Evaluating Users' Satisfaction in Packet Networks Using Random Neural Networks

Gerardo Rubino[1], Pierre Tirilly[1], and Martın Varela[2,⋆]

[1] Inria/Irisa, Rennes
rubino@irisa.fr, pierre.tirilly@ens.insa-rennes.fr
[2] SICS
mvarela@sics.se

Abstract. Quantifying the quality of a video or audio transmission over the Internet is usually a hard task, as based on the statistical processing of the evaluations made by a panel of humans (the corresponding and standardized area is called *subjective testing*). In this paper we describe a methodology called Pseudo-Subjective Quality Assessment (PSQA), based on Random Neural Networks, which is able to perform this task automatically, accurately and efficiently. RNN had been chosen here because of their good performances over other possibilities; this is discussed in the paper. Some new insights on PSQA's use and performance are also given. In particular we discuss new results concerning PSQA–based dynamic quality control, and conversational quality assessment.

1 Introduction

When we need to quantitatively assess the quality of an audio or video transmission over the Internet, the most accurate way to do it is to have a panel of humans perform the assessment on actual test sequences representative of the conditions studied. This is a standard procedure for which norms exist (see Subsection 2.1 for examples on voice multimedia communications). There are some methods to do an automatic quantitative assessment as well, that is, without using subjective tests, but they suffer either from poor accuracy or efficiency, or both (see Section 2). As an alternative, the Pseudo-Subjective Quality Assessment (PSQA) technology has been recently developed. It allows to automatically quantify the quality of a video or audio communication over a packet network, as perceived by the user. The PSQA technique is accurate, which means that it correlates well with the values given by panels of human observers, and efficient, because it can work, if necessary, in real time. It has been tested on video [10] and audio [13] flows. It can be applied in many areas, for instance, for the analysis of the impact of different factors on quality (see the mentioned papers, and [12] for an example of the study of Forward Error Correction techniques

⋆ M. Varela´s work was carried out during the tenure of an ERCIM fellowship.

S. Kollias et al. (Eds.): ICANN 2006, Part I, LNCS 4131, pp. 303–312, 2006.

in audio streaming), or for performance evaluation of communication networks using models [14].

PSQA is based on learning how human observers quantify the quality of a flow under standardized experimental conditions. The learning process consists of training a particular type of Neural Network, a Random Neural Network (RNN), to capture the relation between a set of factors having an *a priori* strong impact on the perceived quality and the latter. Let us briefly recall the structure of the RNN tool, before describing in some detail the technique used for quantitative quality assessment of video and audio flows. For a global presentation of PSQSA with a detailed description of the RNN tool see [11].

RNN have been developed by Erol Gelenbe in a series of papers (for starters, see [1], [3], [2]). In this paper we will use 3-layer feedforward RNN. Such a network can be seen as a parametric function $\nu()$ mapping a vector of size $I+J$, denoted here $(\boldsymbol{C}, \boldsymbol{N}) = (C_1, \cdots, C_I, N_1, \cdots, N_J)$, into a real number. Let us denote by vector $\boldsymbol{W}$ the function's parameters. The input variables $C_1, \cdots, C_I$ are related to the network connection, or to the codec used (example: the bit rate), and the variables $N_1, \cdots, N_J$ correspond to the network state (example: the loss rate). The value of the function is the quality of the audio or video connection. The function's parameters are the weights in the neural network.

As a neural network, our RNN has three layers, the input one with $I + J$ variables, the hidden one with H units, and the output layer with a single node. The mapping can be explicitely written as

$$\nu(\boldsymbol{W}; \boldsymbol{C}, \boldsymbol{N}) = \frac{\sum_{h=1}^{H} \varrho_h w_{ho}^{+}}{r_o + \sum_{h=1}^{H} \varrho_h w_{ho}^{-}},$$

where

$$\varrho_h = \frac{\sum_{i=1}^{I} C_i r_i^{-1} w_{ih}^{+} + \sum_{j=1}^{J} N_j r_j^{-1} w_{jh}^{+}}{r_h + \sum_{i=1}^{I} C_i r_i^{-1} w_{ih}^{-} + \sum_{j=1}^{J} N_j r_j^{-1} w_{jh}^{-}}$$

is the *activity rate* of hidden neuron h. The strictly positive numbers r_o, r_h for $h = 1..H$, r_i for $i = 1..I$ and r_j for $j = 1..J$, correspond to the firing rates of the neurons in the network (respectively, for the output one, the hidden nodes, and the $I + J$ input ones). The weights are the variables tuned during the learning process. We denote by w_{uv}^{+} (resp. by w_{uv}^{-}) the weight corresponding to an exiting (resp. inhibiting) signal going from neuron u to neuron v (observe that both numbers w_{uv}^{+} and w_{uv}^{-} are ≥ 0). For the interpretation and the dynamics of a RNN see the references cited above. For our purposes here, we can just see it as a rational parametric function. Learning will thus consist of finding appropriate values of the weights capturing the mapping from $(\boldsymbol{c}^{(k)}, \boldsymbol{n}^{(k)})$ to the real number $q^{(k)}$ where $q^{(k)}$ is the quality given by a panel of human observers to some audio or video sequence (depending on the application) when the source parameters had the values present in $\boldsymbol{c}^{(k)}$ and the parameters caracterizing the network had the values in vector $\boldsymbol{n}^{(k)}$, for $k = 1..K$.

To be more specific, let us describe how PSQA is used, with a simple example. Assume we have some audio or video stream whose quality, as perceived by the

users, is to be evaluated. Assume we limit the application of PSQA to just $I = 1$ source parameter, the bit rate C_1, and to $J = 2$ network parameters, the end-to-end loss rate N_1 and the mean size of bursts of (consecutive) lost packets, N_2. The goal of the process is to come up with a function $\nu(\boldsymbol{W}; C_1, N_1, N_2)$ mapping the values of these 3 parameters into quality, for instance in a standard MOS range. We start by choosing K series of values for the bit rate, the loss rate and the mean loss burst size, denoted $(c_1^{(k)}, n_1^{(k)}, n_2^{(k)})$ for $k = 1..K$. Then, we select some short (audio/video) sequence (norms recommend to use sequences with a few seconds length) and we send it K times through a controllable network (using a testbed, for instance), where in the kth case the three selected parameters have values $c_1^{(k)}$, $n_1^{(k)}$ and $n_2^{(k)}$. The K resulting sequences are shown to a panel of humans and a subjective testing experiment is performed, following an appropriate norm depending on the type of flow (see below), which allows to obtain the (measured) perceived quality of each received sequence, denoted $q^{(1)}, \cdots, q^{(K)}$. Then, a RNN is trained to build such a $\nu()$ function, using the K input-output values obtained from the testing phase. The usual way to do this is to separate this data into a training part, used to build the approximation $\nu()$, and a validation one, used to test the predictive capacity of $\nu()$.

PSQA has been tested on video and on audio flows. In this paper, we will discuss only the latter, for sake of space. Next section describes previous and current work on the quantitative analyses of the perceived quality of voice communications, with some new results concerning interactive voice sessions. Section 3 will then discuss about the performance of our learning tool, RNN, as compared to other available tools such as standard Artificial Neural Networks. Section 4 concludes the paper.

2 Using PSQA to Analyze Perceived Voice Quality

2.1 On Voice Quality Assessment

When assessing voice quality, there are two very different kinds of subjective tests one can perform. The first kind, which is the one most widely used, is related to the quality of the voice itself, and so it does not take other factors inherent to conversation into account. We refer to these assessments as unidirectional, since the only point of interest is the quality of the received signal. The other kind of tests concern actual conversational quality, and in a way are a superset of the first kind. In these, we not only consider the voice quality itself, but also other factors (mostly delay-related) which affect the perceived quality of an *actual conversation*. There exist standard procedures for performing subjective assessments of both unidirectional (e.g. [6]) and interactive (e.g. [8]) speech. Interactive tests are more difficult to set up and perform than unidirectional ones.

In any case, all subjective tests are expensive in terms of time and money, so a significant research effort has been directed toward developing *objective* tests. These provide a cheaper and more practical alternative to subjective tests. Most

subjective tests operate by comparing the received signal to the original one, and estimating the quality from the difference between the two. The estimation can be done by mechanisms ranging from a simple SNR measurement to very complex psychoacoustic models. Of the methods available in the literature, only the ITU E-model [4] and P.563 algorithm [5] can perform their assessment without requiring the original signal. The accuracy of objective assessment techniques is quite variable, in particular when considering VoIP. The best performance among the purely objective metrics is that of PESQ [7], which attains up to about 0.92 correlation with subjective scores (a typical "best score").

The need for being able to accurately assess VoIP quality in real–time arises in several scenarios, such as the development of dynamic quality–control mechanisms. Traditional methods of voice quality assessment tend to only cover one of the two conditions needed. Either they are adequately accurate (i.e. PESQ) but not able to perform in real–time, or they do work in real–time, but their accuracy is not as high as needed (i.e. the ITU E–model).

In this respect, PSQA offers the best of both worlds, since it is very accurate, and it can work in real–time. This is a consequence of not needing the original signal to perform the assessment, and of the simplicity of the calculations on the RNN.

2.2 Unidirectional Voice Quality Assessment and Its Applications

We have succesfully used PSQA to analize the ways in which different parameters affect the voice quality for VoIP streams [13]. To this end, we used six quality–affecting parameters, and using PSQA, we studied the relations between them and the quality as perceived by the end user. The parameters considered were the codec used, whether error correction was being used, the offset of the error correction, the loss rate and mean loss burst size found in the network, and the packetization interval (i.e. the length of the speech contained in each packet).

In this study, we obtained a 0.94 correlation coefficient between the RNN predictions and subjective scores, which is on par (slightly better, actually) with the best objective assessment techniques currently available. The results obtained allow us to understand, for instance, how the loss rate affects the perceived quality, for different codecs and with or without error correction. Figure 1(a), for example, shows how the packetization interval and the mean loss burst size affect the perceived quality.

An immediate application of these results is modifying application–level parameters to acommodate variations in the network conditions, improving, if possible, the perceived quality. We have developed two simple algorithms which allow to manipulate the codec, forward error correction, and packetization interval values to dynamically optimize the quality. The first algorithm takes a naive approach, trying to keep the quality between two thresholds at all costs. The second algorithm takes bandwidth consumption into account, and tries to keep the quality between the same bounds, but resorting to lower bit rate encodings whenever possible. Both algorithms present a similar improvement on quality

over scenarios lacking dynamic control, and depending on network load, one may perform slightly better than the other. Figure 1(b) shows the performance of the simplest of both algorithms when the network conditions degrade.

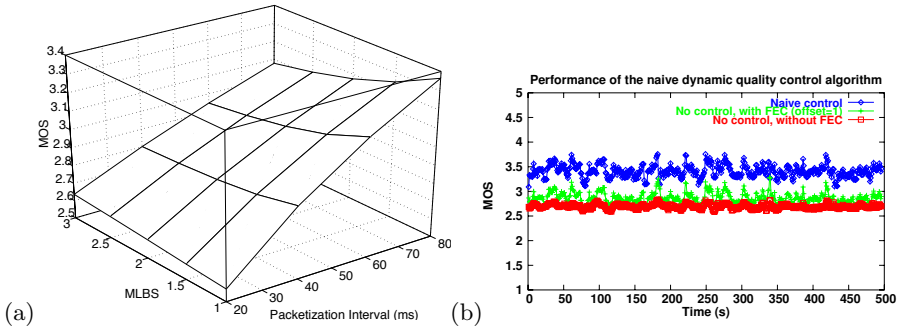

Fig. 1. (a) Variation of the perceived quality as a function of the Mean Loss Burst Size (MLBS) and the Packetization interval, without error correction. Loss rate is 2%. (b) Performance of a PSQA–based control algorithm based on the same RNN. The network parameters found in these trace are as follows: Loss rate: 12.47%, Mean loss burst size: 2.13 packets. Under these network conditions, the (naive) control algorithm offers a noticeably better quality than when no control is used.

2.3 Conversational Quality Assessment

As mentioned above, conversational quality assessment is significantly more difficult to perform than the unidirectional one. To our knowledge, there is no purely objective assessment method available in the literature able to predict conversational quality, nor have there been subjective assessment campaign covering as many parameters as we used in our studies. We used PSQA to develop an understanding of the how the conversational quality evolves with the different parameters which affect it. In particular, we were interested in the relative impacts of delay and loss rate on the overall quality.

The set of parameters we considered for this study were the bit rate (using a single codec), the forward error correction settings, the packet loss rate and mean loss burst size, and the one–way delay and its variability (jitter). As the study was focused on VoIP, we used subjects with previous experience with VoIP applications. The results we obtained with the RNN present a correlation coefficient of 0.95 with subjective scores, which is very good. Given the large parameter space considered, it is worth noting that the use of RNN for the estimations allowed for excellent generalization capabilities.

Among the results we obtained by analizing the RNN behavior, the most interesting one is that VoIP users seem much more tolerant to delay than it is usually reported for PSTN systems (and by extension adopted for VoIP). Moreover, we found that the impact of the delay on the conversational quality was relatively small compared to that of the loss rate, and that one–way delays as big

as 600ms made little difference on the perceived quality for loss rates of about 7% and up. This is good news, since it allows implementors to use better loss concealment or correction techniques, at the expense of delay, resulting in an improved overall quality. Figure 2 shows the impact of delay on the conversational quality for diferent loss rates.

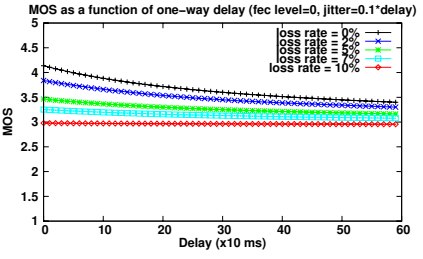

Fig. 2. Perceived quality as a function of delay for some loss rates (without FEC). Jitter was 10% of the delay. Note how the impact of the delay diminishes as the loss rate increases.

3 Comparing RNN Against Other Tools

As mentioned above, the choice of RNN over other tools for implementing PSQA is not arbitrary. In [9], [10], a first performance comparison had been made, establishing that RNN offer two advantages over traditional ANN for our applications. The first one is that of better generalization capabilities (i.e. less over-training). Since our approach only allows for relatively few learning points in a vast input space, this is a very important characteristic. The second advantage was that RNN are less sensitive to variations in the number of neurons in the hidden layer, which allows to obtain good performance without needing to optimize the neural network architecture.

In this section we discuss more in deep about the performances of RNN compared to Bayesian networks and standard ANN, the two families of alternate tools we used. We also present results of new comparison tests between ANN and RNN performance for both unidirectional and interactive VoIP applications.

3.1 Using Naive Bayesian Classifiers

We have tried naive Bayesian classifiers for providing MOS estimations for PSQA. This was meant primarily to test their accuracy more than for production use, since two reasons make them less desirable than RNN:

- they require a larger amount of training data, and
- they only provide classification into discrete values, whereas RNNs provide a continuous, differentiable function.

However, this kind of classifier is easier to implement than RNNs, and is computationally trivial, which, despite the RNN's simplicity, could be useful in certain contexts where computing power or memory are very limited.

Naive Bayesian classification is based on the observation of a sufficiently large amount of data, and the assumption that those observations are statistically independent. We perform the classification by counting, for *a large number of assessments*, the number of times each score happens for each value of each parameter considered. This allows us to estimate, for each value of the selected parameters, the probability that a given score will happen, looking at the quality as a random object. In other words, we estimate the conditional probability of the score (or quality) being equal to q given that $\boldsymbol{C} = \boldsymbol{c}$ and $\boldsymbol{N} = \boldsymbol{n}$, for any possible configuration $(\boldsymbol{c}, \boldsymbol{n})$ of the parameters' values (we are assuming a discrete range for quality as well as for the parameters). Then, we can find the score which is most likely to be associated with each configuration.

As stated above, this approach needs a large body of data for the training process. As not enough subjective assessments were available to train the classifier, we needed to generate data suitable for assessing the performance of this tool. To this end, we generated a large set of assessments (covering the whole parameter space) with a previously trained RNN, and used them instead of actual subjective scores.

We performed several tests, using over 20,000 configurations to train the classifier. Although validation results were reasonably accurate, the performance of this classifier with configurations for which we had subjective scores was consistently and significantly lower than that of the RNN.

Among the possible explanations for this bad performance, the foremost is that the quality–affecting parameters, and their relation to MOS scores are not actually independent. Given the results obtained, it is reasonable to think that they are too correlated for the assumption of statistical independence to hold.

As mentioned before, even if this approach did perform well, it does not offer the advantages of using a RNN, and in any case, it needs the RNN (or another trained estimator) in order to compensate for the usual lack of available subjective data.

3.2 RNN Compared to Standard ANN Tools

In our experiments, we also tested the performances of classical neural networks in the previously described learning tasks for automatic perceived quality assessment in VoIP applications. The global conclusion is that RNN outperforms Artificial Neural Networks in our context. In this subsection we present further results concerning the accuracy of PSQA when implemented with RNN and ANN. To this end, we used 15 different training and validation sets, with varying sizes and also by using different parts of the data for training and for predicting, for both the one-way and interactive cases in VoIP communications. We then trained RNN and an ANN with appropriate architectures and compared their performances.

In order for the comparison to be fair, we tried to optimize the performance of the neural nets by using different number of hidden layers and different sizes for those hidden layers. In the ANN case, we used commercial packages and thus we looked at the results given by different training algorithms. The best results in both types of networks were obtained with three-layer architectures. For the ANN, the optimal hidden layer sizes were between 6 and 8 neurons, depending on the input data, and for the RNN, a 10-neuron hidden layer provided the best performance. For the RNN, we also considered the simplest architecture possible, namely 6 input neurons, and one output neuron, with no hidden units, as [14] reported acceptable results even with this simple structure. The best training algorithms from the quality of the predictions were Levenberg-Marquardt for the ANN. Our implementation of RNN only uses a simple gradient descent algorithm.

Tables 1 and 2 show the sizes of the training, validation and testing sets used (all randomly chosen). As classically done in neural network methodology, the validation set was used to stop the training when the network generalized well, in order to select good candidates in each family (ANN, RNN), and the test set was used to mesure the real performance of the network (for the comparisons).

Table 1. Number of samples used to train, validate and test the neural nets (one-way streams) in order to compare RNN vs ANN performance

Training	Validation	Test	Training	Validation	Test	Training	Validation	Test
92	10	10	82	20	10	72	20	20
92	10	10	82	10	20	72	30	10
92	10	10	82	10	20	72	10	30
82	20	10	82	10	20	62	10	40
82	20	10	72	20	20	62	20	30

Table 2. Number of samples used to train, validate and test the neural nets (interactive streams) in order to compare RNN vs ANN performance

Training	Validation	Test	Training	Validation	Test	Training	Validation	Test
100	10	10	90	20	10	80	20	20
100	10	10	90	10	20	80	30	10
100	10	10	90	10	20	80	10	30
90	20	10	90	10	20	70	10	40
90	20	10	80	20	20	70	20	30

We found that, although the training error reached (calculated as MSE) is about one order of magnitude lower for ANN than for RNN, the validation results are consistently and significantly better for the RNN. Figures 3 (a) through (d) show the results obtained for both one-way and interactive streams, for the 15 validation and test data sets.

It is interesting that for the interactive scenario, the ANNs performance is noticeably better than for the one-way case. However, the MSE obtained with RNN are lower, and for the one-way case, the difference is very noticeable.

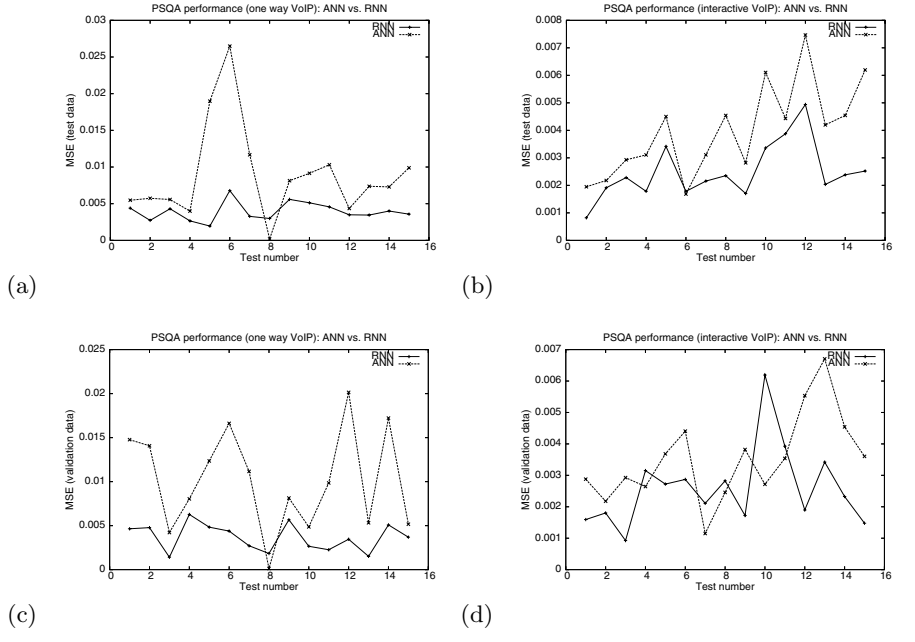

Fig. 3. MSE comparison of RNN and ANN for (a) test data, one-way streams, (b) test data, interactive streams (c) validation data, one-way streams and (d) validation data, interactive streams

4 Conclusions

The PSQA technology allows to provide an accurate and automatic quantitative evaluation of the perceived quality of an audio or video communication over a packet network, where the flow is subject to different kinds of distortions (losses by congestion, delays and jitter, etc.). It has been succesfully tested and used on video and audio streams. In this paper we report novel results on interactive speech quality, obtained by using a RNN-based PSQA tool, and about two proof–of–concept PSQA–based dynamic quality control algorithms. The method comes up with an approximation of the perceived quality as a nice function of measurable parameters, thanks to the RNN technology. It can be applied to the monitoring of an existing network or for control purposes, as well as to the analyzing the impact of specific parameters on the perceived quality.

This paper also shows some results illustrating how RNN outperforms standard ANN as well as Bayesian networks in performing the assessment task.

In particular, we have performed an in-depth performance comparison between ANN and RNN–based PSQA implementations, both for unidirectional nad interactive speech streams.

One of the topics of further research work is to extend these experiments to more cases. Another important research direction for PSQA development is the analysis of the impact of using more efficient learning techniques in the RNN tool.

References

1. E. Gelenbe. Random Neural Networks with Negative and Positive Signals and Product Form Solution. *Neural Computation*, 1(4):502–511, 1989.
2. E. Gelenbe. Stability of the Random Neural Network Model. In *Proc. of Neural Computation Workshop*, pages 56–68, Berlin, West Germany, February 1990.
3. E. Gelenbe. Learning in the Recurrent Random Neural Network. *Neural Computation*, 5(1):154–511, 1993.
4. ITU-T Recommendation G.107. The E-model, a Computational Model for Use in Transmission Planning, March 2005. http://www.itu.int/.
5. ITU-T Recommendation P.563. Single–ended Method for Objective Speech Quality Assessment in Narrow–band Telephony Applications, May 2004.
6. ITU-T Recommendation P.800. Methods for Subjective Determination of Transmission Quality, August 1996.
7. ITU-T Recommendation P.862. Perceptual Evaluation of Speech Quality (Pesq), an Objective Method for End-To-End Speech Quality Assessment of Narrowband Telephone Networks and Speech Codecs, 2001.
8. ITU-T Recommendation P.920. Interactive test methods for audiovisual communications, 2000.
9. S. Mohamed. *Automatic Evaluation of Real-Time Multimedia Quality: a Neural Network Approach*. PhD thesis, INRIA/IRISA, Univ. Rennes I, Rennes, France, jan 2003.
10. 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.
11. G. Rubino. Quantifying the quality of audio and video transmissions over the internet: the psqa approach. In J. Barria, editor, *Design and Operations of Communication Networks: A Review of Wired and Wireless Modelling and Management Challenges*. Imperial College Press, 2005.
12. G. Rubino and M. Varela. Evaluating the utility of media–dependent FEC in VoIP flows. In *LNCS 3266: Proceedings of the Fifth International Workshop on Quality of future Internet Services (QofIS'04)*, Barcelona, Spain, September 2004.
13. G. Rubino, M. Varela, and S. Mohamed. Performance evaluation of real-time speech through a packet network: a random neural networks-based approach. *Performance Evaluation*, 57(2):141–162, May 2004.
14. Gerardo Rubino and Martn Varela. A new approach for the prediction of end-to-end performance of multimedia streams. In *In Proceedings of the Measurement of Speech and Audio Quality in Networks workshop (MESAQIN'04)*, September 2004.