queuing terminology: ϱ_i is the probability that queue i is not empty (under the stationary distribution). When the system is seen as a Neural Network, the backlog N_t^i of queue i at time t is called the *potential* of neuron i at t.

We use the network as a statistical approximator (it "learns" how to evaluate the quality of audio signals as humans do). As an approximator, its input is the vector $\vec{\lambda} = (\lambda_1^+, \cdots, \lambda_M^+)$ and the output is the vector $\vec{\varrho} = (\varrho_1, \cdots, \varrho_M)$. Observe that as a G-network, that is, as an open queuing network, we usually consider as the output process the flows of customers going out of the set of nodes; for instance, the mean rate of customers leaving the network from neuron i is $\varrho_i r_i d_i$).

The loads are obtained by solving a non-linear system of equations built using the data composed of the service rates and the routing probabilities (this follows from the product-form result). If we denote $w_{ij}^+ = r_i p_{ij}^+$ and $w_{ij}^- = r_i p_{ij}^-$, the non-linear equations to solve are

$$\varrho_i = \frac{\lambda_i^+ + \sum_{j=1}^{M} \varrho_j w_{ji}^+}{r_i + \sum_{j=1}^{M} \varrho_j w_{ji}^-}.$$

The w_*^* factors are called *weights* as in the standard neural network terminology, and they play a similar role in this model. More specifically, our training data is composed of the set of pairs $(\vec{v_s}, \mu_s)_{s=1,\cdots,\mathcal{T}}$, with $\vec{v_s} = (v_{1s}, \cdots, v_{Ps})$. What we want is that the G-network behave in the following way: it must have P input neurons, one output one, and when the rate of the arrival flow (of positive customers) at neuron i is $\lambda_i^+ = v_{is}$, then the occupation rate of neuron o is $\varrho_o \approx \mu_s$. To do this, consider the output ϱ_o as a function of the set of weights, that we denote here by $\vec{w}$, and of the input rates $\vec{\lambda^+}$, and we write $\varrho_o(\vec{w}, \vec{\lambda^+})$. Then, we look for a set of weights $\vec{w_0}$ defined by

$$\vec{w_0} = \text{argmin}_{\vec{w}} \ \frac{1}{2} \sum_{s=1}^{S} (\varrho_o(\vec{w}, \vec{v_s}) - \mu_s)^2$$

the minimization taking place on the set of positive multidimensional vectors w_*^*. This optimization problem can be solved used standard techniques as gradient descent (observe that we are able to compute any partial derivative of the output, using the non-linear system of equations satisfied by the occupation rates).

An interesting observation for the reader familiar with standard Neural Networks, is that the "knowledge" of the RNN, once trained, is stored in the set of weights, as in the standard case, but here the weights are mean frequencies of the signals sent by a neuron to the others, which appears to be closer to the functioning of real nervous systems.

In learning applications as in ours, it is usual to use *feedforward* topologies (in the associated G-network, a customer cannot visit more than once any given queue). We use here this class of model. In such a case, the output is a rational fraction in the input variables (and also in the weights), that is, it is the ratio between two multi-variate polynomials in the variables λ_1^+, ..., λ_M^+. This means that its computation is straightforward, and the same for its derivatives. This in turn leads to efficient learning procedures. For details, see [23,20,21,22].

In our RNN there are I neurons receiving positive customers from outside (following [22] we set all external negative flows λ_i^- to zero). Let us call $\mathcal{I}$ this set of neurons. Each external flow corresponds to one of the selected parameters being part of the analysis, and λ_i^+ is the ith parameter's value. There is only one neuron, denoted here by o, sending customers out of the network. Between this output element and the set of input neurons $\mathcal{I}$, there is a set of H neurons, called *hidden* neurons, receiving flows from the set $\mathcal{I}$ and sending customers to neuron o. Finally, there is no connection between $\mathcal{I}$ and o. We have that for all $i \in \mathcal{I}$, $\varrho_i = \lambda_i^+ / r_i$, then, for all $h \in \mathcal{H}$,

$$\varrho_h = \frac{\displaystyle\sum_{i \in \mathcal{I}} \frac{\lambda_i^+}{r_i} w_{ih}^+}{r_h + \displaystyle\sum_{i \in \mathcal{I}} \frac{\lambda_i^+}{r_i} w_{ih}^-}$$

and finally

$$\varrho_o = \frac{\displaystyle\sum_{h \in \mathcal{H}} \varrho_h w_{ho}^+}{r_o + \displaystyle\sum_{h \in \mathcal{H}} \varrho_h w_{ho}^-},$$

the stability conditions being that for every neuron j we have $\varrho_j < 1$. A sufficient condition for this is that for any input neuron $i \in \mathcal{I}$, $\lambda_i^+ < r_i$, that for any hidden neuron $h \in \mathcal{H}$ we have $r_h \geq I$ and, finally, that we have $r_o \geq I$. In practice, we observe that the number of iterations needed in the learning phase are sensitive to the effective values of these parameters, and that often much smaller values are used. Also, it is efficient to scale the inputs and to consider that for every $i \in \mathcal{I}$ we have $0 \leq \lambda_i^+ \leq 1$. As a last observation, observe that once trained, the computation of the network output is a fast process: for instance, it involves exactly $2HI + 3H + I + 1$ products (in our models, $I = 6$ and H is less than 15).

4 Network Parameters

It is well known that network losses play a major role in the perceived quality of a multimedia stream [7,60,55]. It is therefore important to understand how network losses happen, and take this behavior into account when experimenting.

Several authors [7,55,2,60] propose different mathematical models for the Internet's packet loss process, ranging from very simple ones (e.g. assuming independent losses uniformly distributed in time) to more complex structures (e.g. n^{th} order Markov chains). Many authors agree that relatively simple models, such as a 2–state homogeneous Markov chain [2], provide a good approximation of the packet loss process [55,2,60,8].

4.1 Measurements

We have studied several loss traces collected by Sue Moon *et al.* [60]. These traces were taken over a period of several days, and under different conditions (national and international links, different packet sizes, unicast and multicast transmissions). We found the loss rates of these traces to be quite low (approximately between 1.7% and 5.5%) and the average loss burst sizes were between 1 and 5 packets.

We have used the studied traces to calculate the parameters of a two-state Markov model with two degrees of freedom (c.f. Section 4.2) and then used the calibrated model to generate simulated traces. We found that the simulated loss processes were statistically similar to the real traces (even when comparing second order moments), so we decided to use this model for our experiments.

4.2 The Network Loss Model

Previous studies on the impact of network losses on VoIP quality sometimes use very simplistic loss models, such as assuming that losses are practically independent [6], or assuming packet loss bursts of a fixed size [37,51], which is not accurate. It has been stated [55,2,60,8] that a simplified Gilbert model provides a good approximation of the packet loss process as it happens on a big network such as the Internet. We chose this simpler model over the

[2] This model, which we present in detail in Section 4.2 is often mistakenly called Gilbert model. In fact, the real Gilbert model [35] is slightly more complex.

original Gilbert model [35], since it allows us to reduce the number of parameters required to tune it. Instead of the three parameters required for the Gilbert model (the two transition probabilities and the probability of an error occurring in the *"Bad"* state), we used only two probabilities to drive a two–state Markov chain. This model also provides a good approximation of the loss process (cf. [60,55,2] for details), and by having only two parameters, it reduces the number of configurations to evaluate during the subjective tests.

In the model used, each transmitted packet deals with one of two possible results: either the packet arrives successfully at destination or it doesn't (either because some of its bits are corrupted or, more likely, because it is dropped by a router, or because it arrives out of time, etc.) The first event is coded by '1' and the second one by '0'. In this last case, we say that the packet is lost. Let us denote by X_n the result of the transmission of packet n, for $n \geq 1$. The sequence $X = (X_n)_{n\geq 1}$ is then considered to be a Markovian homogeneous stochastic process, thus with values in $\{0, 1\}$, and we denote

$$\Pr(X_{n+1} = 1 \,|\, X_n = 0) = p,$$

$$\Pr(X_{n+1} = 0 \,|\, X_n = 1) = q.$$

In words, p is the probability of a packet being lost when the previous transmission was correct, and $1 - q$ is the probability of losing a packet when the previous transmission was not correct.

Let us now denote by BS the size of a *burst* of losses, that is, a sequence of consecutive losses not strictly included in another burst of losses. Since the holding times of X are geometrically distributed, we have

$$\Pr(\mathrm{BS} = k) = (1 - q)^{k-1} q, \qquad k \geq 1.$$

If MBS is the mean size of a burst of losses, we then have $\mathrm{MBS} = q^{-1}$.

Now, if *PER* is the Packet Error Rate, that is, if *PER* is the (unconditional) probability of losing a packet, we have

$$PER = \pi_1 p + \pi_0 (1 - q),$$

where (π_1, π_0) is the stationary distribution of X, giving $PER = \pi_0$ (as expected), and so,

$$p = q \frac{PER}{1 - PER}.$$

Assume now that we analyze a long trace, and that the *PER* is measured together with the average of the bursts of losses, and denote the obtained values PER_m et MBS_m respectively. If the parametric model described above is chosen to explain these observations, we have

$$p = \frac{1}{\mathrm{MBS}_m} \frac{PER_m}{1 - PER_m}, \qquad q = \frac{1}{\mathrm{MBS}_m}.$$

See that if there are losses (at least one) and if not every transmission is a loss, then necessarily $\text{MBS}_m > 1$ and $0 < PER_m < 1$, leading to $0 < p, q < 1$.

4.3 Other Network Parameters

The delay and jitter in the network are two other important parameters to consider when evaluating real–time speech quality. End-to-end delay plays an important role in interactive (i.e. two-way) applications, since high delay values hinder the conversation [38]. However, for one-way applications, the delay is less relevant in most cases. As we didn't consider interactivity for the tests we carried out, end-to-end delay was not considered as a quality-affecting parameter [3].

Jitter, on the other hand, plays an important role in both types of applications, and its effects are similar to those of packet loss. If a dejittering buffer [15] is implemented, the effect of jitter is reduced, and from the point of view of the application, the effect of jitter can be translated as extra network losses [4]. This was the case in our experiments.

Finally, for interactive applications, the echo effect should also be considered, and an echo suppression or cancellation mechanism should be implemented [17]. For a complete discussion for packet loss, delay, jitter and associated recovering mechanisms, see [7].

5 Description of the Experiments

In order to train the neural networks, we developed a database of subjective test results (MOS scores) for different speech samples transmitted under different conditions.

The distorted samples were generated using the Robust Audio Tool (RAT) [57], over a LAN in which we generated losses according to our loss model (c.f. Section 4.2). Using a LAN context as a testbed to generate the distorted samples allowed us to control precisely the network parameters, which would have been much harder, or even impossible, in a larger network. RAT has been developed as part of a research project, and has proved itself very useful in the experiments undertaken. It has many encoding options, and it is based in the

[3] Although it can be considered if need be (e.g. for evaluating the quality of two–way VoIP). In that case, we must add it to the list of parameters, control it while generating the training database, measure it when our method is in operation, etc.
[4] As packets that arrive after some expiration time are considered as lost.

Mbus architecture [50,42]. This allowed us to automate the generation of the degraded sequences by means of a small Mbus application written in Java and a few Ruby scripts, without needing to modify a single line of RAT's code.

The losses in the network were generated with a modified version of Orion Hodson's packet reflector [39], which we adapted to work with our loss model. As this software can be controlled remotely, we had the same application that drove RAT set the loss parameters for the reflector.

We considered six parameters which affect the perceived speech quality, two of them concerning the network, and four others concerning the encoding schemes used. The network parameters that we used were the loss rate (LR), and the mean size of loss bursts (MBS, for Mean Burst Size). These parameters are enough to set those of the loss model used. As for jitter, its effects can be considered as network losses, since RAT implements a dejittering buffer. As a side note, both delay and jitter were negligible in our testbed, and so it is safe to say that the only network losses seen by RAT after the dejittering process were those that had actually happened at the network level [5]. We used four values for the LR (2, 7, 10 and 15%) and three values for the MBS (1, 1.7 and 2.5 packets).

For the encoding, we considered the following parameters:

Codec – the primary codec (16 bit linear PCM, and GSM),
Redundancy – the secondary codec (GSM), if any,
Redundancy offset – the offset, in packets, of the redundant data (we used offsets of 1 and 3 packets for the Forward Error Correction (FEC) scheme presented in [41,8] [6], which is the default in RAT),
Packetization Interval (PI) – the length (in milliseconds) of audio contained in each packet (we considered packets containing 20, 40 and 80ms).

These six parameters give place to many possible configurations, even when using just a few values for each one, so we chose the values used so as to have a reasonable number of sequences to assess, but at the same time, to have enough data to train the neural networks.

We chose about half of the 216 possible valid configurations for those values, concentrating on higher quality encodings (Linear-16 PCM as the primary codec) and small (20ms) packets, and ended up with 112 configurations to test.

Twelve original samples were used to generate 112 four–sequence groups, one for each of the configurations considered. This is in accordance with [45]. The

[5] That is, there were no lost packets due to the jitter.
[6] From now on, when we talk about FEC, we will be referring to this FEC scheme.

original sequences were sampled at 8kHz (16 bit, mono) and their contents were unrelated. Half of them were male voices and the other half were female. The four samples in each group were chosen randomly between the original ones, and then transmitted with RAT over our test network in order to affect their quality as it would be affected by normal usage of such a tool. Finally, three hidden reference groups (original samples and very degraded ones) were added to the test, in order to help detect subjects who couldn't conduct proper evaluations, and to add dynamism to the test's scale.

The results obtained were screened to eliminate subjects that produced erroneous values, and so from the 17 subjects that originally performed the test, 16 results were used to calculate the MOS, as one was rejected during the screening process. The screening method used was the β_2 test, in accordance to the ITU recommendations followed [45].

We then used randomly chosen subsets of the screened results to train several neural networks, and the rest of the results to validate the trained networks.

6 Results

In this Section we present the results obtained, which are of two kinds. Firstly, Section 6.1 presents an analysis of the neural networks' performance as subjective quality estimators, and secondly, Section 6.2 shows how, according to the estimations, the parameters considered affect the quality as it is perceived by a human subject.

6.1 Performance of the Neural Networks

We conducted experiments with several neural network architectures, and with different sizes of training databases.

The architectures used for the networks ranged from simple 3–layer designs (Fig. 2) to more complex ones, with several hidden layers and feedback between neurons. We didn't observe any significant variation in performance by modifying the architecture. However, the complex architectures showed a small advantage in convergence time over the simpler ones during the first hundred iterations of the training. Fig. 3 shows these differences. Convergence time is not really important for this type of application, as the training is only done once. We believe, however, that better performances (both in the learning process and in the estimation) can be obtained by using a different training algorithm (we used a gradient descent method for our experiments), and tak-

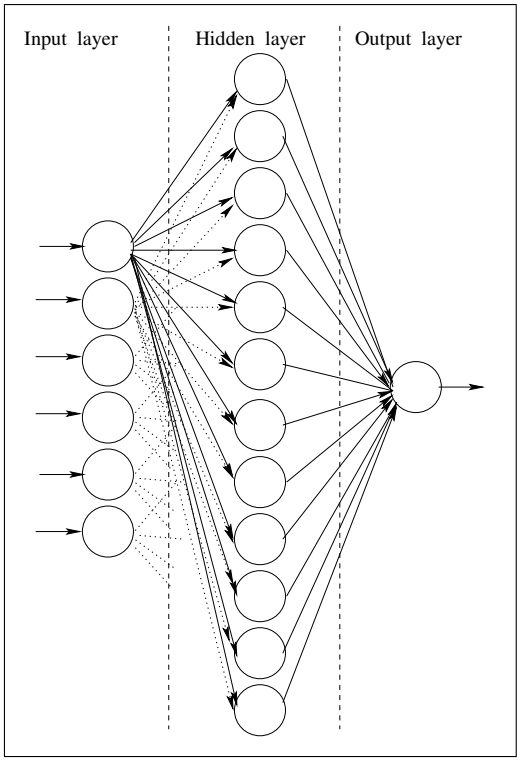


Figure 2. Example of a 3–layer RNN architecture (simplified schematics).

ing into account the possible relationships between different parameters (e.g. loss rate and loss mean burst size) when designing the neural network.

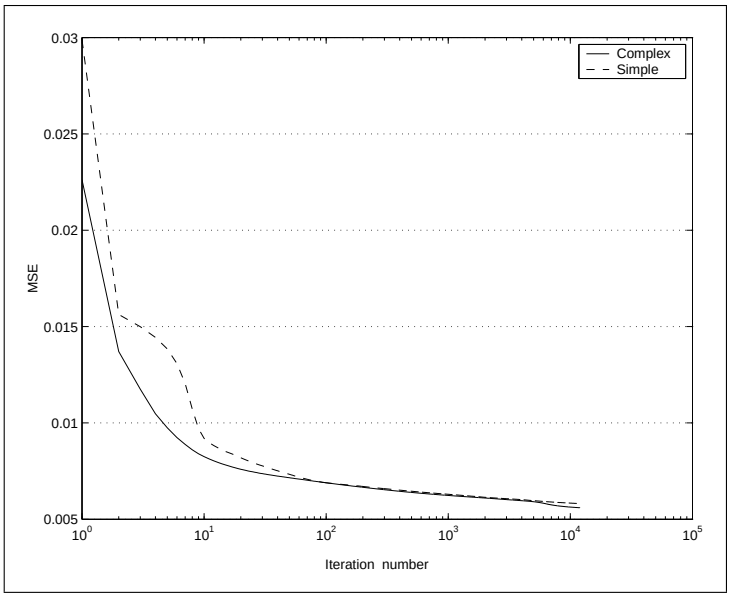
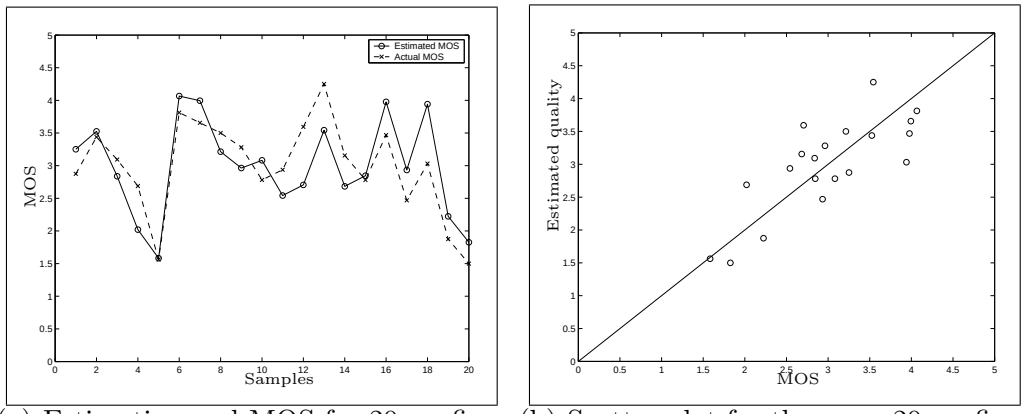


Figure 3. Convergence speed for two different neural network architectures (normalized values).

We used some configurations unknown to the trained networks (that is, not used during the training phase) to verify their accuracy. Fig. 4 shows the results given by a 3–layer network trained with 92 configurations when faced

with the 20 configurations reserved for this validation phase. We also compared the results obtained with some of the human subjects' evaluations, and found the RNN estimation's distance [7] from the MOS to be generally smaller than those of the subjects.



(a) Estimation and MOS for 20 configurations used for validating the RNN

(b) Scatter plot for the same 20 configurations considered in (a) (MSE≈0.21)

Figure 4. Estimations for 20 unknown (randomly chosen) configurations.

We obtained similar results when using only 56 configurations to train the network, and the other 56 to test it. The mean square error for that case was very similar (≈0.23), but there were some outliers.

As for the correlation coefficient, we found it to be dependent on the number of configurations used for the testing, and it was in the 0.71–0.93 range (the lowest value being for the 56–configuration training set, and the highest one for a 102–configuration one). We found that the performance of the RNN was good even when using a very small number of training configurations, which is good, since the main cost of our approach resides in the number of subjective evaluations to be made.

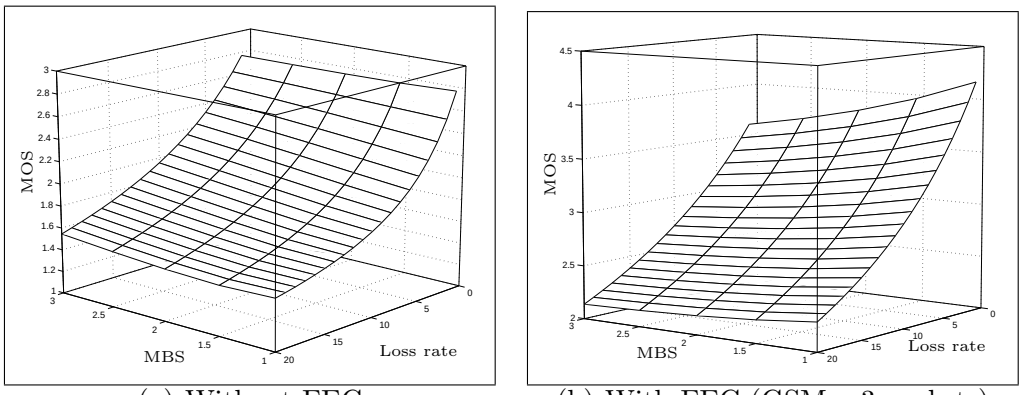
6.2 Effects of the Different Parameters on the Perceived Quality

In this Section we put forward our results concerning the influence of the encoding and the network conditions on the quality as perceived by the end user. To this end, we used a previously trained neural network (trained with a 92–configuration set) to estimate the quality of the speech samples under a number of different combinations of encoding and network parameters.

[7] Measured as the mean square error.

6.2.1 LR and MBS

Let us start by looking at the quality of the received voice stream when the loss rate and the mean loss burst size vary, and the other parameters remain constant. We considered LR values ranging from 1% to 20%, and MBSs ranging from 1 (i.e. mostly independent losses) to 3. The codec used was PCM linear-16, and packet size was 20ms. Fig. 5 shows the evolution of quality as a function of the network loss rate and the MBS, with and without FEC.



(a) Without FEC (b) With FEC (GSM – 3 packets)

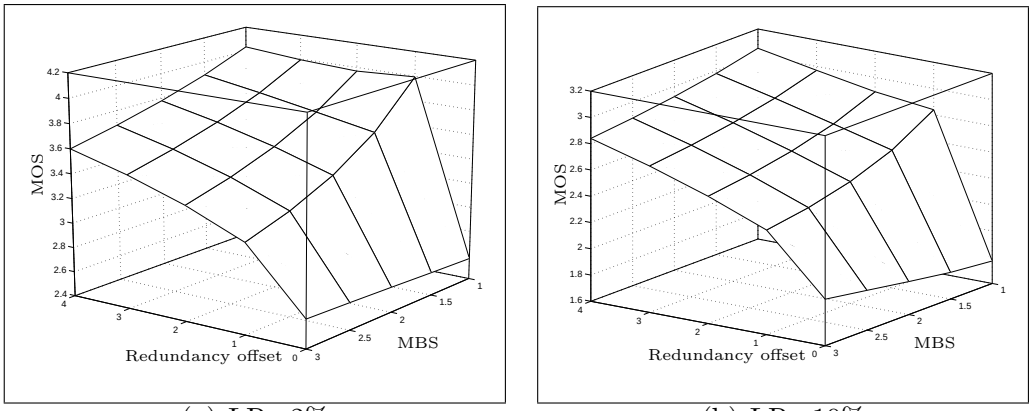
Figure 5. Perceived quality as a function of LR and MBS.

We observed that the fact of using FEC produced an improvement of about one MOS point in the perceived quality. We also verified that when there is no error correction, an increase of the MBS has a slightly positive effect on quality, since there are fewer loss episodes (this had already been noted in [37]). As expected, the FEC scheme used suffers a significant degradation when the MBS is increased, since in this case, more redundant packets are lost in the same loss episode that the original ones.

6.2.2 MBS and Redundancy Offset

Having seen how the MBS affects the performance of the FEC scheme used, it is interesting to see how much does modifying the offset of the secondary encoding improve the perceived quality. For that, we studied the evolution of quality as we varied the MBS and offset parameters. The results are shown in Fig. 6.

We can see that the curves are quite similar regardless of the loss rate, though there is a one point overall difference for the two loss rates considered. As in Fig. 5 (b) we can see that the quality degrades as we increase the MBS. However, we can achieve a significant improvement in quality by increasing the offset of the FEC payload. Again, we observe that when no FEC is used, increasing the MBS produces a slight improvement in quality. It should be noted that although using high values for the redundant data offset counters



(a) LR=2% (b) LR=10%

Figure 6. Variation of perceived quality as a function of MBS and FEC offset (an offset of 0 means that no FEC was used).

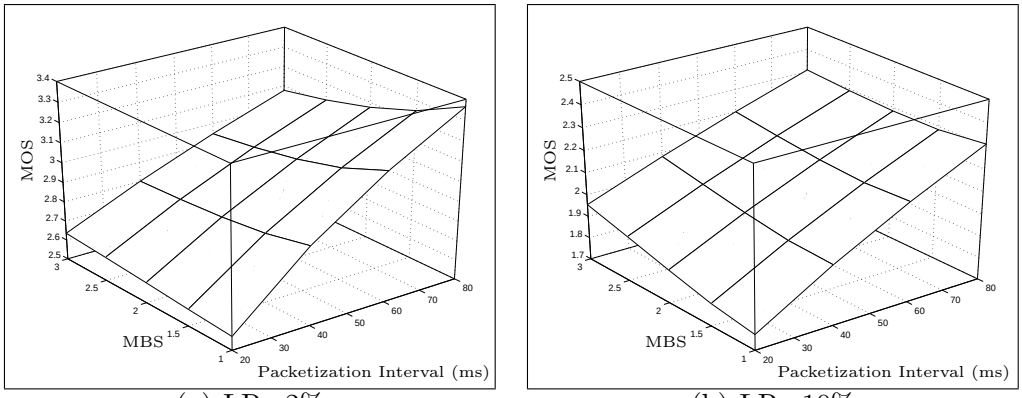
the effects of high MBSs, it also introduces more delay to the transmission, which may decrease the perceived quality for interactive applications, even if the sound quality is better.

6.2.3 MBS and PI

Another interesting aspect to consider is how does the packetization interval affect the quality of the speech streams, under various MBSs. We found that increasing the PI produces an improvement in quality, whether FEC is being used or not, and independently of the MBS size. We haven't found an explanation for this phenomenon in the literature, but we believe that it is due to a lower number of loss episodes being produced. This is reasonable, since when using longer packets, we have fewer ones, and thus, fewer loss episodes. This is interesting, because it allows to have an acceptable[8] quality without adding redundant data, even with relatively high values of LR and MBS. Increasing the PI also produces a better network utilization, since there is less header overhead, and doesn't affect the performance of the FEC scheme used. Fig. 7 shows the results we obtained.

This may not be applicable under certain circumstances, for example, in applications where minimizing the delay is critical, or where receiver based loss concealment techniques such as waveform interpolation are used, in which case a smaller value of PI may be preferred.

[8] Close to 3 points in the 5 point MOS scale.

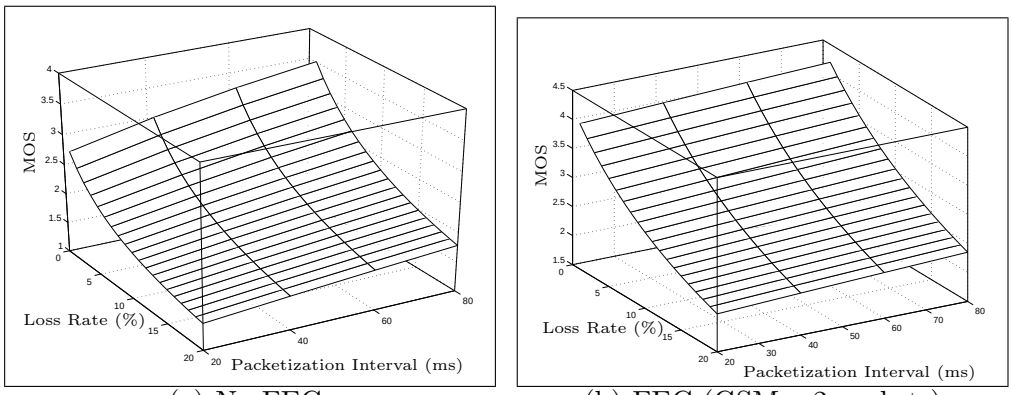


(a) LR=2% (b) LR=10%

Figure 7. Variation of the perceived quality as a function of MBS and PI (without FEC).

6.2.4 LR and PI

To further understand the effect of packetization interval on speech quality, we studied the quality variation as a function of PI and LR, in order to determine in which situations an increase in PI is a good option for improving quality [9]. We found that we can get near–toll quality without FEC at 5% of packet losses, which we believe to be interesting. Figure 8 shows the results obtained.



(a) No FEC (b) FEC (GSM – 2 packets)

Figure 8. Variation of the perceived quality as a function of LR and PI (MBS = 1.3 packets).

We can see that the influence of PI is stronger for low values of LR (almost one MOS point for LR = 1%), and it decreases as the loss rate increases. At LR = 8%, the quality is about 2.5 MOS points, which is not bad considering no error correction is being made. When using FEC, on the other hand, the improvement brought by the increase of PI is not so evident, but it is still there.

[9] We must still consider the remarks made about delay on the previous section.

6.2.5 Codecs and Speech Quality

While the purpose of our study was not the comparison of the codecs' performances, we did study whether the codec used had an important influence on the way the other parameters affect quality. We found that the evolution of quality as a function of the network and redundancy parameters is very similar for both GSM and PCM Linear–16.

We did find, however, that when the network conditions degrade, the difference in quality between both codecs tends to diminish. Figure 9 shows an example of this situation. This difference may be due to GSM being a frame–based codec and PCM being a sample–based one. Frame–based codecs can extrapolate missing data from the decoder's state, and so they may behave better than sample–based codecs under certain network loss conditions. This issue is the subject of current research in our group.

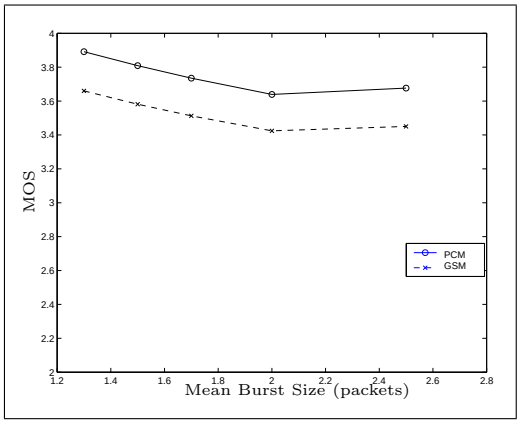


Figure 9. The perceived quality as a function of MBS, for both PCM and GSM (LR=2%, FEC = (GSM – 2)).

7 Applications

This section aims to present some examples of possible applications of the tool presented in this paper, some of which are the subject of current research.

7.1 Quality Control

Probably the most obvious application of being able to quantify the quality as perceived by human subjects as a function of network and coding parameters, is to control it. Attaching a trained RNN to a streaming media player, or a teleconferencing application would allow to optimize the perceived quality by controlling the coding parameters according to the current network state.

Moreover, in a near future when network QoS architectures such as DiffServ become widely available, our tool will help in controlling packet marking, or in negotiating with the network in order to improve transmission quality. Even though in this study we considered only one–way applications, the approach is valid for interactive ones as well, and so other network parameters, such as delay and jitter could also be considered for optimization.

7.2 Pricing

As more real–time multimedia applications become commonplace, the current best–effort Internet architecture is likely to change into a multi–class QoS architecture, in which real–time packets will be treated differently than bulk transfer ones. This will surely imply new pricing schemes, as flat–rates (which are probably the most common ones nowadays) will not be appropriate for this kind of service.

It is likely that the new pricing models will include some form of Service Level Agreements (SLA) which guarantee that the client will receive a minimum quality for the service he is paying. In this context, it is possible to use the approach presented here to verify that the QoS is indeed within the levels specified by the SLA.

Another interesting point which is currently being studied is the adaptation of this method to estimate utility functions, rather than just quality. This would be useful for designing and testing new pricing schemes for QoS–aware networks. Utility functions are monetary valuations of any given product or service, i.e. how much money does the client *feel* that the product or service is worth. Thus, it is important that the perceived utility be well known before establishing a new pricing scheme.

8 Conclusions

This paper proposes an analysis of the combined effect of several quality–affecting parameters on the perceived quality of real–time speech streams sent over a packet network with no provisions for QoS. The proposed approach can be used to analyze the effects of any measurable parameter on the perceived quality of the stream. The goal of this analysis is to help in the understanding of the behavior of real–time speech streams transmitted over a best–effort network such as the Internet. This may be used, for instance, in developing control mechanisms allowing the delivery of the best possible speech quality given the current network state.

Our method aims to overcome the limitations of the available quality measuring techniques in the literature, and it presents several advantages over them: (i) the results obtained correlate well with human perception; (ii) it is not computationally intensive; (iii) it can be used in real–time applications; (iv) some parameters that cannot be easily taken into account by the traditional methods are easy to consider using our approach. To the best of our knowledge, no other approach can currently solve the same problem and achieve the same performances.

Although we based our study on IP networks, analog studies can be done for ATM and wireless technologies as well (the specific type of packet technology has no effect on the relevance of the method we discuss here). Some future research directions for this work include the study of other codecs and parameters (for instance, explicit analysis of jitter effects), two-way interactive sessions, and the extension of the RNN–based quality evaluation approach to general multimedia streams.

References

[1] A.H. Abdelbaki and E. Gelenbe. Random neural network decoder for error correcting codes. In *International Joint Conference on Neural Networks*, volume 5, pages 3241–3245, Washington DC, July 1999. IEEE. ISBN 0780355296.

[2] B. Ahlgren, A. Andersson, O. Hagsand, and I. Marsh. Dimensioning links for IP telephony. Technical Report T2000–09, Swedish Institute of Computer Science (SICS), 2000.

[3] H. Bakircioglu and T. Kocak. Survey of random neural network applications. *European Journal of Operational Research*, 126(2):319–330, 2000.

[4] J. Beerends. Improvement of the p.861 perceptual speech quality measure. ITU-T SG12 COM-34E, December 1997.

[5] J. Beerends and J. Stemerdink. A perceptual speech quality measure based on a psychoacoustic sound representation. *Journal of Audio Eng. Soc.*, 42:115–123, December 1994.

[6] J-C. Bolot. Characterizing end-to-end packet delay and loss in the internet. *Journal of High-Speed Networks*, 2(3):305–323, December 1993.

[7] J-C. Bolot and H. Crépin. Analysis and control of audio packet loss over packet-switched networks. Technical report, France, 1993.

[8] J-C. Bolot, S. Fosse-Parisis, and D.F. Towsley. Adaptive FEC–based error control for internet telephony. In *Proceedings of INFOCOM '99*, pages 1453–1460, New York, NY, USA, March 1999. ISBN 0780354176.

[9] A. Choi and A. Constantinides. Effect of packet loss on 3 toll quality speech coders. In *Second IEE National Conference on Telecommunications*, pages 380–385, York, UK, April 1989.

[10] R. Cole and J. Rosenbluth. Voice over IP performance monitoring. *ACM Computer Communication Review*, 31(2):9–24, April 2001.

[11] C. Cramer and E. Gelenbe. Video quality and traffic QoS in learning-based subsampled and receiver-interpolated video sequences. *IEEE Journal on Selected Areas in Communications*, 18(2):150–167, 2000.

[12] C. Cramer, E. Gelenbe, and H. Bakircioglu. Low bit rate video compression with neural networks and temporal sub-sampling. *Proceedings of the IEEE*, 84 (10):1529–1543, 1996.

[13] C. Cramer, E. Gelenbe, and P. Gelenbe. Image and video compression. *IEEE Potentials*, (1):29–33, March 1998.

[14] A. Cray. Voice over IP: Hear's how. *Data Communications International*, 27 (5):44–59, April 1998.

[15] D. De Vleeschauwer, G.H. Petit, B. Steyaert, S. Wittevrongel, and H. Bruneel. An accurate closed-form formula to calculate the dejittering delay in packetised voice transport. In *Proceedings of International Conference on Networking 2000*, pages 374–385, Paris, 2000.

[16] B. Duysburgh, S. Vanhastel, B. De Vreese, C. Petrisor, and P. Demeester. On the influence of the best-effort network conditions on the perceived speech quality of VoIP connections. In *10th Int. Conf. on Computer Communications and Networks (ICCCN 2001)*, pages 334–339, Scottsdale, Arizona, USA, October 2001. ISBN 0780371283.

[17] K. Eom and J. Jung. A novel echo canceller maintaining high quality of speech under double-talk conditions. In *Fifth Asia-Pacific Conference on Communications and Fourth Optoelectronics and Communications Conference – APCC/OECC*, pages 810–812, Beijing, China, October 1999.

[18] A. Estepa, R. Estepa, and J. Vozmediano. On the suitability of the E-Model to VoIP networks. In *Seventh International Symposium on Computers and Communications, IEEE ISCC 2002*, pages 511–516, Taormina, Italy, July 2002. ISBN 0769516718.

[19] Y. Feng and E. Gelenbe. Adaptive object tracking and video compression. *Network and Information Systems Journal*, 1(4-5):371–400, 1999.

[20] E. Gelenbe. Random neural networks with negative and positive signals and product form solution. *Neural Computation*, 1(4):502–511, 1989.

[21] E. Gelenbe. Stability of the random neural network model. In *Proc. of Neural Computation Workshop*, pages 56–68, Berlin, West Germany, February 1990. ISBN 3540522557.

[22] E. Gelenbe. Learning in the recurrent random neural network. *Neural Computation*, 5(1):154–511, 1993.

[23] E. Gelenbe. G-networks: new queueing models with additional control capabilities. In *Proceedings of the 1995 ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*, pages 58–59, Ottawa, Ontario, Canada, 1995. ISBN 0-89791-695-6.

[24] E. Gelenbe. Towards networks with cognitive packets. In *Proc. IEEE MASCOTS Conference*, pages 3–12, San Francisco, CA, 2000.

[25] E. Gelenbe. Reliable networks with cognitive packets. In *International Symposium on Design of Reliable Communication Networks*, pages 28–35, Budapest, Hungary, October 2001. Invited Keynote Paper.

[26] E. Gelenbe. Cognitive packet networks: QoS and performance. In *Proc. IEEE Computer Society MASCOTS02 Symposium*, pages 3–12, Fort Worth, TX, October 2002. ISBN 0769518400. Opening Keynote.

[27] E. Gelenbe, H. Bakircioglu, and T. Kocak. Image processing with the random neural network. In *Proc. of the IEEE Digital Signal Processing Conference*, volume 1, pages 243 –248, New York, June 1997. ISBN 0780341376.

[28] E. Gelenbe, C. Cramer, M. Sungur, and P. Gelenbe. Traffic and video quality in adaptive neural compression. *Multimedia Systems*, 4(6):357–369, 1996.

[29] E. Gelenbe, T. Feng, and K.R.R. Krishnan. Neural network methods for volumetric magnetic resonance imaging of the human brain. *Proceedings of the IEEE*, 84(10):1488–1496, 1996.

[30] E. Gelenbe and J.M. Fourneau. Random neural networks with multiple classes of signals. *Neural Computation*, 11(3):953–963, 1999.

[31] E. Gelenbe and K. Hussain. Learning in the multiple class random neural network. *IEEE Trans. on Neural Networks*, 13(6):1257–1267, 2002.

[32] E. Gelenbe, T. Kocak, and L. Collins. Sensor fusion for mine detection with the RNN. In *Proceedings of ICANN97, International Conference on Artificial Neural Networks*, pages 937–942, Lausanne, Switzerland, October 1997. ISBN 3540636315.

[33] E. Gelenbe, R. Lent, and Z. Xu. Design and performance of cognitive packet networks. *Performance Evaluation*, 46:155–176, 2001.

[34] E. Gelenbe, Z.-H. Mao, and Y.-D. Li. Function approximation with spiked random networks. *IEEE Trans. on Neural Networks*, 10(1):3–9, 1999.

[35] E. Gilbert. Capacity of a burst–loss channel. *Bell Systems Technical Journal*, 5(39), September 1960.

[36] T. A. Hall. Objective speech quality measures for internet telephony. In *Voice over IP (VoIP) Technology, Proceedings of SPIE*, volume 4522, pages 128–136, Denver, CO, USA, August 2001.

[37] D. Hands and M. Wilkins. A study of the impact of network loss and burst size on video streaming quality and acceptability. In *Interactive Distributed Multimedia Systems and Telecommunication Services Workshop*, October 1999. ISBN 3540665951.

[38] M. Hassan, A. Nayandoro, and M. Atiquzzaman. Internet Telephony: Services, Technical Challenges, and Products. *IEEE Communications Magazine*, 38(4): 96–103, April 2000.

[39] O. Hodson. Packet reflector.

[40] J.B. Hooper and M.J. Russell. Objective quality analysis of a voice over internet protocol system. *IEEE Electronics Letters*, 36(22):1900 –1901, 2000.

[41] IETF Network Working Group. RTP payload for redundant audio data (RFC 2198), September 1997.

[42] IETF Network Working Group. A message bus for local coordination (RFC 3259), April 2002.

[43] ITU-R Recommendation BT.500-10. Methodology for the subjective assessment of the quality of television pictures. In *International Telecommunication Union*, March 2000.

[44] ITU-T Recommendation G.107. The E-model, a computational model for use in transmission planning. `http://www.itu.int/`.

[45] ITU-T Recommendation P.800. Methods for subjective determination of transmission quality. `http://www.itu.int/`.

[46] D. Kirby, K. Warren, and K. Watanabe. Report on the formal subjective listening tests of MPEG-2 nbc multichannel audio coding. In *ISO/IEC JTC1/SC29/WG11/N1419*, November 1996.

[47] N. Kitawaki and K. Itoh. Pure delay effects on speech quality in telecommunicatios. *IEEE Journal on selected Areas in Communications*, 9(4): 586–593, May 1991.

[48] A. Krogh and J. Hertz. A simple weight decay can improve generalization. In *Advances in Neural Information Processing Systems*, volume 4, pages 950–957, 1992.

[49] B. Li and X. Cao. Experimental results on the impact of cell delay varition on speech quality in ATM networks. In *IEEE International Conference on Communications - ICC*, volume 1, pages 477–481, Atlanta, GA, USA, June 1998. ISBN 0780347889.

[50] mbus.org. Mbus web site - `http://www.mbus.org`.

[51] S. Mohamed, F. Cervantes, and H. Afifi. Integrating networks measurements and speech quality subjective scores for control purposes. In *Proceedings of IEEE INFOCOM'01*, pages 641–649, Anchorage, AK, USA, April 2001. ISBN 0780370163.

[52] S Mohamed and G. Rubino. A study of real–time packet video quality using random neural networks. *IEEE Transactions On Circuits and Systems for Video Technology*, 12(12):1071 –1083, December 2002.

[53] S. Pennock. Accuracy of the perceptual evaluation of speech quality (PESQ) algorithm, January 2002.

[54] A. Rix. Advances in objective quality assessment of speech over analogue and packet-based networks. In *the IEEE Data Compression Colloquium*, London, UK, November 1999.

[55] H. Sanneck, G. Carle, and R. Koodli. A framework model for packet loss metrics based on loss runlengths. In *Proceedings of the SPIA/ACM SIGMM Multimedia Computing and Networking Conference*, pages 177–187, San Jose, CA, January 2000.

[56] L. Sun and E. Ifeachor. Perceived speech quality prediction for voice over IP-based networks. In *Proceedings of IEEE International Conference on Communications (IEEE ICC'02)*, pages 2573–2577, New York, USA, April 2002. ISBN 0780374002.

[57] UCL. Robust Audio Tool website.

[58] S. Voran. Estimation of perceived speech quality using measuring normalizing blocks. In *IEEE Workshop on Speech Coding For Telecommunications Proceeding*, pages 83–84, Pocono Manor, PA, USA, September 1997. ISBN 0780340736.

[59] A. Watson and M.A. Sasse. Evaluating audio and video quality in low-cost multimedia conferencing systems. *ACM Interacting with Computers Journal*, 8 (3):255–275, 1996.

[60] M. Yajnik, S. Moon, J.F. Kurose, and D.F. Towsley. Measurement and modeling of the temporal dependence in packet loss. In *Proccedings of IEEE INFOCOM '99*, pages 345–352, 1999.

[61] W. Yang. *Enhanced Modified Bark Spectral Distortion (EMBSD): an Objective Speech Quality Measrure Based on Audible Distortion and Cognition Model*. PhD thesis, Temple University Graduate Board, May 1999.