N° d'Ordre: 2742
de la thése

THÈSE

présentée

DEVANT L'UNIVERSITÉ DE RENNES I

pour obtenir

le grade de : ***DOCTEUR DE L'UNIVERSITÉ DE RENNES I***

Mention: INFORMATIQUE

PAR

Samir Ahmed MOHAMED

Équipe d'accueil : IRISA

École Doctorale : Mathématiques, Informatique, Signal, Électronique, Télécommunications (MATISSE)

Composante universitaire : Institut de Formation Supérieur en Informatique et Communication (IFSIC)

TITRE DE LA THÈSE :

Évaluation automatique de la qualité des flux multimédias en temps réel : une approche par réseaux de neurones

SOUTENUE LE 7 janvier 2003 devant la commission d'Examen

COMPOSITION DU JURY :

MM. :	**Jean-Michel**	**Fourneau**	**Professeur, Univ. de Versailles**	**Rapporteurs**
	Samir	**Tohmé**	**Professeur, ENST de Paris**	
MM. :	**Francisco**	**Cervantes**	**Professeur, ITAM**	**Examinateurs**
	Erol	**Gelenbe**	**Professeur, Univ. of Central Florida**	
	Christine	**Guillemot**	**Directeur de Recherche, IRISA/INRIA**	
	Gerardo	**Rubino**	**Directeur de Recherche, IRISA/INRIA**	

Remerciements

Ce travail n'aurait pu être réalisé sans le soutien de plusieurs personnes. L'humour, la disponibilité et les compétences de chacun m'ont souvent permis de franchir des obstacles en apparence insurmontables. Que soient donc ici remerciés tous ceux qui m'ont communiqué l'énergie et la confiance nécessaires au déroulement de cette thèse.

Je tiens à remercier en tout premier lieu Gerardo Rubino, directeur de recherche à l'INRIA et responsable du projet ARMOR, qui a dirigé cette thèse, pour sa disponibilité, son amabilité, sa gentillesse et sa générosité. Tout au long de ces trois années, il a su orienter mes recherches aux bons moments. Pour tout cela, ainsi que sa confiance et son soutien financier en fin de thèse, je le remercie vivement.

Je remercie les rapporteurs de cette thèse : Jean-Michel FOURNEAU, professeur à l'université de Versailles, et Samir TOHME, professeur à l'ENST de Paris, pour la rapidité avec laquelle ils ont lu mon manuscrit et l'intérêt qu'ils ont porté à mon travail. Merci également aux autres membres du jury qui ont accepté de juger ce travail : Francisco Cervantes, professeur à l'ITAM, Erol Gelenbe, professeur à l'université de *Central Florida* , et Christine Guillemot, directrice de recherche à l'INRIA et responsable du projet TEMICS. En particulier, je voudrais remercier Prof. Francisco Cervantes pour l'aide qu'il m'a apportée ; sans lui une grande partie de ce travail n'aurait vu le jour.

Je tiens à remercier Hossam Afifi, HDR, qui a dirigé avec Gerardo Rubino ce travail pendant la première année.

Je voudrais également remercier tous ceux et celles qui ont eu la patience de m'entourer tout au long de ce travail, notamment mes collègues à l'IRISA et à l'ENST de Bretagne.

Cette thèse est dédiée à ma femme Naglaa ; mes filles :Riham, Nermine et Heba ; mes parents : Ahmed et Salma ; mes frères : Khaled, Adel et Ihab ; et mes sœurs : Nevine et Mona. Je n'oublierai jamais les aides permanentes reçues de ma femme et mes parents. Sans leurs aides nombreuses et variées, ces travaux n'auraient jamais pu aboutir.

Table des matières

Publications **XIX**

Acronyms **XXI**

Glossary **XXIII**

Évaluation automatique de la qualité des flux multimédias en temps réel : une approche par réseaux de neurones **1**

1.1 Introduction . . . 1
1.1.1 Les motivations . . . 1
1.1.2 Les contributions de cette thèse . . . 4
1.1.3 Vue d'ensemble de la dissertation . . . 5
1.2 Descriptions de notre nouvelle méthode en général . . . 6
1.2.1 Vue d'ensemble de la méthode . . . 6
1.2.2 Essais subjectifs de la qualité . . . 7
1.2.3 Calcul de MOS et analyse statistique . . . 8
1.2.4 Comparaison entre ANN et RNN . . . 9
1.2.5 Choix de paramètres . . . 10
1.3 La mesure de la qualité de la parole en temps réel . . . 11
1.3.1 Paramètres affectant la qualité de la parole . . . 12
1.3.2 Une base de données de MOS pour différentes langues . . . 12
1.3.3 Évaluation de la qualité de la parole par le RN . . . 12
1.3.4 Performance d'autres mesures de la qualité de la parole . . . 15
1.4 Évaluation de la qualité de la vidéo . . . 16
1.4.1 Les paramètres influant la qualité de la vidéo . . . 16
1.4.2 Résultats . . . 17
1.4.3 Les performances des autres mesures de la qualité . . . 18
1.5 Études des effets de paramètres sur la qualité . . . 19
1.5.1 Impact des paramètres sur la qualité de la parole . . . 19
1.5.2 Impact des paramètres sur la qualité de la vidéo . . . 21
1.6 Un nouveau mécanisme de contrôle de débit . . . 22
1.6.1 TFRC basé sur une équation (EB-TFRC) . . . 23
1.6.2 Notre protocole de contrôle de débit . . . 23
1.6.3 Résultats de simulation . . . 25
1.7 Une nouvelle méthode pour la prévision du trafic . . . 27
1.7.1 Notre méthode proposée . . . 28
1.7.2 Les résultats et les évaluations expérimentales du nouveau modèle . . . 29
1.8 Nouveaux algorithmes d'apprentissage pour les réseaux de neurones aléatoires . . . 33
1.8.1 La méthode de *Levenberg-Marquardt* pour RNN . . . 33

1.8.2 LM avec *adaptive momentum* pour RNN 34
1.8.3 Évaluation des performances des algorithmes proposés 34
1.9 Conclusions générales 39
1.9.1 Sommaire des contributions principales 39
1.9.2 Extensions possibles 41

English Annex: Automatic Evaluation of Real-Time Multimedia Quality: a Neural Network Approach **45**

2 Introduction **45**
2.1 Motivations 45
2.1.1 Multimedia Quality Assessment 45
2.1.2 Impact of the Quality-Affecting Parameters on Multimedia Quality 46
2.1.3 Rate Control Protocols 47
2.1.4 Traffic Prediction 48
2.1.5 Neural Network Learning 48
2.2 The Contributions of this Dissertation 49
2.3 Overview of the Dissertation 50

I Automatic Real-Time Multimedia Quality Evaluation **51**

3 State of the Art **53**
3.1 Objective Speech Quality Measures 53
3.1.1 Bark Spectral Distortion (BSD) 54
3.1.2 Enhanced Modified BSD (EMBSD) 55
3.1.3 Perceptual Speech Quality Measure (PSQM) 55
3.1.4 PSQM+ 55
3.1.5 Measuring Normalizing Blocks (MNB) 56
3.1.6 Perceptual Analysis Measurement System (PAMS) 56
3.1.7 ITU E-model 56
3.2 Objective Video Quality Techniques 57
3.2.1 ITS Video Quality Measure 58
3.2.2 EPFL Objective Video Quality Measures 59
3.2.3 Other Works in Objective Video Quality Measures 61
3.3 Multimedia Transport Protocols 62
3.3.1 Challenges of Multimedia Transmission 62
3.3.2 Working Around Loss 64
3.4 Video Compression and Standard Codecs 65
3.4.1 Spatial or Block Coding 66
3.4.2 Temporal Coding 66
3.4.3 Standard Video Codecs 67
3.5 Audio Compression and Codecs 69
3.5.1 Waveform Coding 69
3.5.2 Source Codecs 70
3.5.3 Hybrid Codecs 70
3.5.4 Standard Audio Codecs 71
3.6 Neural Networks 72
3.6.1 Artificial Neural Networks (ANN) 72
3.6.2 Random Neural Networks (RNN) 74

4 Descriptions of Our New Method in General **77**
4.1 Introduction . . . 77
4.2 Overview of the Method . . . 78
4.2.1 More than one Media Type . . . 79
4.2.2 Selecting the Parameters' Values . . . 79
4.3 Subjective Quality Tests . . . 81
4.3.1 Source Signal for Audiovisual Tests . . . 81
4.3.2 Number of Subjects and their Selection . . . 81
4.3.3 Instructions for Assessors (Subjects) . . . 81
4.3.4 The Test Sessions . . . 82
4.3.5 Subjective Quality Test Methods . . . 82
4.3.6 Comparison of the Methods . . . 84
4.4 MOS Calculation and Statistical Analysis . . . 84
4.5 On the Use of the Neural Networks . . . 86
4.5.1 Comparison between ANN and RNN . . . 87
4.6 Parameter Selection . . . 88
4.7 Expected Use of Our Method . . . 90
4.7.1 Possible Uses For Speech and Audio . . . 90
4.7.2 Possible Uses for Video Applications . . . 90
4.8 Operation Mode . . . 91
4.9 Conclusions . . . 92

II Measuring Speech and Video Quality, and Applications **95**

5 Measuring Speech Quality in Real-Time **97**
5.1 Introduction . . . 97
5.2 Measuring Network Parameters in a Testbed . . . 98
5.2.1 Speech-Quality-Affecting Parameters . . . 99
5.2.2 Other Effects . . . 101
5.2.3 Our Method and Session Types . . . 101
5.3 A Mean Opinion Score (MOS) Database for Different Languages . . . 101
5.4 Assessment of Speech Quality by Neural Networks . . . 104
5.5 Performance of other Speech Quality Measures . . . 109
5.6 Conclusions . . . 111

6 Measuring Video Quality **113**
6.1 Introduction . . . 113
6.2 Simulator Description . . . 115
6.2.1 H263 encoder . . . 115
6.2.2 H263 decoder . . . 116
6.2.3 Network Transport Simulation . . . 116
6.3 The Quality-Affecting Parameters . . . 117
6.4 Subjective Quality Test and MOS Experiment . . . 118
6.5 Results . . . 122
6.5.1 Training the NN . . . 122
6.5.2 How Well does the NN Perform? . . . 123
6.6 Performance of other Video Quality Measures . . . 124
6.7 A Demonstration Application . . . 126
6.8 Conclusions . . . 127

7 Study of Parameters Effects on the Quality 133
7.1 Introduction . . . 133
7.2 Related Works . . . 134
7.3 Parameters Impact on Speech Quality . . . 134
7.3.1 Loss Rate (LR) . . . 135
7.3.2 Consecutive Lost Packets (CLP) . . . 135
7.3.3 Packetization Interval (PI) . . . 135
7.3.4 Speech Codec . . . 136
7.4 Parameters Impact on Video Quality . . . 136
7.4.1 Bit Rate (BR) . . . 136
7.4.2 Frame Rate (FR) . . . 136
7.4.3 Loss Rate (LR) . . . 140
7.4.4 The number of Consecutively Lost Packet (CLP) . . . 140
7.4.5 Intra-to-Inter Ratio (RA) . . . 141
7.5 Conclusions . . . 142

8 A New Rate Control Mechanism 145
8.1 Introduction . . . 145
8.2 Related Works . . . 147
8.2.1 Limitation of RTCP in Rate Control . . . 148
8.2.2 Equation-based TFRC . . . 149
8.3 Our Proposed Rate Control Protocol . . . 149
8.3.1 The Possible Controlling Parameters . . . 150
8.4 Simulation Results . . . 152
8.4.1 Speech Case . . . 152
8.4.2 Video Case . . . 152
8.5 Discussion . . . 154
8.6 Conclusions . . . 155

III On the Neural Networks Tools 157

9 Using Neural Networks for Traffic Prediction: a New Method 159
9.1 Introduction . . . 159
9.2 Our Method . . . 160
9.3 Experimental Results and Evaluation of the New Model . . . 161
9.3.1 Real Traces for Training . . . 161
9.3.2 Training Database Description . . . 161
9.3.3 Experimental Tests to Identify the Best Length of Each Window . . . 162
9.3.4 Performance of the NN to Predict Traffic . . . 163
9.3.5 More Than One Step in the Future . . . 164
9.3.6 Conditions to Retrain the NN . . . 167
9.3.7 Traffic Flow Type . . . 168
9.4 Comparison with Other Models . . . 168
9.5 Possible Uses of our Model . . . 169
9.6 Conclusion and Discussion . . . 170

10 New Random Neural Network Training Algorithms **171**
10.1 Introduction . . . 171
10.2 Gradient Descent Training Algorithm for RNN . . . 171
10.3 New Levenberg-Marquardt Training Algorithms for RNN . . . 173
10.3.1 Analytical Formulation of the Algorithm . . . 174
10.3.2 Different Variants of the LM Training Algorithm . . . 175
10.4 New LM with Adaptive Momentum for RNN . . . 176
10.4.1 The Algorithm with Adaptive Momentum . . . 176
10.5 Performance Evaluation of the Proposed Algorithms . . . 178
10.5.1 ζ and dP Parameters for AM-LM . . . 179
10.5.2 Algorithms' Performance Comparison . . . 180
10.5.3 Testing the Success Rate and the Performance . . . 181
10.5.4 Learning Performance in the Video Quality Problem . . . 184
10.6 Conclusions . . . 184

General Conclusions of this Dissertation **187**
Summary of the Main Contributions . . . 187
Possible Extensions . . . 188

Table des figures

1 Le problème de surentraînement en utilisant ANN. 10
2 La performance d'un ANN et d'un RNN à interpoler et à extrapoler pour des nombres différents des neurones cachés. 11
3 La corrélation entre les valeurs réelles et prévues de MOS (langage arabe). 13
4 La corrélation entre les valeurs réelles et prévues de MOS (langage espagnol). 14
5 La corrélation entre les valeurs réelles et prévues de MOS (langage français). 14
6 La corrélation entre les valeurs réelles et prévues de MOS pour les deux BD. 18
7 L'architecture du mécanisme proposé de contrôle. 24
8 Les débits suggérés par TFRC et l'économie obtenue en utilisant des règles de contrôle en changeant le codec dans le cas de la parole. 26
9 Les valeurs de MOS avec et sans notre contrôleur en changeant le codec dans le cas de la parole. 26
10 Les débits suggérés par TFRC et ceux de l'expéditeur 27
11 Les valeurs de MOS en changeant FR et ceux en changeant QP. 28
12 Une représentation de boîte noire de notre outil pour prévoir en temps réel le futur trafic 29
13 Le trafic réel contre celui prévu pour la totalité de deux semaines suivantes. 30
14 Le trafic réel contre celui prévu pour les troisième et quatrièmes jours de la troisième semaine. 31
15 La différence entre le trafic réel et prévu pour les suivantes deux semaines complètes. 31
16 Un histogramme de la distribution entre la différence de trafic réel et prévu 32
17 La différence entre le trafic réel et prévu pour les suivantes deux semaines complètes une fois que les échantillons spiky sont retirés. 32
18 L'effet de variables ζ et dP sur la performance de l'algorithme AM-LM pour un RNN. Les résultats sont pour le premier problème. 35
19 Les performances des algorithmes GD, LM, LM2 et AM-AM d'apprentissage du premier problème. 36
20 Les performances des algorithmes GD, LM, LM1, et LM2 d'apprentissage du deuxième problème. 37
21 Comparaison entre les performances de GD et d'AM-LM lors de l'apprentissage du problème de la qualité de la vidéo présenté en section 1.4. 39

3.1 Architecture of a three-layer feedforward neural network. 73

4.1 The overall architecture of the new method to evaluate real-time speech, audio and/or video quality in real time. 80
4.2 A schematic diagram of our method showing the steps in the design phase. 80
4.3 Eleven-point quality scale. 83
4.4 ITU 5-point impairment scale 83
4.5 Stimulus presentation timing in ACR method. 83
4.6 Stimulus presentation timing in DCR method. 84
4.7 A portion of quality rating form using continuous scales. 85

4.8 A portion of quality rating form using continuous scales for DSCQS method. 85
4.9 The problem of overtraining using ANN 89
4.10 Performance of ANN and RNN to interpolate and extrapolate for different number of hidden neurons 89
4.11 The run-time mode of our method. 91
4.12 The current existing model of objective methods. 92
4.13 Operation mode for the tool in real-time video system. 92

5.1 Maximum and minimum percentage loss rates as a function of the playback buffer length between Rennes and the other different sites. 100
5.2 Minimum rates for one-to-ten consecutively lost packets as a function of the playback buffer length between Rennes and the other different sites. 100
5.3 Maximum rates for one-to-ten consecutively lost packets as a function of the playback buffer length between Rennes and the other different sites. 101
5.4 Actual vs. Predicted MOS values on the Arabic training database. 105
5.5 Actual vs. Predicted MOS values on the Spanish training database. 106
5.6 Actual vs. Predicted MOS values on the French training database. 106
5.7 Actual vs. Predicted MOS values on the testing databases. 107
5.8 Scatter plots to show the correlation between Actual and Predicted MOS values (Arabic Language). 108
5.9 Scatter plots to show the correlation between Actual and Predicted MOS values (Spanish Language). 108
5.10 Scatter plots to show the correlation between Actual and Predicted MOS values (French Language). 109
5.11 MNB2 and E-model results against the MOS subjective values in evaluating a set of speech samples distorted by both encoding and network impairments. Source is [57]. . 111

6.1 A screen dump showing an instance during the subjective quality test evaluation. . . . 119
6.2 The 95% confidence intervals before and after removing the rate of two unreliable subjects. 122
6.3 Actual and Predicted MOS scores for the training database. 123
6.4 Actual and Predicted MOS scores for the testing database. 124
6.5 Scatter plots showing the correlation between Actual and Predicted MOS scores. . . . 124
6.6 The performance of the ITS metric to evaluate video quality. 125
6.7 Quality assessment by MPQM as a function of the bitrate and the loss rate. 125
6.8 MPQM quality assessment of MPEG-2 video as a function of the bit rate. 126
6.9 The quality assessment by the ITS model of MPEG-2 video as a function of the bit rate. 126
6.10 CMPQM quality assessment of MPEG-2 video as a function of the bit rate. 126
6.11 NVFM quality assessment of MPEG-2 video as a function of the bit rate. 126
6.12 Comparison of the subjective data against MPQM, CMPQM and NVFM metrics for the video sequence “Mobile & Calendar”. 127
6.13 Comparison of the subjective data against MPQM, CMPQM, NVFM and ITS metrics for the video sequence “Basket Ball”. 127
6.14 A screen dump showing manual mode for Stefan 128
6.15 A screen dump showing manual mode for Children 129
6.16 A screen dump showing manual mode for Foreman 130
6.17 A screen dump showing Automatic Mode for Stefan 131

7.1 On the left, we show the impact of LR and CLP on speech quality for the different codecs and PI=20 ms. On the right we show the effect of LR and PI on speech quality for CLP=1. 137

7.2 The impact of CLP and LR on speech quality when LR=5 % (left) and when LR=10 %(right) for PCM, ADPCM and GSM codecs. 138
7.3 The variations of the quality as a function of the LR and the employed speech codec in both languages for PI=20 ms and CLP=2. 139
7.4 The impact of BR and FR on video quality. 139
7.5 The impact of BR and LR on video quality. 139
7.6 The impact of BR and CLP on video quality. 140
7.7 The impact of BR and RA on video quality. 140
7.8 The impact of FR and LR on video quality. 141
7.9 The impact of FR and CLP on video quality. 141
7.10 The impact of FR and RA on video quality. 142
7.11 The impact of LR and CLP on video quality. 142
7.12 The impact of LR and RA on video quality. 142
7.13 The impact of CLP and RA on video quality. 142
7.14 RA is more benefic than FR for lower values of BR. 143

8.1 Architecture of the proposed control mechanism. 151
8.2 Rates suggested by TCP-friendly and the saving using control rules when changing the codec in the case of Speech 153
8.3 MOS values with and without our control when changing the codec in the case of Speech (CM stands for Control Mechanism) 153
8.4 The supposed rates suggested by TCP-friendly and that of the sender 154
8.5 MOS values when changing frame rate and those when changing the quantization parameter to meet the bit rates shown in Figure 8.4 155

9.1 A black-box representation of our tool to predict in real time the future traffic. 161
9.2 Our best architecture employing both short- and long-range dependencies in traffic prediction for the ENSTB Network. 163
9.3 The actual traffic against the predicted one for the whole complete next two weeks. . . 164
9.4 The Normalized actual against that predicted for the third and fourth days from the third week. 165
9.5 The difference between the actual and the predicted traffic for the complete next two weeks. 165
9.6 The histogram of the distribution of difference between actual and prediction with a step of 0.1. 166
9.7 The difference between the actual and the predicted traffic for the complete next two weeks once the spiky samples are removed. 166
9.8 Predicting 2nd step ahead: the difference between the actual traffic and the predicted one for the complete next two weeks, including the spikes. 167
9.9 The traditional NN model that has been widely used to predict network traffic. This Figure is taken from [56, p. 115], where z^{-1} represents a unit-step delay function. . . . 168
9.10 The actual against the NN prediction when training and testing it by data generated by Eqn. 9.1. This Figure is taken from [56]. 169

10.1 The 7-5-2 feedforward RNN network architecture. 179
10.2 The fully-connected recurrent RNN network architecture. 179
10.3 The RNN network architecture used to solve the XOR problem. 180
10.4 The impact of the two variables ζ and dP on the performance of the adaptive momentum LM training algorithm for RNN. The results for the first problem. 180
10.5 The performance of the GD, LM, LM2 and AM-LM training algorithms on the first problem. 181

10.6 The performance of the GD, LM, LM1, and LM2 training algorithms on the second problem. 182
10.7 Comparsion between the performance of GD and that of AM-LM on the video quality database presented in Chapter 6. 184

Liste des tableaux

1 La performance de RN pour apprendre le problème et sa réaction aux exemples non-vus. 13
2 Le coefficient de corrélation de quelques mesures objectives existantes de la qualité de la parole avec MOS. 15
3 Le coefficient de corrélation de quelques mesures objectives existantes de la qualité de la parole avec MOS en considérant les paramètres de codage et de réseau. 16
4 Les coefficients de corrélation d'EMBSD, de MNB et d'E-model avec MOS en considérant les paramètres de réseau. 16
5 Comparaison entre GD, LM, LM2 et AM-LM pour le premier problème 38
6 Comparaison entre GD, LM, LM1 et LM2 pour le deuxièmes problème. 38

3.1 Standard video resolution formats. 67

4.1 ITU 5-point quality scale 82
4.2 ITU 9-point quality scale 82

5.1 Number of hops and one-way delay statistics between the peers (Site) and Rennes (ENST-B) one. 99
5.2 The quality databases for both Arabic and Spanish languages to train and test the NN. 102
5.3 Performance of the NN to learn the problem and react to non-seen examples. 105
5.4 Correlation coefficient of some existing objective speech quality measures with MOS. Results are taken from the literature. Only the encoding impairments are considered. 110
5.5 Correlation coefficient of some existing objective speech quality measures with MOS for both network and encoding impairments. 110
5.6 Correlation coefficient of EMBSD, MNB and E-model with MOS for VoIP impairments. Source is [57]. 111

6.1 A portion of the scoring sheet distributed to each subject. 119
6.2 The quality database to train and test the NN (before being normalized) 120

9.1 Some entries from the ENSTB trace. 162
9.2 Some entries from the training database after being normalized between 0 ...1. 162

10.1 Comparison between GD, LM, LM2 and AM-LM for the first problem. 183
10.2 Comparison between GD, LM, LM1 and LM2 for the second problem. 183

Publications

This is a list of the publications on the research topics associated with this thesis.

- **Journal Articles**

1. S. Mohamed and G. Rubino. "A study of real-time packet video quality using random neural networks", In *IEEE Transactions On Circuits and Systems for Video Technology*, vol. 12, no. 12, December 2002.
2. S. Mohamed, F. Cervantes and H. Afifi. "Real-Time Audio Quality Assessment in Packet Networks", In *Network and Information Systems Journal*, vol. 3, no. 3-4, 2000, pp. 595-609.

- **Conference Papers**

1. S. Mohamed, G. Rubino, and M. Varela. "The impact of encoding and network impairments on real-time speech quality", *Submitted to IEEE Infocom*, 2003.
2. S. Mohamed and G. Rubino. "Evaluation of Packet Video Quality in Real-time using RNN", Random Neural Networks workshop, 2002, Madrid, Spain.
3. S. Mohamed, G. Rubino and F. Cervantes, H. Afifi. "Real-Time Video Quality Assessment in Packet Networks: A Neural Network Model", In *Proceedings of International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'2001)*, June, 2001, Las Vegas, Nevada, USA.
4. S. Mohamed, F. Cervantes and H. Afifi. "Integrating Networks Measurements and Speech Quality Subjective Scores for Control Purposes", In *Proceedings of the IEEE INFOCOM'01*, April 22 - 26, 2001, Alaska, USA.
5. S. Mohamed, F. Cervantes and H. Afifi. "Audio Quality Assessment in Packet Networks: an Inter-Subjective Neural Network Model", In *Proceedings of the 15th International Conference on Information Networking (ICOIN-15)*, Japan, 2001
6. S. Mohamed, F. Cervantes and H. Afifi. "Une méthode inter-subjective pour l'évaluation de la qualité sonore de voix sur IP", In *4ème édition des Journées Doctorales Informatique et Réseaux (JDIR)*, Paris, France, 6 - 8 novembre 2000 (in French)

- **Technical Reports**

1. S. Mohamed and G. Rubino. "A study of real-time packet video quality using random neural networks", INRIA, research report no. RR-4525, Aug. 2002.
2. S. Mohamed, G. Rubino, F. Cervantes and H. Afifi. 'Real-Time Video Quality Assessment in Packet Networks: A Neural Network Model´, INRIA, research report no. 4186, May 2001. (Long version of PDPTA'2001)

3. S. Mohamed, F. Cervantes and H. Afifi. "Audio Quality Assessment in Packet Switched Networks", IRISA, technical report no. 1325, April 2000

Acronyms

ACR: Absolute Category Rating.
ADPCM: Adaptive Differential PCM.
AM-LM: LM with Adaptive Momentum.
ANN: Artificial Neural Networks.
ARQ: Automatic Repeat reQuest.
ATM: Asynchronous Transfer Mode.
BR: Bit Rate.
BSD: Bark Spectral Distortion.
CIF: Common Interchange Format.
CLP: Consecutive Lost Packets.
CMPQM: Color MPQM.
Codec: COder and DECoder .
DCR: Degradation Category Rating.
DCT: Discrete Cosine Transform.
DSCQS: Double-Stimulus Continuous Quality-Scale.
EB-TFRC: Equation-based TFRC.
EBSD: Enhanced Modified BSD.
FEC: Forward Error Control.
FR: Frame Rate.
GD: Gradient Descent.
GOP: Group Of Pictures.
GSM: Global System for Mobile Communication.
HVS: Human Visual System.
IP: Internet Protocol.
ISDN: Integrated Services Digital Network.
ITU: International Telecommunication Union.
LM: Levenberg-Marquardt.
LR: Loss Rate.
MB: Macro Bloc.
MNB: Measuring Normalizing Blocks.
MOS: Mean Opinion Score.
MPEG: Moving Picture Expert Group.
MPQM: Moving Picture Quality Metric.
MSE: Mean Square Error.
NN: Neural Networks.
NVFM: Normalization Video Fidelity Metric.

PAMS: Perceptual Analysis Measurement System.
PCM: Pulse Code Modulation.
PI: Packetization Interval.
PSQM: Perceptual Speech Quality Measure.
QP: Quantization Parameter.
QoS: Quality of Service.
RA: Ratio of the Encoded Intra Macro-Blocs to Inter Macro-Blocs.
RLC: Run Length Coding.
RNN: Random Neural Networks.
RTCP: Real Time Control Protocol.
RTP: Real time Transport Protocol.
RTT: Round-Trip Time.
SN: Sequence Number.
SNR: Signal-to-Noise Ratio.
SNRseg: Segmental SNR.
TCP: Transport Control Protocol.
TFRC: TCP-Friendly Rate Control.
TS: Time Stamp.
UDP: User Datagram Protocol.
VoD: Video on Demand.

Glossary [1]

Artificial Neural Networks (ANN) – Also referred to as connectionist architectures, parallel distributed processing, and neuromorphic systems, an artificial neural network (ANN) is an information-processing paradigm inspired by the way the densely interconnected, parallel structure of the mammalian brain processes information. Artificial neural networks are collections of mathematical models that emulate some of the observed properties of biological nervous systems and draw on the analogies of adaptive biological learning. The key element of the ANN paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements that are analogous to neurons and are tied together with weighted connections that are analogous to synapses.

Asynchronous Transfer Mode (ATM) – A network technology for both local and wide area networks (LANs and WANs) that supports realtime voice and video as well as data. The topology uses switches that establish a logical circuit from end to end, which guarantees quality of service (QoS). However, unlike telephone switches that dedicate circuits end to end, unused bandwidth in ATM's logical circuits can be appropriated when needed. For example, idle bandwidth in a videoconference circuit can be used to transfer data. ATM is widely used as a backbone technology in carrier networks and large enterprises, but never became popular as a local network (LAN) topology. ATM is highly scalable and supports transmission speeds of 1.5, 25, 100, 155, 622, 2488 and 9953 Mbps. ATM is also running as slow as 9.6 Kbps between ships at sea. An ATM switch can be added into the middle of a switch fabric to enhance total capacity, and the new switch is automatically updated using ATM's PNNI routing protocol.

Audio Codec – A hardware circuit (chip) or software routine that converts sound into digital code and vice versa. The first step is to convert the analog sound into digital samples, using PCM or ADPCM. The next step is to use perceptual audio coding to further compress the amount of digital data. If the codec is specialized for human voice, it is also known as a "speech codec," "voice codec" or "vocoder."

Audio Compression – Encoding digital audio data to take up less storage space and transmission bandwidth. Audio compression typically uses lossy methods, which eliminate bits that are not restored at the other end. ADPCM and MP3 are examples of audio compression methods.

Automatic Repeat reQuest (ARQ) – A method of handling communications errors in which the receiving station requests retransmission if an error occurs.

Bit Rate (BR) – The transmission speed of binary coded data.

COder-DECoder (Codec) – Hardware or software that converts analog sound, speech or video to digital code and vice versa (analog to digital– digital to analog). Codecs must faithfully reproduce the original signal, but they must also compress the binary code to the smallest number of bits possible in order to transmit faster. As network bandwidth increases, so does the demand for more audio and video, so compression is always an issue. Codecs can be software or hardware. Software codecs are installed into audio and video editing programs as well as media players that download audio and video over the Web. Software codecs rely entirely on the PC for processing.

1. Source is `http://www.techweb.com/encyclopedia/` except the definitions of ANN, RNN, MOS and QoS.

Hardware codecs are specialized chips built into digital telephones and videoconferencing stations to maximize performance. Although hardware codecs are faster than software routines, faster desktop machines are increasingly enabling software codecs to perform quite adequately.

Forward Error Correction (FEC) – A communications technique that can correct bad data on the receiving end. Before transmission, the data are processed through an algorithm that adds extra bits for error correction. If the transmitted message is received in error, the correction bits are used to repair it.

IP Telephony – The two-way transmission of audio over a packet-switched IP network (TCP/IP network). When used in a private intranet or WAN, it is generally known as "voice over IP," or "VoIP." When the transport is the public Internet or the Internet backbone from a major carrier, it is generally called "IP telephony" or "Internet telephony." However, the terms IP telephony, Internet telephony and VoIP are used interchangeably. IP telephony uses two protocols: one for transport and another for signaling. Transport is provided by UDP over IP for voice packets and either UDP or TCP over IP for signals. Signaling commands to establish and terminate the call as well as provide all special features such as call forwarding, call waiting and conference calling are defined in a signaling protocol such as H.323, SIP, MGCP or MEGACO

International Telecommunication Union (ITU) – Formerly the CCITT (Consultative Committee for International Telephony and Telegraphy), it is an international organization founded in 1865, now part of the United Nations System, that sets communications standards for global telecom networks. The ITU is comprised of more than 185 member countries. The Union began the 21st century streamlined into three sectors: Telecommunication Standardization (ITU-T), Radiocommunication (ITU-R) and Telecommunication Development (ITU-D). The oldest of these is the ITU-T, which produces more than 200 standards recommendations each year in the converging areas of telecommunications, information technology, consumer electronics, broadcasting and multimedia communications.

Internet Protocol (IP) – The protocol that is used to route Internet traffic. IP is an unreliable protocol; higher layer protocols such as TCP insure that IP successfully delivers all data to the intended recipient. The Internet protocol defines how information gets passed between systems across the Internet.

Jitter – A flicker or fluctuation in a transmission signal or display image. The term is used in several ways, but it always refers to some offset of time and space from the norm. For example, in a network transmission, jitter would be a bit arriving either ahead or behind a standard clock cycle or, more generally, the variable arrival of packets. In computer graphics, to "jitter a pixel" means to place it off side of its normal placement by some random amount in order to achieve a more natural antialiasing effect.

Mean Opinion Score (MOS) – The quality of a digitized voice, audio, video or multimedia signal. It is a subjective measurement that is derived entirely by people scoring the results from 1 to 5, with a 5 meaning that the quality is perfect (other quality scales are also available). The MOS is an average of the numbers for a particular codec. There are several recommendations provided by the ITU concerning MOS and the diffrent ways to carry out subjective quality tests.

Multimedia – Information in more than one form. It includes the use of text, audio, graphics, animated graphics and full-motion video.

Neural Network (NN) – A modeling technique based on the observed behavior of biological neurons and used to mimic the performance of a system. It consists of a set of elements that start out connected in a random pattern, and, based upon operational feedback, are molded into the pattern required to generate the required results. It is used in applications such as robotics, diagnosing, forecasting, image processing and pattern recognition.

Packet Loss – The discarding of data packets in a network when a device (switch, router, etc.) is overloaded and cannot accept any incoming data at a given moment. High-level transport

protocols such as TCP/IP ensure that all the data sent in a transmission is received properly at the other end.

Packet Switching – A networking technology that breaks up a message into smaller packets for transmission and switches them to their required destination. Unlike circuit switching, which requires a constant point-to-point circuit to be established, each packet in a packet switched network contains a destination address. Thus all packets in a single message do not have to travel the same path. They can be dynamically routed over the network as lines become available or unavailable. The destination computer reassembles the packets back into their proper sequence. Packet switching efficiently handles messages of different lengths and priorities. By accounting for packets sent, a public network can charge customers for only the data they transmit. Packet switching has been widely used for data, but not for realtime voice and video. However, this is beginning to change. IP and ATM technologies are expected to enable packet switching to be used for everything.

Quality of Service (QoS) – refers to the capability of a network to provide better service to selected network traffic over various technologies, including Frame Relay, Asynchronous Transfer Mode (ATM), Ethernet and 802.1 networks, SONET, and IP-routed networks that may use any or all of these underlying technologies. The primary goal of QoS is to provide priority including dedicated bandwidth, controlled jitter and latency (required by some real-time and interactive traffic), and improved loss characteristics. Also important is making sure that providing priority for one or more flows does not make other flows fail. QoS technologies provide the elemental building blocks that will be used for future business applications in campus, WAN, and service provider networks.

Random Neural Network (RNN) – The RNN model is introduced by E. Gelenbe in 1989, and 1990. In the RNN model signals travel as voltage spikes. This model represents more closely the manner in which signals are transmitted in a biophysical neural network than widely used artificial neuron models in which signals are represented by fixed signal levels. Signals in the form of spikes of unit amplitude circulate among the neurons. Positive signals represent excitation and negative signals represent inhibition. Each neuron's state is a non-negative integer called its potential, which increases when an excitation signal arrives to it, and decreases when an inhibition signal arrives. Thus, an excitatory spike is interpreted as a "+1" signal at a receiving neuron, while an inhibitory spike is interpreted as a "-1" signal. Neural potential also decreases when the neuron fires. Firing occurs at random according to an exponential distribution of a constant rate and signals are sent out to other neurons or to the outside of the network. A backpropagation type learning algorithm for recurrent RNN model is introduced by Gelenbe in 1993.

Realtime Transport Protocol (RTP) – An IP protocol that supports realtime transmission of voice and video. It is widely used for IP telephony. An RTP packet rides on top of UDP, the non-reliable counterpart of TCP, and includes timestamping and synchronization information in its header for proper reassembly at the receiving end. Realtime Control Protocol (RTCP) is a companion protocol that is used to maintain QoS. RTP nodes analyzes network conditions and periodically send each other RTCP packets that report on network congestion.

Speech Codec – Also called a "voice codec" or "vocoder," it is a hardware circuit (chip) or software rouine that converts the spoken word into digital code and vice versa. A speech codec is an audio codec specialized for human voice. By analyzing vocal tract sounds, a recipe for rebuilding the sound at the other end is sent rather than the soundwaves themselves. As a result, the speech codec is able to achieve a much higher compression ratio which yields a smaller amount of digital data for transmission. However, if music is encoded with a speech codec, it will not sound as good when decoded at the other end.

Streaming Audio – Audio transmission over a data network. The term implies a one-way trans-

mission to the listener, in which both the client and server cooperate for uninterrupted sound. The client side buffers a few seconds of audio data before it starts sending it to the speakers, which compensates for momentary delays in packet delivery. Because of the buffering, streaming audio can be delivered over a slow network. Audio conferencing, on the other hand, requires realtime, two-way transmission, which means the bandwidth must support the speed of both incoming and outgoing audio streams without buffering any of it. Streaming audio differs from downloading an audio file that can be played later, because the latter remains in the computer. The streaming audio is stored as a temporary file that is deleted when the media player is closed.

Streaming Video – Video transmission over a data network. It is widely used on the Web to deliver video on demand or a video broadcast at a set time. In streaming video, both the client and server software cooperate for uninterrupted motion. The client side buffers a few seconds of video data before it starts sending it to the screen, which compensates for momentary delays in packet delivery. Because of the buffering, streaming video can be delivered over a slower network. Videoconferencing, on the other hand, requires realtime, two-way transmission, which means the bandwidth must support the speed of both incoming and outgoing video streams without buffering any of it. Streaming video differs from downloading a video file that can be played later, because the latter remains in the computer. The streaming video is stored as a temporary file that is deleted when the media player is closed.

Transmission Control Protocol/Internet Protocol (TCP/IP) – It is the protocol of the Internet and has become the global standard for communications. TCP provides transport functions, which ensures that the total amount of bytes sent is received correctly at the other end. UDP, which is part of the TCP/IP suite, is an alternate transport that does not guarantee delivery. It is widely used for realtime voice and video transmissions where erroneous packets are not retransmitted. TCP/IP is a routable protocol, and the IP part of TCP/IP provides this capability. In a routable protocol, all messages contain not only the address of the destination station, but the address of a destination network. This allows TCP/IP messages to be sent to multiple networks (subnets) within an organization or around the world, hence its use in the worldwide Internet. IP accepts packets from TCP or UDP, adds its own header and delivers a "datagram" to the data link layer protocol. It may also break the packet into fragments to support the maximum transmission unit (MTU) of the network. Every client and server in a TCP/IP network requires an IP address, which is either permanently assigned or dynamically assigned at startup.

User Datagram Protocol (UDP) – A protocol within the TCP/IP protocol suite that is used in place of TCP when a reliable delivery is not required. For example, UDP is used for realtime audio and video traffic where lost packets are simply ignored, because there is no time to retransmit. If UDP is used and a reliable delivery is required, packet sequence checking and error notification must be written into the applications.

Video Compression – Encoding digital video to take up less storage space and transmission bandwidth.

Video on Demand (VoD) – The ability to start delivering a movie or other video program to an individual Web browser or TV set whenever the user requests it.

Videoconferencing – A video communications session among three or more people who are geographically separated. This form of conferencing started with room systems where groups of people meet in a room with a wide-angle camera and large monitors and conference with other groups at remote locations. Federal, state and local governments are making major investments in group videoconferencing for distance learning and telemedicine.

Évaluation automatique de la qualité des flux multimédias en temps réel : une approche par réseaux de neurones

1.1 Introduction

Au cours des dernières décennies, les réseaux de paquet (par exemple, l'Internet) ont connu une énorme avancée technologique. De très nombreuses applications ont ainsi pu voir le jour et n'auraient pas été possibles sans ces grands et rapides progrès technologiques. En outre, il y a eu des avancées dans les méthodes et les algorithmes de codage numérique (par exemple, les techniques de compression) pour les signaux de la parole, de l'audio et de la vidéo (le multimédia). La compression est une des caractéristiques importantes de la technologie des télécommunications. Ces facteurs ont permis la transmission à bas prix de flux multimédias dans ces réseaux. L'intégration efficace de la voix, de la vidéo, et des services de données dans ces réseaux est également possible de nos jours. Parmi les applications possibles, nous pouvons trouver la vidéoconférence, la téléphonie et la vidéo sur IP, la télémédecine, la téléprésence, la vidéo sur demande, la voix sur IP, la radiodiffusion sur l'Internet, le stockage de donnée multimédia, etc. Ceci donne cependant lieu à quelques problèmes importants qui doivent être abordés.

Ce chapitre est organisé comme suit. Dans la section 1.1.1, nous fournissons les motivations du travail présenté dans cette thèse. Les principales contributions sont récapitulées dans la section 1.1.2. Dans la section 1.1.3, nous donnons une brève vue d'ensemble et l'organisation de ce chapitre. (Nous vous rappelons que l'annexe anglaise est plus complète, et donc, nous vous conseillons vivement de la consulter.)

1.1.1 Les motivations

1.1.1.1 Évaluation de la qualité de multimédia

Un des premiers problèmes à considérer est l'évaluation automatique de la qualité des flux de multimédia transmis en temps réel sur le réseau. L'évaluation de la qualité de la parole dans des applications de la téléphonie sur IP, l'évaluation de la qualité de la vidéo dans des applications de téléconférence ou de diffusion de vidéo sont quelques exemples. C'est un problème puisque, comme nous le verrons en détail dans les sections 1.3.4 et 1.4.3, aucune solution proposée jusqu'alors n'est satisfaisante.

Traditionnellement, la qualité est mesurée en fonction des algorithmes de codage, sans tenir compte de l'effet des paramètres du réseau. Il y a deux approches pour mesurer la qualité : par des méthodes objectives ou des méthodes subjectives. Les méthodes subjectives [72, 70, 59] mesurent la qualité globale perçue ; elles sont effectuées par des sujets humains. La mesure la plus généralement utilisée est la note moyenne d'opinion, ou *Mean Openion Score* (MOS), recommandée par l'union internationale de télécommunication (UIT) [70]. Un groupe de sujets regarde et/ou écoute les échantillons ou les

séquences afin d'évaluer leur qualité selon une échelle de notes prédéfinies de la qualité. D'autre part, les méthodes objectives [154] mesurent habituellement la qualité en comparant les séquences ou les échantillons originaux et aux résultant du traitement par les codeurs. Quelques méthodes usuelles pour la vidéo sont le *Mean Square Error* (MSE) ou le *Peak Signal to Noise Ratio* (PSNR), qui mesurent la qualité en faisant une différence simple entre les images. Il y a d'autres mesures plus complexes comme *Moving Picture Quality Metric* (MPQM), et *Normalized Video Fidelity Metric* (NVFM) [153]. Quelques exemples pour l'évaluation objective de la qualité de la parole sont : le rapport signal à bruit (SNR), le *Segmental* SNR et le *Perceptual Speech Quality Measures* (PSQM) [38, 59, 105]. Il faut observer que pour vérifier l'exactitude de ces mesures (également appelées " essais " dans ce domaine de travail), elles doivent habituellement être corrélées avec des résultats obtenus en utilisant des évaluations subjectives de la qualité.

Limitations des méthodes subjectives et objectives

Bien que les études de MOS aient servi de base à l'analyse de nombreux aspects du traitement des signaux, les méthodes subjectives présentent plusieurs limitations : (a) des environnements rigoureux sont exigés ; (b) le processus ne peut pas être automatisé ; (c) elles sont très coûteuses et exigent beaucoup de temps pour les effectuer ; (d) c'est impossible de les employer dans l'évaluation en temps réel de la qualité. D'autre part, les inconvénients des méthodes objectives existantes sont : (a) une mauvaise corrélation avec la perception humaine ; (b) un besoin élevé de puissance de calcul ; (c) elles ne peuvent pas être employées pour l'évaluation en temps réel de qualité, car elles travaillent à la fois sur les signaux originaux et ceux à évaluer ; (d) la difficulté d'établir un modèle tentant compte de l'effet de multiples paramètres influant sur la qualité simultanément, particulièrement les paramètres de réseau.

Les critères que nous recherchons pour une bonne mesure objective de la qualité sont : (a) la possibilité d'évaluation en temps réel, (b) des calculs rapides, (c) aucun besoin d'accéder au signal original, (d) tous les paramètres influant la qualité, comprenant des facteurs de réseau, peuvent être considérés (c'est-à-dire l'essai ne doit pas être limité à un type spécifique de paramètres) et (e) une bonne corrélation des résultats avec la perception humaine. Il est important de mentionner que, à ce jour, il n'y a aucune mesure objective pouvant répondre à ces conditions. Ainsi, l'un de nos objectifs est de proposer une nouvelle mesure objective qui peut répondre à toutes les exigences indiquées.

1.1.1.2 Impact des paramètres influant la qualité de multimédia

Un deuxième problème important est l'évaluation de l'impact d'un ensemble de paramètres sur la qualité. Le rapport entre la qualité et les différents paramètres possibles est complexe et difficile à appréhender. Il n'y a aucune étude complète dans la littérature montrant l'impact de plusieurs de ces paramètres sur la qualité, principalement parce que les essais subjectifs de la qualité sont chers à effectuer. En effet, pour analyser l'effet combiné, par exemple, de trois ou quatre paramètres, nous devons établir un ensemble très grand d'évaluations humaines afin d'atteindre un niveau minimal de précision.

Les études précédentes sont concentrées sur l'effet des paramètres de réseau sans prêter attention aux paramètres de codage, ou l'inverse. Les articles qui considèrent des paramètres de réseau en utilisant des essais subjectifs pour l'évaluation limitent l'étude seulement à un ou deux des plus importants. Par exemple, dans le cas de la parole, le taux de perte et l'intervalle de mise en paquet sont étudiés dans [61], tandis que [28] étudie principalement l'effet du taux de perte sur plusieurs codecs. Quant à la vidéo, le taux de perte et la taille de rafale de perte sont étudiés dans [58], alors que [153] étudie principalement l'effet du débit et [53] travaille seulement sur l'effet du débit d'images. En ce qui concerne les approches objectives, il n'y a aucune méthode publiée précédemment tenant compte de l'influence directe de tous les paramètres influant sur la qualité simultanément.

Aux critères décrits précédemment à satisfaire par une bonne mesure de la qualité, il faut ajouter qu'elle devrait permettre non seulement d'intégrer beaucoup de paramètres (de la source et aussi du réseau) mais également de travailler avec des plages de valeurs de ces paramètres. Un outil capable de mesurer la qualité en satisfaisant toutes ces propriété devrait aider :

- à développer des mécanismes de contrôle afin de fournir la meilleure qualité étant donnée la situation courante du réseau (cf. la prochaine sous-section),
- à une meilleure compréhension des aspects de QoS dans l'ingénierie du multimédia.

Notre objectif dans ce domaine est de fournir des études des effets combinés de plusieurs paramètres influant sur la qualité de la parole et la qualité de la vidéo. Nous visons également à fournir les orientations qui peuvent être suivies pour réaliser des études semblables dans d'autres types de technologies de réseau et, si nécessaire, pour d'autres paramètres.

1.1.1.3 Protocoles de contrôle de débit

Un troisième problème important dans la l'ingénierie de trafic est le contrôle de débit. Traditionnellement, les protocoles de contrôle de débit sont conçus sur des mesures passives de réseau (la perte, le délai, la bande passante, etc.) [26, 41, 44, 77, 107, 108, 124, 126]. Cependant, la réaction directe des sujets humains, en tant qu'utilisateurs finaux des applications de multimédia temps réel, est souvent négligée dans la conception de ces protocoles, parce qu'il a été difficile de mesurer en temps réel leur perception d'une façon permettant de tenir compte de l'effet direct des paramètres du réseau et des codecs. Les solutions aux problèmes précédemment mentionnés (notamment l'évaluation en temps réel de la qualité en temps réel et la compréhension de l'impact des paramètres sur la perception de l'utilisateur), que nous présentons dans cette thèse, peuvent permettre la conception de nouveaux types de protocoles. En plus des mesures passives de réseaux utilisées dans les protocoles traditionnels, ces nouveaux protocoles pourraient prendre en compte l'influence directe de la perception des utilisateurs en temps réel. Comme illustrations, nous proposons dans cette thèse à proposer un nouveau protocole de contrôle de débit qui intègre les mesures de réseau et la perception des utilisateurs. Parmi les avantages d'un tel protocole, mentionnons les suivantes :

- Nous pouvons maximiser ou garantir la livraison de la meilleure qualité de multimédia étant donné l'état de réseau.
- On peut faciliter la réduction de la congestion dans le réseau.
- Il permet de maintenir les flux dans un état dit *TCP-friendly*, et par conséquent ne contribue pas à congestionner le réseau.

1.1.1.4 Prévision du trafic

Un quatrième problème important est celui de la *prévision du trafic*. Ce problème est d'importance parce que la prévision du trafic est cruciale pour le contrôle de congestion, voire pour éviter la congestion, l'attribution dynamique de la bande passante, la négociation dynamique de contrat, l'ingénierie de trafic, etc. Des mécanismes plus précis pour éviter la congestion peuvent être mis en application en considérant à la fois la prévision du trafic et le contrôle de débit présentés précédemment. Nous avons également exploré cette problématique pour les raisons suivantes :

- à cause de sa relation directe avec les techniques de contrôle,
- parce que les réseaux de neurones (voir après) fonctionnent également bien pour ce type de tâche.

Cependant, il y a quelques difficultés qui ont empêché de réaliser de bons outils de prévision du trafic qui peuvent travailler dans des conditions réelles, principalement liées au fait que le trafic du réseau est, en général, un processus complexe, variable avec le temps, et non stationnaire. En outre, ce processus est régi par des paramètres et des variables qui sont très difficiles à mesurer. Parfois,

il y a des parties complètement non prévisibles de ce processus (les fragments dites "spikes", voir la section 1.7.2.3). Par conséquent, un modèle précis de ce processus devient difficile à établir.

En dépit de ces problèmes, la prévision du trafic est possible parce que des mesures ont montré que des dépendances à longue portée et à courte portée coexistent dans les réseaux. Par exemple, la quantité du trafic diffère le week-end de celle des jours de la semaine. Cependant, elle est souvent statistiquement semblable pour tous les week-ends, également pendant le même jour, les matins, les nuits, à quelques parties spécifiques du jour, etc.

Il y a beaucoup de propositions dans la littérature pour la prévision du trafic [85, 27, 121, 120, 156, 39]. Ces propositions se concentrent sur les dépendances à courte portée, négligeant d'une façon ou d'autre celles à longue portée. En examinant ces modèles sur de vraies traces durant plusieurs minutes ou même plusieurs heures, ils donnent des bons résultats. Cependant, en les utilisant pour prévoir le trafic pour des jours ou des semaines, leur performance se dégrade de manière significative. Notre but dans la thèse est de proposer un nouveau modèle pour la prévision du trafic qui tient compte simultanément des dépendances à longue portée et à courte portée.

1.1.1.5 Étude de réseaux de neurones

Les réseaux de neurones (RN) ont été employés avec succès pour résoudre beaucoup de problèmes [117, 149, 150, 2, 4, 7, 50, 128, 10] qui ne pourraient pas être résolus, par exemple, par des algorithmes "classiques". L'un des motivations derrière l'utilisation des RN est qu'ils sont très efficaces en apprenant et en prévoyant des fonctions non linéaires et complexes. En outre, nous n'avons pas besoin de connaître les détails du modèle mathématique du système à étudier (il est dans certains cas très difficile à trouver ou bien complètement inaccessible). Les problèmes mentionnés ci-dessus sont complexes et, pour le cas du trafic, non stationnaires. Notre travail s'appuie sur les RN. Nous avons employé deux genres de RN : les RN artificiels (ANN) [117, 118, 149, 150] et les RN aléatoires (RNN) [49, 89, 48, 45, 101, 51]. Nous avons comparé leurs performances respectives et nous avons constaté que les RNN ont plusieurs avantages par rapport aux ANN. Cependant, un inconvénient des RNN est que l'algorithme disponible d'apprentissage est assez lent et parfois peu efficace. Ceci nous a motivés pour proposer de nouveaux algorithmes d'apprentissage pour surmonter ce type de problème.

1.1.2 Les contributions de cette thèse

Dans cette section nous résumons brièvement les contributions principales de cette thèse. Nous les classifions selon le domaine auquel elles appartiennent naturellement. Dans le domaine de l'évaluation de la qualité de flux multimédia, les contributions principales sont les suivantes :

- Nous proposons une nouvelle méthodologie pour mesurer la qualité en temps réel de ces flux ou "streams" (la parole, l'audio, et/ou la vidéo) (les publications associées sont [90, 92, 95, 91]). Les propriétés principales de notre méthode sont les suivantes : (i) aucun accès au signal original n'est exigé, (ii) les calculs relatifs à cette méthode sont rapides, puisque, une fois qualifié, le RN donne ses évaluations dans des temps négligeables, (iii) les résultats obtenus sont bien corrélés bien avec le MOS (c.-à-d. la perception humaine), (iv) beaucoup de paramètres influant la qualité peuvent être pris en considération, (v) elle peut fonctionner en temps réel et, par exemple, elle peut être facilement intégrée avec une application de multimédia.
- Nous étudions l'applicabilité de notre méthode au cas de l'évaluation de la qualité de la parole. Nous présentons les résultats d'expériences effectuées entre des sites nationaux et internationaux. Le but de ces expériences est d'identifier les plages typiques des paramètres de réseau. Nous employons trois codecs[2] différents de la parole à savoir PCM, ADPCM et GSM. Trois langues (l'arabe, l'espagnol et le français) sont considérées. En particulier, le taux de perte, la taille de

2. C'est un terme qu'on utilise pour désigner l'ensemble d'un encodeur et un décodeur.

rafales des pertes aussi bien que l'intervalle de la mise en paquet sont considérés (les publications associées sont [92, 90]).

- Nous explorons l'applicabilité de notre méthode pour évaluer la qualité de la vidéo temps réel. Nous employons un codec H.263 et nous avons choisi cinq paramètres importants qui ont un impact fort sur la qualité visuelle. Ces paramètres sont : le débit, le débit d'images, le taux de perte, la taille des rafales des pertes, et la quantité de l'information redondante utilisée pour remédier à la perte (la publication associée est [95]).
- Nous fournissons une étude de l'impact des paramètres sur la qualité perçue. Notre étude est pour les applications vidéo et audio. À notre connaissance, c'est la première analyse des effets combinés de plusieurs paramètres de réseau et de l'encodage sur la qualité (les publications correspondantes sont [94, 96]).

Dans le domaine de contrôle de débit, notre contribution est la suivante :

- Nous présentons un nouveau protocole de contrôle de débit qui intègre des mesures passives de réseau (par exemple, la perte) et la perception de l'utilisateur (la qualité) de flux multimédias transmis sur le réseau. Les objectifs de notre dispositif de contrôle sont doubles : d'abord, une meilleure utilisation de la bande passante, et en second lieu, la livraison de la meilleure qualité étant donnée la situation actuelle du réseau (la publication associée est [93]). On peut voir cette contribution comme une application de la méthode d'évaluation de qualité décrite précédemment. Nous avons examiné l'applicabilité de notre protocole pour la transmission de la parole et la vidéo via l'Internet.

Dans le domaine de la prévision du trafic, notre contribution est la suivante :

- Nous proposons une nouvelle méthode pour la prévision du trafic qui tient compte non seulement de la dépendance à court terme du processus du trafic, mais également celle à long terme. Nous étudions et évaluons notre modèle en employant de vraies traces de trafic et nous proposons quelques applications possibles qui peuvent tirer profit de notre nouvelle technique.

Dans le domaine de l'apprentissage d'un RN, la contribution principale est la suivante :

- Nous proposons deux nouveaux algorithmes d'apprentissage pour les réseaux de neurones aléatoires. Ces deux algorithmes sont inspirés des algorithmes d'apprentissage les plus rapides pour le ANN, à savoir *Levenberg-Marquardt* (LM) et une variante récemment proposée, référée comme *LM avec adaptive momentum*. Nous évaluons la performance de ces algorithmes et les comparons avec l'algorithme disponible de type "gradient descente". En outre, nous proposons quelques éléments de comparaison entre ANN et RNN.

1.1.3 Vue d'ensemble de la dissertation

Dans cette section, nous avons présenté les motivations et le contexte des problèmes que nous abordons dans la thèse. Le reste de ce chapitre est organisé comme suit. Dans la section 1.2, nous présentons une nouvelle méthode pour mesurer en temps réel la qualité de multimédia transmis en temps réel sur un réseau de paquets. Nous fournissons la méthodologie générale qui peut être appliquée pour la parole, l'audio et/ou la vidéo. Nous présentons également les techniques les plus utilisées pour évaluer la qualité subjective de multimédia. Nous nous concentrons sur l'utilisation des réseaux de neurones et nous comparons ANN et RNN dans le contexte de nos problèmes. Nous décrivons également quelques utilisations et applications possibles de notre méthode.

En section 1.3, nous évaluons l'applicabilité de notre technique pour évaluer en temps réel la qualité de la parole. Nous présentons les résultats des expériences que nous avons effectuées entre plusieurs sites pour identifier les plages typiques des paramètres de réseau. Dans la section 1.4, nous analysons l'applicabilité de notre technique pour évaluer la qualité de la vidéo. Dans la section 1.5, nous étudions l'impact des paramètres sur la qualité perçue. En 1.6, nous proposons un protocole de contrôle de débit

qui tient compte de la perception d'utilisateur aussi bien que des mesures traditionnelles de réseau pour fournir la meilleure qualité en temps réel étant donné l'état actuel de réseau. Nous fournissons une liste des paramètres possibles qui peuvent être employés pour les applications de type vidéo et audio.

En section 1.7, nous fournissons une nouvelle méthode pour la prévision du trafic. Nous comparons notre approche à celles existantes et nous l'évaluons en employant de vraies traces. Dans la section 1.8, nous présentons deux nouveaux algorithmes d'apprentissage pour améliorer la performance et la précision des RNN. Nous évaluons ces algorithmes et nous les comparons également à l'algorithme traditionnel de type "gradient descente".

En conclusion, nous donnons les conclusions de cette thèse et quelques perspectives de recherches liées aux problèmes abordés au cours de ce travail dans la section 1.9.

1.2 Descriptions de notre nouvelle méthode en général

Dans cette section, nous proposons une nouvelle méthode pour évaluer la qualité des flux multimédias transmis en temps réel sur des réseaux de paquet. Cette méthode peut être utilisée en général avec n'importe quel type de média (la parole, la voix, la vidéo ou le multimédia). La validation de cette méthode est remise aux prochaines sections. La qualité de multimédia est affectée par beaucoup de paramètres comme les pertes de réseau, le délai, la gigue, le type de codage, le débit, etc. En outre, la qualité n'est pas linéairement proportionnelle à la variation d'aucun de ces paramètres. La détermination de la qualité est, donc, un problème complexe, et il n'a pas été possible de le résoudre en développant des modèles mathématiques qui incluent simultanément les effets de tous ces paramètres.

Nous pensons que les réseaux de neurones (RN) sont des outils appropriés pour résoudre ce problème [117, 19], voir la section 3.6 pour plus de détails. Nous illustrons notre approche en construisant un système qui tire profit des avantages offerts par les RN pour capter la relation non linéaire entre plusieurs mesures non subjectives (c.-à-d. les paramètres des réseaux de paquet et de codage) et la qualité perçue par un groupe de sujets d'humains.

1.2.1 Vue d'ensemble de la méthode

Nous allons décrire notre méthode indépendamment du type de multimédia à disposition. Voir les figures 4.1 et 4.2 en lisant le texte qui suit. D'abord, nous définissons un ensemble d'informations statiques qui affecteront la perception de qualité. Nous devons choisir les paramètres les plus importants *a priori* correspondant aux applications et au réseau qui assure la transmission. Puis, pour chaque paramètre nous devons choisir plusieurs valeurs couvrant la gamme de valeurs possibles. Par exemple, si on s'attend à ce que le taux de perte varie de 0 à 10%, nous pouvons employer 0, 1, 2, 5, et 10% en tant que valeurs typiques pour ce paramètre. Notons que tous les types de média ne tolèrent pas les mêmes valeurs d'un paramètre donnée de la même manière. Par exemple, la parole peut tolérer des taux de perte jusqu'à 20%, alors que la vidéo seulement jusqu'à 5%.

Pour produire une base de données (BD) composée des échantillons correspondant à différentes configurations des paramètres choisis, un environnement de simulation ou un banc d'essai doit être mis en application. Ce dernier est employé pour envoyer des échantillons de médias de la source à la destination et pour contrôler le réseau de transmission. Chaque configuration dans les données d'entrée définies doit être appliquée au système composé du réseau, de la source et du récepteur. Par exemple, considérons les réseaux IP et le cas de la vidéo; la source contrôle le débit, le débit d'images et l'algorithme de codage, et elle envoie des paquets RTP de vidéo; le routeur se manifeste à travers le taux de perte et la distribution de perte, ainsi que les conditions du trafic dans le réseau; la destination stocke le flux vidéo transmis et rassemble les valeurs correspondantes des paramètres. Puis, en actionnant le banc d'essai ou en employant la simulation, nous produisons et stockons un ensemble d'échantillons déformés avec les valeurs correspondantes de ces paramètres.

Ensuite, un essai subjectif de la qualité doit être effectué. Il y a plusieurs méthodes subjectives dans les recommandations de l'UIT-R ou de l'UIT-T selon le type de média. Les détails sur cette étape se trouvent dans la section 1.2.2. En général, un groupe de sujets humains est invité à évaluer la qualité des échantillons (c.-à-d. chaque sujet donne à chaque échantillon une note selon une échelle prédéfinie). Les sujets ne devraient établir aucune relation entre les échantillons et les valeurs de paramètres correspondants.

En s'appuyant sur les notes donnés par les sujets humains, des analyses et filtrages statistiques doivent être effectués pour retirer les notes des individus suspectés de donner des résultats incertains [73]. Voir la section 1.2.2 pour plus de détails au sujet de cette étape. L'étape suivante est de calculer les valeurs de MOS pour tous les échantillons. Ensuite, nous stockons les valeurs de MOS et les valeurs de paramètres correspondants dans une BD.

Dans la troisième étape, une architecture appropriée de RN et un algorithme d'apprentissage doivent être choisis. La BD construite précédemment est divisée en deux parties : une pour entraîner le RN et l'autre pour vérifier son exactitude. Ce RN peut alors évaluer la qualité pour toutes les valeurs données des paramètres.

1.2.1.1 Plus d'un type de média à la fois

Notons que, dans les cas où il y a plusieurs types de média (la parole + la vidéo par exemple), nous recommandons qu'un module séparé de RN est utilisé pour mesurer la qualité du chacun des médias. Un module supplémentaire de RN devrait être utilisé pour évaluer la qualité globale en fonction de toutes les mesures d'autres modules de RN. Notons qu'il n'y a pas beaucoup de travail à faire dans ces cas, car les sujets sont invités à évaluer chaque type de média à part ainsi que la qualité de multimédia globale de chaque échantillon pendant l'essai subjectif. Ceci parce que, comme précédemment mentionné, tous les types de média ne tolèrent pas de la même manière les paramètres de réseau, aussi bien que la diversité des caractéristiques des algorithmes de codage.

1.2.2 Essais subjectifs de la qualité

Pour évaluer la qualité de la parole, de l'audio ou de la vidéo (codec, télécommunication, images de télévision, téléphonie sur IP, etc.), un essai subjectif de qualité est typiquement employé. Dans cet essai, un groupe de sujets humains est invité à juger la qualité de l'échantillon déformé. Il y a plusieurs recommandations qui indiquent des conditions strictes à suivre afin d'effectuer ces essais. Ces recommandations sont spécifiques au type de média (la parole [70], l'audio [63], la vidéo [72] ou le multimédia [73] en général).

Les recommandations de l'UIT suggèrent que la durée des échantillons doit être d'environ 10 secondes pour l'audiovisuel, et d'environ la durée de 5 petites phrases pour la parole. Le nombre de sujets exigés pour effectuer ces essais peut être typiquement entre 4 et 40. Quatre est le minimum absolu pour des raisons statistiques. Le nombre de sujets requis dépend de la sensibilité et de la fiabilité de la méthode adoptée. Le nombre moyen de sujets est environ 15. Avant d'effectuer l'un de ces essais, des instructions doivent être données aux sujets. Ces instructions incluent la méthode d'évaluation et les types de déformations qui se probablement produiraient et l'échelle de notes. Des séquences d'entraînement peuvent être données aux sujets pour les familiariser avec la tâche qu'ils accompliront.

D'après les recommandations de l'UIT, les essais subjectifs qui durent plus de 30 minutes doivent être divisés en sessions multiples et chaque session ne doit pas durer plus de 30 minutes. Au début de chaque session, on doit ajouter quelques échantillons factices (environ quatre ou cinq). Ces échantillons ne devraient pas être considérés dans le calcul. Leur but est d'être employés pour familiariser les sujets à donner des notes significatives.

1.2.2.1 Les méthodes subjectives d'essai de qualité

Dans les recommandations de l'UIT, quelques méthodes subjectives d'essai de la qualité existent. Ici nous détaillons les méthodes qui peuvent être employées avec notre approche. Elles sont l'évaluation par catégories absolues (ACR), l'évaluation par catégories de dégradation (DCR) et la mesure continue de la qualité en double stimulus (DSCQS).

Évaluation par catégories absolues (ACR)

Dans cette méthode, les échantillons sont présentés un par un et sont évalués indépendamment sur une échelle de catégorie. Le modèle de temps pour la présentation est illustré par la figure 4.5. L'échelle de notes à 5 points pour l'estimation de la qualité est l'une des plus utilisées, voir le tableau 4.1. Si une puissance distinctive plus élevée est exigée, une échelle à 9 points peut être employée comme indiqué dans le tableau 4.2. Enfin, il y a une échelle générale, qui est continue est indiqué en figure 4.3. La quantité évaluée d'après les notes est représentée par le symbole MOS.

Évaluation par catégories de dégradation (DCR)

Dans cette méthode, les échantillons sont présentés en paires. Le premier échantillon à présenter de chaque paire est toujours la référence, c'est à dire, le signal sans aucune déformation. Le second est le même signal mais altéré par les conditions d'essai. Le modèle de temps pour la présentation est montré sur la figure 4.6. Dans ce cas-ci, les sujets sont invités à évaluer la déformation de deuxième échantillon par rapport à la référence. L'échelle de notes à 5 points pour évaluer la déformation est la plus largement répandue (voir le tableau 4.4). Cependant, toutes les échelles utilisées dans la méthode ACR peuvent être employées avec la méthode DCR à condition de remplacer les adjectifs de qualité par les adjectifs correspondants de déformation.

La mesure continue de la qualité en double stimulus (DSCQS)

DSCQS est utile quand il n'est pas possible de fournir les conditions d'essai qui montrent la gamme complète de la qualité. De même que la méthode de DCR, les échantillons sont présentés en paires: la référence et l'altéré. Des sujets sont invités à évaluer la qualité des deux échantillons (et non pas seulement l'échantillon altéré par rapport à la référence comme dans DCR), mais les sujets ne savent pas quelle est la référence. Les sujets sont invités à évaluer la qualité globale d'échantillon de chaque présentation en insérant une marque sur une échelle verticale (voir la figure 4.8).

1.2.3 Calcul de MOS et analyse statistique

Soit $S = \{\sigma_1, \sigma_2, \sigma_s\}$ l'ensemble des échantillons, et désignons par N le nombre de sujets dans la méthode subjective choisie et par u_{is} l'évaluation de l'échantillon σ_s faite par l'utilisateur i. L'ensemble de valeurs $(u_{is})_{i=1,\cdots,N}$ présenterait probablement des variations dues aux différences dans le jugement entre les sujets. D'ailleurs, il est possible que quelques sujets ne fassent pas assez attention pendant l'essai. Ceci peut mener à des données contradictoires pour la phase d'apprentissage de RN. Le filtrage statistique est ainsi nécessaire sur l'ensemble de données brutes. La référence la plus largement répandue en la matière est la recommandation UIT-R BT.500-10 [67]. Le procédé décrit permet d'enlever les estimations de ces sujets qui ne pourraient pas conduire à notes conformes.

D'abord, désignons par $\bar{u}_s$ la moyenne des évaluations de l'échantillon σ_s sur l'ensemble de sujets, c.-à-d.,

$$\bar{u}_s = \frac{1}{N} \sum_{i=1}^{N} u_{is}. \tag{1}$$

Désignons par $[\bar{u}_s - \Delta_s, \bar{u}_s + \Delta_s]$ l'intervalle de confiance à 95% obtenu à partir de (u_{is}), c.-à-d., $\Delta_s = 1.96\delta_s/\sqrt{N}$, où

$$\delta_s = \sqrt{\sum_{i=1}^{N} \frac{(u_{is} - \bar{u}_s)^2}{N-1}}.$$

Comme indiqué dans [67], on doit établir si cette distribution de notes est normale ou pas, en utilisant l'essai β_2 (en calculant le coefficient de *kurtosis*). Si β_2 est entre 2 et 4, la distribution peut être considéré normale. Dénotant par $\beta_{2s} = m_{4s}/m_{2s}^2$ où

$$m_{xs} = \frac{1}{N}\sum_{i=1}^{N}(u_{is} - \bar{u}_s)^x,$$

si $2 \leq \beta_{2s} \leq 4$ alors on peut supposer que la distribution $(u_{is})_{i=1,\cdots,N}$ est normale. Pour chaque sujet i, nous devons calculer deux valeurs de nombre entier L_i et R_i, d'après le procédé suivant :

soit $L_i = 0$ et $R_i = 0$
pour chaque échantillon $\sigma_s \in \mathcal{S} = \{\sigma_1, \cdots, \sigma_S\}$
  si $2 \leq \beta_{2s} \leq 4$, alors
    si $u_{is} \geq \bar{u}_s + 2\,\delta_s$ alors $R_i = R_i + 1$
    si $u_{is} \leq \bar{u}_s - 2\,\delta_s$ alors $L_i = L_i + 1$
  sinon
    si $u_{is} \geq \bar{u}_s + \sqrt{20}\,\delta_s$ alors $R_i = R_i + 1$
    si $u_{is} \leq \bar{u}_s - \sqrt{20}\,\delta_s$ alors $L_i = L_i + 1$

Enfin, si $(L_i + R_i)/S > 0,05$ et $|(L_i - R_i)/(L_i + R_i)| < 0,3$ alors les notes du sujet i doivent être supprimées. Pour plus de détails à ce sujet et d'autres méthodes d'essai subjectifs, voir [67]. Après l'élimination des notes de ces sujets qui ne pourraient pas conduire des estimations cohérentes en utilisant la technique ci-dessus, les notes moyennes devraient être calculées en utilisant (1). Ceci constituera la BD de MOS que nous emploierons pour entraîner et pour valider le RN.

1.2.4 Comparaison entre ANN et RNN

Dans le contexte de notre problème spécifique, nous comparons les deux types de réseaux de neurones : les réseaux de neurones artificiels (ANN) et les réseaux de neurones aléatoires (RNN). Nous avons employé la boîte à outils de RN MATLAB pour travailler avec les ANN, et un logiciel tournant sous MATLAB [1] pour les RNN. Nous avons observé que le processus d'apprentissage des ANN était relativement plus rapide que celui des RNN (pour surmonter ce problème, nous avons proposé plusieurs algorithmes d'apprentissage pour les RNN dont le but est d'accélérer le processus d'apprentissage, voir la section 1.8). Cependant, pendant la phase d'exécution, les RNN ont surpassé les ANN dans le temps de calcul.

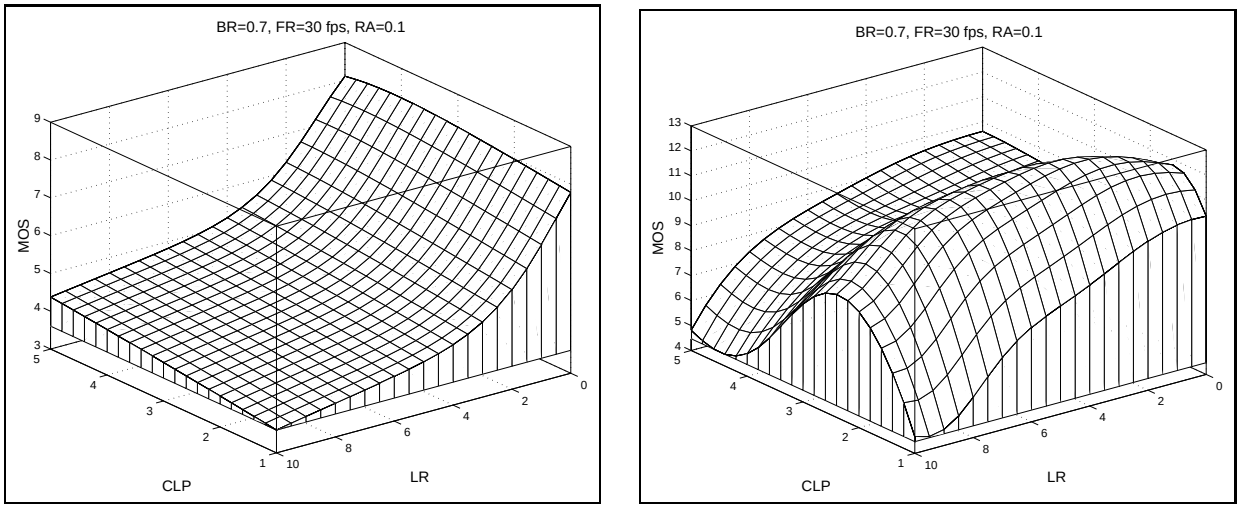
Les calculs des ANN sont plus lents parce qu'ils impliquent des calculs de fonctions non linéaires (*sigmoïde, tanh*, etc.) pour chaque neurone dans l'architecture. Ceci rend les RNN particulièrement attrayants dans des applications avec contraintes en temps réel ou pour des applications légères. Ceci peut être important dans certaines applications en réseaux, par exemple, dans [83], un prédicteur de perte de paquet par les ANN a été proposé pour des flux multimédias temps réel. La précision est bonne, mais le temps de calcul est beaucoup plus long que le délai d'inter-arrivée de paquets. Ceci rend le système inutile à moins de disposer d'ordinateurs très puissants.

Un avantage très important des RNN que nous avons trouvé pour notre problème est qu'il capte très bien la relation entre les valeurs des paramètres et la qualité. Ceci concerne également leur capacité à extrapoler d'une manière logique les valeurs de paramètres hors des gammes utilisées pendant la phase

d'apprentissage. Les problèmes les plus communs des ANN sont *le surentraînement* et la sensibilité au nombre de neurones cachés. Le problème de surentraînement fait le RN mémorise les modèles d'apprentissage, mais donne des généralisations faibles pour de nouvelles entrées. D'ailleurs, si nous ne pouvons pas identifier un certain nombre "quasi optimal" de neurones cachés, la performance peut être mauvaise. La figure 4.9(b) montre un exemple d'un réseau ANN surentraîné, où nous pouvons voir des irrégularités, c'est-à-dire de mauvaises généralisations.

Nous avons entraîné différentes architectures (en changeant le nombre de neurones cachés) pour ANN et RNN, avec les mêmes données (décrites dans la section 1.4.1) et la même valeur d'erreur. Regardons, par exemple, le comportement de la qualité en fonction du débit normalisé (BR) (les autres 4 variables ont été fixés). Dans la BD, le BR varie entre 0,15 et 0,7. Dans les figures 4.10(a) et 4.10(b), nous montrons la capacité des deux types de réseau à interpoler et à extrapoler les résultats quand le BR change de zéro à sa valeur maximale 1. Ces deux figures montrent que les RNN captent correctement la relation entre les variables d'entrée (les paramètres) et la qualité, et que ce model n'est pas très sensible au nombre de neurones cachés ; tandis que les ANN donnent des approximations tout à fait différentes pour de petits changements de la taille dans la couche cachée.

Par ailleurs, si l'architecture optimale d'ANN ne pouvait pas être identifiée, son exactitude pourrait être mauvaise. Regardons maintenant les valeurs extrêmes : si BR=0,0, la qualité doit être autour le 1, alors que pour BR=1.0, la qualité doit être entre 8,5 et 9,0. Pour le cas d'ANN, comme représenté sur la figure 4.10(b), quand le nombre de neurones cachés varie de quatre (optimal expérimentalement) à cinq, la généralisation est mauvaise, particulièrement quand le BR tendre à zéro, où le résultat est 3,2 au lieu de 1. Ceci donne à RNN une meilleure capacité à généraliser.



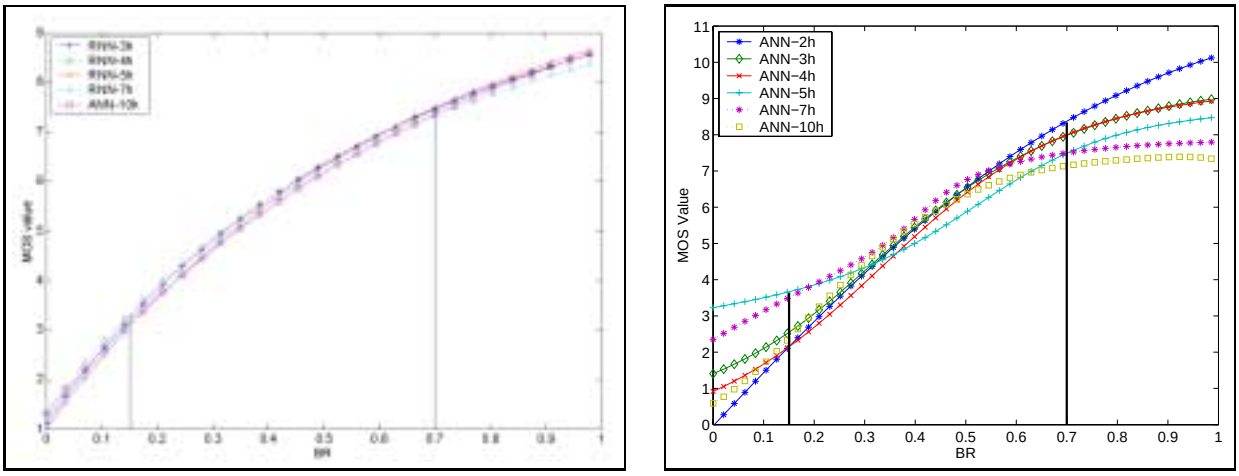
(a) Un ANN bien entraîné

(b) Exemple d'un ANN surentraîné

FIG. 1 – *Le problème de surentraînement en utilisant ANN.*

1.2.5 Choix de paramètres

Il est important d'identifier et de choisir correctement les paramètres influant la qualité. Si un paramètre important est négligé, l'exactitude de l'outil sera dégradée et ne se comportera pas comme prévu. Cependant, le nombre de paramètres à prendre en considération devrait être réduit au minimum. Ceci parce que ce nombre détermine le nombre minimum d'échantillons dans la BD de qualité pour entraîner et tester le RN, sachant que ces échantillons devraient être évalués par essai subjectif, ce qui est très coûteux et difficile à effectuer. Par conséquent quelques mécanismes peuvent être employés



(a) RNN (b) ANN

FIG. 2 – *La performance d'un ANN et d'un RNN à interpoler et à extrapoler pour des nombres différents des neurones cachés.*

pour fusionner deux paramètres ou plus en un seul. Par exemple, la gigue peut être incluse dans le paramètre de perte si on emploie un buffer de dejittering.

Nous pouvons classifier les paramètres influant sur la qualité en trois catégories :

- **Paramètres de réseau** : paramètres dus au processus de mise en paquet et à la transmission à travers d'un réseau de type best effort. Les paramètres *a priori* les plus importants sont le taux de perte de paquet, la gigue, la taille de paquet, la distribution de perte, et le délai de bout en bout).
- **Paramètres de codage** : les paramètres dus à la capacité d'encoder ou de compresser le signal original pour réduire son débit. Par conséquent, certains des paramètres dus à la déformation de codage peuvent être classifiés pour la parole comme le type du codec utilisé, la fréquence d'échantillonnage, le débit, le nombre de couches dans le cas des codecs à multicouches. Pour la vidéo, il y a le débit, le débit d'images, les types de codage de chaque image, etc.
- **D'autres paramètres** comme l'écho (qui peut se produire en raison du long retard ou le nombre de sources participantes. Ces effets se produisent dans des sessions bi-directionnelles. Il y a également l'aspect de synchronisation de l'audio avec la vidéo pour les applications de multimédia [40]. En outre, les effets des techniques de dissimulation des erreurs (FEC ou ARQ) [38] peut affecter la qualité.

1.3 La mesure de la qualité de la parole en temps réel

Les utilisations récentes de la téléphonie sur IP [32], de la voix sur IP [60], et de la voix et la téléphonie sur ATM [152] ont déterminé un fort besoin d'évaluer la qualité de l'audio, en temps réel, quand celle-ci est transmise sur n'importe quel réseau de paquet. Le nombre d'applications de réseau qui exigent l'évaluation de la qualité de l'audio augmente rapidement. En dépit de l'importance de ce point, peu de méthodes sont disponibles; en outre, les quelques contributions dans ce domaine se concentrent principalement sur la différentiation des algorithmes de codage sans tenir compte des paramètres de réseau.

Dans la section précédente, nous avons présenté une nouvelle méthode pour évaluer la qualité des flux multimédias transmis en temps réel sur des réseaux de paquet en général, indépendemment du

type de média. Dans cette section, nous visons à valider notre approche dans le cas de l'évaluation de qualité de la parole.

1.3.1 Paramètres affectant la qualité de la parole

Pour identifier les plages et les valeurs typiques des paramètres de réseau, nous avons effectué une expérience dont les détails sont dans la section 5.2. En s'appuyant sur ce résultat, nous avons choisi des valeurs pour les paramètres de réseau et de codage comme suit :

1. Taux de perte : 0, 5, 10, 20 et 40%.
2. Distribution de perte : nous avons choisi le nombre de paquets consécutivement perdus comme distribution de perte, qui varie de 1 à 5 paquets à la fois.
3. Intervalle de mise en paquet : nous avons choisi pour valeurs pour ce paramètre 20, 40, 60 et 80 ms. En fait la majorité des applications temps réel existantes emploient une ou plusieurs de ces valeurs.
4. Algorithmes de codage : nous avons choisi le PCM (64kbps), le G726 ADPCM (32kbps) et le GSM-FR (13.2kbps). Nous n'avons pas considéré le codec ADPCM dans le cas de la langue française.

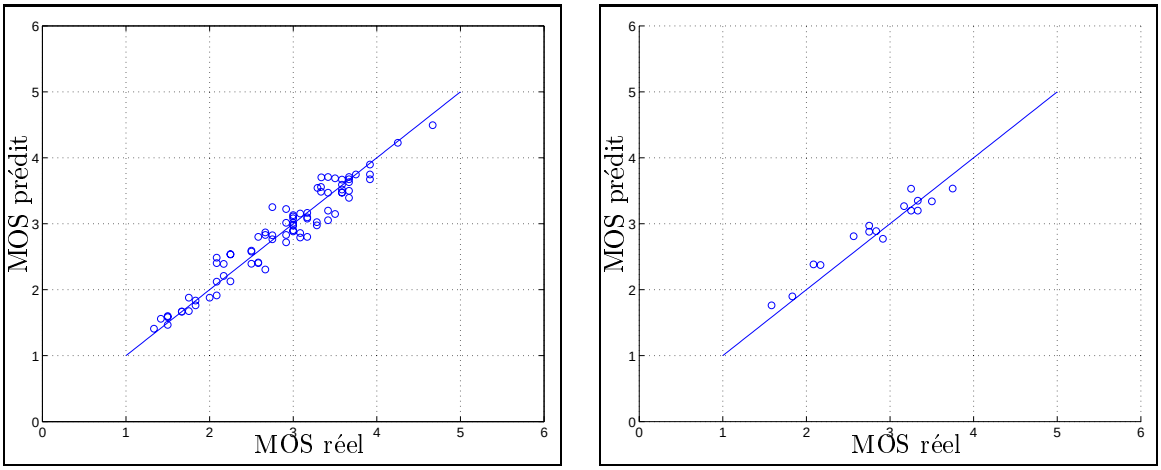
1.3.2 Une base de données de MOS pour différentes langues

Nous avons réalisé une BD ayant un ensemble d'échantillons de parole ayant subi différentes déformations par le jeu de paramètres choisis comme montré dans le tableau 5.2. L'essai subjectif effectué est conforme aux recommandations de l'UIT et réalisé en suivant la méthode ACR [70]. Ces échantillons ont été produits en employant un banc d'essai de réseau IP composé d'un expéditeur, d'un routeur, et d'un récepteur. L'expéditeur a contrôlé l'intervalle de mise en paquet et le choix de l'algorithme de codage. Le routeur a contrôlé le taux de perte, et la distribution de perte. Enfin, le récepteur a stocké les signaux reçus, les a décodé, et a calculé le taux de perte et le nombre de paquets consécutivement perdus. De plus, il a été établi que les codecs de voix dépendent de la langue utilisée [79] ; ainsi, le type de langue est une autre variable incluse dans notre BD. Nous avons effectué des essais de qualité subjective dans trois langues différentes : l'espagnol, l'arabe et le français.

Trois groupes de sujets ont été invités pour évaluer la qualité des échantillons pour les trois langues considérées sur une échelle de qualité à 5 notes. Les nombres des échantillons ont été 96, 96 65 pour respectivement la langue arabe, espagnole et française. Les nombres de sujets sont 15 pour la langue arabe, 15 pour la langue espagnole et 7 pour la lange française. Ensuit, des analyses et filtrages statistiques ont été effectués pour identifier les sujets qui ne pouvaient pas fournir des estimations fiables suivant la description de la section 1.2.3. En conséquence, nous avons enlevé toutes les notes d'un sujet pour la lange espagnol et de trois pour la langue arabe. Ensuite, la valeur de MOS pour chaque échantillon a été calculée. Voir les résultats dans le tableau 5.2.

1.3.3 Évaluation de la qualité de la parole par le RN

On utilise un RN de type *feedforward* à trois couches pour chaque ensemble d'échantillons (chaque langue) se composant de 4 entrées (les paramètres) dans la couche d'entrée, 5 neurones cachés, et un neurone de sortie (la mesure de qualité). Nous avons entraîné chaque RN en utilisant les 80 premiers échantillons des bases de données montrées dans le tableau 5.2 dans les cas des langues arabe et espagnole et 50 échantillons dans le cas de la langue française. Les échantillons restants ont été employés pour examiner les RN. En comparant les valeurs réelles de la qualité avec celles évaluées par les RN, nous les montrons dans les figures 5.4, 5.5 et 5.6 pour les BD arabe et espagnole respectivement. Des nuages de points sont aussi montrés pour ces cas sur les figures 3(a), 4(a) et 5(a). De même, nous avons utilisé les RN qualifiés pour évaluer les qualités pour les échantillons de test. Ces échantillons ne sont pas parmi ceux servant à entraîner ces RN. Nous traçons sur les figures 5.7(a), 5.7(b) et 5.7(c)



(a) BD d'apprentissage (b) BD de teste

FIG. 3 – *La corrélation entre les valeurs réelles et prévues de MOS (langage arabe).*

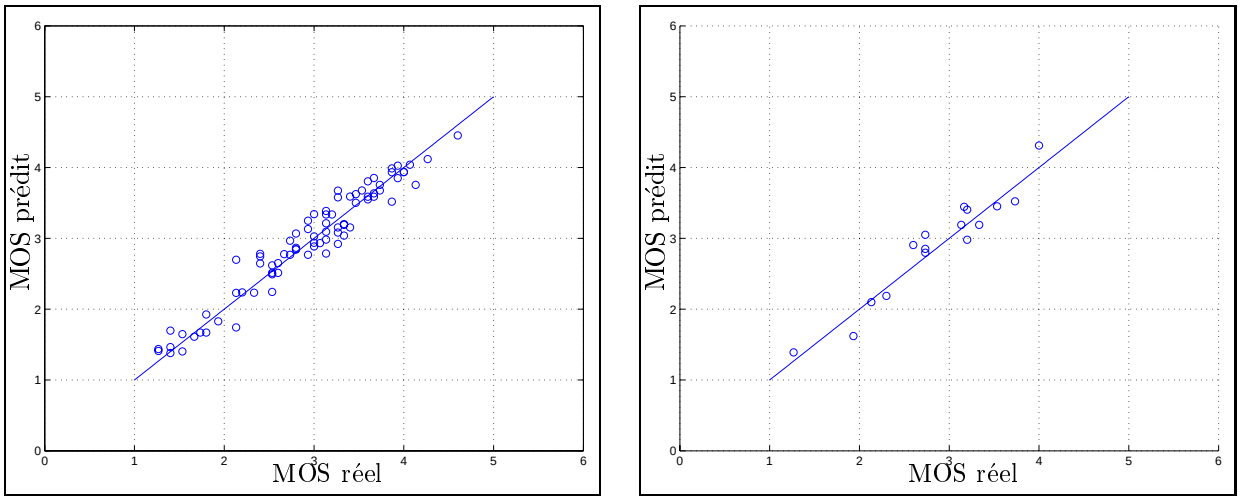
les valeurs prévues contre les réelles. Nous montrons également des nuages de points pour ces cas sur les figures 3(b), 4(b) et 5(b).

De ces figures, nous pouvons constater deux résultats importants : d'abord, les RN ont la capacité d'apprendre très exactement l'évaluation du MOS pour un ensemble donné de paramètres d'entrée. Ceci est clairement montré dans les figures en utilisant les BD d'apprentissage. En second lieu, les RN peuvent avoir une évaluation très précise de MOS pour toutes les nouvelles valeurs des paramètres d'entrée. Ceci est montré dans les figures en utilisant les BD de test.

On peut observer de ces figures que les notes de la qualité de la parole produites par le modèle de RN sont en parfait accord avec les résultats obtenus par les sujets participant à l'expérience d'essai subjectif de la qualité. On peut établir de ces résultats que les algorithmes d'apprentissage donnent aux RN l'avantage de l'adaptabilité, qui leurs permet de fonctionner dans des environnements dynamiques. Les statistiques sont comme montrées dans le tableau 1. Comme nous pouvons observer, les résultats sont très encourageants, l'approche via les RN nous a permis d'obtenir un modèle très bon de la relation non linéaire entre la qualité et les paramètres qui l'affectent.

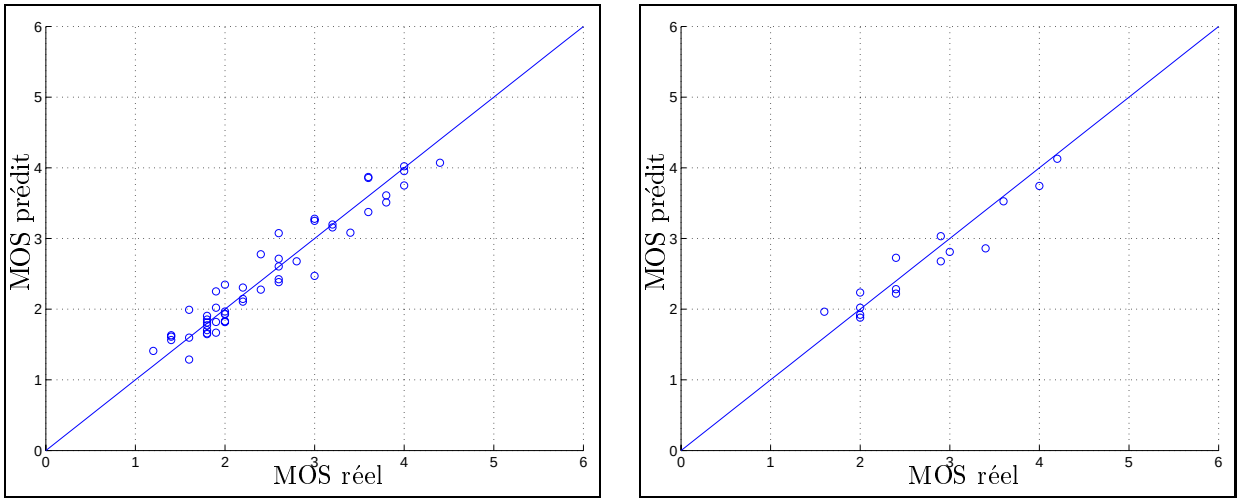
TAB. 1 – *La performance de RN pour apprendre le problème et sa réaction aux exemples non-vus.*

Langage	Bases de données d'apprentissage		Bases de données de test	
	Coefficient de corrélation	MSE	Coefficient de corrélation	MSE
Arabe	0,966	0,035	0,967	0,035
Espagnol	0,969	0,035	0,961	0,045
Français	0,965	0,048	0,957	0,055



(a) BD d'apprentissage (b) BD de teste

FIG. 4 – *La corrélation entre les valeurs réelles et prévues de MOS (langage espagnol).*



(a) BD d'apprentissage (b) BD de teste

FIG. 5 – *La corrélation entre les valeurs réelles et prévues de MOS (langage français).*

1.3.4 Performance d'autres mesures de la qualité de la parole

À présent, pour avoir une idée de la performance des mesures objectives connues de qualité de la parole; nous avons rassemblé dans cette section quelques données et figures de la littérature pour l'illustration. Les résultats rapportés sont présentés en deux groupes : le premier montre la performance de ces mesures à évaluer la qualité quand seulement les paramètres de codage sont considérés ; dans le second elles sont employées en prenant en compte les paramètres de codage et de réseau. Selon les données disponibles, les mesures comparées sont SNR, SNRseg, BSD, MBSD, EMBSD, PSQM, PSQM+, MNB(1,2), E-modèle et PAMS. Voir la version anglaise, section 3.1, pour plus de détails.

Dans le cas où ces mesures sont employées pour évaluer seulement les déformations dues aux paramètres de codage (voir le tableau 2), comme nous le voyons, les mesures simples comme SNR et SNRseg donnent une corrélation faible avec les essais subjectifs de qualité (entre 0,226 et 0,523 pour SNR et entre 0,221 et 521 pour SNRseg). MBS, MBSD et EMBSD donnent de meilleurs résultats que SNR ou SNRseg. Pour PSQM et sa version améliorée PSQM+, la performance est meilleure et elle est comparable à celle de PAMS et de MNB. Le coefficient de corrélation peut atteindre jusqu'à 0,98 pour certaines mesures.

D'autre part, quand ces mesures sont employées pour évaluer la qualité en tenant compte des effets de réseau de codage, la performance se dégrade. Les seules données disponibles comparant la majorité de ces mesures sont trouvées dans [155, p. 106], voir tableau 3. Cependant, en raison de l'accord de projet, ils n'ont pas indiqué les noms de ces mesures comparées, exceptés MBSD et EMBSD. Nous pouvons voir qu'EMBSD donne un coefficient de corrélation de 0,54 dans ce cas-ci contre 0,87 dans le cas où il est appliqué au cas de codage seulement. La performance de MBSD est très mauvaise, environ 0,24 contre 0,760 pour le cas de codage seulement. La meilleure mesure est celle d'un coefficient de corrélation de 0,90 avec des résultats de MOS.

Parmi ces mesures, seulement l'E-modèle de l'UIT n'a pas besoin de l'accès au signal original pour calculer la qualité. Ainsi, c'est la seule mesure disponible qui pourrait théoriquement être employée dans des applications temps réel comme notre méthode. Cependant, utilisée en prenant en compte les effets de codage et de réseau, elle est peu performante : le coefficient de corrélation est environ 0,70 [57]. En outre, le même travail présenté dans [57] compare les performances d'EMBSD, d'E-modèle et de MNB ; les résultats obtenus prouvent que la corrélation avec des données de MOS est très mauvaise comme le montre le tableau 4.

Nous montrons également des nuages de points pour MNB2 et E-modèle en comparant les résultats obtenues par ces mesures avec celles de MOS sur les figures 5.11(a) et 5.11(b). Une mesure optimale doit donner de résultats qui corrèlent parfaitement avec le MOS (c.-à-d. tous les résultats devraient être dessinés sur une même ligne). Cependant, de ces deux figures, nous constatons trop de variations :

TAB. 2 – *Le coefficient de corrélation de quelques mesures objectives existantes de la qualité de la parole avec MOS. Les résultats sont issus de la littérature. Seuls les paramètres de codages sont considérés. Les sources sont [155, p. 103], [139, p. 84], [113, p. 1517] et [112, p. 10/7].*

Les mesures objectives	corrélation avec MOS
SNR	0.226-0.523
SNRseg	0.221-521
BSD	0.367-0.919
PSQM	0.830-0.980
PSQM+	0.874-0.980
MNB2	0.740-.98
PAMS	0.640-0.895
MBSD	0.760
EMBSD	0.870

pour la même valeur du MOS, il y a beaucoup de valeurs de ces deux mesures. Par exemple, pour MOS=2.5, le résultat de MNB2 varie de 0,1 à 0,7 (la plage de résultats est de 0 à 1) ; en ce concerne E-modèle, on observe une variation de 20 à 80 (la plage de résultat est 0/100). De même, pour le même résultat de mesure, il y a aussi beaucoup de valeurs de MOS qui peuvent le satisfaire. Par exemple, pour le résultat de MNB2 de 0,4, la valeur correspondante de MOS varie de 1,2 à 4,5 ; et pour le résultat d'E-modè de 80, la valeur de MOS varie de 1,1 à 4,5 (avec l'échelle de notes à 5). Comme nous pouvons voir, ceci réduit de manière significative la confiance de ces mesures pour évaluer la qualité.

TAB. 3 – *Le coefficient de corrélation de quelques mesures objectives existantes de la qualité de la parole avec MOS en considérant les paramètres de codage et de réseau. Les résultats sont issus de la littérature : la source est [155, p. 106]. Les lettres de A à F représentent les mesures objectives de qualité présentées en section 3.1. Le nom de chaque mesure n'a pas été mentionné en raison de l'accord de projet.*

Les mesures objectives	corrélation avec MOS
A	0.87
B	0.85
C	0.56
D	0.86
E	0.90
F	0.86
MBSD	0.24
EMBSD	0.54

TAB. 4 – *Les coefficients de corrélation d'EMBSD, de MNB et d'E-model avec MOS en considérant les paramètres de réseau. La source est [57].*

Les mesures objectives	corrélation avec MOS
EMBSD	0.38
MNB1	0.51
MNB2	0.54
E-model	0.70

1.4 Évaluation de la qualité de la vidéo

La complexité élevée des processus d'encodage et de décodage de la vidéo par rapport à ceux de l'audio et de la parole est proportionnelle à la complexité des mesures de la qualité. De plus, en raison de la diversité des algorithmes de codage et leur complexité, il est difficile d'avoir une mesure objective de qualité qui tient compte de toutes les déformations possibles de codage et de réseau. Nous voudrions appliquer notre méthode pour évaluer son applicabilité au cas de la vidéo. Ainsi, nous montrons comment les RN peuvent être employés pour imiter la manière suivant laquelle un groupe de sujets humains évalue la qualité quand la vidéo est déformée par certains paramètres.

1.4.1 Les paramètres influant la qualité de la vidéo

Pour produire des séquences vidéos déformées, nous avons utilisé un outil qui encode des flux vidéo temps réel au format H263 [69], simule le processus de mise en paquet, simule la transmission sur un réseau IP et qui nous permet de manipuler le modèle de perte. Nous avons employé une séquence

vidéo standard appelé "stefan". Elle contient 300 images, est encodée à 30 images par seconde, avec une durée de 10 secondes et au format CIF (352 lignes X 288 pixels). La longueur maximale de paquet est 536 octets, afin d'éviter la fragmentation des paquets entre les routeurs. Nous présentons ici les paramètres que nous pensons avoir l'impact le plus important sur la qualité :

- Le débit (BR) : c'est le débit de flux sortant de l'encodeur. Nous avons choisi quatre valeurs : 256, 512, 768 et 1024 Koctets/s.
- Le débit d'images en images par seconde (FR) : ce paramètre prend l'une des 4 valeurs 5, 10, 15 et 30. Ceci est fait dans l'encodeur en laissant tomber des images uniformément.
- Le rapport des blocs encodés en mode intra à ceux encodés en mode inter (RA) : ceci est fait par l'encodeur afin de rendre la séquence encodé plus ou moins sensible à la perte de paquet [84]. Ce paramètre prend des valeurs qui changent entre 0,053 et 0,4417 selon le BR et le FR. Nous avons choisi cinq valeurs pour lui.
- Le taux de perte de paquet (LR) : le simulateur peut éliminer des paquets aléatoirement et uniformément pour satisfaire un taux de perte choisi. Ce paramètre prend cinq valeurs 0, 1, 2, 4, et 8%.
- Le nombre de paquets consécutivement perdus (CLP) : nous avons choisi d'éliminer des paquets à la rafale de 1 à 5 paquets.

Si nous choisissons de considérer toutes les combinaisons de ces valeurs de paramètres, nous devons tenir compte de 4x4x5x5x5 = 2000 combinaisons différentes. Ceci nécessiterait énormément de travail pour évaluer leurs qualités. Mais heureusement, le RN n'a pas besoin de toutes ces combinaisons pour "apprendre" le problème. Son rôle est d'interpoler la qualité des parties manquées de cet espace d'entrée potentiellement grand. Par conséquent, nous avons choisi de donner des valeurs par défaut et de composer différentes combinaisons en changeant seulement deux paramètres à la fois. Ceci a mené à 94 combinaisons (comme montré dans le tableau 6.2). Nous verrons prochainement que ce nombre est suffisant pour former et tester le RN.

1.4.2 Résultats

Nous avons employé la méthode DCR pour évaluer la qualité de ces 94 échantillons en utilisant l'échelle de notes à 9 points et en se conformant aux recommandations de l'UIT-R [67]. Nous avons invité 20 personnes à réaliser l'essai subjectif. Ensuite, des analyses statistiques ont été exécutées (voir la section 1.2.3 pour des détails) et nous avons rejeté les notes de deux sujets.

1.4.2.1 L'apprentissage de réseaux de neurones

Nous avons employé l'architecture dite *feedforward* à trois couches. Le nombre de neurones dans la couche d'entrée est égal au nombre de paramètres choisis (cinq). Il y a seulement une seule sortie du RN, qui est la valeur du MOS correspondant. L'algorithme d'apprentissage pour l'ANN est le célèbre "Backpropagation". Pour le RNN, nous avons employé l'algorithme proposé par Gelenbe [47] ainsi que d'autres algorithmes que nous avons développés. Nous avons remarqué qu'à précision identique, le nombre d'itérations exigés par ces nouveaux algorithmes est plus faible que pour l'algorithme de base (le gradient). Le nombre de neurones cachés est de 5 pour le RNN et de 4 pour l'ANN.

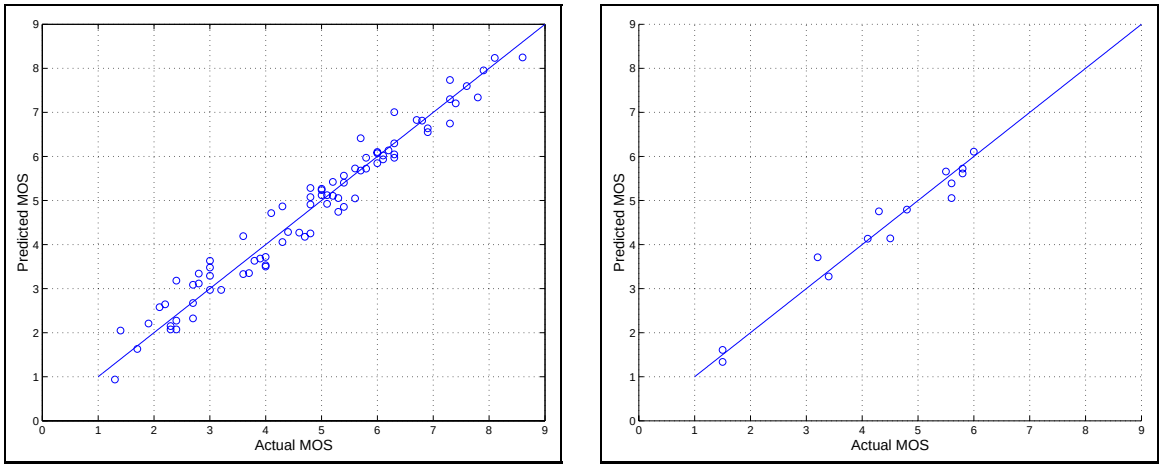
Après la mise en œuvre de l'expérience de MOS pour les 94 échantillons, nous avons divisé notre base de données (BD) en deux parties : une contenant 80 échantillons (les premiers dans le tableau 6.2) pour entraîner les RN, et l'autre contenant 14 autres échantillons pour la phase de validation. Après l'apprentissage de deux RN, nous comparons les valeurs réelles de la qualité contre celles prédites par les RN. Nous avons obtenu un facteur de corrélation de 0,9801, et une erreur quadratique moyenne de 0,108 ; c.-à-d., le RN s'adapte tout à fait bien à la manière dont les humains ont évalué la qualité de la vidéo. Le résultat est montré sur les figures 6.3 et 6(a). Il faut noter que pour le nombre

optimal de neurones cachés pour un ANN et un RNN, les précisions obtenues sont presque les mêmes. Cependant, un ANN peut être surentraîné facilement. Dans ce cas, l'ANN ne généralise pas bien pour les échantillons non vus pendant la phase d'apprentissage.

1.4.2.2 Comportement de réseaux de neurones

Afin de répondre à la question : "À quel point peut-on se fier au RN ?", nous avons appliqué la BD de test qui contient les échantillons qui n'ont jamais été vus pendant le processus d'apprentissage. Les résultats étaient : un coefficient de corrélation de 0,9821 et une erreur quadratique moyenne de 0,070. De nouveau les performances de RN sont excellentes, comme on peut l'observer dans la figure 6.4. Nous montrons également des nuages de points pour ce cas dans la figure 6(b).

En observant les figures 6.3, 6.4, et 6, on constate que notre méthode peut-être utilisée par des réseaux avec un comportement dynamique. Une fois que le RN est bien formé, il peut interpoler les résultats même si les entrées ne sont pas nécessairement des échantillons employés pour le former. En d'autres termes, le RN peut imiter la façon dont les sujets humains peuvent estimer la qualité subjective de la vidéo.



(a) DB d'apprentissage (b) DB de teste

FIG. 6 – *La corrélation entre les valeurs réelles et prévues de MOS pour les deux BD.*

1.4.3 Les performances des autres mesures de la qualité

Les mesures de l'évaluation de la qualité de la vidéo les plus connues sont la métrique ITS et les métriques de l'EPFL. En utilisant la première, le coefficient de corrélation obtenu avec des résultats subjectifs lorsqu'elle a été employée pour évaluer quelques scènes vidéos déformés sans tenir compte de paramètres de réseau (voir la figure 6.6) est environ 94% comme rapporté dans [143]. Cependant, quand cette métrique est employée pour évaluer la qualité en changeant seulement le débit, sa performance devient très mauvaise. Ceci peut être vu aux figures 6.9 et 6.13. En outre, même si cette mesure a été conçue fondamentalement pour évaluer la qualité de la vidéo en fonction des déformations de codage, elle ne peut pas évaluer la qualité en changeant la variable de codage la plus fondamentale, à savoir le débit. Plus de détails au sujet de cette mesure sont donnés dans [138, 142, 151].

Les autres métriques considérées ont été développées à l'École Polytechnique Fédérale de Lausanne (EPFL). Il s'agit du *Moving Picture Quality Metric* (MPQM), du *Color MPQM* (CMPQM), et du *Normalisation Video Fidelity Metric* (NVFM) [135]. Même si ces métriques sont conçues pour

l'évaluation de séquences vidéo de haute qualité comme indiqué dans [135, p. 90], à partir des résultats obtenus (de la figure 6.7 et les figures suivantes) nous pouvons voir que ce n'est pas toujours vrai : ces métriques ne peuvent pas évaluer la qualité en présence de la perte. Logiquement, la qualité doit devenir meilleure quand on augmente le débit (ceci réduit les artifices de codage) mais s'il y a des pertes, les résultats obtenus par le MPQM affirment le contraire : la qualité devient mauvaise avec l'augmentation de débit.

Un autre inconvénient de ces métriques est que la complexité des calculs est trop importante [135, p. 88] ce qui peut limiter leur utilisation pour l'évaluation en temps réel du multimédia dans l'Internet. A cause de cet inconvénient, les auteurs de ces métriques n'ont pu évaluer la qualité que pour seulement les 32 premières images (environ 1 seconde de la vidéo). De même, comme la métrique ITS, le MPQM et le CMPQM ne peuvent pas fonctionner correctement pour de bas débits, cf. les figures 6.8, 6.10, 6.12 et 6.13. Comme nous pouvons voir, le MPQM et le CMPQM donnent une note de 3 au lieu de 1 sur l'échelle de la qualité à 5 points. La métrique NVFM donne de meilleurs résultats pour quelques séquences, mais elle échoue également pour d'autres ; comme représenté sur la figure 6.11, elle donne une note de 2 au lieu de 1. Concernant la corrélation avec les essais subjectifs de qualité, elle n'est pas très bonne comme nous pouvons voir à la figure 6.13. Malheureusement, il n'y a aucune valeur numérique disponible pour le coefficient de corrélation avec ces mesures.

1.5 Études des effets de paramètres sur la qualité

Dans cette section, notre but est d'étudier et d'analyser l'impact des paramètres influant la qualité pour les deux types de médias considérés. Pour la parole, les paramètres considérés sont le codec utilisé, le taux de perte (LR), le nombre de paquets consécutivement perdus (CLP) et l'intervalle de mise en paquet (PI). De même, pour la vidéo, nous considérons le débit (BR), le débit d'images (FR), le rapport d'intra-blocs aux inter-blocs (RA), le LR et le CLP. Nous employons dans cette section un RNN pour mesurer la qualité de la parole et de vidéo en fonction des ces paramètres.

1.5.1 Impact des paramètres sur la qualité de la parole

Pour illustrer l'impact des paramètres sur la qualité de la parole, nous avons dessiné un ensemble de figures 3D. Pour produire les données nécessaires pour dessiner ces figures, nous avons utilisé un RNN qualifié et examiné par la BD "espagnole" qui est décrite et évaluée en section 1.3. Nous avons fait les mêmes tests en utilisant la BD "arabe", mais les résultats étaient approximativement identiques, donc, nous avons omis les figures résultantes car ils n'apporteront peu d'informations supplémentaire à notre étude.

Pour toutes les figures représentées dans la figure 7.3.4, sur le côté gauche, nous montrons la variation de la qualité en fonction de LR et de CLP pour les différents codecs et en fixant PI à 20ms ; sur le côté droit, nous montrons la variation de la qualité en fonction de LR et de PI pour les différents codecs et en fixant CLP à 1. Dans la figure 7.2 sur le côté gauche, nous montrons la variation de la qualité en fonction de CLP et de PI en fixant le LR à 5% ; sur le côté droit, nous faisons la même chose mais pour le LR à 10%. Nous montrons également dans la figure 7.3 la variation de la qualité en fonction de LR et des différents codecs pour PI de 20ms et CLP de 2.

1.5.1.1 Taux de perte (LR)

Comme représenté sur toutes les figures, l'effet le plus dominant sur la qualité est dû au LR. La qualité se dégrade sévèrement quand LR augmente et en fixant les valeurs des autres paramètres. Comme représenté sur la figure 7.1(a), pour le codec PCM et PI=20ms, la qualité se dégrade de 3,7 à 1,5 (environ 2,2 points) et de 4,1 à 2,2 (environ 1,9 points) lorsque le LR augmente de 5 à 40% pour CLP de 1 et de 5 respectivement. Les résultats sont qualitativement les mêmes pour les autres codecs.

De la figure 7.2, nous pouvons voir que les effets de CLP et de PI sont plus importants dans la situation où LR = 10% que dans la situation où LR=5%. Nous pouvons obtenir la même qualité en changeant ces deux paramètres si le LR change. Par exemple, pour CLP=2 et en utilisant le codec ADPCM, la qualité lorsque LR=5% et PI=80 ms est identique que lorsque LR=10% et PI=20ms (environ 3 points). La même observation est pour le CLP. De plus, si nous avons des états faibles de réseau, en changeant le codec, la qualité peut être améliorée. On observe une différence de 0,5 points en général en changeant d'ADPCM en PCM.

1.5.1.2 Paquets perdus consécutifs (CLP)

Comme représenté sur les figures 7.2, 7.1(a), 7.1(c) et 7.1(e) l'effet d'augmenter CLP est bénéfique à la qualité de la parole en maintenant les autres paramètres constants. On constate une amélioration absolue de 0,5, de 0,7, et de 0,5 pour les codecs PCM, ADPCM et GSM lorsque CLP augmente de 1 à 5 dans les figures montrées sur le côté gauche sur la figure 7.2, où LR=5%. Tandis que, nous pouvons voir une amélioration de 0,8, 0,7, 0,6 lorsque LR=10%. Cependant, nous pouvons voir dans la figure 7.2, qu'il n'y a aucune amélioration sur la qualité lorsque PI est important (environ 60 ms). Ainsi, l'amélioration devient plus importante pour de plus petites valeurs de PI.

Ce phénomène s'explique ainsi : lorsqu'on fixe tous les autres paramètres y compris le LR et qu'on fait augmenter le CLP, le nombre d'événements de perte diminue. Et donc, le nombre de mots déformés dans le discours diminue.

1.5.1.3 Intervalle de mise en paquet (PI)

Le PI représente combien de trames de parole il y en a dans chaque paquet. Comme pour le CLP, l'augmentation du PI tend à améliorer la qualité. Ceci est clairement montré dans les figures 7.3.4 et 7.2. Cependant, pour une plus petite valeur de LR, il n'y a aucune amélioration apparente alors que l'effet d'augmenter le PI devient plus important pour de fortes valeurs du LR et de petites valeurs du CLP. Pour une valeur plus élevée de CLP, il n'y a aucun effet du PI sur la qualité. L'effet de PI dépend du codec utilisé et de la valeur du LR. Par exemple, pour le PCM, lorsque LR=5%, il y a seulement une amélioration de 0,2 contre 0,4 pour LR=10%. Cependant, en utilisant le GSM, les améliorations sont 0,3 et 0,6. En outre, quand les conditions de réseau sont mauvaises, nous pouvons améliorer la qualité en augmentant le PI. Par exemple, en utilisant le codec ADPCM et lorsque CLP=1, la qualité est la même (3 points) pour le cas où LR=5% et PI=20ms et le cas où LR=10% et PI=40 ms. Ces observations sont importantes, parce qu'à la différence des paramètres de réseau (LR et CLP), qui ne pourraient pas être modifiés excepté en employant le FEC ou en améliorant l'infrastructure du réseau, ce paramètre peut être modifié pour améliorer la qualité. Le résultat que nous obtenons ici est en accord avec celui dans [28, 61]. Cependant, dans [28, 61] on n'explique pas pourquoi ceci se produit.

1.5.1.4 L'effet de codecs

Il est évidant que les codecs de la parole donnent différentes valeurs de qualité subjective, selon l'algorithme de codage utilisé. Pour pas de déformation de réseau (i.e., pour LR=0), la qualité subjective obtenue est 4,6, 4,1 et 3,7 dans la langue arabe en utilisant des codecs PCM, ADPCM et GSM respectivement. En langue espagnole, les résultats obtenus sont 4,7, 4,2 et 3,8. (référez-vous à la figure 7.3). Dans la littérature, pour la langue anglaise, les valeurs de MOS pour ces codecs sont respectivement 4,4, 4,1 et 3,6 pour l'anglais [60, 79].

Nous pouvons voir également à la figure 7.2 que les codecs considérés ne tolèrent pas la perte de la même manière. Ainsi, pour les mêmes conditions de réseau, en changeant le codec, nous pouvons obtenir une meilleure qualité. Par exemple, il y a une amélioration de 0,6 en changeant de GSM en PCM pour les valeurs de LR à 5% et à 10%.

1.5.2 Impact des paramètres sur la qualité de la vidéo

Dans cette section, nous étudions l'effet des cinq paramètres (BR, LR, CLP, FR et RA) sur la qualité de la vidéo temps réel. Les valeurs du MOS ont été calculées en fonction des valeurs de tous les cinq paramètres en employant un RNN qualifié en utilisant la BD de qualité décrite et évaluée en section 1.4.

1.5.2.1 Le débit (BR)

Le débit a une grande influence sur la qualité (cf. les figures 7.4 ... 7.7). Dans la figure 7.4, en variant le BR de 0,15 à 0,8, les valeurs de MOS obtenues par le RNN changent de 3 à 6,7 (pour FR=6, LR=0%, CLP=2, et RA=0,1) et de 3,7 à 7,7 lorsque le FR est fixé à 30. L'amélioration de la qualité est significative pour de petites valeurs de LR et diminue pour de grandes valeurs de LR. De la figure 7.7, pour des valeurs plus élevées de BR, il n'y a aucun effet des variations du paramètre RA sur la qualité ; cependant, son effet augmente et devient bénéfique pour des valeurs plus basses de BR.

Lorsque l'encodeur doit diminuer le BR d'un flux donné, il augmente le paramètre de quantization afin de compresser davantage les images. Ce processus augmente les déformations du flux encodé et par conséquent augmente la déformation qui devient plus apparente par des humains. Ce paramètre est étudié dans [153, 102] ; cependant, l'interaction avec les autres paramètres n'a pas été étudiée.

1.5.2.2 Débit d'images (FR)

L'effet de ce paramètre sur la qualité n'est pas aussi significatif que dans les cas de BR ou de LR (voir ci-dessous). Ceci est clairement montré dans les figures 7.4, 7.8, 7.9 et 7.10. Pour BR=0,8, LR=0%, CLP=2, et RA=0,1 ; une amélioration de 6,6 jusqu'à 7,6 est réalisée pour des variations de FR de 6 à 30. Nous pouvons également noter que l'amélioration de la qualité pour de valeurs de FR supérieures à 16 est négligeable, comme déjà observé dans des travaux précédents [119, 53]. En diminuant le BR, l'effet de FR devient plus petit (voir la figure 7.14).

1.5.2.3 Taux de perte (LR)

L'effet de LR sur la qualité visuelle était le but principal de plusieurs études précédentes car il est le paramètre de réseau le plus important [23, 53, 58]. Comme représenté sur la figure 7.5, la qualité diminue lorsqu'on augmente le LR de 0 jusqu'à 10%. Mais, cela dépend de valeurs d'autres paramètres : environ 1,3 pour BR=0,15, tandis que 5,0 pour BR=0,8. Comme représenté sur la figure 7.8, la détérioration absolue est d'environ 3,8 pour FR=6, mais de 4,6 pour FR=30. La variation de la qualité est d'environ 4,4 pour RA=0,5 et de 3,8 pour RA=0,05, comme représenté sur la figure 7.12.

Le fait qu'il n'y ait aucun changement sur la qualité en fonction de CLP en fixant le LR à zéro (les figures 7.6, 7.9 et 7.11) montre que le RNN capte avec précision et fiabilité les caractéristiques de la qualité en fonction des paramètres. On pourrait penser que c'est dû à de grandes valeurs de BR et de FR, mais nous avons changé tous ces paramètres et nous avons obtenu le même comportement. Dans le cas où on emploie un ANN, ce n'est pas toujours facile d'obtenir ce résultat (voir la figure 4.9(b)).

1.5.2.4 Le nombre de paquets consécutivement perdus (CLP)

Comme le résultat obtenu pour ce paramètre peut sembler étrange, nous allons d'abord expliquer ses effets. En maintenant tous les autres paramètres constants et en augmentant la valeur de CLP, la distance entre deux événements de perte consécutives augmente. Ceci a deux conséquences. D'abord, le nombre de ces événements diminue et par conséquent la détérioration des images devient plus petite (c.-à-d. moins d'images sont partiellement déformées par la perte). En second lieu, comme il est connu et également montrée dans notre analyse précédente de FR, l'œil est moins sensible à des valeurs plus élevées de FR. D'ailleurs, pour chaque paquet perdu, les blocs passés (une partie de l'image) sont

encore montrés sur l'écran comme résilience d'erreur. Par conséquent, de plus grandes valeurs de CLP peuvent présenter des détériorations sur un plus petit nombre d'images. Ceci est équivalent de petites valeurs de LR et des valeurs légèrement plus basses de FR. Comme on a précédemment montré que l'effet du LR est beaucoup plus grand que qui celui du FR, l'effet des valeurs plus élevées du CLP sur la qualité est bénéfique. Ce résultat est en accord avec ce qui est obtenu dans [58], bien que les auteurs n'aient pas expliqué les raisons pour lesquelles cet effet se produit.

1.5.2.5 Rapport Intra à Inter (RA)

Le RA est le rapport des blocs encodés en mode intra à ceux encodé en mode inter, par conséquent on s'attend à ce que ce paramètre ait un effet bénéfique lorsqu'il augmente. Ceci est clairement montré dans les figues 7.7, 7.10, 7.12 et 7.13. Son effet est plus important que celui du FR, et encore pour des valeurs plus basses de BR. Ceci est montré sur la figure 7.7, où pour BR=0,15 la qualité augmente de 3,4 jusqu'à 4,6. Cette amélioration est relativement importante lorsque la bande passante de réseau est petite. Pour un plus grand BR, l'effet est négligeable.

Les deux paramètres LR et RA sont connexes, cf. la figure 7.12. Pour un plus petit LR, il y a une amélioration sur la qualité pour l'augmentation en RA. Cependant, cette amélioration disparaît lorsque le LR augmente. Ceci signifie que nous pouvons obtenir une meilleure qualité visuelle quand la bande passante disponible (BR) est petite et pour de plus petites valeurs de LR en augmentant la valeur de RA.

1.6 Un nouveau mécanisme de contrôle de débit

On sais que le protocole TCP n'est pas approprié au transport des flux multimédias temps réel. Le protocole le plus approprié pour ces derniers est UDP, associé à d'autres protocoles (RTCP, RTP, etc.). En utilisant UDP, qui est non fiable dans le sens que l'expéditeur ne peut pas garantir que les paquets arriveront à la destination (dans l'ordre correct ou sans pertes), la source envoie des paquets au taux qu'elle veut sans faire attention si ceci encombrera le réseau ou pas.

Permettre aux sources d'envoyer au taux qu'elles ont besoin peut engendrer des conséquences graves (par exemple, congestion de réseau). Ainsi, l'utilisation des protocoles pour régulariser le débit de sources UDP a les objectives suivants:

- éviter d'encombrer le réseau dû à ses ressources limitées (bandes passante, capacité de routeurs, ...).
- fournir l'équité à d'autres flux TCP. En réponse aux pertes dans le réseau, les flux TCP partageant les mêmes liens encombrés avec des flux UDP réduisent leurs taux de transmission. Cependant, sans aucune réduction de débit de sources UDP, les flux TCP recevraient une plus petite part de la bande passante totale. Par conséquent, les sources UDP doivent être dotées de mécanismes de contrôle adaptatifs qui visent non seulement à réduire les taux de perte et améliorent l'utilisation des bandes passantes mais également à se comporter d'une manière plus juste envers les flux TCP partageant les même liens (un tel protocole s'appelle *TCP-Friendly*). Ce terme indique que si un flux TCP et un autre UDP adaptatif ont les même comportements, les deux devraient recevoir les mêmes valeurs de bande passante [108].
- les mécanismes de contrôle de débit sont employés pour maximiser l'utilisation des ressources du réseau par toutes les sources.

La majorité des protocoles existants visent à atteindre tous ces buts. Ils sont généralement référés comme *TCP-Friendly Rate Control* (TFRC). Ces protocoles emploient les mêmes mécanismes existants dans TCP pour régler le taux de transmission de sources UDP. Il est clair que la performance de ces protocoles peut mener, dans quelques situations, aux raisons pour lesquelles TCP n'est pas employé pour transporter des flux multimédias temps réel. Une source TCP-Friendly peut adapter son

débit selon les états de réseau. Ceci signifie qu'il enverra moins de données qu'une application UDP normale. Cela signifie également qu'elle peut réduire nettement la qualité perçue de ces flux [129]. Par conséquent, une fois que l'expéditeur est obligé de diminuer son débit, il doit prendre une décision sur la façon dont il le fait afin de maximiser la qualité perçue par les utilisateurs. Ceci est le but du mécanisme de contrôle que nous proposons ici.

La diversité des algorithmes de codage existants permet d'avoir un codec qui donne une meilleure qualité que les autres pour les mêmes conditions (l'état de réseau, le débit, etc.) comme nous avons montré dans la section précédente. En outre, pour le même encodeur, nous pouvons considérablement réduire l'utilisation de réseau (débit) en changeant quelques variables de codage (par exemple, le paramètre de quantification (*quantization parameter*) ou le débit d'images dans l'encodeur vidéo). Après des analyses et des études des flux de la parole et de la vidéo déformés par des plusieurs valeurs de paramètres de réseau et d'encodeur, ce deux observations nous ont menés à concevoir un nouveau protocole pour garantir la livraison de la meilleure qualité tout en maintenant le comportement de *TCP-friendly*.

Une autre motivation de notre travail a été l'établissement de quelques éléments pour la conception d'un certain genre de protocoles de réseau qui tiennent compte de la perception d'utilisateur de la qualité au lieu de baser ces protocoles sur des mesures passives de réseau. Jusqu'ici, cela n'a pas été possible, parce qu'il n'y avait aucun mécanisme qui puisse mesurer la qualité en temps réel sans avoir l'accès au signal original, en donnant des résultats qui se corrèlent bien avec la perception humaine. Notre outil répond jusqu'à certains degrés à ces exigences.

1.6.1 TFRC basé sur une équation (EB-TFRC)

Les motivations derrière notre choix d'EB-TFRC [41] parmi les autres protocoles existant sont :

- Il ne cherche pas agressivement la bande passante disponible, c'est-à-dire, il augmente le débit lentement en réponse à une diminution du taux de perte.
- Il ne réduit pas le débit par la moitié en réponse à une perte. Cependant, il le réduit par la moitié en réponse à plusieurs pertes successives.
- Le récepteur devrait rapporter les informations de *feedback* à l'expéditeur au moins une fois par temps d'aller-retour s'il a reçu des paquets dans cet intervalle.

Par conséquent, l'EB-TFRC convient aux applications multimédias en raison de la qualité perçue par les utilisateurs [108, 124, 126]. Comme beaucoup d'autres protocoles TFRC, l'EB-TFRC emploie une équation de contrôle qui donne explicitement le débit acceptable maximum en fonction des événements récents de perte. L'expéditeur adapte son taux d'envoi, guidé par cette équation de contrôle, en réponse à la rétroaction du récepteur.

1.6.2 Notre protocole de contrôle de débit

Nous proposons d'employer les RN que nous avons présentés dans les sections précédentes afin de décider sur la meilleure adaptation qu'un expéditeur pourrait employer lorsqu'il reçoit des informations de congestion du réseau. Nous prouvons que l'utilisation des RN aide à améliorer la perceptive qualité. Le nouveau protocole (parties ombragées sur la figure 7) se compose de trois parties :

1. La première est un TFRC qui calcule périodiquement le débit suggéré. La périodicité du calcul peut dépendre des recommandations de RTCP (toutes les 5 secondes) si l'adaptation à long terme est nécessaire ou peut se comporter comme EB-TFRC (chaque RTT) si une adaptation plus fine est nécessaire.
2. La deuxième partie est un RN travaillant du côté du récepteur, pour mesurer en temps réel la qualité en fonction des conditions de réseau et des paramètres de codage. Il envoie les informations, qui contiennent les statistiques du réseau aussi bien que le résultat du MOS évalué au

récepteur, à l'expéditeur (cf. la description de l'approche en section 1.2 et les validations pour la parole et la vidéo en sections 1.3 et 1.4.)

3. La troisième partie prend ces résultats et décide, basée sur des règles internes, des nouveaux paramètres à employer. Dans ce module, un ensemble de paramètres de contrôle doit être défini (voir ci-après pour une liste des paramètres possibles). L'impact de ces paramètres sur la qualité devrait être connu (voir la section 1.5 pour une directive). Le but de ce module est de choisir les meilleures valeurs des paramètres de contrôle pour maximiser la qualité comme si elle était perçue par l'utilisateur (représenté par le RN) en fonction de l'état du réseau mesuré dans la première partie. Un compromis entre la stabilité des paramètres et l'utilisation de la bande passante devrait être établi. Une fréquence trop lente de changement des paramètres aurait comme conséquence une mauvaise qualité pendant les périodes d'encombrement du réseau et une injustice envers les flux TCP partageant les même liens. De même, une fréquence trop élevée de changement pourrait avoir comme conséquence trop de fluctuations de la qualité perçue, ce qui ne convient pas les utilisateurs.

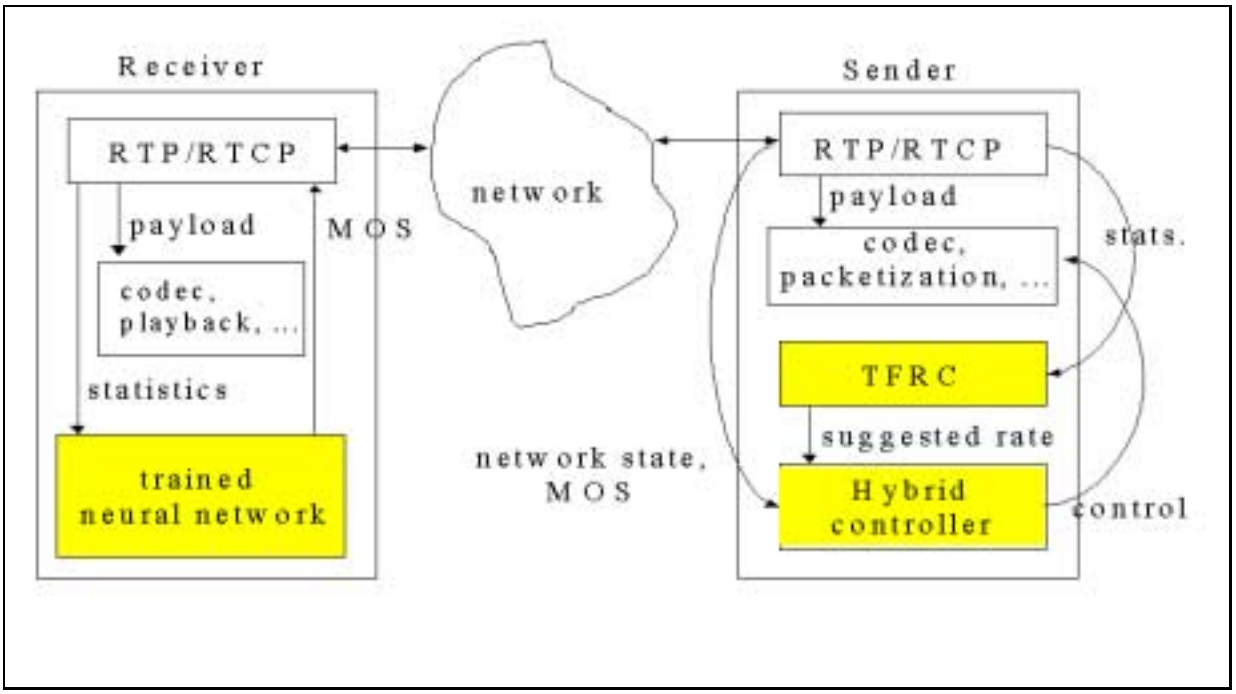


FIG. 7 – *L'architecture du mécanisme proposé de contrôle.*

1.6.2.1 Les paramètres de contrôle possibles

Les paramètres suivants peuvent participer au protocole de contrôle de débit :

- **Le débit (BR).** L'expéditeur peut changer dynamiquement le BR, quand un encodeur de débit variable (VBR) est employé, en utilisant un encodeur à plusieurs couches, et/ou en utilisant d'autres algorithmes de codage. Le codage multicouche est un des mécanismes les plus réussis dans la littérature [106, 107, 109, 110].
- **L'algorithme de codage.** Le type de codec ou certains des paramètres spécifiques de l'encodeur (fréquence d'échantillonnage, paramètres de quantification, etc.) peuvent être changés pour

adapter le débit nécessaire. Pour la même débit et le même état du réseau, on peut choisir le codec (ou changer ses paramètres) pour obtenir la meilleure qualité perçue.

- **Le débit d'images (FR).** Dans le cas de la vidéo, des expériences et des études sur HVS ont montrés que l'œil humain n'est pas trop sensible à la variation du FR supérieure à 16 images/s. Ainsi, en sautant quelques images, le débit peut adapter la valeur suggérée par TFRC et ne pas dégrader trop la qualité. Ce cas est intensivement étudiée dans la littérature car c'est une manière très efficace pour contrôler le débit [42, 127, 78, 76].
- ***Forward Error Correction* (FEC).** le FEC peut être employé pour se protéger contre la perte. Ainsi, en choisissant la quantité et le type de données superflues [20, 18, 19, 21, 22, 116], l'effet de la perte peut être réduit et par conséquent la qualité peut être améliorée. Cependant, trop de FEC augmente le BR global et le délai. Ainsi, un compromis devrait être trouvé.
- **Intervalle de mise en paquet.** Dans le cas de la parole, en changeant cet intervalle (et par conséquent la taille de paquet), le débit de transmission changera (comme suggéré par EB-TFRC, car il est directement proportionnel à la taille de paquet dans l'équation 8.1). Cependant, d'autre part, une taille trop grande de paquet augmentera le délai global, ce qui n'est pas tolérable dans la majorité des applications multimédias temps réel.

1.6.3 Résultats de simulation

Dans cette section, nous allons valider notre approche pour la parole et la vidéo.

1.6.3.1 Cas de la parole

Nous avons utilisé une pile de RTP/RTCP et une implantation EB-TFRC [5]. La topologie du réseau utilisé se compose d'un expéditeur, d'un routeur et d'un récepteur. Nous avons employé seulement un paramètre (le type de codec) pour illustrer l'intérêt de notre mécanisme.

Un développement local (logiciel de type "bypass" développé à l'INRIA Sophia Antipolis) dans le routeur simule un goulot d'étranglement qui se produit après une minute d'opération normale. En plus, des délais sont appliqués aux paquets passant par le routeur. Dans la figure 8, nous traçons le comportement de la bande passante calculée suggéré par TFRC et celle économisée du moment où le récepteur décide de changer du PCM (64Kbps) en GSM (13.2Kbps) en se basant sur une décision prise par l'expéditeur pour maintenir une bonne qualité. Nous pouvons voir que quand le goulot d'étranglement est détecté, l'expéditeur adapte son débit de transmission avec une valeur inférieure à celle de la bande passante disponible. Dans la figure 9, nous comparons les MOS mesurés du côté du récepteur quand l'expéditeur utilise notre mécanisme en changeant le codec et quand l'expéditeur suit aveuglement TFRC et continue à employer le codec de PCM dans les périodes de goulot d'étranglement. Clairement, le MOS est amélioré même en présence de congestion. D'ailleurs, notre mécanisme peut également être utile en absence de congestion, en décidant sur la combinaison des paramètres conduisant à la meilleure qualité. Enfin, en employant ce mécanisme, et pour maintenir un certain niveau de qualité, l'expéditeur peut décider d'employer la largeur de bande exacte exigée par l'application en l'absence de la congestion et de ne pas donner seulement une borne supérieure comme fait TFRC (cf. la première partie sur la figure 8).

1.6.3.2 Cas de la vidéo

En se basant sur les études expérimentales et de simulation d'EB-TFRC données dans [41], et basées dans l'équation 8.1, nous supposons pour des raisons de simplicité que pour différents états du réseau, le débit de transmission suggéré par l'EB-TFRC est donné dans la figure 10. Les fluctuations sont dues à l'effet des différentes variables dans cette équation. Ceci a pour but de tenir compte des variations du délai de bout en bout, la taille des paquets, la perte, le nombre de flux partageant les même liens, etc.

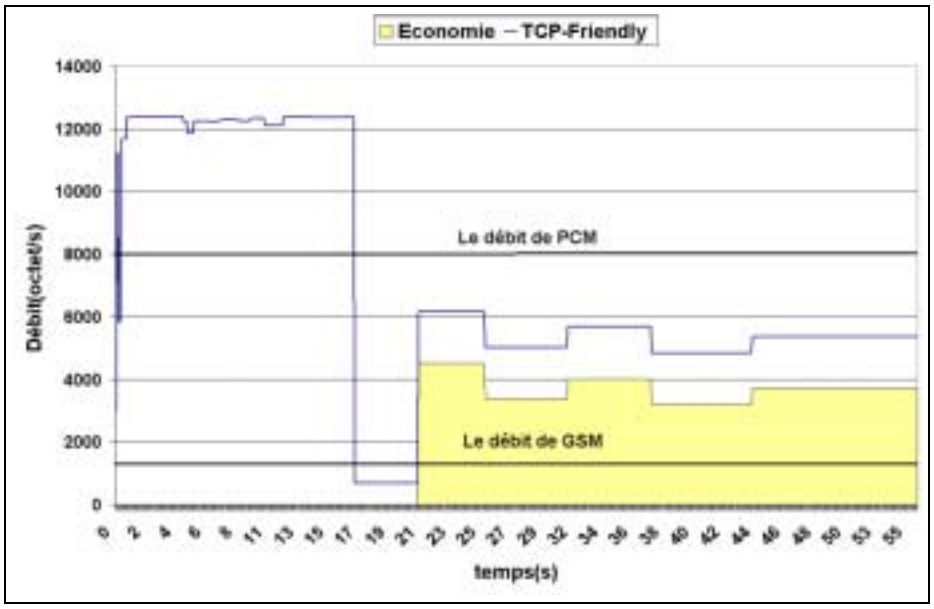


FIG. 8 – *Les débits suggérés par TFRC et l'économie obtenue en utilisant des règles de contrôle en changeant le codec dans le cas de la parole.*

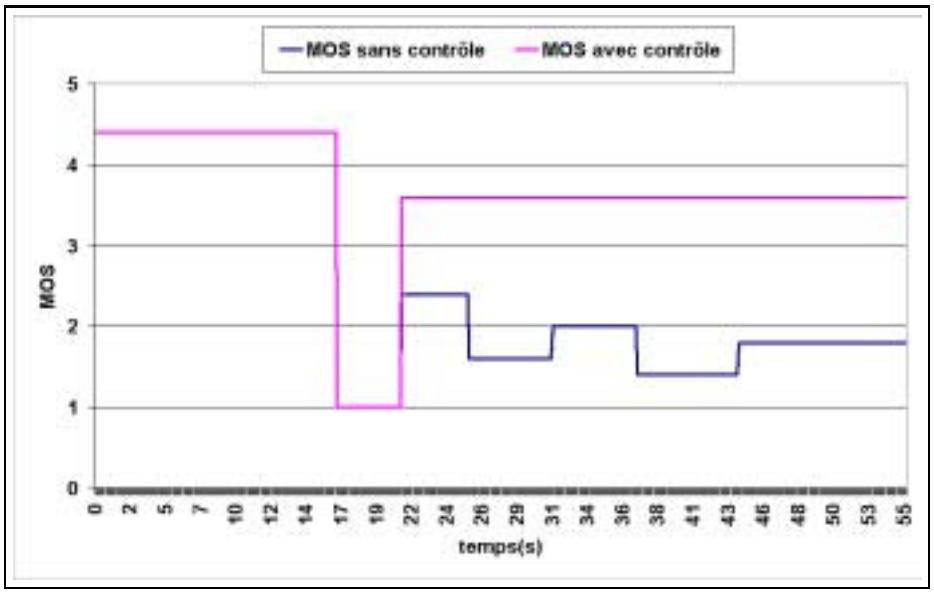


FIG. 9 – *Les valeurs de MOS avec et sans notre contrôleur en changeant le codec dans le cas de la parole.*

Comme nous avons précédemment mentionné, l'expéditeur ne devrait pas (ne peut pas dans certains cas) suivre les fluctuations de TFRC pour éviter de dégrader la qualité. Afin de clarifier l'idée, nous supposons également que l'expéditeur change son débit selon la ligne continue représentée dans la même figure. Nous avons choisi deux paramètres de contrôle afin de satisfaire les débits suggérés, à savoir le paramètre de quantification (QP) et le débit d'images (FR).

En changeant le QP, l'encodeur peut compresser le signal original pour produire le débit désiré (BR). En "sautant" quelques images du signal original, et par conséquent en changeant le FR, le BR peut également être changé pour satisfaire les débits exigés. Pour le cas où on emploie seulement le

QP en tant que variable de contrôle, nous avons fixé FR à 30 images/s et RA à 0,05. Dans les périodes encombrées, nous supposons qu'il y a une perte de 1% avec CLP de 1. De même, pour le FR, nous avons fixé tous les paramètres comme avant, excepté le BR et le FR. En ce concerne le PQ, il a été fixé pour produire la meilleure qualité. Nous avons employé le RNN qualifié (voir les sections 1.4 et 1.5) pour mesurer la qualité pour chaque cas. Nous montrons sur la figure 11 l'évaluation de la qualité pour les deux cas. Comme nous pouvons le voir, le changement du FR donne une meilleure qualité que si l'on permettait seulement le QP de changer.

L'amélioration de la qualité est plus importante dans le cas où le débit est très petit. L'explication de ce fait a déjà été mentionnée avant. Brièvement, en fixant le FR à 30 images/s et en changeant seulement le QP pour réduire le BR exigé, les déformations du signal encodé augmentent en raison des artifices de codage. D'autre part, en fixant le QP à sa meilleure valeur, et en réduisant seulement le FR, le niveau de déformation n'est pas si ennuyant sur la perception de l'utilisateur que dans l'autre cas. En effet les yeux humains sont moins sensibles au changement de FR qu'aux déformations de codage.

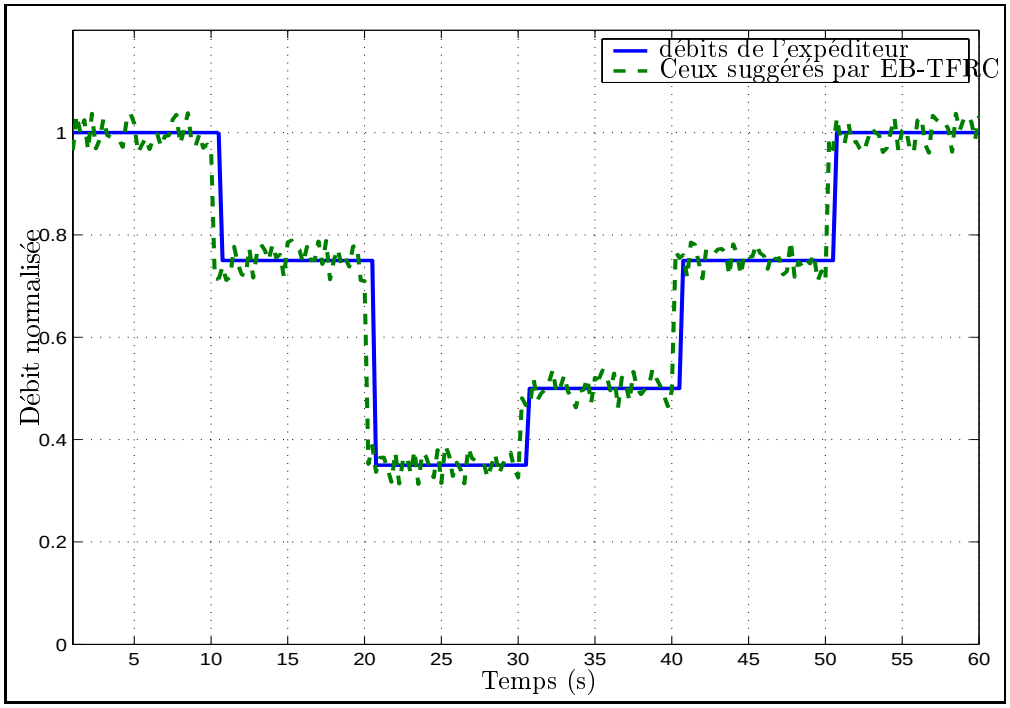


FIG. 10 – *Les débits suggérés par TFRC et ceux de l'expéditeur*

1.7 Une nouvelle méthode pour la prévision du trafic

Le problème de prévision du trafic a été abordé par plusieurs techniques : *least mean square filter* ou des techniques semblables [85], les modèles tels qu'AR, ARIMA, FARIMA [27, 121, 85], et la logique floue avec la régression [120, 27]. Il y a plusieurs propositions en utilisant les RN pour modéliser le processus du trafic [156, 157, 39, 88, 133]. Cependant, les propositions existantes se concentrent seulement sur les dépendances à courte portée et négligent les dépendances à longue portée. Ainsi, en examinant ces modèles sur de vraies traces durant plusieurs minutes ou heures, ils donnent de bonnes performances. Cependant, en les utilisant pour prévoir le trafic pour des jours ou des semaines, leur performance se dégrade d'une manière significative [56].

Notre but dans cette étude est de présenter un nouveau modèle pour la prévision du trafic qui fonctionne bien pour des réseaux IP et ATM. Cependant, nous l'avons examiné seulement sur de vraies

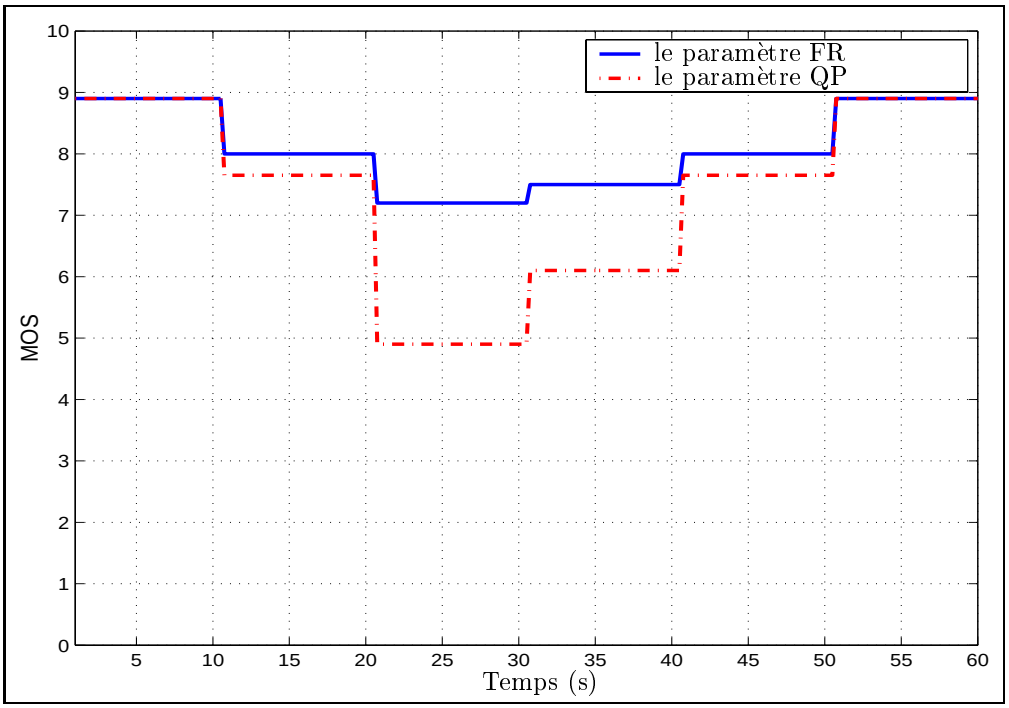


FIG. 11 – *Les valeurs de MOS en changeant FR et ceux en changeant QP.*

traces IP. L'avantage de notre modèle est qu'il se sert de l'information à longue portée aussi bien que de l'information à courte portée dans l'historique pour prévoir efficacement les futures arrivées du trafic. Ceci améliore la précision de prévision et par conséquent la performance globale des applications.

1.7.1 Notre méthode proposée

Le modèle proposé est représenté dans la figure 12. Pour construire un tel outil, trois étapes doivent être effectuées. Premièrement, une BD doit être construite à partir de vraies traces du trafic pendant une période suffisamment longue. Deuxièmement, une architecture et un algorithme d'apprentissage appropriés de RN doivent être choisis. Troisièmement, ce RN doit être entraîné et validé. De plus, une stratégie de "recyclage" doit être choisie (voir la section 1.7.2.5 pour des résultats expérimentaux concernant ce point).

Chacune des BD d'apprentissage et de test devrait se composer de 5 parties. La première est la fenêtre "aujourd'hui", qui se compose des valeurs $F(T) \ldots F(T-n)$, où $F(T)$ est la valeur courante du trafic et $F(T-n)$ représente la valeur à la $n^{\text{ème}}$ étape précédente. La deuxième partie est la fenêtre "jour précédent", qui se compose des valeurs $F_y(T) \ldots F_y(T-j)$, où $F_y(T)$ est la valeur du trafic le jour précédent au même instant que "aujourd'hui". De même, la troisième partie est la fenêtre "semaine précédente", qui se compose des valeurs $F_w(T) \ldots F_w(T-k)$, où $F_w(T)$ est la valeur du trafic dans la semaine passée, le même jour et au même instant que "aujourd'hui". La quatrième partie est le "jour" de la semaine et le "temps" d'aujourd'hui. Le "jour" prend de valeurs entre 0 et 6, et le "temps" prend de valeurs entre 0 et 24 heures. La partie finale est la prochaine valeur du trafic, $F(T+1)$, qui représente la sortie de RN. Toutes ces données devraient être normalisées entre 0 et 1. La fenêtre "aujourd'hui" représente l'information à courte portée du processus du trafic. Toutes les autres entrées représentent l'information à longue portée.

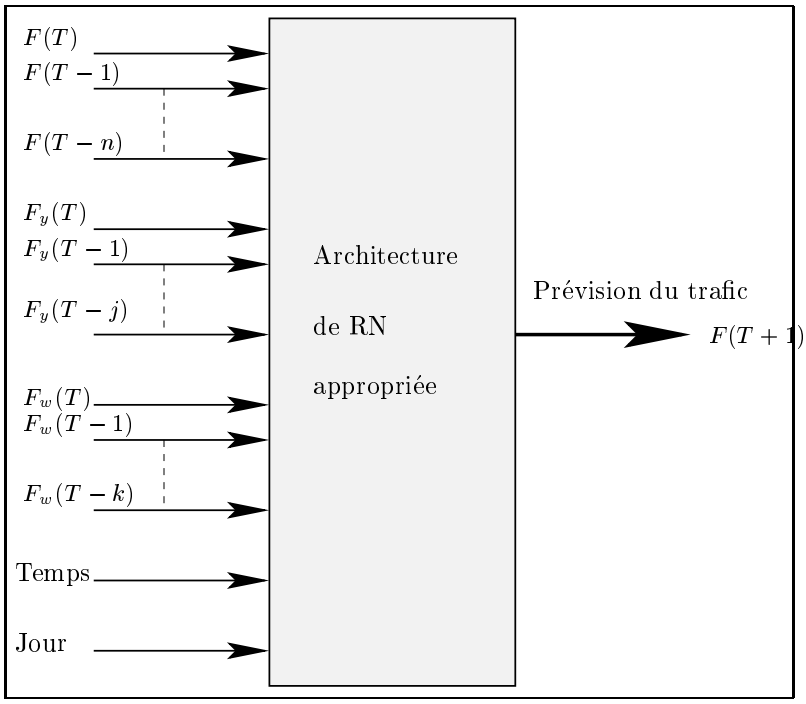


FIG. 12 – *Une représentation de boîte noire de notre outil pour prévoir en temps réel le futur trafic, où, $F(T)$ est le trafic actuel, $F(T-n)$ est celui à la $n^{ème}$ étape précédente. $F_y(T)$ est celui pour le même instant mais le jour précédent, $F_w(T)$ est celui dans la semaine précédente.*

1.7.2 Les résultats et les évaluations expérimentales du nouveau modèle

1.7.2.1 Vraies traces pour la formation

Nous avons employé pour toutes les expériences représentées ici des vraies traces du trafic de l'ENST de Bretagne (ENSTB) à Rennes pendant 6 mois (d'octobre 1999 à mars 2001). Il contient le trafic entrant et sortant du routeur principal reliant l'ENSTB au monde externe. Les traces sont prélevées et échantillonnées toutes les 5 minutes. Par conséquent, chaque échantillon représente la moyenne pendant les 5 dernières minutes. Nous montrons dans le tableau 9.1 une partie de ce trafic.

1.7.2.2 Essais expérimentaux pour identifier la meilleure longueur de chaque fenêtre

Pour identifier la meilleure architecture (les longueurs de chaque fenêtre, et le nombre de neurones cachés) pour le RN, nous avons effectué l'expérience suivante. En utilisant la trace d'ENSTB, nous avons créé un ensemble de BD pour former et tester différentes architectures de RN. Chaque BD est caractérisée par la longueur des fenêtres (voir tableau 9.2 pour un exemple) ; nous avons varié les longueurs respectives ("aujourd'hui", "jour précédent" et "semaine précédente") de 2 à 5, de 0 à 3, et de 0 à 3. Pour l'architecture de RN, nous avons varié le nombre de neurones cachés de 1 à 10. Ceci nous a donné 36 architectures différentes ; donc, nous avons construit 36 pairs de BD pour entraîner et valider ces RN. Pour chaque RN qualifié, nous l'avons utilisé pour prédire tous les échantillons de la semaine suivante. Nous avons rapporté le nombre de neurones cachés, la taille de chaque fenêtre et la valeur de MSE.

De ces expériences, nous avons constaté que les valeurs optimales sont 2 neurones cachés et les tailles de fenêtre sont 3 pour "aujourd'hui", et 2 pour "jour précédent", et 2 pour "semaine précédente". Pour ces valeurs optimales nous obtenons MSE=0,0041. Observez la bonne performance de l'outil dans sa configuration optimale.

1.7.2.3 Performance du RN pour prédire le trafic

Le RN qualifié est employé pour prédire le trafic pour les des deux semaines suivantes à l'instant courant. Dans la figure 13 nous montrons les trafics réels et prédits. (Notons que nous avons choisi 1 échantillon tous les 4 pour rendre la figure suffisamment claire.) Nous montrons également sur la figure 14 les trafics réels et prédits pour le troisième et le quatrième jour de la troisième semaine prédite. Pour cette prédiction nous avons obtenu un MSE de 0,0046. Ceci montre que la précision de notre modèle est très bonne et que le RN a bien appris le processus du trafic à partir de son historique. Dans la figure 15, nous montrons la différence entre le trafic réel et celui prédit pour cette expérience ; un histogramme est aussi montré dans la figure 16. Dans ces figures, nous pouvons voir qu'environ 78% des échantillons sont prédits avec une précision entre 0 et 0,05 ; environ 8% des échantillons sont prédits avec une précision entre 0,05 et 0,15, où le réel est supérieur que la prédiction. D'autre part, environ 10% des échantillons sont prédits avec une précision entre 0,05 et 0,15 (dans ces cas, la prédiction surévalue le trafic réel, ce qui est plutôt intéressant).

Dans la figure 15, nous pouvons voir que quelques échantillons sont prédits avec une précision entre 0,2 et 0,4. Ceci dû aux phénomènes connus sous le nom "*spikes*". Un échantillon *spiky* se produit quand la différence entre les échantillons futurs et courants est assez grande et qu'il n'y a aucune information dans les échantillons passés qui pourraient indiquer l'occurrence possible de la transition. Ce sont des phénomènes connus dans le processus du trafic [56]. Il faut noter, cependant, que les transitoires sont non prévisibles. Aucun outil de prévision ne peut les prévoir de façon aussi précises que pour les autres échantillons. Donc, nous recommandons que les *spikes* soient enlevés de la BD d'apprentissage, afin de ne pas dégrader la performance du RN en présentant des mauvaises informations pendant l'apprentissage. Heureusement, leur présence dans le vrai trafic n'est pas fréquente.

Si les *spikes* n'étaient pas pris en considération pour le calcul du MSE, la performance trouverait considérablement améliorée. Par exemple, en enlevant 124 échantillons *spikes* des 4200 échantillons des semaines prédites (en considérant simplement qu'un échantillon est *spiky* s'il diffère au moins 0,3 de l'échantillon passé), le MSE est amélioré de 0,0046 à 0,00163. La figure 17 montre la différence entre le trafic réel et prédit après la suppression de ces échantillons. Il est clair que la performance est meilleure qu'à la figure 15.

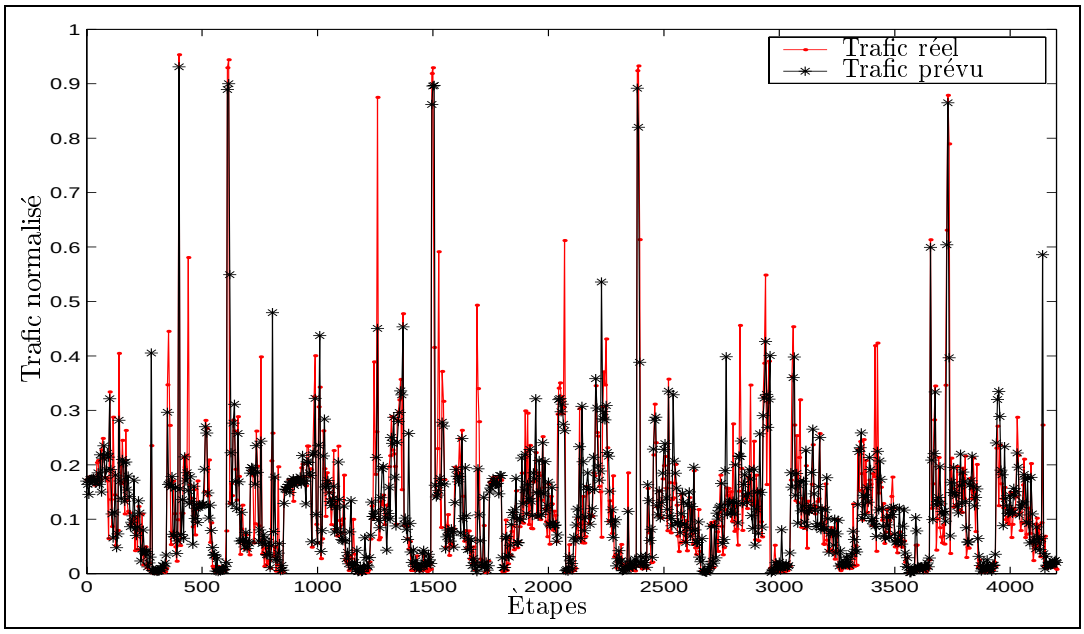


FIG. 13 – *Le trafic réel contre celui prévu pour la totalité de deux semaines suivantes.*

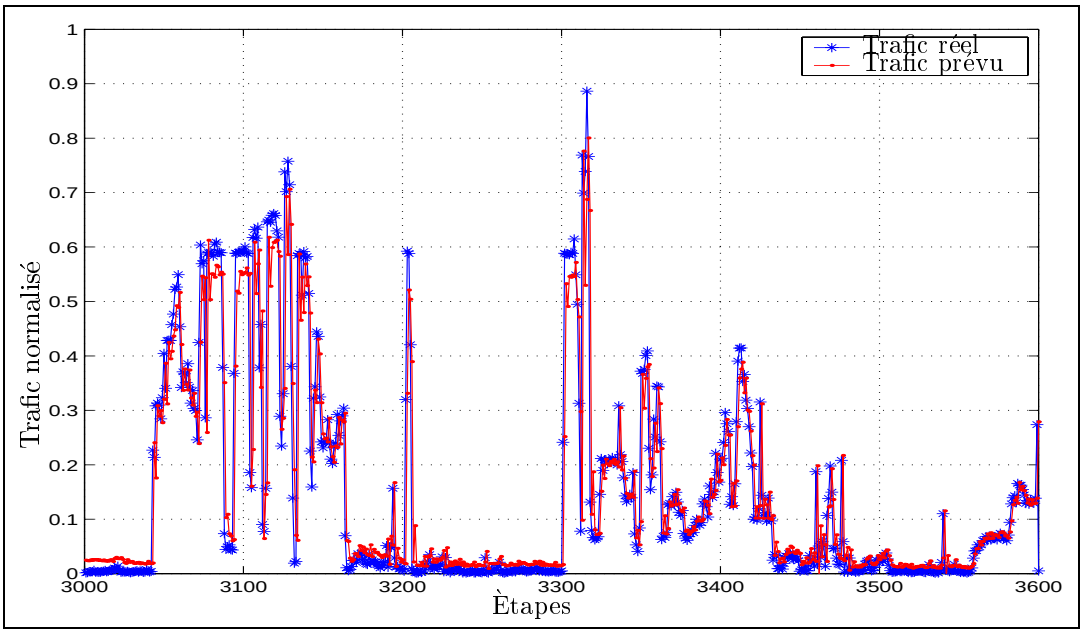


FIG. 14 – *Le trafic réel contre celui prévu pour les troisième et quatrièmes jours de la troisième semaine.*

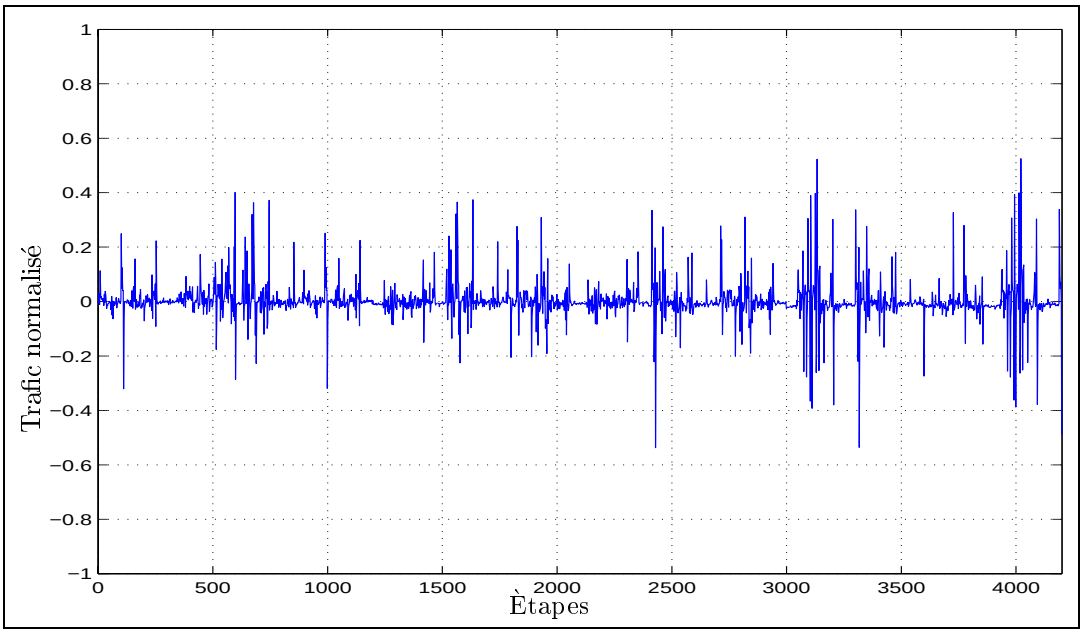


FIG. 15 – *La différence entre le trafic réel et prévu pour les suivantes deux semaines complètes.*

1.7.2.4 Plus d'une étape à l'avenir

Dans l'analyse précédente, nous avions utilisé le RN pour prédire seulement une étape future. Cependant, il est possible d'employer notre modèle pour prédire plus d'une étape à l'avenir. Il y a deux manières de le faire. La première est d'entraîner un RN ayant comme sorties les futures étapes et les mêmes entrées que notre modèle (temps, jour, fenêtres). Ce modèle est très naïf et sa performance est plutôt mauvaise.

Nous proposons une autre manière de procéder. Nous employons le même modèle de RN que nous

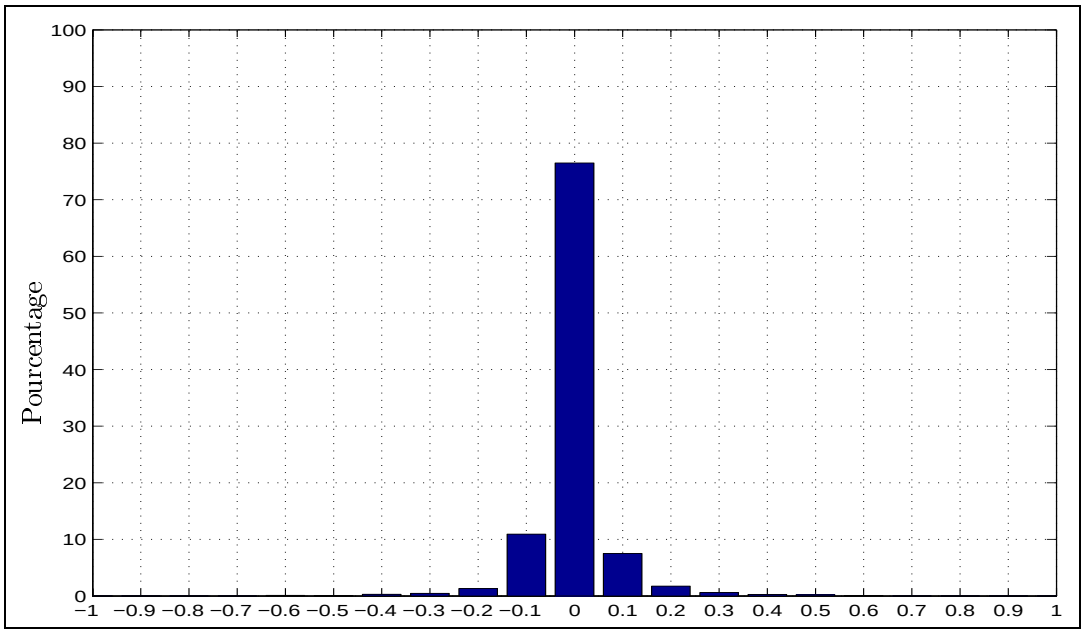


FIG. 16 – *Un histogramme de la distribution entre la différence de trafic réel et prévu*

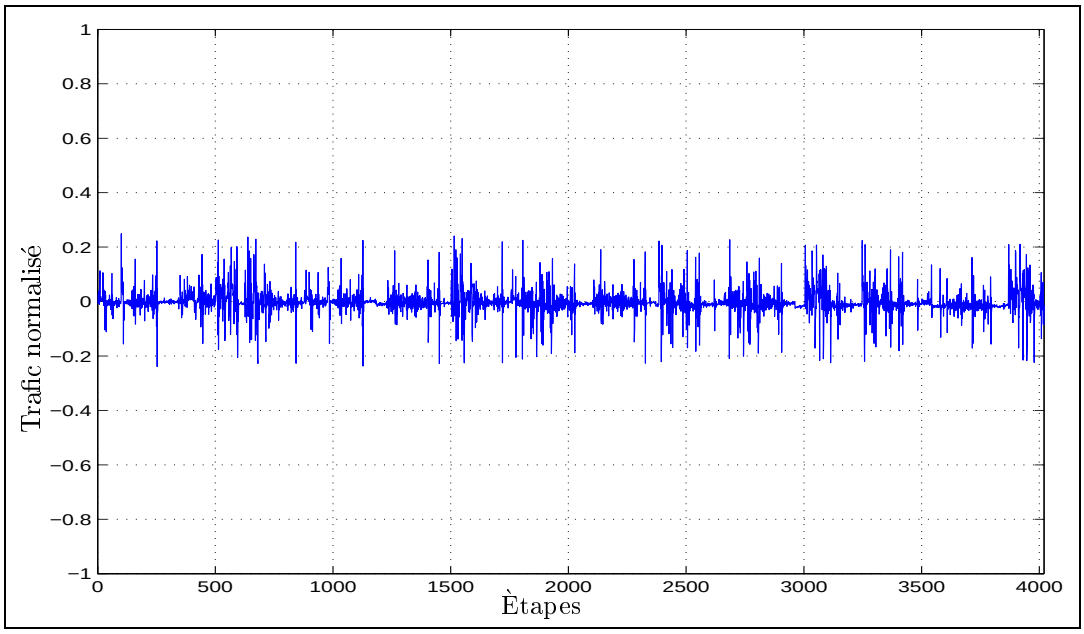


FIG. 17 – *La différence entre le trafic réel et prévu pour les suivantes deux semaines complètes une fois que les échantillons spiky sont retirés.*

avons décrit dans la section 1.7.2.3. Nous l'employons pour prédire la valeur de trafic de la prochaine étape. Ensuit, nous utilisons la valeur prédite en tant que la valeur actuelle du trafic, et l'utilisons pour prédire la deuxième étape future. Nous pouvons continuer récursivement à prédire plus de deux étapes de la même façon.

Nous avons employé cette méthode pour prédire la deuxième étape à venir pour le trafic des deux semaines suivantes en utilisant le même RN qualifié décrit dans la section 1.7.2.3 ; le MSE est 0,006. Nous montrons sur la figure 9.8, la différence entre le trafic réel et celui prédit. Comme nous pouvons

voir la performance de notre modèle est bonne.

1.7.2.5 Conditions pour recycler le RN

La performance d'un outil basé sur des RN peut se dégrader quand il y a des changements dans le système. Le RN peut tenir compte de cet effet. Ceci peut être fait par l'apprentissage en ligne ou l'apprentissage hors ligne. Quand nous avons utilisé le même RN qualifié pour prévoir la sixième semaine, nous avons obtenu MSE=0,012. Cela signifie que la performance se dégrade et que le RN doit être recyclé pour s'améliorer. Un recyclage aveugle du RN qualifié n'améliorera pas trop la performance. Les échantillons de recyclage devraient être choisis comme suit. L'enlèvement de tous les échantillons de *spikes* est une nécessité pour obtenir une meilleure performance. Nous devons choisir quelques échantillons qui donnent de bonnes prévisions et également quelques échantillons qui sont mal prédits dans le passé. Le choix des ces dernières permet au RN l'apprentissage des changements de processus du trafic. Le choix des échantillons bien prédits est important pour tenir au courant le RN de l'historique du processus du trafic.

Nous avons effectué deux expériences pour recycler le RN. Dans la première, nous l'avons entraîné aveuglement avec une BD contenant des *spikes*. Nous avons obtenu MSE=0,009 sur la BD de test. Une fois que nous enlevions les *spikes* de la BD d'apprentissage dans la deuxième expérience, le MSE s'est amélioré à 0,0054 sur la même BD de test.

Il est important de mentionner que dans [56], les auteurs ont argumenté du fait que ni le recyclage en ligne ni hors ligne n'améliorent la performance. Nous expériences ont montré que ceci est du aux échantillons dits *spikey*.

1.8 Nouveaux algorithmes d'apprentissage pour les réseaux de neurones aléatoires

Nous avons montré que les réseaux de neurones aléatoires (RNN) [45, 51, 46] se comportent bien pour l'évaluation de la qualité de multimédia. Cependant, dans le cas des grandes applications (un grand nombre de neurones et d'échantillons d'apprentissage), la phase d'apprentissage peut être longue. L'algorithme d'apprentissage dans le logiciel disponible mettant en œuvre les RNN est le *Gradient Descent* (GD) proposé par l'inventeur des RNN, Erol Gelenbe [47]. Cet algorithme est plutôt lent et peut exiger un nombre élevé d'itérations pour atteindre la performance voulue. Il souffre également du comportement de zigzag (l'erreur diminue à un minimum local, puis augmente encore, puis diminue à nouveau, et ainsi de suite).

Ce problème a orienté notre recherche afin de trouver de nouveaux algorithmes d'apprentissage pour les RNN. Nous avons développé deux nouveaux algorithmes d'apprentissage pour RNN. Le premier est inspiré de l'algorithme d'apprentissage de *Levenberg-Marquardt* (LM) pour un ANN [55]. Le second est inspiré d'un algorithme récemment proposé pour les ANN. On parle de LM avec *adaptive momentum* (LM-AM) [8]. Cet algorithme vise à surmonter certains des inconvénients de la méthode traditionnelle de LM pour les RN de type *feedforward*.

1.8.1 La méthode de *Levenberg-Marquardt* pour RNN

Dans l'algorithme de GD, la fonction d'erreur à minimiser doit être continue et différentiable par rapport à ses paramètres. Une fois que les gradients ont été calculés, l'approche la plus simple pour la minimisation de l'erreur est la méthode du gradient, où à chaque point, la direction de mise à jour du vecteur de paramètres est l'opposé du gradient en ce moment. Cette approche, due à sa généralité, a des inconvénients : (a) le comportement en zigzag, (b) la difficulté de choisir la valeur du paramètre dit taux d'apprentissage, (c) le fait qu'elle peut être lente : elle exige habituellement un grand nombre d'itérations.

Pour alléger ces problèmes, nous avons essayé d'appliquer la méthode de LM, qui est une approximation de la méthode de Newton pour un RNN. La technique LM est connue pour être le meilleur algorithme pour des problèmes d'optimisation appliqués à un ANN [55]. (Voir la partie anglaise, la section 10.3 en page 173, pour plus de détails et la dérivation analytique pour notre méthode).

1.8.2 LM avec *adaptive momentum* pour RNN

Quoique la méthode de LM soit l'un des algorithmes les plus puissants pour l'apprentissage des réseaux de feedforward, elle a plusieurs inconvénients. D'abord, la nécessité de calculer la matrice jacobienne et d'inverser la matrice hessienne avec des dimensions égales au nombre de poids du réseau. Cependant, ce problème est compensé par le plus grand taux de convergence de l'algorithme, qui devient quadratique lorsque l'on s'approche d'une solution. Un autre inconvénient est qu'on ne garantit pas la convergence vers le minimum global de la fonction de coût. Quand l'algorithme converge vers un minimum local, il n'est pas possible de s'échapper de celui-ci. Ainsi, dans [8], les auteurs ont proposé un algorithme que traite les minimums locaux avec une robustesse accrue et maintient toujours un taux de convergence rapide.

L'idée principale d'aider le LM à échapper d'un minimum local est d'inclure une terme de *momentum* que rajoute comme information la dérivée seconde dans le processus d'apprentissage et fournit des itérations dont la forme est semblable à celle dans la méthode gradient conjuguée (CG). La différence principale avec la méthode du CG est que les coefficients réglant les poids entre le gradient et le terme de *momentum* sont heuristiquement déterminés tandis que dans CG, ils le sont d'une manière adaptative. (Voir la partie anglaise, la section 10.4.1 en page 176, pour plus de détails et la dérivation pour notre méthode).

1.8.3 Évaluation des performances des algorithmes proposés

Dans cette section, nous fournissons les résultats expérimentaux obtenus pour évaluer les algorithmes proposés et nous les comparons avec ceux obtenus pour l'algorithme GD. Nous avons employé deux problèmes qui ont été déjà utilisés dans [1] pour la validation : le problème de XOR est également considéré. La description de ces problèmes est donnée dans la partie anglaise, la section 10.5.

Nous avons mis en application les algorithmes proposés en MATLAB. Pendant l'évaluation des ces algorithmes, nous avons noté que quand nous permettons des valeurs négatives pour les poids, nous obtenons de meilleurs résultats. Ainsi, pour traiter ce point, nous avons considéré trois procédures différentes quand un poids négatif est produit. Dans la première, chaque fois qu'il y a une valeur négative du poids, nous l'avons simplement mis à zéro. La seconde suit les proposition dans [87]. Pour s'assurer que les poids sont toujours positifs, l'équation suivante est employée $p^2 = w_{i,j}$. Dans le troisième cas, nous autorisons les poids négatifs. Par conséquent, nous fournissons une étude comparative pour ces cas. Dans l'analyse, les figures et les tableaux suivants, nous employons ces abréviations :

GD : la méthode du gradient, algorithme de base, comme proposé dans [1].

LM : l'algorithme de Levenberg-Marquardt sans contrainte sur les poids (ils peuvent être négatifs).

LM1 : l'algorithme de LM avec la contrainte de positivité sur les poids : une fois qu'une valeur négative de n'importe quel poids est produite, nous la mettons simplement à zéro.

LM2 : l'algorithme de LM avec la contrainte de positivité sur les poids, en employant $p^2 = w_{i,j}$ et en modifiant les dérivés des équations.

AM-LM : l'algorithme de LM avec *adaptative momentum* sans contrainte sur les poids.

1.8.3.1 ζ et dP paramètres pour AM-LM

Pour, l'algorithme AM-LM, il y a deux paramètres (ζ et dP) à choisir, au lieu d'un paramètre (η) pour la méthode GD. Dans [87], les auteurs ont étudié ce problème et ils ont déterminé par des

simulations les plages de valeur idéales de ces paramètres : $\zeta \in [0,85\,,0,95]$ et $dP \in [0,1\,,0,6]$. Nous allons montrer la performance de cet algorithme une fois adapté au RNN avec ces plages en l'utilisant pour apprendre le premier problème. Pour chaque paire de valeurs, nous avons compté le nombre de fois que l'algorithme converge à une valeur de MSE prédéfinie (5×10^{-5}). Pour chaque cas, tout le nombre d'essais est fixé à 100. Nous montrons sur la figure 18 les résultats. Comme nous pouvons voir de ces figures, les valeurs de ces paramètres affectent considérablement la performance de l'algorithme. La meilleure gamme pour dP est [0,4, 0,6]. Cependant, l'impact du ζ semble être aléatoire et pas facile à caractériser. Nous pouvons également noter que l'impact de dP est plus significatif que celui de ζ. Ainsi, nous suggérons que ce paramètre doit être fixé à 0,90 et qu'on ne change que la valeur de dP.

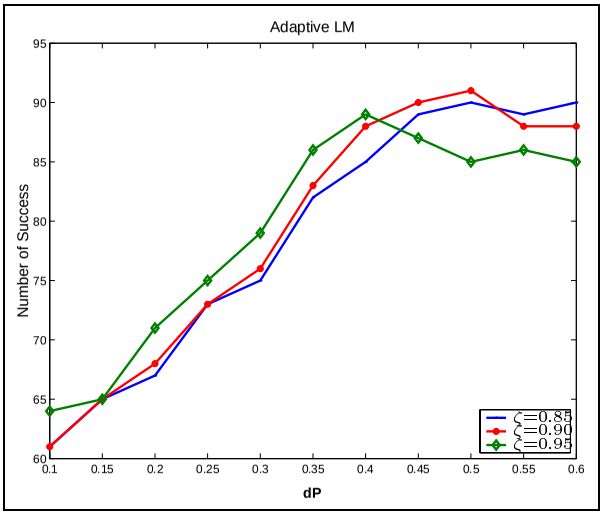


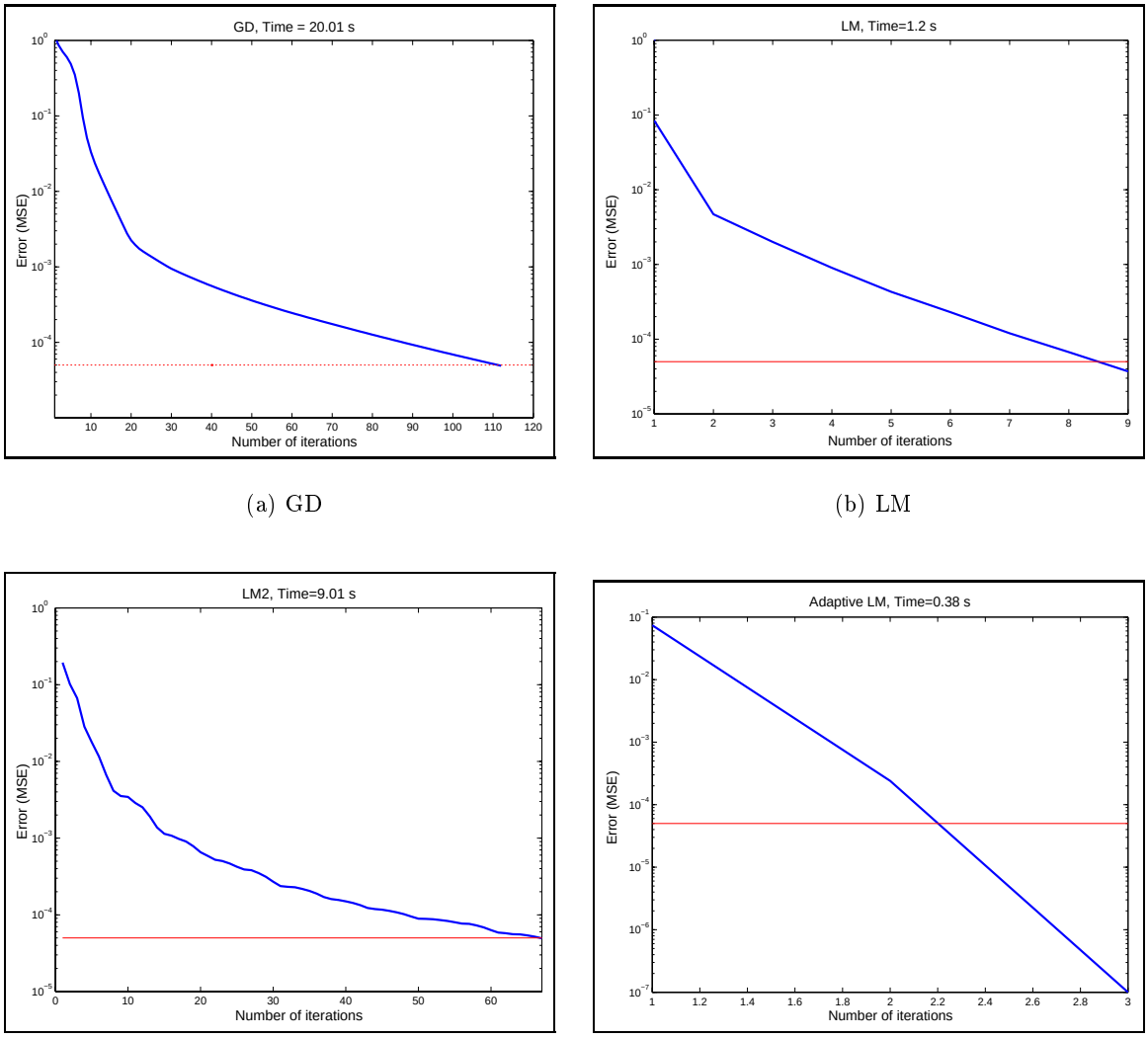
FIG. 18 – *L'effet de variables ζ et dP sur la performance de l'algorithme AM-LM pour un RNN. Les résultats sont pour le premier problème.*

1.8.3.2 Comparaison de la performance des algorithmes

Nous allons comparer la performance des différents algorithmes (GD, LM, LM1, LM2 et AM-LM) pour un RNN. Pour les deux problèmes définis précédemment, nous examinons la période de convergence et le nombre d'itérations requises pour atteindre une certaine valeur fixe de MSE. Les algorithmes ont été considérés convergeant si le MSE est de 5×10^{-5} (resp. 4×10^{-7}) pour le premiers (resp. le deuxième) problème. Pour l'algorithme GD, nous avons fixé η à 0,1. Pour l'algorithme LM, nous avons fixé β à 2 et la valeur initiale de μ à 0,1. Pour l'algorithme AM-LM, les valeurs de ζ et dP sont fixées à 0,90 et à 0,5 respectivement. Nous montrons dans l'ensemble des figures 19 et 20 la variation du MSE en fonction du nombre d'itérations et du temps de convergence pour les premier et deuxième problèmes.

Sur les figures 19(a) et 19(b), nous pouvons voir que LM surpasse considérablement GD. Ce dernier prend 20,01s en 112 itérations contre 1,2s en 9 itérations pour LM, pour atteindre le même niveau d'erreur ; ainsi, LM converge 8 fois plus vite. Pour le deuxième problème, GD prend 38s en 270 itérations contre 0,87s en 7 itérations pour LM. Dans ce problème, LM est 43 fois plus rapide que GD.

Maintenir les poids positifs dégrade la performance des algorithmes. On constate ceci à la lecture des figures 19(b), et 19(c) pour le premier problème, et des figures 20(b), 20(c), et 20(d) pour le deuxième problème. Comme nous pouvons le voir, LM s'exécute mieux que LM1 et LM2. Nous pouvons également voir que LM1 s'exécute mieux que LM2. Pour le premier problème, LM1 ne converge pas. C'est parce que la plupart des poids sont mis à zéro et par conséquent la matrice hessienne devient singulière. Cependant, en général, LM et LM2 exécutent mieux que GD. Par conséquent, nous pouvons conclure que la contrainte de positivité sur les poids ne devrait pas être employée car elle dégrade trop



(a) GD (b) LM

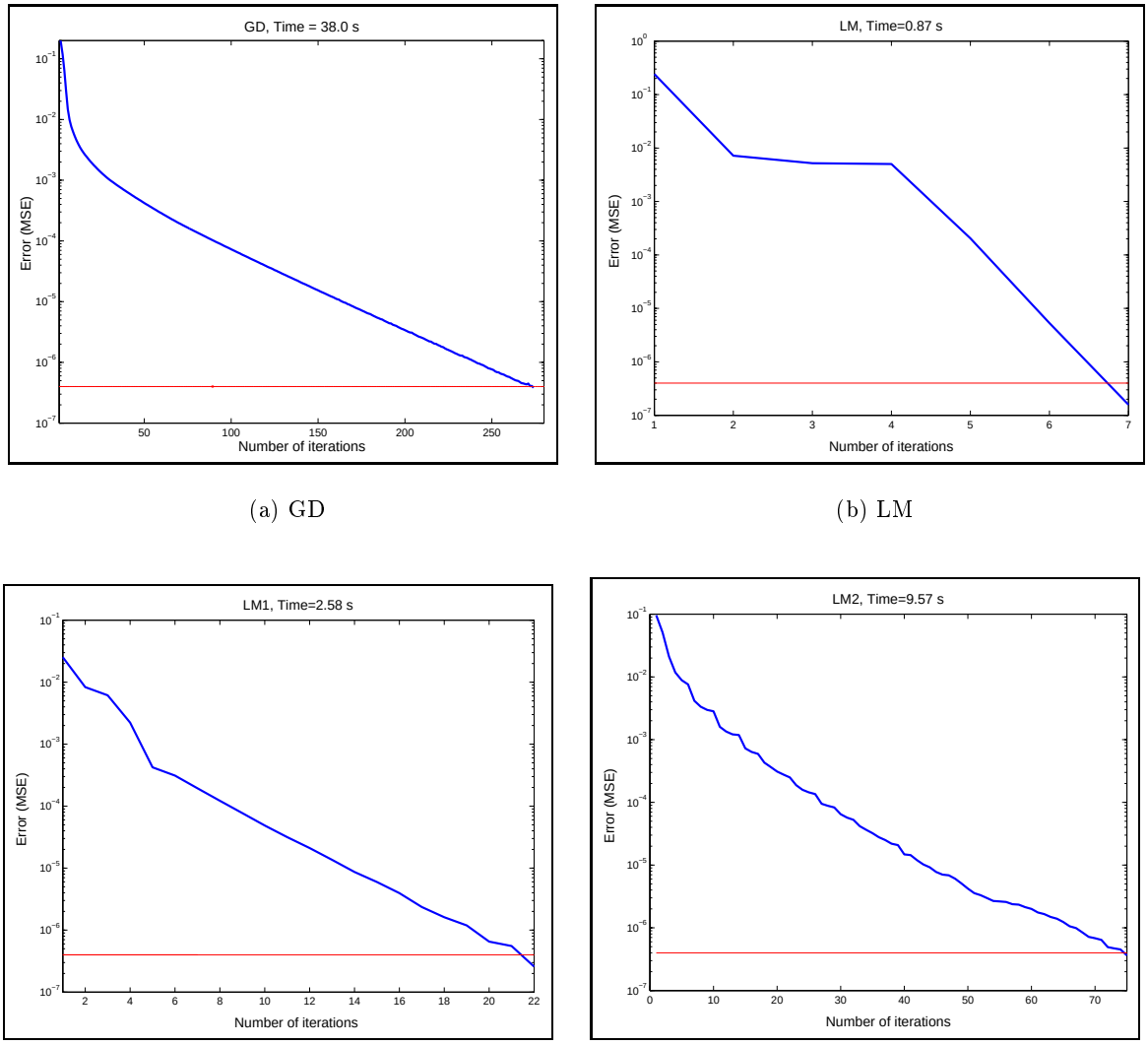
(c) LM2 (d) AM-LM

FIG. 19 – *Les performances des algorithmes GD, LM, LM2 et AM-AM d'apprentissage du premier problème.*

la performance des algorithmes d'apprentissage.

Dans la figure 19(d), nous constatons qu'AM-LM surpasse tous les autres algorithmes d'apprentissage. Cet algorithme est environ 3 fois plus rapide que LM et 52 fois plus rapide que GD pour le premier problème. Nous pouvons aussi voir que dans trois itérations, il a convergé à 10^{-7} contre 5×10^{-5} pour les autres algorithmes d'apprentissage pour le premier problème. Nous verrons après, pour le premier problème, que LM peut converger dans seulement 2 itérations et prendre seulement 0,22s. D'ailleurs, l'AM-LM peut converger dans seulement une itération et prendre 0,14s. Dans certains cas, les deux algorithmes peuvent atteindre une erreur pratiquement nulle.

Le problème de XOR est résolu par l'algorithme AM-LM. Selon l'initialisation des poids, la convergence nécessite une à huit itérations. Dans certains cas, il converge vers un niveau d'erreur quasiment zéro. La méthode du gradient (GD) prend 2140 itérations pour atteindre un MSE de 5×10^{-5} [87].



(a) GD

(b) LM

(c) LM1

(d) LM2 training algorithm

FIG. 20 – *Les performances des algorithmes GD, LM, LM1, et LM2 d'apprentissage du deuxième problème.*

1.8.3.3 Essai du taux de succès et de la performance

Nous avons effectué deux expériences pour examiner le taux de succès et la performance des algorithmes. Nous avons limité le nombre maximum des itérations à 100. Chaque fois que l'algorithme converge, le temps et l'erreur sont stockés. Pour chaque algorithme, dans les premier et deuxième problèmes, le taux de succès, les statistiques du nombre d'itérations, les statistiques du temps exigé et les statistiques de MSE atteint (le minimum, le maximum, l'écart type et la moyenne) sont calculés. Les résultats sont donnés dans les tableaux 10.1 et 10.2 respectivement pour le premier et le deuxième problèmes. Pour l'algorithme AM-LM, nous avons fixé $dP = 0,6$ et $\zeta = 0,90$.

Comme nous pouvons voir au tableau 10.1, les algorithmes LM et GD donnent approximativement le même taux de succès ; mais, LM converge en moins d'itérations et de temps. Les temps minimum et maximum nécessaires pour LM sont 0,33 et 3,58 avec une moyenne de 1,15 s contre 2,92 et 13,94 avec une moyenne de 8,63 s pour GD. Cependant, en appliquant la contrainte de positivité sur poids dans LM, la performance se dégrade significativement. En ce qui concerne l'algorithme AM-LM, il donne

les meilleurs résultats parmi l'ensemble des techniques. Le taux de succès est de 99%. En outre, le temps et les itérations exigés sont les meilleurs. La moyenne du temps nécessaire est 0,67s contre 8,63s, 1,15s, 14,25s respectivement pour GD, LM et LM2. Nous pouvons noter également qu'AM-LM peut converger en une seule itération contre 21, 2 et 48 itérations pour GD, LM et LM2 respectivement. Concernant l'erreur, quelques fois LM et AM-LM peuvent converger vers une erreur nulle.

Il faut mentionner qu'AM-LM ne fonctionne pas pour le cas d'une topologie complètement connectée (comme dans le cas du deuxième problème). Ainsi, nous ne fournissons pas les résultats dans ce cas. Nous pouvons noter dans le tableau 6 que LM converge toutes les fois (100% de succès). La convergence de GD est très pauvre, seulement 22% de succès. En outre, pour LM sans contraintes de positivité sur les poids, les résultats sont meilleurs que dans tous les autres cas (LM1 et LM2). Cependant, la performance est meilleure que celle de GD. LM peut converger dans seulement 5 itérations contre 64, 9 et 38 itérations pour GD, LM1 et LM2 respectivement. D'ailleurs, LM prend moins de temps à converger (environ 1,05s en moyenne contre 9,56s, 3,78s et 9,40s pour GD, LM1 et LM2 respectivement). De même, les niveaux d'erreur sont les minimaux dans le cas de LM.

TAB. 5 – *Comparaison entre GD, LM, LM2 et AM-LM pour le premier problème .*

	GD	LM	LM2	AM-LM
Succès	81	83	50	99
Min(itérations)	21	2	48	1
Max(itérations)	97	24	100	6
Moyen(itérations)	60.27	7.14	82.52	2.83
Std(itérations)	21.26	4.85	12.01	0.95
Min(temps)(s)	2.92	0.33	8.4	0.22
Max(temps)(s)	13.94	3.58	17.57	1.70
Moyen(temps)(s)	8.63	1.15	14.25	0.67
Std(temps)	3.06	0.72	3.06	0.31
Min(erreurs)	3.0537×10^{-5}	0	4.7736×10^{-5}	0
Max(erreurs)	4.9987×10^{-5}	4.9293×10^{-5}	4.9991×10^{-5}	3.8730×10^{-5}
Moyen(erreurs)	4.8643×10^{-5}	2.4402×10^{-5}	4.9153×10^{-5}	1.0993×10^{-6}
Std(erreurs)	2.2703×10^{-6}	1.7641×10^{-5}	5.6558×10^{-7}	5.0233×10^{-5}

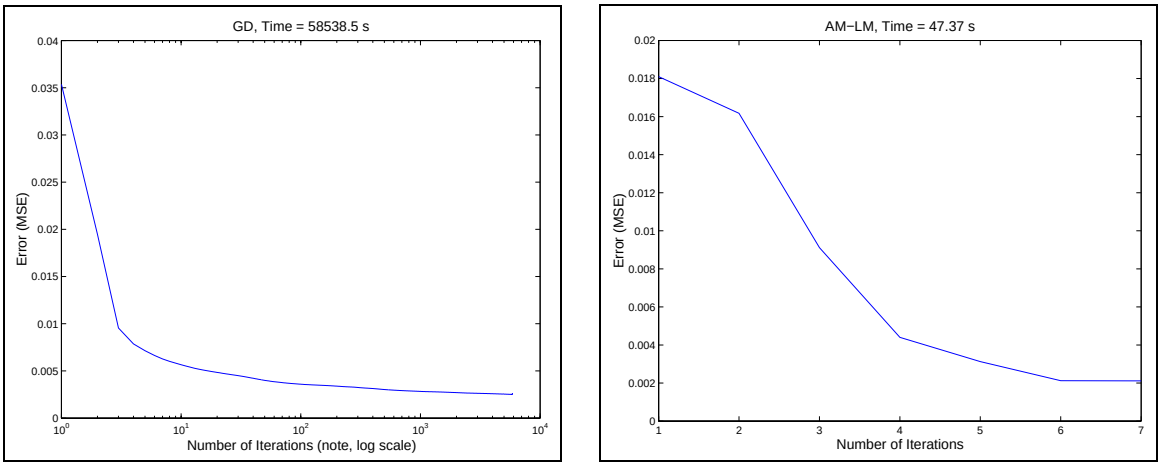
TAB. 6 – *Comparaison entre GD, LM, LM1 et LM2 pour le deuxièmes problème.*

	GD	LM	LM1	LM2
Succès	22	100	95	84
Min(itérations)	64	5	9	38
Max(itérations)	100	9	34	78
Moyen(itérations)	86.45	5.48	18.51	56.32
Std(itérations)	11.76	0.88	4.93	9.27
Min(temps)(s)	5.84	0.54	0.94	4.35
Max(temps)(s)	9.56	1.05	3.78	9.40
Moyen(temps)(s)	8.14	0.67	2.05	6.69
Std(temps)	1.12	0.10	0.53	1.17
Min(erreurs)	3.3503×10^{-7}	4.6590×10^{-9}	5.3220×10^{-8}	2.8499×10^{-7}
Max(erreurs)	3.9810×10^{-7}	3.8989×10^{-7}	3.9886×10^{-7}	3.9001×10^{-7}
Moyen(erreurs)	3.7638×10^{-7}	1.1062×10^{-7}	3.1342×10^{-7}	3.6046×10^{-7}
Std(erreurs)	1.8052×10^{-8}	8.2365×10^{-8}	6.5777×10^{-8}	2.5932×10^{-8}

1.8.3.4 Capacité d'apprendre le problème de la qualité de la vidéo

Nous donnons les résultats de GD et AM-LM pour apprendre le problème de l'évaluation de la qualité de la vidéo présentée en section 1.4. La BD d'apprentissage contient 80 échantillons (chacun contient 5 entrées et une sortie). Le RNN utilisé est de type *feedforward* à trois couches ayant 5 neurones dans la couche d'entrée, 5 neurones cachés et un neurone de sortie. Nous avons entraîné ce réseau par les deux algorithmes avec, comme critère de convergence, un MSE de 0,0025. Pour GD, nous avons employé $\eta = 0,1$ et pour AM-LM, nous avons employé $dP = 0,7$ et $\zeta = 0,90$.

Nous montrons sur la figure 21 la variation de l'erreur avec le nombre d'itérations. Comme nous pouvons le voir, l'AM-LM donne une meilleure exécution que le GD en terme de vitesse : il prend seulement 7 itérations dans 47,37 secondes, alors que le GD atteint la même erreur après 5000 itérations dans environ 16 heures.



(a) GD (b) AM-LM

FIG. 21 – *Comparaison entre les performances de GD et d'AM-LM lors de l'apprentissage du problème de la qualité de la vidéo présenté en section 1.4.*

1.9 Conclusions générales

1.9.1 Sommaire des contributions principales

Le travail présenté dans cette thèse peut être récapitulé comme suit. Nous avons proposé une méthodologie pour évaluer automatiquement la qualité des flux multimédias temps réel, tenant compte du fait qu'il y a beaucoup de facteurs affectant la qualité, notamment les déformations dues au codage et la transmission en temps réel sur des réseaux de paquet. Le but de cette méthode est de dépasser les limites des techniques disponibles dans la littérature. Les avantages de notre méthode sont les suivantes : (i) les résultats obtenus se corrèlent bien avec la perception humaine car notre procédé est établi partiellement en utilisant des données subjectives ; (ii) elle est rapide en terme de temps de calcul, car un réseau de neurones simple (RN) est employé pour mesurer la qualité ; (iii) il n' y a aucun besoin d'accéder aux signaux originaux du multimédia, qui servent seulement pendant la phase de développement à identifier quelques facteurs objectifs qui sont employés pour permettre au RN d'apprendre la relation entre les signaux originaux et traités en se basant sur ces facteurs objectifs ; (iv) beaucoup de facteurs qui ne peuvent pas être pris en considération par les méthodes traditionnelles peuvent être facilement inclus (par exemple, le débit d'images ou l'utilisation de FEC).

Nous avons employé notre méthode avec succès pour évaluer la qualité en temps réel de la parole en tenant compte de plusieurs paramètres de codage et de réseau. Les paramètres considérés sont : le taux de perte, la distribution de perte, l'intervalle de mise en paquet, plusieurs codecs et l'effet de la langue.

Nous avons suivi la même approche pour évaluer la qualité de la vidéo. Les paramètres considérés sont : le taux de perte, la distribution de perte, le débit, le débit d'images, et le type de codage pour le codec H.263. L'application de notre technique aux flux vidéo a mené à des résultats semblables à ceux obtenus dans le cas de la parole.

Les caractéristiques mentionnées ci-dessus de notre méthodologie permettent de l'employer en temps réel avec un temps de calcul négligeable. Le fait que la qualité puisse être mesurée avec une grande confiance, sans avoir accès aux signaux originaux, permet de développer des nouveaux protocoles du réseau de transmission.

Comme exemple de tels protocoles, nous avons conçu un nouveau protocole de contrôle de débit. Ce mécanisme combine une évaluation en temps réel de la qualité de multimédia avec un contrôleur de débit *TCP-Friendly*. Il permet de fournir une meilleure qualité de multimédia et d'économiser la largeur de bande pour une situation donnée de réseau (en déterminant le débit de transmission exact à employer, au lieu de donner juste une borne supérieure). En se basant sur la qualité mesurée par le RN et sur les états de réseau (valeurs données par un contrôleur *TCP-Friendly*), le contrôleur décide des paramètres qui devraient être modifiés pour réaliser cette tâche.

Avant de mettre en application ce protocole, il est nécessaire de comprendre l'impact des paramètres, *a priori* importants, sur la qualité du multimédia. Les conditions à respecter sont que la qualité doit bien se corréler avec la perception humaine et que l'analyse doit inclure l'effet combiné de tous les paramètres influant la qualité à la fois. Deux solutions possibles existent pour répondre à ces exigences : en utilisant les essais subjectifs de qualité ou celle de mesures objectives. Malheureusement, la première solution n'est pas pratique car elle est très chère et difficile à effectuer. En outre, pour atteindre un niveau minimum de précision, un nombre énorme d'évaluations est nécessaire. En ce qui concerne la seconde, la complexité du problème fait qu'il n'existe pas de telle mesure objective.

Notre méthode fournit un RN prenant en entrée plusieurs paramètres ayant une influence sur la qualité, et il est alors aisé de réaliser des études sur l'effet combiné de ces paramètres. Nous avons développé de telles études dans cette thèse.

Dans notre travail, nous avons employé deux types de RN, à savoir les RN artificiels (ANN) et les RN aléatoires (RNN). Nous recommandons l'utilisation des RNN car ils offrent quelques avantages. Cette recommandation se base sur de nombreuses expériences que nous avons effectuées pour comparer les performances de ces outils. Parmi ces avantages, nous pouvons mentionner ce qui suit : (i) les RNN peuvent capter avec plus de précision la relation non linéaire entre la qualité et les paramètres ; (ii) les RNN ne souffrent pas trop du problème de *surentraînement* ; (iii) dans la phase d'exécution, les RNN sont beaucoup plus rapides que les ANN à architecture égale.

L'algorithme disponible d'apprentissage d'un RNN (de type gradient descente) peut être trop lent pour converger à une précision minimale. Par exemple, dans le problème de la vidéo, pour atteindre une erreur quadratique moyenne de 0,0025, il prend environ 5000 itérations dans 16 heures sur une station de travail standard.

Ce problème nous a motivés pour explorer des nouvelles techniques d'apprentissage pour les RNN. Ceci nous a menés à proposer deux nouveaux algorithmes. Le premier est basé sur la méthode de *Levenberg-Marquardt* pour les ANN, l'une des plus puissantes pour les ANN (temps requis pour converger, précision, nombre d'itérations, ...). Le second est une amélioration de la méthode générale de *Levenberg-Marquardt* pour les réseaux de type *feedforward*. Il s'agit d'une technique adaptative pour accélérer le processus d'apprentissage en ajoutant l'information des dérivés d'ordre 2 dans la fonction de coût. Nous avons étudié ces deux algorithmes et nous avons fourni les étapes et la dérivation mathématique à employer avec un RNN. Nous avons trouvé que ces deux algorithmes surpassent l'algorithme de base. Pour le problème de la vidéo discuté dans ce document et la même architecture, le

deuxième algorithme proposé a pris seulement 7 itérations en moins d'une minute pour arriver à la même erreur. Nous avons évalué leurs performances pour d'autres problèmes et nous avons également fourni quelques variantes des deux techniques.

Nous avons également présenté un nouveau modèle pour la prédiction du trafic qui se sert de la périodicité à longue portée et à courte portée du processus du trafic pour fournir un modèle plus réaliste. Les modèles existants se concentrent seulement sur l'information à courte portée tout en négligeant l'information à longue portée.

1.9.2 Extensions possibles

Nous décrivons ici certaines extensions possibles du travail présenté dans cette thèse.

- **Évaluation de la qualité du multimédia.** La qualité du multimédia (une fois encodé et transmis en temps réel en utilisant des réseaux de paquet) est affectée par un grand nombre de paramètres. Notre méthode est donc *a priori* opte à s'appliquer aussi dans le cas plus général.
- **Compréhension de l'impact des paramètres.** Sur ce point, quelques futures directions de recherche pour ce travail incluent l'étude d'autres codecs comme MPEG2/4 ou l'analyse de l'effet de la synchronisation entre l'audio et la vidéo. Les effets combinés d'autres paramètres peuvent être étudiés en suivant l'approche présentée dans cette dissertation. La perte dans le réseau est un des facteurs les plus ennuyants sur la qualité du multimédia, et il y a plusieurs techniques proposées pour traiter ce problème. Une solution générale est l'utilisation de FEC. Il y a également quelques techniques de dissimulation des erreurs selon le type de support (par exemple, la substitution de forme d'onde pour la parole ou l'interpolation linéaire pour la vidéo). Il est intéressant d'étudier l'impact de ces différentes techniques sur la perception humaine.
- **Technologies de réseau.** Dans cette dissertation, nos travaux sont basés sur les réseaux IP, mais ils peuvent être également appliqués à d'autres technologies, par exemple les réseaux sans fil. Le développement de ce type de réseau augmente rapidement de nos jours, avec la mise au point d'applications en temps réel et multimédia (téléconférences, visiophones, vidéo streaming, ldots).
- **Conception d'un encodeur.** On propose dans [161] une méthode appelée *Perception Based Spectrum Distortion* (PBSD). Un résultat important de ce travail est que les auteurs avaient utilisé leur métrique au lieu de MSE pour remodeler l'encodeur de la parole ADPCM afin de choisir le niveau de quantification. Ceci a amené à une amélioration de la qualité subjective. De même, il y a des travaux similaires [98] pour le codage de la vidéo afin de trouver d'autres métriques qui donnent de meilleurs résultats que PSNR dans la conception de codage. Un résultat préliminaire montre une meilleure amélioration. Mais comme nous avons montré, la plupart des mesures objectives disponibles ont trop de limites (principalement, le temps de calcul considérable et surtout, l'exigence de l'accès au signal original). Ainsi elles ne peuvent pas être employées en temps réel pour améliorer le codage basé sur la perception humaine. Nous croyons que notre méthode est un meilleur candidat pour effectuer un tel travail. En outre, elle peut tenir compte de l'impact de la déformation du réseau lorsqu'on l'intègre à un contrôleur.
- **Contrôle de débit.** Nous avons employé, avec notre contrôleur de débit, le protocole de congestion TCP-Friendly basé sur une équation qui a quelques avantages pour des applications multimédias et temps réel. Il est important d'étudier l'utilisation d'autres protocoles avec le nôtre. Nous avons fourni une liste des paramètres de contrôle possibles à employer avec notre proposition. En employant ces paramètres, nous pensons que la qualité peut être encore améliorée. L'effet d'autres paramètres et la réalisation d'un contrôleur complet sont des directions possibles de recherche future.
- **Modèle de prédiction du trafic.** Basé sur le modèle proposé ici, plusieurs applications peuvent être considérées : attribution dynamique de largeur de bande, négociation dynamique du contrat à long terme, facturation, mise en place d'espaceurs et l'ingénierie de trafic, par exemples.

English Annex: Automatic Evaluation of Real-Time Multimedia Quality: a Neural Network Approach

Chapter 2

Introduction

In the recent times, there has been an enormous advance in packet-based networks (for example, the global Internet) technology. This gives rise to a high number of promising applications that would not have been possible without these great and rapid technological achievements. In addition, there have been advances in digital speech, audio, and video encoding (compression) methods. Compression is one of the most important features in current telecommunications technology.

These factors have made it possible to transmit multimedia data over packet-based networks at low prices. In addition, efficient integration of voice, video, and data services in telecommunication networks is possible today. Among the endless number of applications, we can find videoconferencing, video streaming, video telephony, telemedicine, telepresence, video on demand, IP telephony, voice over IP, multi-party conferencing, Internet radio and broadcasting, multimedia storage and retrieving etc. This gives also rise to some important problems that have to be tackled.

This Chapter is organized as follows. In Section 2.1, we provide the motivations behind the work presented in this thesis. The major contributions are summarized in Section 2.2. In Section 2.3, we give a brief overview and the organization of the report.

2.1 Motivations

2.1.1 Multimedia Quality Assessment

One of the problems referred to before is the automatic *quality assessment*, and in real time, of multimedia streams transmitted through the network. For example, *speech quality assessment* in IP-telephony applications, *video quality assessment* in video teleconferencing or video streaming applications, etc. It is a problem since, as we will see in detail in Chapter 3, no previously published satisfactory solution to it exists.

Traditionally, the quality is measured as a function of the encoding algorithms, without taking into account the effect of packet networks' parameters. There are two approaches to measure the quality: either by *objective* methods or by *subjective* methods. Subjective quality assessment methods [72, 70, 59] measure the overall perceived quality. They are carried out by human subjects. The most commonly used measure is the Mean Opinion Score (MOS), recommended by the International Telecommunication Union (ITU) [70]. It consists of having a group of subjects viewing and/or listening to processed samples or sequences in order to rate their quality, according to a predefined quality scale. That is, human subjects are trained to "build" a mapping between the quality scale and a set of processed multimedia samples.

On the other hand, objective methods [154] usually measure the quality by comparing the original and distorted sequences. Some existing methods for video are MSE (Mean Square Error) or PSNR (Peak Signal to Noise Ratio), which measure the quality by performing some kind of simple difference between frames. There are other more complicated measures like the moving picture quality metric

(MPQM), and the normalized video fidelity metric (NVFM) [153]. Some examples for objective speech quality assessment are: Signal to Noise Ratio (SNR), Itukura-Saito distortion, Log-likelihood ratio, Segmental SNR and Perceptual Speech Quality Measures (PSQM) [38, 59, 105]. It must be observed that to verify the accuracy of these measures (also called "tests" in this area), they usually have to be correlated with results obtained using subjective quality evaluations.

2.1.1.1 Limitations of Subjective and Objective Quality Tests

Although MOS studies have served as the basis for analyzing many aspects of signal processing, they present several limitations:

(a) stringent environments are required;

(b) the process cannot be automated;

(c) they are very costly and time consuming to repeat it frequently;

(d) they are impossible to use in real-time quality assessment.

On the other hand, the disadvantages of objective methods are:

(a) they do not correlate well with human perception;

(b) they require high calculation power, and are then time consuming;

(c) they cannot be used for real-time quality assessment, as they work on both the original video sequence or speech sample and the transmitted/distorted one;

(d) it is difficult to build a model that takes into account the effect of many quality-affecting parameters at the same time, especially network parameters.

The requirements of the kind of good objective quality measure we are looking for are: the evaluation must be possible in real time, it must be computationally simple (not time consuming), there is no need to access the original signal, all the quality-affecting parameters including network factors can be considered (that is the test must not be limited to a specific kind of parameters) and the obtained results correlate well with human perception. It is important to mention that, so far, there is no objective measure that can fulfill these requirements. Thus, one of our objectives is to propose a new objective measure that can satisfy all the stated requirements.

2.1.2 Impact of the Quality-Affecting Parameters on Multimedia Quality

A second important problem is the evaluation of the *impact* of many basic parameters on multimedia quality. The relationship between quality and these parameters is complex and difficult to apprehend. Multimedia quality is affected by many factors. However, there is no complete study in the literature that shows the impact of these parameters on the quality, mainly because subjective quality tests are expensive to carry out. Moreover, to analyze the combined effect, for instance, of three or four parameters, we need to build a very large set of human evaluations in order to reach a minimal precision level. Previous studies are either concentrated on the effect of network parameters without paying attention to encoding parameters, or the contrary. 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. For example, in the case of speech, the loss rate and packetization interval metrics are studied in [61], while [28] studies mainly the effect of the loss rate on several speech codecs. In the case of video, loss rate and loss burst size metrics are studied in [58], while [153] studies mainly the effect of bit rate and [53] works only on the effect of frame rate. Concerning objective approaches, there is

no previously published objective quality test that can take into account the direct influence on the quality of, for instance, the whole set of previously mentioned parameters simultaneously.

To attack this problem, we need a tool that can measure multimedia quality by taking into account the direct influence of a large range variation of these parameters. The measured quality should correlate well with human perception. The tool should predict the quality within the range of each parameter's values without having to do the subjective tests for all their possible combinations. By understanding the impact of the parameters on the quality, this may help in:

- developing control mechanisms in order to deliver the best possible video quality given the current network situation (cf. the next subsection),
- and in better understanding of QoS aspects in multimedia engineering.

Our objective in this area is to provide studies of the combined effects of several quality-affecting parameters on both speech and video quality. We aim also to provide general guidelines that can be followed to perform similar studies in other kinds of network technologies and, if useful, other parameters.

2.1.3 Rate Control Protocols

A third important problem in traffic engineering is *rate control*. Traditionally, rate control protocols are designed based on passive network measurements (loss, delay, bandwidth, etc.) [26, 41, 44, 77, 107, 108, 124, 126]. However, human subjects are the final end-users of real-time multimedia applications. The direct reaction of these end-users is often neglected in the design of these protocols, because it has been difficult to quantify in real time their perception in a way allowing to take into account the direct effect of both network and codecs's parameters. The solutions to the previously mentioned problems that we present in this thesis (namely the real-time quality assessment of real time multimedia transmitted over packet networks and the understanding of the impact of the quality-affecting parameters on user's perception) may make it possible to think in designing new kinds of ones. These protocols, in addition to the passive networks measurements employed in the traditional protocols, could take into consideration the direct influence of the end-users' perception in real time. We concentrate in this dissertation on providing a new rate control protocol that integrates both network measurements and users' perception. Among the benefits of having such a protocol, let us mention the following:

- We can maximize or guarantee the delivery of the best possible multimedia quality given the network state.
- It helps in reducing network congestion.
- It keeps the stream in a TCP-friendliness state and hence does not contribute to collapse the network.
- Based on the rate controller and the real-time multimedia quality assessment, multimedia codecs could be re-engineered to compress the original signal based on users' perception and the network state rather than using only the simplistic SNR or PSNR objective quality measures.

It is important to underline that the design of these protocols was not possible without having a tool that measures or evaluates in real time the user's perception of real-time multimedia quality given the parameters' values. In this dissertation, a third objective of our work is to propose such a protocols.

2.1.4 Traffic Prediction

A fourth important problem is *traffic prediction*. This problem is of extreme importance because traffic prediction is crucial for successful congestion control, congestion avoidance, dynamic bandwidth allocation, dynamic contract negotiation, traffic shaping and engineering, etc. More precise congestion avoidance mechanisms can be implemented by considering both traffic prediction and the rate control previously presented. We have also explored this issue for the following reasons:

- because of its direct relation with control techniques,
- because neural network tools work quite well for the case of rate control and the quality assessment problems.

However, there are some difficulties that so far prevented providing good traffic predictors that can work in real situations, mainly related to the fact that network traffic is, in general, a complex, time-variant, nonlinear, and nonstationary process. Furthermore, this process is governed by parameters and variables that are very difficult to measure. Sometimes, there are completely nonpredictable portions of this time-variant process (the so-called "spiky" fragments, see Section 9.3.4). Therefore, a precise model of this process becomes difficult as its complexity increases.

Despite these problems, traffic prediction is possible because, as the measurements have shown, there coexist both long-range and short-range dependencies in network traffics. For example, the amount of traffic differs from the weekend to that in the weekdays. However, it is statistically similar for all the weekends, also during the same day, in the morning, in the nights, at some specific parts of the day, etc.

There are many proposals in the literature for traffic prediction [85, 27, 121, 120, 156, 39]. These proposals concentrate on the short-range dependencies, somehow neglecting the long-range ones. When testing these models on real traces lasting for several minutes or even hours, they give good performance. However, when employing them for predicting the traffic for days or weeks, their performance degrades significantly.

Our aim in this part of our dissertation is to propose a new model for traffic prediction that takes into account both long-range and short-range dependencies.

2.1.5 Neural Network Learning

It is known that Neural Networks (NN) have been successfully used to solve many problems [117, 149, 150, 2, 4, 7, 50, 128, 10] that could not be solved by, say, mathematical algorithms. However, performance depends not only on NN properties but alos how well we are modelling our system.

One of the motivations behind the use of NN is that they are very efficient in learning and predicting nonlinear and complex functions. In addition, we do not need to know the details of the mathematical model of the system under study, which in some cases is very difficult to do or completely unavailable. The aforementioned problems are complex, nonlinear and, the case of network traffic, nonstationary. NN has been found to be powerful method to efficiently describe a real, complex and unknown process with nonlinear and time-varying properties.

Our study and work is based on NN. We have used two kinds of NN: Artificial NN (ANN) [117, 118, 149, 150] and Random NN (RNN) [49, 89, 48, 45, 101, 51]. We have compared the performance of both of them. We have found that RNN have several advantages over ANN. However, a major disadvantage of RNN is that the available training algorithm is very slow and not efficient. This motivated us to propose new training algorithms for RNN to overcome the problem.

2.2 The Contributions of this Dissertation

In this Section we briefly resume the major contributions of this dissertation. We classify them according to the area they naturally belong to.

In the area of multimedia quality assessment, the main contributions are:

- We provide a new methodology to measure real-time multimedia (speech, audio, and/or video) quality in real time (the associated publications are [90, 92, 95, 91]). The main properties of our method that we want to emphasize are the following ones: (i) no access to the original signal is required, (ii) it is not computationally intensive since, once trained, the neural network gives its evaluations in negligible time, (iii) the obtained results correlate well with MOS (i.e. human perception), (iv) all the quality-affecting parameters can be taken into account, (v) it can operate in real time, and, for instance, it can be easily integrated with a multimedia application.

- We investigate the applicability of our method to the case of speech quality assessment. We present the results of real experiments we carried out between national-wide and international-wide sites. The purpose of these experiments is to identify the most typical ranges of the network parameters. We use three different speech encoders, namely PCM, ADPCM and GSM. Three different spoken languages (Arabic, Spanish and French) are considered. In particular, loss rate, consecutive loss duration as well as the packetization interval of the speech are also considered. (The associated publications are [92, 90].)

- We explore the applicability of our method to evaluate real-time video in real time transmitted over packet-based networks. We use a H.263 encoder and we selected five important parameters that have the most dominant impact on video quality. These parameters are: bit rate, frame rate, loss rate, consecutive loss duration, and the amount of redundant information to protect against loss. (The associated publication is [95].)

- We provide a study of the impact of quality-affecting parameters on the perceived quality. Our study is for both video and audio/speech applications. To the best of our knowledge, this is the first analysis of the combined effects of several parameters of both the network and the encoder on the perceived quality. (Corresponding publications submitted for possible publication are [94, 96].)

In the area of rate control, our contribution is the following:

- We present a new rate control protocol that integrates both network passive measurements (e.g., loss rates and delays) and user's perception of multimedia quality transmitted over that network. The objectives of our rate controller is twofold: first, a better use of bandwidth, and second, delivery of the best possible multimedia quality given the network current situation. (The associated publication is [93].) It can be seen as an application of the method of quality assessment described before. We tested the applicability of our protocol for the case of both speech and video transmission over the public Internet. We relay the basis of designing similar kind of protocols that take benefits of having a tool able to measure in real time multimedia quality.

In the area of traffic prediction, the contribution is following:

- We provide a new method for traffic prediction that takes into account not only short-term dependency of the traffic process, but also long-term one. We study and evaluate our model by using real traffic traces and we provide some possible applications that can make use of our new technique.

In the area of neural network training, the main contribution is the following:

- We provide two new training algorithms for random neural networks. These two algorithms are inspired from the fastest training algorithms for ANN, namely *Levenberg-Marquardt* (LM) and a recently proposed referred as *LM with adaptive momentum*. We evaluate the performance of these algorithms and the available *gradient descent* one. In addition, we provide some elements of comparison between ANN and RNN.

2.3 Overview of the Dissertation

In this Chapter, we have introduced the motivations and the context of the problems that we tackle in this dissertation.

The rest of the report is divided into three parts. In the first one, we focus on the automatic multimedia quality assessment problem. In Chapter 3, a state-of-the-art of the issues related to our work is presented. More specifically, we discuss the current technology concerning objective multimedia quality measures, real-time transmission of multimedia streams over packet networks, the challenges and the existing methods to overcome them, audio and video compression and several standard codecs, and neural networks are overviewed. In Chapter 4, we present a new method to measure real-time multimedia quality when it is transmitted in real time over packet network. We provide the general methodology as it can be applied for speech, audio and/or video. We present also the most used techniques to evaluate subjective multimedia quality. We emphasize on the use of neural networks and we compare both ANN and RNN in the context of our problems. We describe also some possible uses and applications of our method.

In the second part, we focus on speech and video quality as well as our rate control mechanism. In Chapter 5, we evaluate the applicability of our technique to evaluate, in real time, real-time speech transmitted over packet-based networks. We present the results of real experiments we carried out between national-wide and international-wide sites. The purpose of these experiments is to identify the most typical ranges of the main network parameters. We use three different speech encoders, namely PCM, ADPCM and GSM. Three different spoken languages (Arabic, Spanish and French) are considered. Other important parameters we take into account are loss rate, consecutive loss duration and the packetization interval of the speech. In Chapter 6, we evaluate the applicability of our technique to evaluate real-time video transmitted over packet-based networks in real time. We use the H.263 encoder and we selected five important parameters that have *a priori* an impact on video quality. These parameters are bit rate, frame rate, loss rate, consecutive loss duration, and the amount of redundant information to protect against loss. In Chapter 7, we study the impact of quality-affecting parameters on the perceived quality. Our study is for both video and audio/speech applications. This study can help in understanding the behavior of both packet video and audio in the Internet (or in any packet-based network). In Chapter 8, we propose a rate control protocol that takes into account the end-user perception as well as the traditional network measurements to deliver the best possible quality in real time given the network current state. We provide a study of the possible parameters that can be used for the both video and audio applications. The new protocol can, in certain situations, save network resources while maintaining TCP-friendness behavior.

The third and last part of this report deals with traffic prediction and random neural network learning algorithms. In Chapter 9, we provide a new method for traffic prediction that takes into account not only the short term dependency of the traffic process, but also the long term. We compare our approach with the existing ones and we evaluate it using real traffic traces. In Chapter 10, we present two new training algorithms to improve the performance and the accuracy of RNN. We evaluate these algorithms and we also compare them with the traditional gradient descent algorithm.

Finally, we give the conclusions of this dissertation and some future research directions related to the problems addressed during this work.

Part I

Automatic Real-Time Multimedia Quality Evaluation

Chapter 3

State of the Art

In this Chapter, we provide an overview of the most popular objective speech and video quality measures. We give a brief description and the main features of each one; but we provide a comparative analysis for them in Chapters 5 and 6. We also highlight the basic multimedia transport protocols. The challenges and some solutions, including error concealment techniques for audio and video streams, of transferring real-time multimedia over packet networks are presented. Then, we introduce both speech and video compression algorithms. We also describe some of the standard speech and video codecs. In This thesis, we use two types of neural networks (NN); namely, Artificial NN (ANN) and Random NN (RNN). We give in this Chapter a general description of both of them.

3.1 Objective Speech Quality Measures

There are several objective speech quality measures. The most simple one is the Signal to Noise Ratio (SNR) that compares the original and processed speech signals sample by sample. There are also more complex ones that are built based on Human Auditory System model involving complex mathematical calculations. We present the most famous measures in this section. All of them with the exception of the ITU E-model operates on both the original and the processed speech sample. This limitation makes it impossible to work in real time and to include these metrics in designing new mechanisms (rate control or speech codecs design to take into account the user's perception and the network factors). A second disadvantage is that the obtained results do not correlate always with subjective data (thus they cannot measure correctly user's perception). A third drawback is that some of them are computationally extensive. This point limits their usage in lightweight applications including mobile phones. Some of these metrics are designed and optimized basically to consider encoding impairments and restricted conditions, but they do not work efficiently when they used in other conditions (ex. distortion due to the transmission over the network). Some of these methods require a perfect synchronization between the original and processed signals otherwise the performce degrades considerably. In this case several factors including the delay variation's effect cannot be taken into account by these methods.

There are three types of objective speech quality measures: time domain, spectral domain, and perceptual domain measures [155]. The time domain measures are usually applicable to analog or waveform coding systems in which the goal is to reproduce the waveform itself. SNR and segmental SNR (SNRseg) are the most known methods. Since the waveform are directly compared in time domain, synchronization of the original and distorted signals is a must. However, synchronization is difficult; if not performed well, the performance is poor.

The most simple possible measure is the Signal-to-Noise (SNR) ratio. Its goal is to measure the

distortion of the waveform coders that reproduce the input waveform. It is calculated as follows:

$$SNR = 10 \log_{10} \frac{\sum_{i=1}^{N} x^2(i)}{\sum_{i=1}^{N} (x(i) - y(i))^2},$$

where $x(i)$ and $y(i)$ are the original and processed speech samples indexed by i and N is the total number of samples.

Segmental Signal-to-Noise Ratio (SNRseg), instead of working on the whole signal, calculates the average of the SNR values of short segments (15 to 20 ms). It is given by:

$$SNRseg = \frac{10}{M} \sum_{m=0}^{M-1} \log_{10} \sum_{i=Nm}^{Nm+N-1} \left(\frac{\sum_{i=1}^{N} x^2(i)}{\sum_{i=1}^{N} (x(i) - y(i))^2} \right),$$

where N and M are the segment length and the number of segments respectively. SNRseg gives better results than SNR for waveform encoders, but it gives very bad results for vocoders (see Section 3.5).

The second type of measures are the spectral domain ones [155]. They are generally computed using speech segments typically between 15 and 30 ms long. They are much more reliable than time domain measures and less sensitive to the misalignments between the original and distorted signals. However, these measures are closely related to speech codec design and use the parameters of speech production modules. Hence their ability to adequately describe the listener's auditory response is limited by the constraints of the speech production modules. They include the log likelihood ratio, the Linear Predictive Coding (LPC) parameter distance measures, the cepstral distance, and the weighted slope spectral distance measures (for more details and descriptions see [155]). In general, all these methods gives good results for some encoding distortion, but they are not valid for the case when the original speech is passed through a communication system that significantly changes the statistics of the original speech.

The third type of objective measures is constituted by the perceptual domain measures [155]. In contrast to the spectral domain measures, perceptual domain measures are based on models of human auditory perception. They transform speech signal into a perceptually relevant domain such as bark spectrum or loudness domain, and incorporate human auditory models. They give better prediction of the quality under the condition that the used auditory model used truly describes the human auditorial system. It is clear that this task is very complex and it is not possible to implement exact model of such system. However, by using approximations of the human auditorial system, the obtained results correlate better than that of the other two types of speech measures. Another important point to underline is the fact that these models are optimized for a specific type of speech data; the performance is not good for different speech data. In addition, they have the risk of not describing perceptually important effects relevant to speech quality but simply a curve fitting by parameter optimization. These measures are the most known and used in the literature. We provide a brief description of these metrics as given in [155]. The evaluation of their performances is given in Section 5.5.

3.1.1 Bark Spectral Distortion (BSD)

BSD was developed at the University of California [145]. It was essentially the first objective measure to incorporate psychoacoustic responses. Its performance was quite good for speech coding distortions as compared to traditional objective measures (in time domain and in spectral domain). The BSD measure is based on the assumption that speech quality is directly related to speech loudness, which is a psychoacoustical term defined as the magnitude of auditory sensation. In order to calculate loudness, the speech signal is processed using the results of psychoacoustics measurements. BSD estimates the overall distortion by using the average Euclidean distance between loudness vectors of the reference and of the distorted speech. BSD works well in cases where the distortion in voiced regions represents

the overall distortion, because it processes voiced regions only; for this reason, voiced regions must be detected.

3.1.2 Enhanced Modified BSD (EMBSD)

EMBSD [155] metric is based on the BSD algorithm. The EMBSD algorithm consists of a perceptual transform followed by a distance measure that incorporates a cognition model. The original and received signals are normalized and divided into 50% overlapping frames of 320 samples. Frames that exceed an energy threshold are transformed to the Bark frequency domain. The Bark coefficients are transformed to dB to model perceived loudness, then scaled. The first 15 Bark spectral components in each frame are used to compute the loudness difference. Only distortion that exceeds a noise masking threshold and lasts longer than 200 ms (10 frames) is considered. The distortion is computed as the average distortion for all valid frames.

Both the original and distorted signals should be strictly synchronized, otherwise the performance becomes poor. Tests showed that its correlation with subjective results is relatively good for encoding impairments, however it is not able to evaluate the quality when network impairments are applied as we will show in Section 5.5.

3.1.3 Perceptual Speech Quality Measure (PSQM)

PSQM was developed by PTT Research in 1994 [15, 14]. It can be considered as a modified version of the Perceptual Audio Quality Measure (PAQM) which is an objective audio quality measure also developed at PTT Research [14]. Recognizing that the characteristics of speech and music are different, PSQM was optimized for speech by modifying some of the procedures of PAQM. PSQM has been adopted as ITU-T Recommendation P.861 [71]. Its performance has been shown to be relatively robust for coding distortions.

PSQM transforms the speech signal into the loudness domain, modifying some parameters in the loudness calculation in order to optimize performance. PSQM does not include temporal or spectral masking in its calculation of loudness. It applies a nonlinear scaling factor to the loudness vector of distorted speech. The scaling factor is obtained using the loudness ratio of the reference and the distorted speech in three frequency bands. The difference between the scaled loudness of the distorted speech and the loudness of the reference speech is called noise disturbance. The final estimated distortion is an averaged noise disturbance over all the frames processed. PSQM disregards or applies a small weight to silence portions in the calculation of distortion.

PSQM uses psychoacoustic results of loudness calculation to transform speech into the perceptually relevant domain. It also considers the role of distortions in silence portions on overall speech quality. Even though its performance is relatively robust over coding distortions, it may not be robust enough to apply to a broader range of distortions.

3.1.4 PSQM+

PSQM+ was developed by KPN Research in 1997 [13]. The performance of PSQM+ is an improvement over that of the P.861 (PSQM) for loud distortions and temporal clipping by some simple modifications to the cognition module. PSQM+ can be applied to a wider range of distortions as an objective measure than PSQM.

PSQM+ uses the same perceptual transformation module as PSQM. Similar to PSQM, PSQM+ transforms the speech signal into the modified loudness domain, and does not include temporal or spectral masking in its calculation of loudness. In order to improve the performance for the loud distortions like temporal clipping, an additional scaling factor is introduced when the overall distortion is calculated. This scaling factor makes the overall distortion proportional to the amount of temporal clipping distortion. Otherwise, the cognition module is the same as PSQM.

PSQM+ adopts a simple algorithm in the cognition module to improve the performance of PSQM. The poor performance of PSQM for distortions like temporal clipping is due to the procedure calculating a scaling factor. The scaling factors are determined by the ratio of the energy of the distorted speech and the reference speech. This scaling factor scheme of PSQM works very well when a distortion results in additional energy. However, if a distortion results in reduced energy such as temporal clipping, which removes some of the signal energy, the estimate of distortion is proportionally smaller, and PSQM underestimates the actual distortion. Therefore, PSQM+ uses a simple modification to adopt an additional scaling factor to compensate for this effect.

PSQM+ resolves one performance issue of PSQM on distortions such as temporal clipping. However, the performance of PSQM+ may be questionable for other different types of distortions.

3.1.5 Measuring Normalizing Blocks (MNB)

MNB was developed at the US Department of Commerce in 1997 [139, 140, 141]. It emphasizes the important role of the cognition module for estimating speech quality. It models human judgment on speech quality with two types of hierarchical structures. It has showed to be relatively robust over an extensive number of different speech data sets.

MNB transforms speech signals into an approximate loudness domain through frequency warping and logarithmic scaling. It assumes that these two factors play the most important role in modeling human auditory response. The algorithm generates an approximated loudness vector for each frame. MNB considers human listener's sensitivity to the distribution of distortion, so it uses hierarchical structures that work from larger time and frequency scales to smaller time and frequency scales. MNB employs two types of calculations in deriving a quality estimate: time measuring normalizing blocks (TMNB) and frequency measuring normalizing blocks (FMNB). Each TMNB integrates over frequency scales and measures differences over time intervals while the FMNB integrates over time intervals and measures differences over frequency scales. After calculating 11 or 12 MNBs, these MNBs are linearly combined to estimate the overall speech distortion. The weights for each MNB are optimized with a training data set.

3.1.6 Perceptual Analysis Measurement System (PAMS)

PAMS was developed by British Telecom (BT) in 1998 [112, 113, 114, 115]. PAMS uses a model to compare the original signal with the degraded one. It parameterizes the distortions into parameters and maps them into an objective quality measure.

A parameter set, in which each parameter increases with increasing degradation, is generated. The parameter set used in PAMS has not been specified in the literature. A linear mixture of monotonic quadratic functions for the selected parameters is used for mapping to subjective quality. The quadratic functions are constrained to be increasing with their parameters.

The performance of the PAMS depends upon the designer's intuition in extracting candidate parameters. Since the parameters are usually not independent of each other, it is not easy to optimize both the parameter set and the associated mapping function, all this resulting in an extensive computation.

3.1.7 ITU E-model

The E-model [68] was originally designed to evaluate the quality of the public switched telephony network (PSTN) by taking into account the encoding distortion, delay and delay variation, echo etc. However, people use this metric to evaluate the quality of Voice over IP (VoIP) applications [36]. The advantage of this metric over all the other mentioned ones is that it does not require the access to the original speech signal. Thus, it is computationally simple and can be used in real-time applications. However, as we will show in Section 5.5, this metric is not reliable as the correlation coefficient is very

bad with subjective quality tests. This is not surprising because the metric output is calculated from this very simple equation:

$$R = R_0 - I_s - I_d - I_e + A,$$

where, R is the metric output, R_0 corresponds to the quality with no distortion at all, I_s corresponds to the impairment of the speech signal itself, I_d corresponds to the impairment level caused by the delay and jitter, I_e corresponds to the impairments caused by the encoding artifacts and A is the expectation factor, which expresses the decrease in the rating R that a user is willing to tolerate because of the access advantage that certain systems have over traditional wire-bound telephony.

As we can see, this metric evaluates the quality by linear transformation, however the quality is nonlinear with respect to all the affecting parameters.

3.2 Objective Video Quality Techniques

Similar to speech, there are several objective video quality measures. They vary from a simple difference between original and distorted sequences to very complicated ones that are based on Human Vision System (HVS) models and include too complex mathematical calculations. All video quality measures we found in the literature and presented afterwards operate on both the original and the distorted video sequences. This limitation makes it impossible to work in real time and to include these metrics in designing new mechanisms (rate control or video codecs to take into account the user's perception and the network factors). A second disadvantage is that the obtained results do not always correlate with subjective data (thus they cannot measure correctly user's perception). A third drawback is that they are very computationally extensive, especially the ones built based on VHS model (some of them cannot be used to evaluate the quality for video sequences of length greater than 1 sec.). Some of these metrics are designed and optimized basically to consider encoding impairments and restricted conditions, but they do not work efficiently when they used in other conditions (ex. distortion due to the transmission over the network). Another observation is that not all the quality-affecting parameters can be considered. For example, the frame rate effect cannot be considered as they compare the original and distorted. This means that both sequences must have the same frame rate and the decoded picture of the processed sequence must correspond to the encoded picture in each frame of the original sequence, otherwise the results will degrade (cf. signals synchronization in the case of speech).

The most commonly used objective quality metric is the Peak Signal to Noise Ratio (PSNR). For a video sequence of K frames each having NxM pixels with m-bit depth, first the Mean Square Error(MSE) is calculated as follows:

$$\text{MSE} = \frac{1}{N.M.K} \sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{m=1}^{M} [x(i,j,k) - \bar{x}(i,j,k)]^2,$$

where $x(i,j,k)$ and $\bar{x}(i,j,k)$ are the pixel luminance value in the i,j location in the k frame for the original and distorted sequences respectively. The Root MSE (RMSE) is calculated using RMSE = $\sqrt{\text{MSE}}$. The PSNR can be calculated as follows:

$$\text{PSNR} = 10.\log \frac{m^2}{RMSE^2}.$$

MSE and RMSE measure the difference between the original and distorted sequences. PSNR measures the fidelity (how close a sequence is similar to an original one). Compared to other objective measures, PSNR is easy to compute and well understood by most researchers. However, the correlation with subjective measure is bad. In addition, the above measures consider only the luminance component, and neglect the chrominance one, which is important for human perception.

Among the available objective video quality measures in the literature, there are the ITS and EPFL metrics. The first one is developed in the "Institute for Telecommunication Sciences", from which comes the acronym ITS. The other ones are developed in "École Polytechnique Fédérale de Lausanne (EPFL)". We describe these metrics in this Section. Regarding the performance evaluation of these methods, see Section 6.6.

3.2.1 ITS Video Quality Measure

ITS Model [138, 142, 143, 151] maps image imperfections onto measurable mathematical parameters. It is based only on the luminance values, the chrominance values are not considered. From that, we can see that the correlation with subjective tests cannot be very good, as the chrominance values are important to the HVS, even if its importance is not as high as that of luminance values.

The method is based on the extraction and classification of features from both the original and distorted video sequences. The extracted features quantify the impairments of the distorted sequence. An algorithm that segments each video frame into motion and still parts is used to quantify each of them separately. The features correspond to the impairments of the sequence. Each feature is evaluated objectively. The classification part is used to assign a quality measure based on the features objective values.

The quality measure is a linear combination of three quality impairment measures. These measures were selected among a number of candidates such that their combination matched best the subjective evaluations. One of these measures quantifies the Spatial Information (SI), while the other two measures quantify the Temporal Information (TI). The SI for a frame F_n is defined as

$$SI(F_n) = STD_{space}\{Sobel[F_n]\},$$

where STD_{space} is the standard deviation operator over the horizontal and vertical spatial dimensions (the *space*) in a frame, and *Sobel* is the Sobel filtering operator, which is a high pass filter used for edge detection.

The TI is based upon the motion difference between two successive frames, $\triangle F_n = F_n - F_{n-1}$, which is composed of the differences between pixel values at the same location. TI is computed using

$$TI(F_n) = STD_{space}\{\triangle F_n\}.$$

Both SI and TI are to be computed for each frame. To obtain a single scalar quality estimate for each video sequence, SI and TI values are then time-collapsed as follows. The three measures, m_1, m_2, and m_3, are defined, which are to be linearly combined to get the final quality measure. Measure m_1 is a measure of spatial distortion, and is obtained from the SI features of the original and degraded video as follows:

$$m_1 = RMS_{time}\left\{5.81\left|\frac{SI(O_n) - SI(D_n)}{SI(O_n)}\right|\right\},$$

where O_n and D_n are the frame n of the original and degraded video sequences respectively. RMS_{time} denotes the root mean square function performed over the time, for the duration of each test sequence.

m_2 and m_3 are measures of temporal distortion and are given by

$$m_2 = f_{time}\left(0.108 \max\left\{TI(O_n) - TI(D_n), 0\right\}\right),$$

with

$$f_{time}(x) = STD_{time}\{CONV(x, [-1, 2, -1])\},$$

where STD_{time} is the standard deviation across time, and $CONV$ is the convolution operator.

$$m_3 = \max_{time} \left\{ 4.23 \log \frac{TI(D_n)}{TI(O_n)} \right\},$$

where $\max_{time}$ denotes the maximum value of the time history for each test sequence. This measure selects the video frame that has the largest added motion.

Finally, the quality measure q is given in terms of m_1, m_2, and m_3 by

$$q = 4.77 - 0.992m_1 - 0.272m_2 - 0.356m_3.$$

From the above equation, we can see that this metric computes the quality from a linear combination of certain objective quality measures. As we show in Section 6.6, the correlation factor with subjective data is variable from 0.73 to 0.94, depending on the distortion. In addition, it is not consistent with all the bit rates, that is it gives poor results for low bit rates.

3.2.2 EPFL Objective Video Quality Measures

Researchers at EPFL have proposed three objective video quality measures [136, 137, 154, 135]. These measures are Moving Picture Quality Metric (MPQM), Color MPQM (CMPQM), and Normalization Video Fidelity Metric (NVFM). Unlike the ITS model, which considers only the luminance values, EPFL measures consider both luminance and chrominance values. The correlation with subjective tests is reasonable for some video sequences, but gives bad correlation for others (see Section 6.6). In addition, as explained next, they are not suitable for real-time evaluation of the quality as they operate on both the original and distorted sequences. Another disadvantages are: they are not consistent over all video bit rate ranges and they are computationally complex.

We give some details about these metrics bellow (for a complete description of each of them see [135]).

3.2.2.1 Moving Pictures Quality Metric (MPQM)

MPQM is based on a basic vision model. The input to the metric is an original video sequence and a distorted version of it. The distortion is first computed as the difference between the original and the distorted sequences. The original and the error sequences are then decomposed into perceptual channels by the linear transform stage. It uses a bank of Gabor filters [135]. There are seventeen spatial filters and two temporal filters. From this stage, a total of 34 perceptual components for each sequence are to be computed.

The next stage models pattern sensitivity, using a contrast sensitivity function estimation and a linear summation model of masking. They are to be computed for every pixel of each perceptual component of the error sequence. The contrast threshold values are then used to divide the perceptual component of the error sequence (on a pixel-by-pixel basis).

The data is then gathered together to yield a single figure and to account for higher levels of perception, which is termed pooling. This step is computed as follows. First, the pooling is computed over blocks of the sequence. Such blocks are three-dimensional (one to consider the temporal information and the other two to consider the spatial information). The spatial dimension is chosen to consider focus of attention, i.e. the size is computed so that a block covers two degrees of visual angle. The distortion measure is computed for each block by pooling the error over the channels. The actual computation of the distortion E for a given block is given by

$$E = \left(\frac{1}{N} \sum_{c=1}^{N} \left(\frac{1}{N_x N_y N_t} \sum_{t=1}^{N_t} \sum_{y=1}^{N_y} \sum_{x=1}^{N_x} |e(x,y,y,c)| \right)^{\beta} \right)^{\frac{1}{\beta}},$$

where $e(x, y, t, c)$ is the masked error signal at position (x, y) and time t in the current block and in the channel c; N_x, N_y and N_t are the horizontal and vertical dimensions of the blocks; N is the number of channels. β is a constant having the value 4. The final quality measure can be expressed either using the Masked PSNR (MPSNR) as follows

$$MPSNR = 10 \log \frac{255^2}{E^2},$$

or can be mapped to the 5-point ITU quality scale using the equation

$$Q = \frac{5}{1 + \gamma E},$$

where γ is a constant to be obtained experimentally. This can be done if the subjective quality rating Q is known for a given sequence, E has to be computed and then one solves for γ from the last equation.

3.2.2.2 Color Moving Pictures Quality Metric (CMPQM)

CMPQM is an extension of the MPQM metric to consider the effect of color on the quality. The first step in the computation of the metric is to transform the uncorrected color components into RGB values that are linear with luminance. The second step converts the linear RGB values into coordinate values in the chosen opponent-colors space. The three coordinates of this color-opponent space correspond to luminance (B/W), red-green (R/G), and blue-yellow (B/Y) channels. Then, each color component of the original and error images (B/W, R/G, and B/Y) is analyzed by a filter bank created by Gabor functions tuned in spatial frequency and orientation. The bank uses the same filters as the MPQM, except in numbers. The B/W pathway is processed as the luminance data are processed by MPQM, i.e. it is decomposed into 34 components by the seventeen spatial and two temporal filters. It is known that the R/G and B/Y pathways are much less sensitive than the B/W one. Therefore, a restricted number of filters is used for such pathways. Nine spatial filters are used with a single temporal low-pass filter. The remaining of the computation is as in the MPQM. Contrast thresholds are computed and the data is pooled over the channel to yield a distortion measure.

3.2.2.3 Normalization Video Fidelity Metric (NVFM)

NVFM is implemented based on a visibility prediction employing a normalization mode. The perceptual decomposition is done separately in time and space in the pixel domain, without using Fourier transforms. Spatial (or space) decomposition uses the steerable pyramid [135]. The transform decomposes an image into spatial frequency and orientation bands. It is implemented as a recursive pyramidal scheme. The number of bands has been chosen to be equal to the Gabor implementation, as in MPQM. The orientation decomposition is steerable for each frequency band to be invariant by rotation. A temporal filter bank is used to do the temporal decomposition. The bank also has to approximate the Gabor temporal bank using infinite impulse response (IIR) filters. Then, an excitatory-inhibitory stage is used to measure the energy by squaring the output of the linear transform stage. After that, a normalization stage consisting of calculating the ratio between the excitatory and inhibitory mechanisms is carried out.

The final measure of the metric is a squared-error norm, calculated as the squared vector sum of the difference of the normalized responses. Let R_o be a vector collecting the sensor outputs for the original sequence and R_d be that of the distorted one. The detection mechanism computes the vector distance $\triangle R$ using

$$\triangle R = |R_o - R_d|^2.$$

3.2.3 Other Works in Objective Video Quality Measures

In [153], ITS, MPQM and NVFM quality assessment metrics are evaluated for a wide range variation of bit rates. The study showed that the metrics based on HVS give better results than those based on pure mathematical calculations like PSNR. However, the study showed that MPQM and NVFM give unreliable results for low bit rates.

In [104], a methodology for video quality assessment using objective parameters based on image segmentation is presented. Natural scenes are segmented into plane, edge and texture regions, and a set of objective parameters are assigned to each of these contexts. A perception-based model that predicts subjective ratings is defined by computing the relationship between objective measures and the results of subjective assessment tests, applied to a set of natural scenes and MPEG-2 video codecs. A logistic curve is used to approximate the relationship between each objective parameter and the subjective impairment level. Thus an impairment level is estimated for each of the objective parameters. The final result is computed by a linear combination of the estimated impairment levels, where the weight of each impairment level is proportional to its statistical reliability.

Another objective video quality assessment method, which is similar to that presented in [104], is given in [35]. In the article, a morphological video segmentation is used to estimate the subjective quality. Each frame is segmented into three regions: homogeneous, border and texture. The segmentation process uses a collection of morphological tools such as connected smoothing filters, morphological gradients, and watershed. This method has the same characteristics of that presented in [104]. However, the objective results do not correlate with subjective ones, as the MSE ranges from 24.5 to 67.3 % for the 5 tested video sequences. In addition, the only tested impairments are those resulting from encoding/decoding, and no other kind of impairments are studied.

In [130, 131], an objective picture-quality measurement model for MPEG video is presented. The theory of this method is to design a picture-quality model for each kind of known distortion, and to combine the results from the models according to the perceptual impact of each type of impairment. The model consists of quantifying both the blurring and the blockiness effects resulting from the encoding/decoding processes. The blockiness is estimated by using a blockiness detector, and the blurring is measured by a perceptual model. The outputs are then combined to give the perceptibility of the distortion. The final rating is either the output of the blurring model or the blockiness detector. The selection is based on the average amount of blockiness measured in the decoded sequence. When it is high, the blockiness detector's score is selected. Otherwise, the score of the perceptual model is chosen.

In [62], another objective quality measure is based on the time transition of quality of each frame. By obtaining the quality of each frame and applying a weighting function of all the frames in the sequence, the quality is obtained for the whole sequence. The employed frame-quality assessment method is similar to that in [130, 131].

In [33], a study of video quality metrics for MPEG-2 video is given. The authors are more interested in end users' points of view for packet video. A state-of-the-art in the definition of video quality metrics is given. They evaluate also ITS and MPQM metrics versus simply the bit rate as as the work presented in [153]. From that the ITS gives very bad results for low bit rates.

In [119], an objective video quality assessment method for videoconferencing and videophone systems is introduced. It is based on the Temporal Frequency Response measurement technique. The authors use only four test conditions to evaluate the performance of their method. In addition, the only parameter investigated is the bit rate.

From [143], it is clear that the correlation is not always good, and some times there is no correlation at all (the same as the results obtained from [24, 130, 131]).

A state-of-the-art for some objective video quality assessment methods for MPEG-2 video is given in [102].

3.3 Multimedia Transport Protocols

The goal of multimedia transport protocols is to transmit multimedia signals from one point to another point. These points are connected by communication network employing specific protocols. Generally, multimedia original signals are encoded to reduce the bit rate. When the encoded stream is to be sent to another location in the network, the transport protocols are responsible for the packetization and the delivery of that stream. At the other side, the encoded multimedia stream is reconstructed from the stream of delivered packets and then decoded to produce a useful multimedia signal to be played back or stored for further use.

The Internet Protocol (IP) is a packet-based network protocol used to exchange data over networks. It is the underlying network protocol, i.e. other protocols are built over IP. The most used higher level protocol is the Transport Control Protocol (TCP), which is a reliable transport protocol designed for data transmission and extensively used in Internet services. TCP is not suitable for real-time applications as the retransmissions can lead to high delay and cause delay jitter, which significantly degrades the quality. In addition, it does not support multicast. Also, congestion control mechanisms, namely slow start, are not suitable for audio or video media transmission.

On the other hand the transport protocol that is generally used for real-time multimedia transmission is the User Datagram Protocol (UDP). UDP does not guarantee the arrival of the packet, it is up to the application or higher level protocols to take care of the sent data. The most used, for real time applications, protocol which is built over UDP is Real time Transport Protocols (RTP). The most important variables governing the operation of RTP are the *Time Stamp* (TS) and the *Sequence Number* (SN). The TS is responsible for placing the incoming packets in correct timing order. The initial value of the TS is selected randomly and independently for each RTP stream. The TS value is increased by the time indicated by each packet. For example, for the case of audio transmission with 20 ms as packetization interval, the TS may take the values 0, 20, 40, 60, ... for packet numbers 1, 2, 3, ... respectively. The SN is used to detect packet loss occurrences. It is increased by one for each packet in the stream. It should be mentioned that for a video frame that is split into multiple RTP packets, these packets share the same value of TS but use different SN.

There is a separate control protocol that is generally used with RTP, which is named Real Time Control Protocol (RTCP). RTCP synchronizes across different media streams by feedback messages (e.g. the sender report). It also provides feedback on the quality of data transmission by using lost packet counts in the Receiver Report. In addition, it identifies and keeps track of the participants. RTCP reports are sent periodically (every 5 sec.) between participants with the restriction that its traffic should not exceed 5% of the total data traffic.

RTP supports multicasting, payload type identification, time stamping, sequence numbering, delivery monitoring. In addition, the underlying UDP protocol supports multiplexing and checksum services.

Even if RTP is the most used protocol for real-time applications, it has some problems. First, it does not support the timely delivery of data or any other QoS guarantee. In-time delivery requires lower layers that have control over resources in switches or routers (e.g. Resource Reservation Protocol, RSVP). Second, it does not guarantee delivery, so packets may be delivered out of order or get lost. In addition, there is no mechanism to recover from packet loss.

3.3.1 Challenges of Multimedia Transmission

Some of the challenges of carrying multimedia traffic over the Internet are: the lack of guarantee in terms of the bandwidth, packet loss, delay, and jitter, which affect the quality [60]. In this Section, we discuss the major problems caused by the network.

- **Packet Loss** – Packets are sent from the source to the destination across several routers. Each router may get packet streams from many sources at the same time. When the packet arrival

process fills the buffer of any of the routers, some packets are dropped. The network in this case is said to be congested. Packet loss may produce great damage to the received multimedia signal. For example, for the case of speech, each packet contains 40-80 ms of speech information to match the duration of critical units of speech called *phonemes*. When a packet is lost, a phonemes is lost in continuous speech. While the human brain is capable of reconstructing a few lost phonemes in speech, too many packet losses make a voice unintelligible. The same problem exists for video, some portions (blocks) of the image can not be decoded and displayed. The result is severe degradation of the reconstructed signal quality.

The problem becomes more severe when the signal is compressed too much. Natural signal contains in general too much redundant information that can be interpolated by human brain when there is loss. However, to reduce the bitrate, the signal is compressed to remove the redundancy, and thus, packet loss becomes more annoying. There are two types of techniques to mask the effects of packet loss: some focus on reducing packet loss, others concentrate on repairing the damage caused by loss. To reduce packet loss; Network upgrade (increasing the bandwidth, using ATM, SONET [Synchronous Optical NETwork] for gigabits/s, Wavelength Division Multiplexing [WDM] for terabit/s) is needed. This solution is very expensive and does always available. We overview the other methods dealing with packet loss in the next Subsection.

- **Packet Delay** – To understand the effect of packet delay on multimedia quality, we take the case of voice transmission. Timing is an important characteristics of voice. Two syllables of a word are uttered with an interval. This interval is as much a part of the voice as the uttered syllable. If additional delay is inserted between syllables, the rhythm of voice is lost. Long delay may force conversation to be half-duplex or introduce echo. Delay below 150 ms is acceptable. For long distance communication delay between 150 and 400 ms is also acceptable.

  The end-to-end delay typically consists of: codec delay, serialization delay (the time it takes to place a packet in the transmission line), queuing delay (in routers, etc.), propagation delay (time to travel from point to another). From that it is clear that to reduce the delay effect we have several possibilities: (i) using codecs that can run in real time without too much delay, (ii) reduce the serialization delay, (iii) increase the routers speed to decrease the queuing delay or using some kind of Differentiated Services, (iv) reduce the length of the phisical material to reduce the propagation delay (this can be done by selecting the shortest pass from the source to the destination in the network layer). It should be mentioned that delay effect is not relevant in the case of one-way sessions (video streaming, etc.). However, the effect of delay play an important role in the case of interactive two-way sessions [60].

- **Network Jitter** – It is the variance of the inter-packet arrival time at the receiver. Network jitter occurs due to the variability in queuing and propagation delays. To alleviate from jitter, a jitter buffer is used. The receiver holds the first packet in the buffer for a while before playing it out. The amount of the hold time is the measure of the size of the jitter buffer. For example, a 50 ms hold time means 50 ms jitter buffer.

  Jitter, on the other hand, plays an important role in both types of applications (interactive or one-way), and its effects are similar to those of packet loss. If a dejittering buffer [37] 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. As packets that arrive after some expiration time are considered lost. Finally, for interactive applications, the echo effect should also be considered, and an echo suppression or cancellation mechanism should be implemented [40].

3.3.2 Working Around Loss

To protect against packet drop (channel error) there are two types for coding error resiliency: Forward Error Correction (FEC) and Automatic Repeat reQuest (ARQ). FEC is used for error correction at data link layer. The goal is to localize and to correct errors (packet drops) by adding redundancy codes to the original data. However, it cannot handle burst error. In addition, encoding delay is very large as the encoder should process block of packets before start sending them. Some existing FEC coding schemes are Reed Solomon, Turbo Codes, Hamming Codes, etc.

On the other hand, ARQ is done by requesting the retransmission of the data (dropped packets) when an incorrect packet is received. This can be detected by negative acknowledgment (NACK) after a timer expiration, receiving packets out of order, or by implicit acknowledgment (ACK). It maybe suitable for one way real time video applications such as VoD. However, it is not very effective for interactive services such as video conferencing, due to the too long delay that can be required to retransmit the lost packets.

There is also a hybrid channel coding between FEC and ARQ. For a survey of packet loss protection techniques, namely ARQ and FEC for video transmission, see [160].

Unlike FEC and ARQ, which try to protect against error occurrence or assure the transmission of data correctly, error concealment techniques' goal is to mask or to minimize the effect of error after its occurrence. There are many error concealment techniques. However, due to the different nature of speech or audio and video signals, there are specific techniques for each media type.

3.3.2.1 Error Concealment for Speech or Audio

The repairing or loss recovery techniques, as stated in [60], for speech are summarized as follows.

(i) **Silence Substitution** – substitution of the lost packet with silence. That technique causes voice clipping. This deteriorates voice quality when packet size is large, and loss rate is high.

(ii) **Noise Substitution** – substitution of lost packet by white (background) noise is better than silence substitution. This has been attributed to the ability of the human brain to repair the received message if there is some background noise (known as *phonemic restoration*).

(iii) **Packet Repetition** – replaying the last correctly received packet in place of the lost one. The Global System for Mobile (GSM) recommends that the repeated signal be damped or faded to ensure better quality.

(iv) **Packet Interpolation** – Interpolation-based repairs use the characteristics of the packets in the neighborhood of the lost one to produce a replacement. This technique is also known as waveform repetition. This ensure that the replacement will follow the changing characteristics of the whole voice stream, and hence yield better quality.

(v) **Frame Interleaving** – The effect of packet loss can be reduced by interleaving voice frames across different packets. The loss of a single packet will produce multiple short gaps in different streams, which is more tolerable than long gaps. But this technique produces longer delays.

3.3.2.2 Error Concealment for Video

The main purpose of error concealment schemes for video is to limit the impact of loss over the image quality. These schemes take advantages of the redundancy present in the image in the frequency, spatial, and temporal domains. Under these schemes part of the lost information can be recovered by regenerating most of it by means of interpolation in each of these domains [34].

From the literature [52, 158, 34], error concealment techniques for video codecs are as given below.

(i) **Block Replacement** – The simplest method to interpolate the lost areas is to replace them by the corresponding areas of the previous frame or field. This temporal extrapolation works quite well in still parts of the pictures (e.g., backgrounds) but fails in areas where there is a lot of motion.

(ii) **Linear Interpolation** – It is based on the Intra-frame linear interpolation that replaces the lost areas with the linearly interpolated values, calculated from the neighboring areas of the same frame. This method works quite well in a uniform surface with smooth gray-level changes inside the block. This is provided that the surrounding areas are correctly received. If this is not true, which is the case in networks, this methods does not give good results.

(iii) **Motion Vector** – The lost areas are replaced with pixel blocks of the previous frame shifted by the average motion vector of the neighboring blocks. The picture quality is greatly improved, but the performance drops when the surrounding blocks have different motion vectors.

(iv) **Hybrid Technique** – Concealing by simple motion vector estimation and simply averaging the top and bottom correctly received MBs method gives better results. In addition, concealing by more than one motion vector and averaging all the neighboring MBs can do an even better job but of course at the cost of more computational complexity and delay. The hybrid techniques use both spatial redundancy and more than one motion vectors information to predict the lost MB.

3.4 Video Compression and Standard Codecs

The goal of video compression [9] is to remove the redundancy in the original source signal. This help in reducing the bandwidth necessary for transmitting the signal from point to another or allowing smaller space to store the video information. In multimedia real-time codecs, the goal of video compression is not only to minimize the bit rate of the given video signal, but also to guarantee some required levels of signal quality, to minimize the complexity of the used codec, and to reduce the overall encoding/decoding delay. However, it is not easy to satisfy these four requirements in the same time. Hence, a tradeoff between them is used to choose a compression scheme.

Video compression is possible because of the following points:

- There is spatial redundancy among neighboring pixels in a given picture (frame),
- There is spectral redundancy among color images within frames,
- There is temporal redundancy among pixels or blocks of pixels in different frames,
- There is considerable irrelevant (from a perceptual viewpoint) information contained in video data.

A good compression scheme can exploit these four points to produce high compression rates and in the same time maintains a good quality level.

Regarding the irrelevant information in video data, it is found that human eyes are not so sensitive to some non-redundant information. This is possible by employing lossless encoding techniques (like Huffman coding) and approximating the information, which is not important for the sake of human perception of the quality. For example, human eyes do not distinguish between two video sequences one of them encoded at 30 frames per second and the other at 60 frames per second. In addition, they are more sensitive to the variation of luminance than chrominance information. Thanks to the

research in Human Visual System (HVS), there are nowadays many video encoders that make use of the above facts and provide reasonable quality and compression rates.

In some situations (most of the current Internet video applications, for example), even by employing the previously mentioned encoding techniques, the output stream is not compressed enough to be fitted in the available bandwidth or the storage device. In this case the above techniques are mixed with another lossy encoding techniques to achieve certain compression ratio. In this case, the quality of the resulting stream is degraded with respect to the original video signal. This is good for some applications like the cheep Internet videoconferencing, video phones, or when some of the details of the video signal are not very important (video archiving for instance). In general, the more the quality degradation is accepted, the better the compression ratio is.

The characteristics of the three types of video compression (lossless, lossy and hybrid) coding are given here.

- **Lossless coding**: It is a reversible process with the ability of perfect recovery of original data. Thus, video signal before lossless encoding is identical to that after decoding the encoded signal. Therefore, there is no quality degradation due to lossless coding. However, only low compression ratios can be obtained. An example of lossless coding is the *"Entropy coding"*. The data is taken as a simple digital sequence. Some Entropy coding algorithms are run-length coding (RLC), Huffman coding, and Arithmetic coding.

- **Lossy coding**: It is an irreversible process in which the recovered data is degraded. Hence, the reconstructed video quality is sometimes degraded with respect to that of the original video. An example of lossy coding is the *"Source coding"*.

- **Hybrid coding**: It combines both lossy and lossless coding. It is used by most multimedia systems. Examples of video codecs that use Hybrid coding are JPEG, H.263, and MPEG1/2/4.

We provide here the two kind of algorithms used to remove both spatial and temporal redundancies.

3.4.1 Spatial or Block Coding

Spatial coding exploit the redundancy within the same frame. Each frame is decomposed into blocks of 8x8 pixels, where pixel values range from 0 to 255 (8 bits per pixel). These values are shifted to -127 and -128 (centered around zero). After that, Discrete Cosine Transform (DCT) is used to map the spatial data to the frequency domain (64 coefficients). The (0,0) coefficient represents the DC value. The DCT coefficients are then quantized to reduce their spread. The aim of this process is to zero out the higher frequencies. After that, a block-to-vector conversion is used in preparation for the entropy coding. This conversion is done by a zigzag pattern to scan the block to create a 64-element vector.

After that, Run Length Coding (RCL) is used to replace the sequence of same consecutive bytes with the number of occurrences. Then a *"variation-zero suppression"* process is used to encode long binary bit strings containing mostly zeros.

3.4.2 Temporal Coding

The goal of temporal coding is to remove the temporal redundancies among frames. The basic unit is a macro block of 16x16 pixels. The process consists in searching the neighboring macro blocks of the given macro block in the different subsequent frames to find the location of the current macro block in these frames. If a good match (some times approximated when privileging the compression rate to the quality) is found, then instead of encoding this macro block in the current frame and all the frames containing a good match of it, simply a motion vector indicating the location of that block is encoded for each matching frame.

Table 3.1: Standard video resolution formats.

Acronym	Meaning	Resolution (pixels)
CIF	Common Interchange Format	352x288
QCIF	Quarter CIF	176x144
SQCIF	Sub-QCIF	128x96
4CIF	Four CIF	704x576
16CIF	16 CIF	1408x1152

Search windows can be any size. However, the larger the window size is the better the motion estimate, but the higher the computational cost and hence longer encoding delay (too much delay is not acceptable for real-time applications). This process reduces considerably the bit rate. This is because most of natural video scenes have many redundant parts spread into several frames. The situation where the redundant blocks are very small occurs when there is scene cuts (the picture changes abruptly and is replaced by another one). In addition, it occurs when the scene contains too much motion with motion frequency is very large compared to the encoding frame rate. There are some approximations of that step depending on the compression factor required, the tolerated encoding latency, etc.

3.4.3 Standard Video Codecs

The most important video codec standards for streaming video are H.261, H.263, MPEG-1, MPEG-2 and MPEG-4. In this subsection, we focus on their principle functionality and usage.

3.4.3.1 H.261 Video Coding

H.261 is an ITU video-coding standard, designed original to suit ISDN lines. Its output bit rates is aimed to be multiples of 64Kbits/s. The encoding algorithm employed is a mixture of temporal and spatial coding to remove the redundancies in the video. The resulting data rate was designed to be set between 40 Kbits/s and 2 Mbits/s. Motion vectors are used to help the codec compensates for motion.

The encoding algorithm used is similar to that of MPEG. However, H.261 has smaller computational power than MPEG. H.261 is constant-bit-rate codec and not constant-quality, variable-bit-rate encoding. This is because the encoding algorithm trades the picture quality against motion. Hence, scenes having a large amount of motion will have lower quality than those having small amount of motion. Compared to MPEG picture types, it has I and P frames. The output stream is IPPPP... of encoded pictures (frames).

H.261 was targeted at teleconferencing applications and specially for face-to-face videophone applications and for videoconferencing. There are only two supported resolutions CIF (Common Interchange format), and QCIF (Quarter CIF) (see Table 3.1).

3.4.3.2 H.263 Video Coding

H.263 is another ITU-T video encoding standard. It was designed for low bitrate communication, less than 64 Kbits/s, however this limitation has now been removed, and it supports higher bit rates.

The coding algorithm of H.263 is similar to that used by H.261, however there are some improvements and changes to improve performance and error recovery. The differences between the H.261 and H.263 coding algorithms are that H.263 uses half pixel precision for motion compensation rather than full pixel. H.263 can be configured for lower bit rates or alternatively for better error recovery.

Like H.261, it has IPPPP... frames, however, some macroblocks of the P frames are encoded in Intra mode without motion prediction. This feature makes it very attractive to be used in real-time video transmission over the Internet, because the output stream is more resilient to packet loss.

H.263 supports five resolutions. In addition to QCIF and CIF that were supported by H.261 there is SQCIF, 4CIF, and 16CIF. SQCIF is approximately half the resolution of QCIF. 4CIF and 16CIF are 4 and 16 times the resolution of CIF respectively.

3.4.3.3 Overview of MPEG-1/2/4 Coding

MPEG is an abbreviation for Moving Picture Expert Group. There are three standards already completed by MPEG, namely MPEG-1, MPEG-2, and MPEG-4. Both MPEG-1 and MPEG-2 standards are similar in basic concepts. They both are based on motion compensated block-based transform coding techniques similar to those employed in H.263. However, MPEG-4 employs more sophisticated approaches including the use of software image construct descriptors, for target bit-rates in the very low range, $\leq$ 64Kb/sec (as described in the initial specifications of MPEG-4). However, the success of MPEG-4 and the impractical ability to fit the bit rate below 64Kb/sec, make it possible now to use it for higher bit rates.

MPEG uses the notation of layers. This is to help with error handling, random search and editing, and synchronization, for example with an audio bitstream. The top layer in the hierarchy is the video sequence layer. It is a self-containing bitstream such as a movie or a football game. The second layer down is the group of pictures (GOP), which is composed of 1 or more groups of Intra (I) frames and/or non-Intra (P and/or B) pictures. The most famous GOP is IBBPBBPBBPBB... The third layer is the picture layer, the processing of each kind of these pictures. The next layer is the slice layer, which is a contiguous sequence of macroblocks. Each macroblock is 16x16 arrays of luminance pixels, and two 8x8 arrays of associated chrominance pixels. The macroblocks can be further divided into distinct 8x8 blocks, for further processing such as transform coding.

Intra frames (I for short) are encoded without any temporal compression techniques. Only lossy and lossless coding are employed on the current picture without any reference to the adjacent frames. On the other side, Inter frames (P or B) are encoded by taking into account the motion prediction techniques (removing the temporal redundancy among frames) as well as those employed to encode I frames. Hence, the bit rate of I frame are very large compared to both P and B frames.

In a GOP, each P frame is predicted from either I or P frames immediately preceding it. Thus temporal compression techniques are employed between these frames to generate the P frames. The major disadvantage of using P frames is the error propagation effect. Any error (transmission data loss) in any I frame will propagate to all the pictures in the GOP, as all the P pictures are based on I frames. One of the advantages of H.263 and H.263+ codecs is that they encode P frames from the previous P frames, and choose randomly (in a cyclic way) some macroblocks in each P frames to encode them completely without any motion prediction (only spatial redundancy are removed). The advantage of doing that is to generate near constant bit rate and maintain good quality with sufficient error resiliency. Knowing that (in average the size of I frame is at least the double of P frame). Hence, quality smoothing techniques against bit rate in real-time video is easier and more powerful in H.26x codecs than MPEG codecs.

Regarding the B frames (bi-directional interpolated prediction frames), they are generated by using forward/backward interpolated prediction. For a simple GOP of I, B, P, B, P, B frames, as previously mentioned, I frames are encoded spatially only and the P frames are forward predicted based on previous I or P frames. The B frames, however, are coded based on a forward prediction from a previous I or P frame, as well as a backward prediction from a succeeding I or P frame. As such, the example sequence is processed by the encoder such that the first B frame is predicted from the first I frame and first P frame, the second B frame is predicted from the second and third P frames, and the third B frame is predicted from the third P frame and the first I frame of the next GOP. Consequently,

it is clear that to decode any B frame, the future P or I frame used to encode the B frame should be encoded and sent to the decoder to be able to decode the B frame.

As it may be seen, the B frames are predicted from two images (previous and next frames), the encoder has more ability to correctly identify the motion vectors. This increase the efficiency of the compression rate. Thus, B frame size is typically very small with respect to that of P frame, which in turn small compared to that of I frame. It should be noted that, since B frames are not used to predict future frames, errors generated in B frames will not be propagated further within the sequence. Only those generated in I or P frames will propagate in the sequence of GOP.

One disadvantage is that the frame reconstruction memory buffers within the encoder and decoder must be doubled in size to hold the two frames required to encode or decode B frame. Another disadvantage is that there will necessarily be a delay throughout the system as the frames are delivered out of order.

The MPEG-1 encoder demands very high computational power, normally it requires dedicated hardware for real-time encoding. However, decoding can be done in software. MPEG-2 requires even more expensive hardware to encode video in real time.

Both MPEG-1 and MPEG-2 are well suited to the purposes for which they were developed. For example, MPEG-1 works very well for playback from CD-ROM, and MPEG-2 is great for high-quality archiving applications and for TV broadcast applications. However, for existing computer and Internet infrastructures, MPEG-based solutions are too expensive and require too much bandwidth.

To overcome this problem (among others), MPEG provided another standard named MPEG-4. It is aimed to be suitable for video conferencing (low bit rate, and can work in real time). MPEG-4 is based on the segmentation of audiovisual scenes into audio/visual objects that can be multiplexed for transmission over heterogeneous networks. The MPEG-4 framework currently being developed focuses on a language called MSDL (MPEG-4 Syntactic Description Language). MSDL allows applications to construct new codecs by composing more primitive components and providing the ability to dynamically download these components over the Internet.

3.5 Audio Compression and Codecs

Unlike video data, audio and speech data is hard to compress effectively. This is because audio signals do not contain as much redundant information as video, namely temporal correlation between macroblocks in consecutive frames.

Commonly used speech coding techniques may be classified into three classes - waveform, source and hybrid coding. Waveform coding is used at high bit rates. The resulting encoded signal has very good quality. Source coding, on the other hand, operates at very low bit rates. Thus, as expected, the resulting signal quality is degraded but intelligible, the signal sounds synthetic. A mixture between these two coding techniques is generally used which is the hybrid coding. Hybrid codecs give good quality at moderate bit rates. We discuss all these coding schemes as well as the most used audio codecs in this Section.

3.5.1 Waveform Coding

Waveform coding is some kind of approximately lossless coding, as it deals with speech signal as any kind of ordinary data. The resulting signal is close as possible as the original one. Codecs using this techniques have generally low complexity and give high quality at rates $\geq$ 16 Kbps.

The simplest form of waveform coding is Pulse Code Modulation (PCM), which involves sampling and quantizing the input waveform. Narrow-band speech is typically band-limited to 4 KHz and sampled at 8 KHz.

Many codecs try to predict the value of the next sample from the previous samples. This is because there is correlation between speech samples due to the nature of speech signal. An error

signal is computed from the original and predicted signals. As in most cases, this error signal is small with respect to the original one, it will have lower variance than the original one. Hence, fewer bits are required to encode them. This is the basis of Differential Pulse Code Modulation (DPCM) codecs. They quantize the difference between the original and predicted (from the past samples) signals.

The notion of adaptive coding is an enhancement to DPCM coding. This is done by making the predictor and quantizer adaptive so that they change to match the characteristics of the speech being coded. The most known codec using this technique is the Adaptive DPCM (ADPCM) codecs.

It is also possible to encode in the frequency domain instead of the time domain (as the above mentioned techniques). In Sub-Band Coding (SBC), the original speech signal is divided into a number of frequency bands, or sub-bands. Each one is coded independently using any time domain coding technique like ADPCM encoder. One of the advantages of doing this is that all sub-band frequencies do not influence in the same way the perceptual quality of the signal. Hence, more bits are used to encode the sub-bands having more perceptually important effect on the quality than those where the noise at these frequencies is less perceptually important. Adaptive bit allocation schemes may be used to further exploit these ideas. SBC produces good quality at bit rates ranging form 16 to 32 Kbps. However, they are very complex with respect to the DPCM codecs.

As in video spatial coding, Discrete Cosine Transformation (DCT) is used in speech coding techniques. The type of coding employing this technique is the Adaptive Transform Coding (ATC). Blocks of speech signal is divided into a large numbers of frequency bands. The number of bits used to code each transformation coefficient is adapted depending on the spectral properties of the speech. Good signal quality is maintained using ATC coding at bit rates of about 16 Kbps.

3.5.2 Source Codecs

As in video coding, high compression ratio is achieved by understanding the video signal and the Human Vision System (HVS), the same idea is used in source speech coding. The main idea is based on the understanding of how the speech signal is produced. Instead of sending the coded signal, some parameters of the signal are sent to the decoder. Original speech signal contains two types of time-varying signals (constituting the majority of the speech). They involve voiced sounds that have a high degree of periodicity at the pitch period and unvoiced sounds that have little long-term periodicity but contain short-term correlations. Thus the vocal tract is represented as a time-varying filter and is excited with either a white noise source, for unvoiced speech segments, or a train of pulses separated by the pitch period for voiced speech.

Therefore the information that must be sent to the decoder is the filter specification, a voiced/unvoiced flag, the necessary variance of the excitation signal, and the pitch period for voiced speech. This is updated every 10-20 ms to follow the non-stationary nature of speech. This kind of coders are also referred as vocoders. Vocoders operate at bit rates $\leq$ 2.4 Kbps. The resulting speech is intelligible and almost synthetic as most of the natural characteristics of the speech signal is removed.

3.5.3 Hybrid Codecs

Hybrid coding is an intermediate between waveform and source coding. Analysis-by-Synthesis (AbS) coding, which is performed in time domain, is the most famous type of hybrid coding. AbS codecs operate exactly as vocoders (source coding), but instead of using only two-state voiced-unvoiced model to determine the parameters of the linear prediction filter, the excitation signal is chosen by attempting to match the reconstructed speech waveform approximately the same as the original speech waveform. The speech signal is split into 20ms frames. For each frame, parameters are determined for a synthesis filter, and then the excitation to this filter is determined. This is done by finding the excitation signal which when passed into the given synthesis filter minimizes the error between the input speech and the reconstructed speech. Finally, for each frame, the encoder transmits information representing the

synthesis filter parameters and the excitation to the decoder, and at the decoder the given excitation is passed through the synthesis filter to give the reconstructed speech.

There are several types of AbS codecs. We can cite Multi-Pulse Excited (MPE), Regular-Pulse Excited (RPE), and the Code-Excited Linear Predictive (CELP) codecs. MPE and RPE codecs give good quality at rates of about 10 Kbps. CELP codecs can give the same quality at rates in the range 2.4 and 10 Kbps.

3.5.4 Standard Audio Codecs

In this Section, we discuss the most used audio and speech codecs that can be used in real-time streaming. The list is not exhaustive as there are many working audio codecs, but we focus on the most popular, namely PCM, ADPCM, GSM, CELP, and LPC. The sampling rate distinguishes codecs used for speech from those used for audio (music) codecs. Speech (voice) codecs use in general 8000 sample per second. In most cases, it is sufficient to use mono channel. Whereas, in audio demanding high fidelity (like music), higher sampling rates are used (up to 44.1 KHz, for CD quality), and stereo channels are used some times.

3.5.4.1 Pulse Code Modulation (PCM)

PCM encoding is from the waveform encoding family. Speech is sampled at 8KHz. Each sample is then quantized either by linear or non-linear (logarithmically) quantization. For linear quantization, a 12 bits is used to encode each sample that gives 96 Kbps. On the other hand, when using non-linear quantization, only 8 bits are sufficient to encode each sample giving a bit rate of 64 Kbps. The quality of the encoded signal, in the latter, is approximately the same as the original one (about 4.6 on the 5-point ITU subjective quality scale).

There are two variant of PCM codecs: A-law and u-law codecs, which are used in America and Europe respectively. They are also referred as the ITU-T G711 codec. They are widely used because of their simplicity and the very good quality they offer without significant complexity. It is easy to use them in software encoder as they do not demand much CPU power. Hence they are widely used in real-time Internet audio conferencing applications. The only disadvantage of PCM codecs is the resulting bit rate, which is sometimes high with respect to other codecs. Thus it is not suitable for archiving too long speech or audio material.

3.5.4.2 Adaptive Differential Pulse Code Modulation (ADPCM)

Like PCM codecs, ADPCM codecs use also waveform encoding. However instead of quantizing the speech signal directly, the difference between the speech signal and a prediction that has been made of the speech signal is quantized. This is based on the fact that the next speech sample can be predicted from the past samples. If the prediction is accurate then the difference between the real and predicted speech samples will have a lower variance than the real speech samples, and will be accurately quantized with fewer bits than would be needed to quantize the original speech samples. At the decoder the quantized difference signal is added to the predicted signal to give the reconstructed speech signal.

Adaptive prediction and quantization are used to increase the performance of the encoder. The goal is to adapt the changing characteristics of the speech with the predictor and difference quantizer.

The standard G721 has as bit rate of 32 Kbps and gives subjective quality almost as that of PCM, about 4.1 score on the ITU-5-point quality scale. ITU recommendations G721, G726 and G727 codecs which provide 40, 30, 24 and 16 Kbps as bit rates. Each speech sample is encoded by 2, 3, 4, 5 bits using ADPCM codecs. However, the quality of the compressed signal decreases as expected with the decrease of the output bit rate.

3.5.4.3 CELP Codec

CELP is a hybrid codec using both waveform and source coding techniques. Thus, as expected, the process of determination of the filter coefficients introduces high delays (cf. PCM codecs). The delay of a speech codec is defined as the time from when a speech sample arrives at the input of its encoder to when the corresponding sample is produced at the output of its decoder, assuming the bit stream from the encoder is fed directly to the decoder. For a typical hybrid speech codec this delay will be of the order of 50 to 100 ms, and such a high delay can cause problems.

Thus, many efforts have been focused in providing a standard codec that has as bit rate 16 Kbps, while providing a quality comparable to that provided by the ADPCM 32 Kbps. The major challenge is to reduce the delay to about 5 ms. This is satisfied by a backward adaptive CELP codec, standardized in 1992 as ITU-T G728 recommendation. Another CELP codec is standardized in 1991 by the American Department of Defense (DoD). It operates at a 4.8 Kbps.

3.5.4.4 LPC codec

Linear Predictive Coding (LPC) is another standard of the DoS. It uses sources coding techniques. Thus, instead of sending the speech signal, a best fit between the original signal and a filter is calculated. Then the best matching parameters of this filter are sent to the decoder. An LPC decoder uses those parameters to generate synthetic speech that is usually more-or-less similar to the original. The result is intelligible but sounds like a machine is talking. The output bit rate is 2.4 Kbps.

3.5.4.5 GSM Codec

The "Global System for Mobile communications" (GSM) is a digital mobile radio system. GSM employs a Regular Pulse Excited (RPE) coding technique, which is a hybrid coding. The speech signal is divided into 20-ms frames. A set of 8 short term predictor coefficients are found for each frame. Each frame is then further divided into four 5-ms sub-frames, and for each sub-frame the encoder finds a delay and a gain for the codec's long term predictor. Finally the residual signal after both short and long term filtering is quantized for each sub-frame.

GSM full rate (GSM-FR) codecs give bit rates of 13 Kbps, with a subjective quality score of 3.6 on the 5-point ITU quality scale. The main advantage of GSM over other low rate codecs is it simplicity. A software encoder can operate in real time, while CELP needs a dedicated hardware to run in real time.

3.6 Neural Networks

As we use in all our work in this dissertation the neural networks (NN) as a tool, we provide in this Section an overview of them. We use both Artificial Neural Networks (ANN) and Random Neural Networks (RNN). Thus, both types are discussed.

3.6.1 Artificial Neural Networks (ANN)

An ANN is a parallel distributed associative processor, comprised of multiple elements (neural models) highly interconnected [117, 149, 150, 118]. Each neuron carries out two operations. The first is an inner product of an input vector (carrying signals coming from other neurons and/or other inputs) and a weight vector, where the weights represent the efficiencies associated with those connections and/or from external inputs. The second, for each neuron, a nonlinear mapping between the inner product and a scalar is to be computed, usually given by a non-decreasing continuous function (e.g., *sigmoid*, or *tanh*). When building ANN, an architecture and a learning algorithm must be specified. There are multiple architectures (e.g., multi-layer feedforward networks, recurrent networks, bi-directional

networks, etc.), as well as learning algorithms (e.g., Backpropagation, Kohonen's LVQ algorithm, Hopfield's algorithm, etc.).

As pattern classifiers, ANN work as information processing systems that search for a non-linear function mapping from a set $\mathcal{X}$ of input vectors (patterns) to a set $\mathcal{Y}$ of output vectors (categories). This mapping is established by extracting the *experience* embedded in a set of examples (training set), following the learning algorithm. Thus, in developing an application with ANN, a set of N known examples must be collected, and represented in terms of patterns and categories (i.e., in pairs (X_n, Y_n), $n = 1, 2, \ldots N$, where $X_n \in \mathcal{X}$ and $Y_n \in \mathcal{Y}$). Then, an appropriate architecture should be defined. Last, a learning algorithm needs to be applied in order to build the mapping.

Highly nonlinear mappings can be obtained using the backpropagation algorithm for learning and adaptation, and a three-layer feedforward neural network consisting of an input layer, a hidden layer and an output layer.

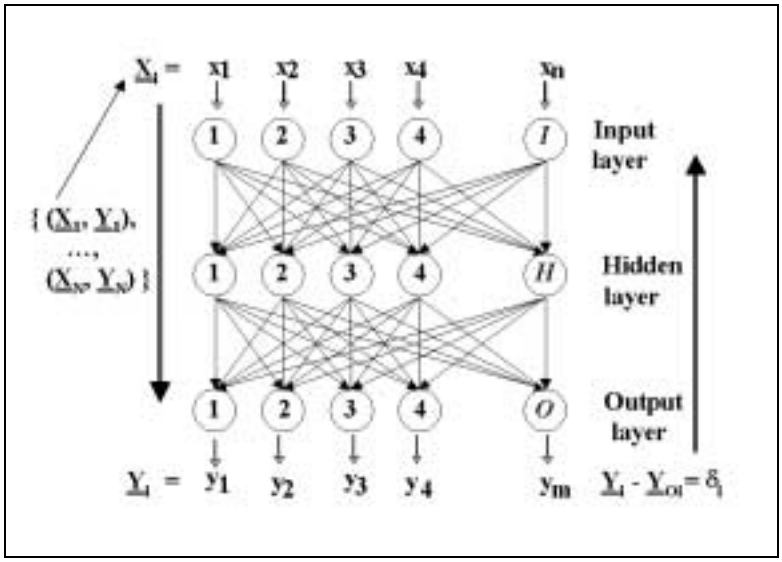


Figure 3.1: Architecture of a three-layer feedforward neural network.

In this architecture (see Figure 3.1), external inputs are the inputs for the neurons in the input layer. The scalar outputs from those neural elements in the input layer are the inputs for the neurons in the hidden layer. The scalar outputs in the hidden layer become, in turn, the inputs for the neurons in the output layer.

When applying the backpropagation algorithm, all the weights' initial values are given randomly. Then, for each pair (X_n, Y_n) in the database, the vector X_n (i.e., pattern) is placed as input for the input layer, and the process is carried out forward through the hidden layer, until the output layer response Y_{on} is generated. Afterwards, an error δ is calculated by comparing vector Y_n (classification) with the output layer response, Y_{on}. If they differ (i.e., if a pattern is misclassified), the weight values are modified throughout the network accordingly to the generalized delta rule:

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta x_j,$$

where, w_{ij} is the weight for the connection that neuron i, in a given layer, receives from neuron j from the previous layer; x_j is the output of neuron j in that layer; η is a parameter representing the learning rate; and δ is an error measure. In case of the output layer, $\delta = ||Y_n - Y_{on}||$, whereas in all hidden layers, δ is an estimated error, based on the backpropagation of the errors calculated for the output layer (for details refer to [117, 118]). In this way, the backpropagation algorithm minimizes a global error associated with all pairs (X_n, Y_n), where $n = 1, 2, \ldots N$ in the database. The training

process keeps on going until all patterns are correctly classified, or a pre-defined minimum error has been reached.

3.6.2 Random Neural Networks (RNN)

The Random Neural Networks (RNN) has been proposed by Erol Gelenbe in 1989 [45]. RNNs have been extensively studied, evaluated and analyzed in [49, 89, 48, 45, 101, 51]; and analog hardware implementations of RNN are detailed in [3, 25]. Low bit rate video compression using RNN is presented in [30]. A list of RNN applications are found in [2, 4, 6, 7, 50, 99, 123, 128, 148, 10].

Gelenbe's idea can be described as a merge between the classical Artificial Neural Networks (ANN) model and queuing networks. Since this tool is a novel one, let us describe here its main characteristics. RNN are, as ANN, composed of a set of interconnected neurons. These neurons exchange signals that travel instantaneously from neuron to neuron, and send and receive signals to and from the environment. Each neuron has a *potential* associated with, which is an integer (random) variable. The potential of neuron i at time t is denoted by $q_i(t)$. If the potential of neuron i is strictly positive, the neuron is *excited*; in that state, it randomly sends signals (to other neurons or to the environment), according to a Poisson process with rate r_i. Signals can be positive or negative. The probability that a signal sent by neuron i goes to neuron j as a positive one, is denoted by $p^+_{i,j}$, and as a negative one, by $p^-_{i,j}$; the signal goes to the environment (that is, it leaves the network) with probability d_i. So, if N is the number of neurons, we must have for all $i = 1, \cdots, N$,

$$d_i + \sum_{j=1}^{N} \left(p^+_{i,j} + p^-_{i,j} \right) = 1.$$

When a neuron receives a positive signal, either from another neuron or from the environment, its potential is increased by 1; if it receives a negative one, its potential decreases by 1 if it was strictly positive and it does not change if its value was 0. In the same way, when a neuron sends a signal, positive or negative, its potential is decreased by one unit (it was necessarily strictly positive since only excited neurons send signals)[1]. The flow of positive (resp. negative) signals arriving from the environment to neuron i (if any) is a Poisson process which rate is denoted by λ^+_i (resp. λ^-_i). It is possible to have $\lambda^+_i = 0$ and/or $\lambda^-_i = 0$ for some neuron i, but to deal with an "alive" network, we need $\sum_{i=1}^{N} \lambda^+_i > 0$. Finally, we make the usual independence assumptions between these arrival processes, the processes composed of the signals sent by each neuron, etc.

The discovery of Gelenbe is that this model has a *product form* stationary solution. This is similar to the classical Jackson's result on open networks of queues. If process $\vec{q}(t) = (q_1(t), \cdots, q_N(t))$ is ergodic (we will say that the network is *stable*), Gelenbe proved that

$$\lim_{t \to \infty} \Pr(\vec{q}(t) = (n_1, \cdots, n_N)) = \prod_{i=1}^{N} (1 - \varrho_i) \varrho_i^{n_i} \tag{3.1}$$

where the ϱ_is satisfy the following non-linear system of equations:

$$\text{for each node } i, \quad \varrho_i = \frac{T^+_i}{r_i + T^-_i}, \tag{3.2}$$

$$\text{for each node } i, \quad T^+_i = \lambda^+_i + \sum_{j=1}^{N} \varrho_j r_j p^+_{j,i}, \tag{3.3}$$

[1] In the general mathematical model, loops are allowed, that is, it is possible to have $p^+_{ii} > 0$ or $p^-_{ii} > 0$. In our application, we set $p^+_{ii} = p^-_{ii} = 0$.

and

$$\text{for each node } i, \quad T_i^- = \lambda_i^- + \sum_{j=1}^{N} \varrho_j r_j p_{j,i}^-. \tag{3.4}$$

Relation (3.1) tells us that ϱ_i is the probability that, in equilibrium, neuron i is excited, that is,

$$\varrho_i = \lim_{t\to\infty} \Pr(q_i(t) > 0).$$

Observe that the non-linear system composed of equations (3.2), (3.3) and (3.4) has $3N$ equations and $3N$ unknowns (the ϱ_is, the T_i^+s and the T_i^-s). Relations (3.3) and (3.4) tell us that T_i^+ is the mean throughput of positive signals arriving to neuron i and that T_i^- is the corresponding mean throughput of negative signals (always in equilibrium). Finally, Gelenbe proved, first, that this non-linear system has a unique solution, and, second, that the stability condition of the network is equivalent to the fact that, for all node i, $\varrho_i < 1$.

Let us describe now the use of this model in statistical learning. Following previous applications of RNN, we fix the λ_i^-s to 0 (so, there is no negative signal arriving from outside). As a learning tool, the RNN will be seen as a black-box having N inputs and N outputs. The inputs are the rates of the incoming flows of positive signals arriving from outside, i.e. the λ_i^+s. The output values are the ϱ_is. In fact, in applications, most of the time some neurons do not receive signals from outside, which simply corresponds to fixing some λ_i^+s to 0; in the same way, users often use as output only a subset of ϱ_is.

At this point, let us assume that the number of neurons has been chosen, and that the topology of the network is selected; this means that we have selected the pairs of neurons that will exchange signals, without fixing the values of the rates r_i and the branching probabilities $p_{i,j}^+$ and $p_{i,j}^-$.

Our learning data is then composed of a set of K input-output pairs, which we will denote here by $\{(\vec{x}^{(k)}, \vec{y}^{\,(k)}),\ k = 1, \cdots, K\}$, where $\vec{x}^{(k)} = (x_1^{(k)}, \cdots, x_N^{(k)})$ and $\vec{y}^{\,(k)} = (y_1^{(k)}, \cdots, y_N^{(k)})$. The goal of the learning process is to obtain values for the remaining parameters of the RNN (the rates r_i and the branching probabilities $p_{i,j}^+$ and $p_{i,j}^-$) such that if, in the resulting RNN, we set $\lambda_i^+ = x_i^{(k)}$ for all i (and $\lambda_i^- = 0$), then, for all i, the steady-state occupation probability ϱ_i is close to $y_i^{(k)}$. This must hold for any value of $k \in \{1, \cdots, K\}$.

To obtain this result, first of all, instead of working with rates and branching probabilities, the following variables are used:

$$w_{i,j}^+ = r_i p_{i,j}^+ \quad \text{and} \quad w_{i,j}^- = r_i p_{i,j}^-.$$

This means that $w_{i,j}^+$ (resp. $w_{i,j}^-$) is the mean throughput in equilibrium of positive (resp. negative) signals going from neuron i to neuron j. They are called *weights* by analogy to standard ANN. The learning algorithm proceeds then formally as follows. The set of weights in the network's topology is initialized to some arbitrary positive value, and then K iterations are performed which modify them. Let us call $w_{i,j}^{+\,(0)}$ and $w_{i,j}^{-\,(0)}$ the initial weights for the connection between i and j. Then, for $k = 1, \cdots, K$, the set of weights at step k is computed from the set of weights at step $k-1$ using a *learning scheme*, as usual with neural networks. More specifically, denote by $\mathcal{R}^{(k-1)}$ the network obtained after step $k-1$, defined by weights $w_{i,j}^{+\,(k-1)}$ and $w_{i,j}^{-\,(k-1)}$. When we set the input rates (external positive signals) in $\mathcal{R}^{(k-1)}$ to the $x_i^{(k)}$s values, we obtain the steady-state occupations $\varrho_i^{(k)}$s (assuming stability). The weights at step k are then defined by

$$w_{i,j}^{+\,(k)} = w_{i,j}^{+\,(k-1)} - \eta \sum_{l=1}^{N} c_l (\varrho_l^{(k)} - y_l^{(k)}) \frac{\partial \varrho_l}{\partial w_{i,j}^+}, \tag{3.5}$$

$$w_{i,j}^{-\,(k)} = w_{i,j}^{-\,(k-1)} - \eta \sum_{l=1}^{N} c_l (\varrho_l^{(k)} - y_l^{(k)}) \frac{\partial \varrho_l}{\partial w_{i,j}^-}, \tag{3.6}$$

where the partial derivatives $\partial\varrho_h/\partial w^*_{m,n}$ ("$*$" being '+' or '−') are evaluated at $\varrho_h = \varrho_h^{(k)}$ and $w^*_{m,n} = w^{*(k-1)}_{m,n}$. Factor c_l is a cost positive term allowing to give different importance to different ouput neurons. If some neuron l must not be considered in the output, we simply set $c_l = 0$. In our cases, we have only one output neuron, so, this factor is not relevant; we put it in (3.5) and (3.6) just to give the general form of the equation. This is a gradient descent algorithm, corresponding to the minimization of the cost function

$$\frac{1}{2}\sum_{l=1}^{N} c_l(\varrho_l^{(k)} - y_l^{(k)})^2.$$

Once again, the relations between the output and the input parameters in the product form result allows to explicitly derive a calculation scheme for the partial derivatives. Instead of solving a non-linear system as (3.2), (3.3) and (3.4), it is shown in [47] that here we just have a linear system to be solved. When relations (3.5) and (3.6) are applied, it may happen that some value $w^{+(k)}_{i,j}$ or $w^{-(k)}_{i,j}$ is negative. Since this is not allowed in the model, the weight is set to 0 and it is no more concerned by the updating process (another possibility is to modify the η coefficient and apply the relation again; previous studies have been done using the first discussed solution, which we also adopt).

Once the K learning values have been used, the whole process is repeated several times, until some convergence conditions are satisfied. Remember that, ideally, we want to obtain a network able to give output $\vec{y}^{\,(k)}$ when the input is $\vec{x}^{(k)}$, for $k = 1, \cdots, K$.

As in most applications of ANN for learning purposes, we use a 3-level network structure: the set of neurons $\{1, \cdots, N\}$ is partitioned into 3 subsets: the set of *input* nodes, the set of intermediate or *hidden* nodes and the set of *output* nodes. The input nodes receive (positive) signals from outside and don't send signals outside (that is, for each input node i, $\lambda_i^+ > 0$ and $d_i = 0$). For output nodes, the situation is the opposite: $\lambda_i^+ = 0$ and $d_i > 0$. The intermediate nodes are not directly connected to the environment; that is, for any hidden neuron i, we have $\lambda_i^+ = \lambda_i^- = d_i = 0$. Moreover, between the nodes inside each level there are no transitions. Last, input neurons are only connected to hidden ones, and hidden neurons are only connected to output ones.

This is a typical structure for neural networks used as a learning tool. Moreover, RNN mathematical analysis (that is, solving the non-linear and linear systems) is considerably simplified in this case. In particular, it can easily be shown that the network is always stable. See [47] or [46] and Chapter 10 for the details.

Chapter 4

Descriptions of Our New Method in General

4.1 Introduction

This Chapter proposes a new method to evaluate the quality of real-time multimedia streams, eventually transmitted over packet networks (IP or ATM networks). We present the methodology in general, which is applicable for whatever media type (speech, audio, video or multimedia). In addition, real-time application configuration such as session types (one way, interactive two ways, or multiparty conferences) can be taken into consideration. The validation of this method is left to the next subsequent Chapters.

It is known that subjective quality measures are the most reliable methods because, by definition, they are carried out by human subjects; that is, their results correspond to human (end-users) perception. However, they are very difficult to carry out, expensive and time-consuming tasks. Furthermore, they are not suitable to nowadays real-time applications running over packet networks (e.g. VoIP, videoconferencing, etc.). On the other hand, objective quality methods can be, in certain cases, used to evaluate the quality in real-time, as they are computer programs. But their major disadvantage is that their results do not always correlate well with human perception (subjective quality measures).

It is known that media-type quality is affected by many parameters (we name them as the quality-affecting parameters), such as the network loss rate, the delay variation, the encoding type, the bit rate, etc. In addition, the quality is not linearly proportional to the variation of any of these parameters. The determination of the quality is, therefore, a complex problem, and it has not been possible to solve by developing mathematical models that include the effects of all these parameters simultaneously.

Our problem has two aspects: first, a classification one, the mapping between the parameters' values and the quality; second, a prediction one, the evaluation of the quality as a function of the quality-affecting parameters in an operational environment. We believe that neural networks (NN) are an appropriate tool to solve this two-fold problem [117, 19]. We illustrate our approach by building a system that takes advantage of the benefits offered by NN to capture the nonlinear mapping between several non-subjective measures (i.e. the quality-affecting parameters) of media-type sequences transmitted over packet switched networks and the quality scale carried out by a group of humans subjects. Hence, a suitable NN is used to learn the mapping between the quality measures and the parameters' values. The NN then can be used to automatically predict the quality for any given combination of the quality-affecting parameters.

The rest of this Chapter is organized as follows. In Section 4.2, we provide a general description of our method. After that, we discuss in Section 4.3 the subjective quality tests to be used with our method for each type of media. We focus in Section 4.4 on the mean opinion score calculation. As our new method is based mainly on neural networks (NN), we present in Section 4.5 their use in our

method, as well as a comparison between the two models of NN used (ANN and RNN). The goal of Section 4.6 is to overview the different quality-affecting parameters for multimedia types that can be used with our method. Some possible uses and application of our method are outlined in Section 4.7. Section 4.8 is on the run-time mode of our method when integrated to measure in real time multimedia quality. Finally we provide the conclusions of this Chapter in Section 4.9

4.2 Overview of the Method

In this Section, we describe the overall steps to be followed in order to build a tool to automatically assess in real time the quality of real-time media transmitted over packet networks. We henceforth mean by the term "media" any speech, audio, video, or multimedia streams. Here our goal is to describe our method regardless of the media type in hand. See Figure 4.1 and Figure 4.2.

First, we define a set of static information that will affect the general quality perception. We must choose the most effective quality-affecting parameters corresponding to the media-type applications and to the network that will support the transmission. Then, for each parameter we must select several values covering all the possible range for that parameter. More values should be given in the range of the most frequent occurrences. For example, if the percentage loss rate is expected to vary from 0 to 10 %, then we may use 0, 1, 2, 5, and 10 % as typical values for this parameter. This is provided that the loss rate is generally between 0 and 5%, and the highest allowed value is 10% in the range. (In this case, it is supposed that the quality is the worst for 10% loss and for normal values of the other parameters.) Note that, not all media types tolerate the parameters' values in the same way. For example, speech can tolerate up to 20% loss rate, while video may tolerate only up to 5 % of loss rate without error resiliency in the encoder side. More values should be used in the range where it is expected to be more frequent in reality. If we call *configuration* of the set of quality-affecting parameters, a set of values for each one, the total number of possible configurations is usually large. We must then select a part of this large cardinality set, which will be used as (part of) the input data of the NN in the learning phase.

To generate a media database composed of samples corresponding to different configurations of the selected parameters (called "Distorted Database"), a simulation environment or a testbed must be implemented. This is used to send media sequences 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. For example, working with IP networks and video streams, the source controls the bit rate, the frame rate and the encoding algorithm, and it sends RTP video packets; the routers' behavior contribute to the loss rate or the loss distribution, together with the traffic conditions in the network. The destination stores the transmitted video sequence and collects the corresponding values of the parameters. Then, by running the testbed or by using simulations, we produce and store a set of distorted sequences along with the corresponding values of the parameters.

After completing the "Distorted Database", a subjective quality test must be carried out. There are several subjective quality methods in the recommendations of the ITU-R or ITU-T depending on the type of media in hands. Details on this step come in Section 4.3. In general, a group of human subjects is invited to evaluate the quality of the sequences (i.e. every subject gives each sequence a score from a predefined quality scale). The subjects should not establish any relation between the sequences and the corresponding parameters' values.

The next step is to calculate the MOS values for all the sequences. Based on the scores given by the human subjects, screening and statistical analysis may be carried out to remove the grading of the individuals suspected to give unreliable results [73]. See Section 4.4 for more details about this step. After that, we store the MOS values and the corresponding parameters' values in a second database (which we call the "Quality Database").

In the third step, a suitable NN architecture and a training algorithm should be selected. The

Quality Database is divided into two parts: one to train the NN and the other one to test its accuracy. The trained NN will then be able to evaluate the quality measure for any given values of the parameters. More details about this part are given in Section 4.5.

To put this more formally, we build a set $\mathcal{S} = \{\sigma_1, \sigma_2, \cdots, \sigma_S\}$ of media sequences that have encountered varied conditions when transmitted and that constitute the "training part" of the Quality Database. We also define a set $\mathcal{P} = \{\pi_1, \pi_2, \cdots, \pi_P\}$ of parameters such as the bit rate of the source, the packet loss rate in the network, etc. Then, we denote by v_{ps} the value of parameter π_p in sequence σ_s, and by V the matrix $V = (v_{ps})$. For $s = 1, 2, \cdots, S$, sequence σ_s receives the MOS evaluation $\mu_s \in [N, M]$ from the subjective test phase. The goal of the NN is to find a real function f having P real variables and with values in $[N, M]$, such that

(i) for any sequence $\sigma_s \in \mathcal{S}$, $f(v_{1s}, \cdots, v_{P,s}) \approx \mu_s$,

(ii) and such that for *any other* vector of parameter values $(v_1, \cdots, v_P)$, $f(v_1, \cdots, v_P)$ is close to the MOS that would receive any media sequence for which the selected parameters would have those specific values $v_1, \cdots, v_P$.

Once all the above steps 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 NN that will take the given values of the quality-affecting parameters and correspondingly computes instantaneously the MOS quality score.

All the above steps are summarized into the four parts shown in Figure 4.1. In the first part, we have to identify the quality-affecting parameters, their ranges and values, to choose the original sequences, and to produce the distorted database. In the second part, the subjective quality test is carried out for the distorted database and the MOS scores (together with their statistical analysis) are calculated in order to form the quality database. In the third part, we select the NN architecture and learning algorithm, and train and test it using the quality database. Finally, in the fourth step, we implement the final tool that consists of the two parts (parameters collection and MOS evaluation), in order to obtain a real-time quality evaluator. These steps are also shown in a block diagram in Figure 4.2.

4.2.1 More than one Media Type

Note that, in the cases where there are more than more media type (speech + video for example), we recommend that a separate NN module could be used to quantify the quality of each media. In addition, an extra NN module should be used to map all the scores measured for each media type into an overall score if needed. Note that there is not much work to do for these cases, as the subjects are asked to rate each media type as well as the overall quality of the given sequence once during the subjective quality tests. This is because, as previously mentioned, not all media types tolerate network impairments in the same way, as well as the diversity of the characteristics of both audio and video encoding algorithms.

4.2.2 Selecting the Parameters' Values

Suppose that we have P parameters to be taken into account, namely ($\mathcal{P} = \{\pi_1, \pi_2, \cdots, \pi_P\}$). Denote by n_p the number of different selected values of the parameter π_p. If we choose the whole set of configurations of the P parameters, the total number of samples may be very large ($n_1 \times n_2 \times \cdots \times n_P$). Carrying out subjective quality tests for such a number of samples may be impractical. Fortunately, the NN does not need all these samples to capture the relationship between the quality and the parameters' values, it is its goal to predict the quality in the missed configurations [1]. Therefore, we

[1] The trained NN can predict also the quality of any value within the allowed ranges of all the parameters.

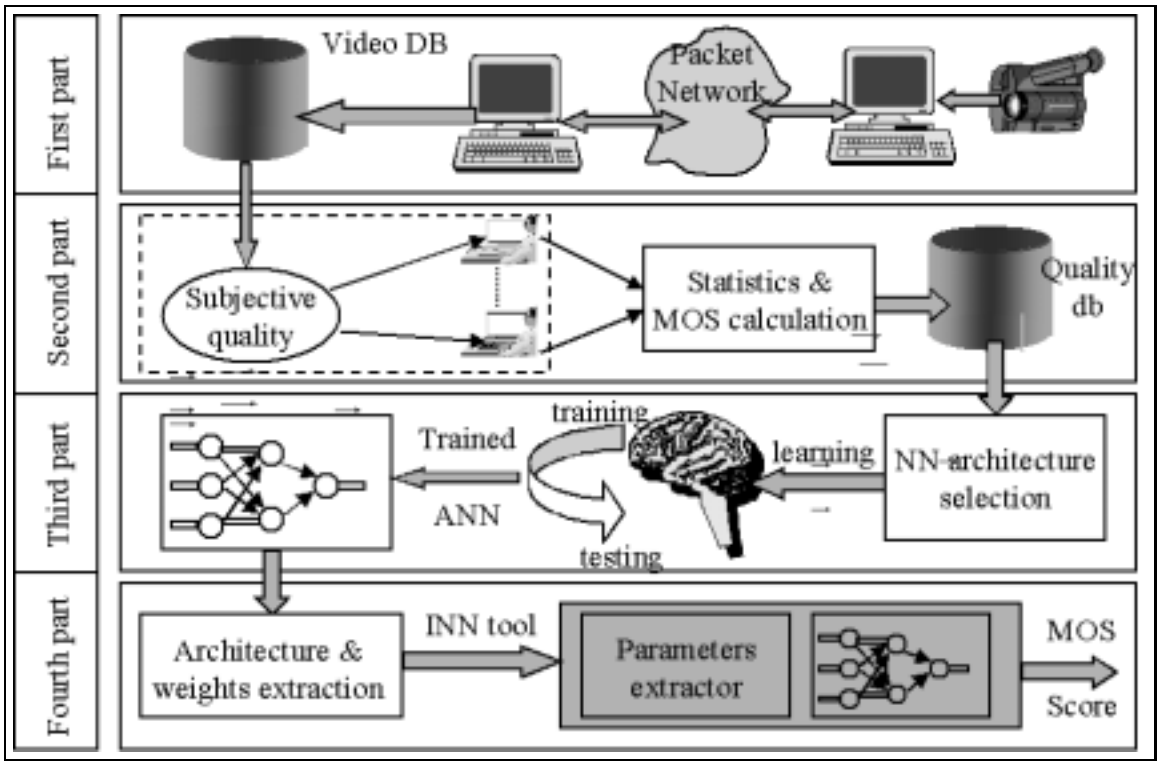


Figure 4.1: The overall architecture of the new method to evaluate real-time speech, audio and/or video quality in real time.

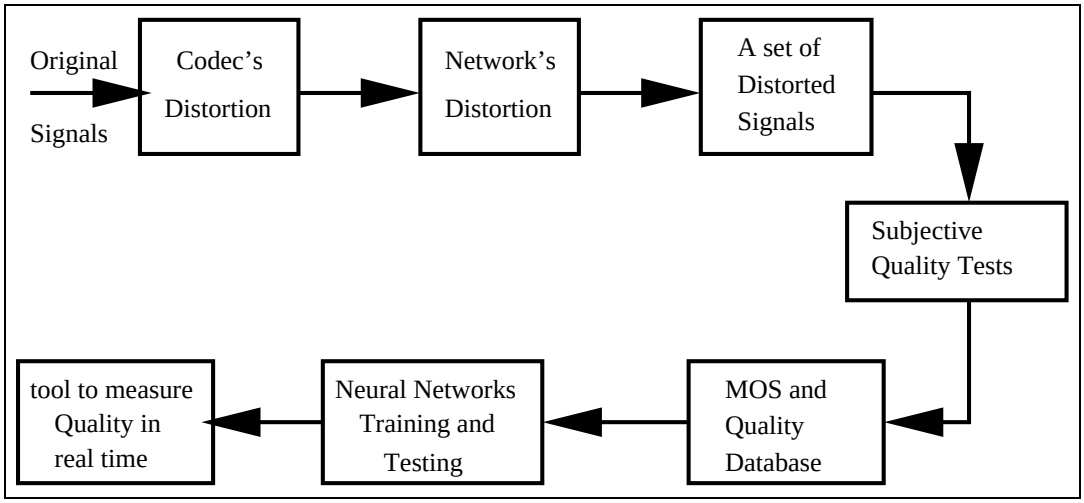


Figure 4.2: A schematic diagram of our method showing the steps in the design phase.

propose the following method to generate the minimum number of samples needed to train and test the neural networks. We should define default values for all the parameters, $\mathcal{P}_\circ = \{\pi_{01}, \pi_{02}, \cdots, \pi_{0P}\}$). These values could be, for instance, the most frequently observed values. We change the values of two parameters at a time and give the default values to the others. We repeat this step for all the parameters. In this way, a list of configurations is defined, containing many duplications. Once these duplicates are removed, the remaining ones constitute the minimum samples required to train and test the NN.

4.3 Subjective Quality Tests

To evaluate the quality of speech, audio or video systems (codec, telecommunication, television pictures, IP telephony, etc.), a subjective quality test is used. In this test, a group of human subjects is invited to judge the quality of the sequence under a predefined system conditions (distortions). There are several recommendations that specify strict conditions to be followed in order to carry out subjective tests. These recommendations are specific to the media type (speech, audio, video, or multimedia in general) in study.

In this Section, we focus on how to use subjective quality tests in our method. In addition, we summarize the recommendations of ITU regarding this point. Depending on the media type, there exist some ITU recommendations that define how to carry out subjective quality tests.

ITU-T Rec. P.800 [70] presents some methods for subjective determination of transmission speech quality. For audio in general, ITU-R Rec. BS.1116 [63] recommends methods for the subjective assessment of audio systems. ITU-R Rec. BS.1284 [65] presents the general requirements for subjective assessment of sound quality. How to assess subjective audio quality mixed with pictures is the subject of ITU-R Rec. BS.1286 [66]. A guide to existing recommendations that deal with audio quality assessment is provided in ITU-R Rec. BS.1283 [64]. For video quality assessment, there exist some recommendations for assessing television picture quality. These recommendations are also valid for video in general. We can cite for example, ITU-T P.910 [72]. For multimedia quality assessment, the most used recommendation is ITU-T P.911 [73]. These recommendations provide some methods for non-interactive subjective assessment methods for evaluating the one-way overall audiovisual quality of multimedia applications. For interactive applications, there exists ITU-T P.920 [74], which provides the methods recommended for subjective interactive quality tests.

4.3.1 Source Signal for Audiovisual Tests

ITU recommendations suggest that the duration of the source sequences should be about 10 seconds for audiovisual, and the length of 5 short different sentences for speech. The termination of the sequences should not cause an incomplete sentence or musical phrase. An initial and a final silent period or gray scene, not longer than 500 ms, can be used to make the sequence be more natural.

For the case of pair presentations, the references should have the best possible quality without any impairments. For audiovisual applications, speech and video should be perfectly synchronized.

4.3.2 Number of Subjects and their Selection

As stated in ITU recommendations, the number of subjects required to carry out the subjective quality test can vary from 4 to 40. Four is the absolute minimum for statistical reasons. The number of assessors needed depends upon the sensitivity and reliability of the test procedure adopted. The average number of subjects is about 15.

They should not be directly involved either in picture or audio quality evaluation as part of their work and should not be experienced assessors. Prior to a session, subjects should usually be screened for normal visual acuity or corrected-to-normal acuity and for normal color vision (in the case of video quality evaluation).

4.3.3 Instructions for Assessors (Subjects)

Before carrying out the experiments, some instructions should be given to the assessors. These instructions include the method of assessment, the types of impairment or quality factors likely to occur, the grading scale, the sequence and timing. This information should be explained and given to the subjects in a written form. The range and type of impairments should be presented in preliminary trials. Training trials may be given to subjects to familiarize them with the task they will perform.

4.3.4 The Test Sessions

Following the ITU recommendations, overall subjective tests should be divided into multiple sessions and each session should not last more than 30 minutes. For each session, we should add several dummy sequences (about four or five) at the beginning. These sequences should not be taken into account in the calculation. Their aim is to be used as training samples for the subjects to learn how to give meaningful rates. Furthermore, the reliability of subjects can be qualitatively evaluated by checking their behavior when reference/reference pairs are given. In these cases, reliable subjects are expected to give evaluations very close to the maximum point in the quality scale. Hence, in these dummy samples, at least one of samples should be the reference. More advanced technique to check for the reliability of subjects is given in the next Section.

4.3.5 Subjective Quality Test Methods

In the ITU recommendations, there are many subjective quality test methods. Here we detail the methods that can be used with our approach. Namely, they are the Absolute Category Rating (ACR), the Degradation Category Rating (DCR) and the Double-Stimulus Continuous Quality-Scale (DSCQS).

4.3.5.1 Absolute Category Rating (ACR)

ACR is a category judgment method where the test sequences are presented one at a time and are rated independently on a category scale. (This method is also called Single Stimulus [2] Method.) Subjects are asked to rate the quality of the presentation based on the level of the quality they have in their opinion for it after viewing or listening it. This phase is named the voting time. The time pattern for the stimulus presentation can be illustrated by Figure 4.5. The voting time should be less than or equal to 10 seconds. The five-level scale for rating overall quality is the most used scale, see Table 4.1. If higher discriminative power is required, a nine-level scale may be used as shown in Table 4.2. There is another variant of this scale which is the 11-point scale, depicted in Figure 4.3. Finally, there is a general scale, which is the continuous quality scale that is depicted in Figure 4.7.

Table 4.1: ITU 5-point quality scale

Grading value	**Estimated Quality**
5	Excellent
4	Good
3	Fair
2	Poor
1	Bad

Table 4.2: ITU 9-point quality scale

Grading value	**Estimated Quality**
9	Excellent
8	
7	Good
6	
5	Fair
4	
3	Poor
2	
1	Bad

4.3.5.2 Degradation Category Rating (DCR)

The second suitable method for our tool is DCR, where test sequences are presented in pairs. The first stimulus presented in each pair is always the source reference without any impairments. The second

[2] The **stimulus** is the sequence presented to the subjects in a test.

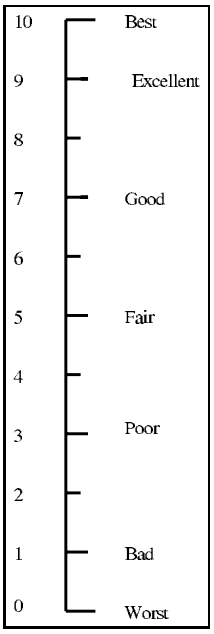


Figure 4.3: Eleven-point quality scale. 10 score for the sequence(s) that is identical to the reference one. Similarly, 0 score is for the sequence that has no similarity with the reference.

Figure 4.4: ITU 5-point impairment scale

Grading value	**Estimated impairment level**
5	Imperceptible
4	Perceptible, but not annoying
3	Slightly annoying
2	Annoying
1	Very annoying

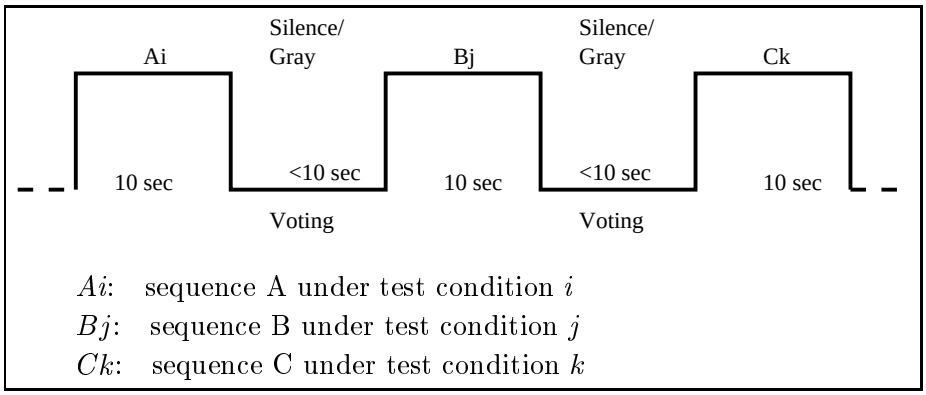


Figure 4.5: Stimulus presentation timing in ACR method.

one is the same source but impaired by the test conditions. This method is also called the Double Stimulus Impairment Scale (DSIS) method.

The time pattern for the stimulus presentation is depicted in Figure 4.6. The voting time should be less than or equal to 10 sec. In this case the subjects are asked to rate the impairment of the second stimulus in relation to the reference. The five-level scale for rating the impairment given in Table 4.4 is the most widely used one. However, all the quality scales used of ACR method can be used for DCR method but by replacing the quality adjectives by the corresponding impairment adjectives.

4.3.5.3 The Double-Stimulus Continuous Quality-Scale method (DSCQS)

DSCQS is useful when it is not possible to provide stimulus test conditions that exhibit the full range of the quality. This method may be of special interest for our method to be used in some kind of

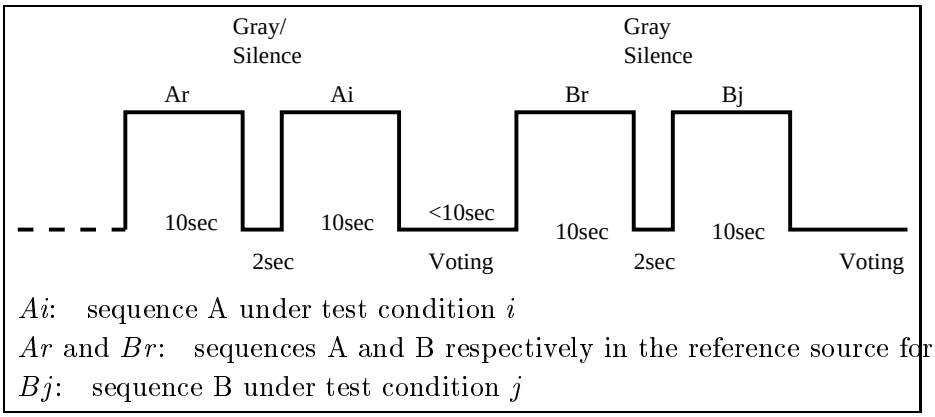


Figure 4.6: Stimulus presentation timing in DCR method.

networks guaranteeing certain level of QoS, for example, when MOS is greater than 3 on the 5-point quality scale.

As in DCR method, the sequences are presented in pairs: the reference one and the impaired one. Subjects are asked to assess the quality of both (and not the impaired one with respect to the reference as in DCR). The unimpaired one is included to serve as a reference, but the observers are not told which is the reference sequence. In the series of tests, the position of the reference is changed randomly.

The subjects are asked to assess the overall sequence quality of each presentation by inserting a mark on a vertical scale (see Figure 4.8). The vertical scales are printed in pairs to accommodate the double presentation of each test sequence. The scales are continuous to avoid quantizing errors, but they are divided into five equal lengths, which correspond to the normal ITU five-point quality scale. The associated terms categorizing the different levels are the same as those normally used. Here they are included for general guidance. Figure 4.8 shows a section of a typical score sheet.

4.3.6 Comparison of the Methods

An important issue in choosing a test method is the fundamental difference between methods that use explicit references (e.g. DCR or DSCQS) and methods that do not use any explicit reference (e.g. ACR). The latter does not test fidelity. The former, on the other hand, should be used when testing the fidelity of transmission with respect to the source signal.

Thus, when it is important to check the fidelity with respect to the source signal, the DCR method should be used. Discrimination of imperceptible/perceptible impairment in the DCR scale supports this, as well as comparison with the reference quality. DSCQS, in addition, is used in the cases when the quality range is not completely covered.

On the other hand, ACR is easy and fast to implement and the presentation of the stimuli is similar to that of the common use of the systems. Thus, ACR is well suited for qualification tests.

4.4 MOS Calculation and Statistical Analysis

Let us denote by N the number of subjects in the chosen subjective method and by u_{is} the evaluation of sequence σ_s made by user i. The set of values $(u_{is})_{i=1,\cdots,N}$ will probably present variations due to the differences in judgment between subjects. Moreover, it is possible that some subjects do not pay enough attention during the experiment, or behave in some unusual way face to the sequences; this can lead to inconsistent data for the training phase. Some statistical filtering is thus necessary on the set of raw data. The most widely used reference to deal with this topic is the ITU-R BT.500-10

recommendation, [67]. The described procedure allows to remove the ratings of those subjects who could not conduct consistent scores.

First, denote by $\bar{u}_s$ the mean of the evaluations of sequence σ_s on the set of subjects, that is,

$$\bar{u}_s = \frac{1}{N}\sum_{i=1}^{N} u_{is}. \tag{4.1}$$

Denote by $[\bar{u}_s - \Delta_s, \bar{u}_s + \Delta_s]$ the 95%-confidence interval obtained from the (u_{is}), that is, $\Delta_s = 1.96\delta_s/\sqrt{N}$, where

$$\delta_s = \sqrt{\sum_{i=1}^{N} \frac{(u_{is} - \bar{u}_s)^2}{N-1}}.$$

As stated in [67], it must be ascertained whether this distribution of scores is normal or not using the β_2 test (by calculating the "kurtosis" coefficient of the function, i.e. the ratio of the fourth order moment to the square of the second order moment). If β_2 is between 2 and 4, the distribution may be taken to be normal. In symbols, denoting $\beta_{2s} = m_{4s}/m_{2s}^2$ where

$$m_{xs} = \frac{1}{N}\sum_{i=1}^{N}(u_{is} - \bar{u}_s)^x,$$

if $2 \leq \beta_{2s} \leq 4$ then the distribution $(u_{is})_{i=1,\cdots,N}$ can be assumed to be normal. For each subject i, we must compute two integer values L_i and R_i, following the following procedure:

set $L_i = 0$ and $R_i = 0$
for each sequence $\sigma_s \in \mathcal{S} = \{\sigma_1, \cdots, \sigma_S\}$
    if $2 \leq \beta_{2s} \leq 4$, then
        if $u_{is} \geq \bar{u}_s + 2\,\delta_s$ then $R_i = R_i + 1$
        if $u_{is} \leq \bar{u}_s - 2\,\delta_s$ then $L_i = L_i + 1$
    else
        if $u_{is} \geq \bar{u}_s + \sqrt{20}\,\delta_s$ then $R_i = R_i + 1$
        if $u_{is} \leq \bar{u}_s - \sqrt{20}\,\delta_s$ then $L_i = L_i + 1$

Finally, if $(L_i + R_i)/S > 0.05$ and $|(L_i - R_i)/(L_i + R_i)| < 0.3$ then the scores of subject i must be deleted. For more details about this topic and the other methods of subjective tests see [67].

After eliminating the scores of those subjects who could not conduct coherent ratings using the above technique, the mean score should be recomputed using Eq. 4.1. This will constitute the MOS database that we will use to train and test the NN.

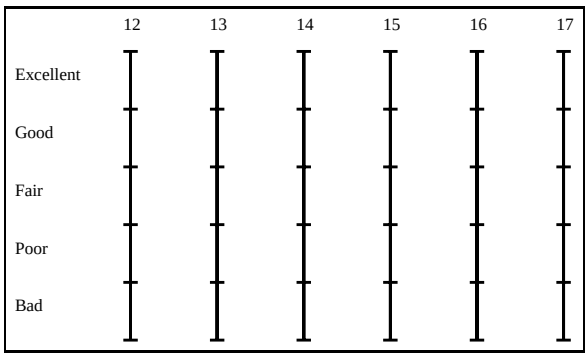


Figure 4.7: A portion of quality rating form using continuous scales.

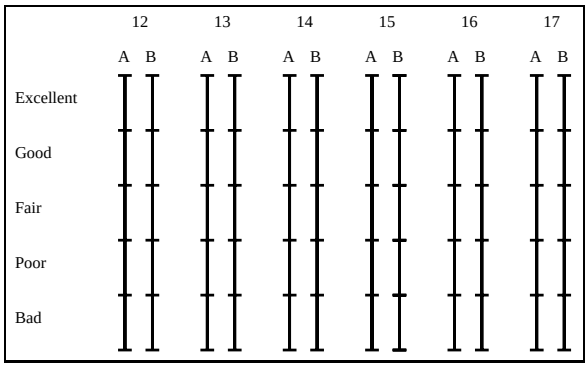


Figure 4.8: A portion of quality rating form using continuous scales for DSCQS method.

4.5 On the Use of the Neural Networks

It is important to mention that since the 60's, NNs have been used to solve problems in communication networks, mostly those involving ambiguity, and an environment that varies over time. There have been very successful applications, always addressing problems difficult to tackle with traditional methods [149], which go from using the NNs as adaptive echo cancellers and adaptive line equalizers on telephone channels, to the development of complex control schemes for traffic control in ATM networks (e.g., call admission control, traffic parameters prediction, traffic flow control, quality of service parameters estimation, traffic prediction, and dynamic allocation of network resources).

Regarding RNN, they are used in video compression with compression ratio that goes from 500:1 to 1000:1 for moving gray-scale images and full-color video sequences respectively [31]. They are also used as decoders for error correcting codes in noisy communication channels, which reduce the error probability to zero [2]. Furthermore, they are used in a variety of image processing techniques, which go from image enlargement and fusion to image segmentation [50]. A survey of RNN applications is given in [12].

After performing the suitable subjective quality test and calculating the MOS scores (see Sections 4.3, 4.4), these scores should be normalized. The goal of the normalization is to achieve better performance from the NN. The normalization formulae is as follows:

$$F_n = \frac{F - F_{min}}{F_{max} - F_{min}}, \tag{4.2}$$

where F_n and F are the normalized and original values of the MOS scores respectively. F_{max} is the maximum score value available on the used quality scale for the best quality. F_{min} are the minimum score value available of the used quality scale for the worst quality.

In addition, all the quality-affecting parameters' values should be normalized between 0 and 1. The quality database, the quality-affecting parameters and the corresponding quality scores (which represent the learning patterns or examples), will be used to train and test the NN. This database should be divided into two parts. The first part is used to train the NN, while the second one is used to test its performance.

For the NN, there are many architectures (feedforward, recurrent, etc.) and many training algorithms (backpropagation, Levenberg-Marquardt, etc.). The two types of NN considered here are the Artificial Neural Networks (ANN) and the Random Neural Networks (RNN). We recommend the use of RNN as it offers some advantages over ANN, at least in our context. (See Section 4.5.1 for a comparison between ANN and RNN and Chapter 10 for new training algorithms for RNN.) For the NN architecture, the number of input neurons is equal to the number of parameters, while there is only one output neuron, which represents the corresponding, estimated quality score (all normalized).

There are some known problems for using NN. One of these problems is overtraining. The network is said to be overtrained, when the performance of network is very good for the training database, while it gives very poor performance for the testing database (the inputs-output examples that have not been seen during the training process).

Another problem is the choice of the number of neurons in the hidden layer. With too few hidden neurons, the NN will not learn correctly the problem. On the other hand, with too many hidden neurons, there is a risk of overtraining, and the system will be more complex. It is stated in [118] that whenever several networks can solve the problem, it is preferable to select the most simple one to make the network generalize well. Indeed, the most difficult problem (yet not efficiently solved) is how to find the optimal number of hidden neurons. There exist some heuristic methods aimed to find a rough approximation, however they work for some problems and give bad results for others. This is because these methods cannot take into consideration the complexity of the given problem. Instead, they try to map the problem complexity as a function of the number of inputs and outputs as well as the number of the training examples.

We recommend to use the following procedure to identify the best NN architecture, provided that a three-layer feedforward architecture is used.

1. Start by one hidden neuron.
2. Train the NN by the training database until a minimum error goal is achieved, and measure its performance to predict the outputs.
3. Test it by the testing database, and measure its performance.
4. Increment the hidden neurons by one and if the number of hidden neurons is less than a maximum value, go to step 2.
5. Select the architecture that gives the best performance for both the training and testing databases.

The weights of the NN are initialized randomly, hence care should be taken to initialize to the same value to correctly identify the optimal number of hidden neurons by the above procedure. The minimum error goal to stop the training may be made variable to precisely identify the hidden neurons and hence get the best performance of the neural networks.

4.5.1 Comparison between ANN and RNN

In this subsection, we compare the two considered types of neural networks: Artificial Neural Networks (ANN) and Random Neural Networks (RNN), in the context of our specific problem. This is based on our experience with both models.

We used the Neural Networks MATLAB Toolbox when working with ANN, and a MATLAB package [1] for RNN. We observed that ANN training process was relatively faster than that of RNN (to overcome this problem, we proposed several training algorithms for RNN with the aim of accelerating the training process, see Chapter 10). However, during run-time phase, RNN outperformed ANN in the total calculation time. This is because in the RNN's three-level architecture, the computation of the output ϱ_o for the unique output neuron o is done extremely fast: the non-linear system of equations (3.2), (3.3) and (3.4) allows, in this topology, to compute the ϱ_is of the input layer directly, and of hidden layer neurons from the values for input layer ones. To be more specific, for each input neuron i we have

$$\varrho_i = \frac{\lambda_i^+}{r_i + \lambda_i^-},$$

(where, actually, we choose to set $\lambda_i^- = 0$), and for each hidden layer neuron h,

$$\varrho_h = \frac{\sum_{\text{input neuron } i} \varrho_i w_{i,h}^+}{r_h + \sum_{\text{input neuron } i} \varrho_i w_{i,h}^-}.$$

The output of the black box is then ϱ_o, given by

$$\varrho_o = \frac{\sum_{\text{hidden neuron } h} \varrho_h w_{h,o}^+}{r_o + \sum_{\text{hidden neuron } h} \varrho_h w_{h,o}^-}.$$

(The cost of computing the output ϱ_o is exactly $2IH + 3H + I + 1$ products (or divisions) and the same number of sums, where I is the number of input neurons and H is the number of hidden ones.)

ANN's computations are slower because they involve nonlinear function calculations for each neuron in the architecture (sigmoid, tanh, etc.). This makes RNN particularly attractive for using them in contexts with real-time constraints, or for lightweight applications. This can be important in some kind of network applications, for example, in [83], an ANN packet-loss predictor is proposed for real-time multimedia streams. The prediction precision is good, but the calculation time is much more than the next packet arrival time which makes the system useless except for very powerful computers.

The most important feature of RNN we found for our problem is that they capture very well the mapping from parameters' values to the quality evaluation. This concerns also their ability to extrapolate in a coherent way for parameters' values out of the ranges used during the training phase. For instance, this led in [2] to build a zero-error channel decoder.

As previously mentioned, the most common problems of ANN's learning are the over-training and the sensitivity to the number of hidden neurons. The over-training problem makes the NN memorize the training patterns, but gives poor generalizations for new inputs. Moreover, if we cannot identify some near-optimal number of hidden neurons, the performance may be bad for both the training set and the new inputs. Figure 4.9(b) shows an example of an over-trained ANN network, where we can see irregularities, bad generalizations and bad capturing of the function mapping (see Subsection 7.4.3 for comparison with RNN).

We trained different architectures (varying the number of hidden neurons) for both ANN and RNN, with the same data (described in Section 6.3) and the same *mean square error*. Let us look, for instance, at the behavior of the quality as a function of the normalized bit rate BR (the 4 remaining variables were set to their most frequent observed values). In the database, BR varies between 0.15 and 0.7. In Figure 4.10(a) and Figure 4.10(b), we depict the ability of both networks to interpolate and extrapolate the results when BR varies from zero to its maximum value 1. These Figures show that RNN captures the mapping between the input and output variables, and that RNN is not very sensitive to the number of hidden neurons. While ANN gives quite different approximations for small changes in the size of the hidden layer.

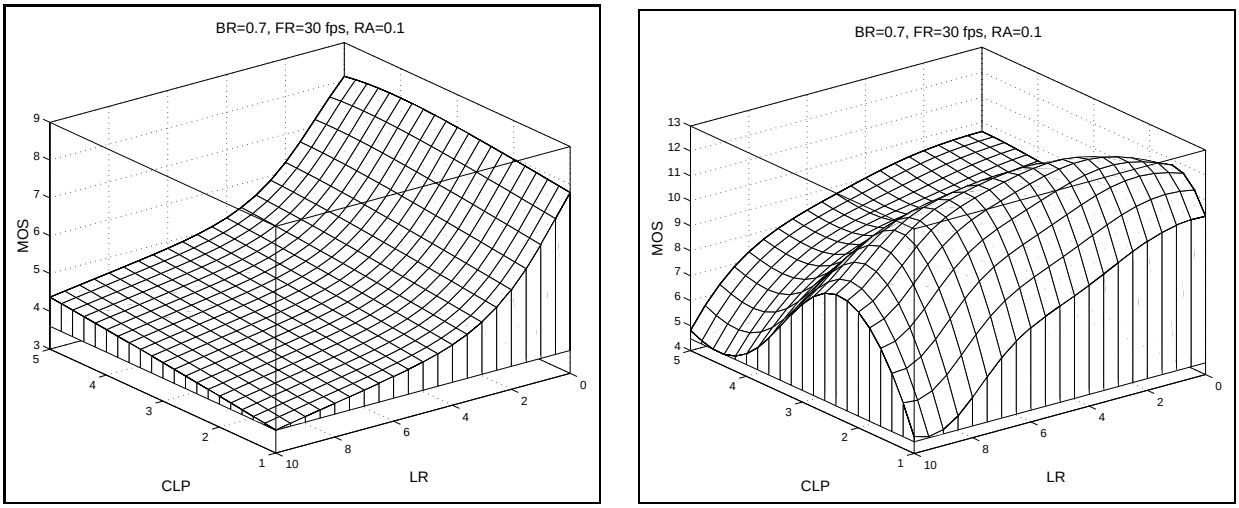
Moreover, if the optimal ANN architecture could not be identified, its accuracy could be bad. Let us look now at the extreme values. If BR=0.0, the output should be around one, while for BR=1.0, the output should be between 8.5 and 9.0 on the y-axis. For the case of ANN, as shown in Figure 4.10(b), when the number of hidden neurons changes from four (optimal experimentally) to five, the generalization is bad, specially when BR goes to zero, where the output is 3.2 instead of 1. This gives RNN a better ability to generalize.

4.6 Parameter Selection

It is important to correctly identify and select the quality-affecting parameters. If an important parameter is neglected, the accuracy of the tool will be degraded and it will not behave as expected. However, the number of parameters to be taken into account should be minimized. This is because this number determine the minimum number of samples (see Sec. 4.3) in the quality database to train and test the NN, knowing that this database is built by subjective quality tests, which are very expensive and hard to carry out. Hence some mechanisms may be used to merge two or more parameters in one. For example, the case of delay variation (jitter) can be mapped to be included in the loss parameter by using dejittering buffer.

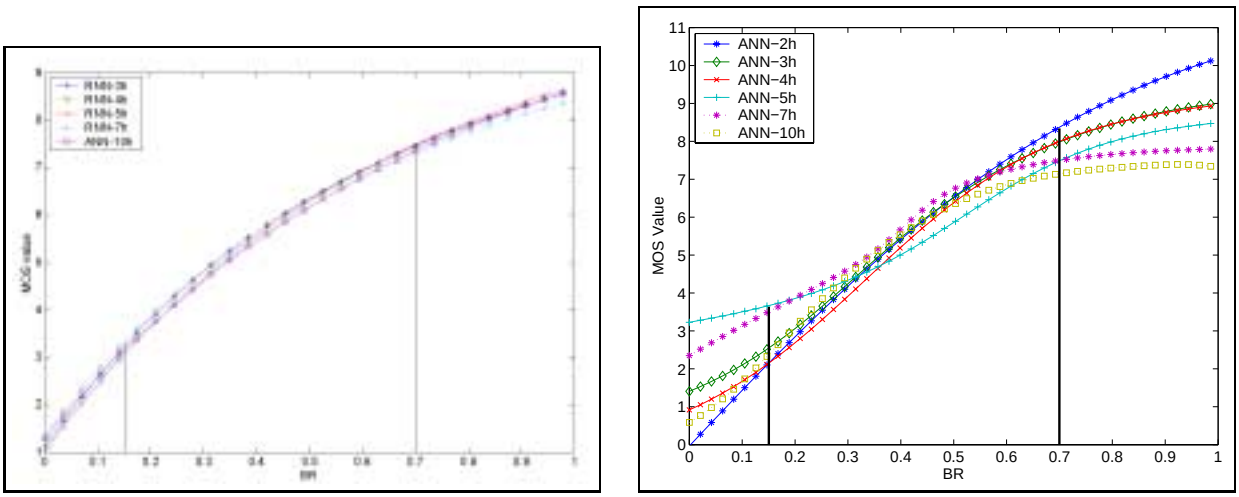
We can classify the quality-affecting parameters into three categories:

- **Network parameters:** Parameters concerning the packet network that carrying media-type signals. The most known parameters are packet loss rate, arrival jitter, packetization interval, loss distribution, and end-to-end delay. Furthermore, effects of error concealment techniques (FEC or ARQ) (e.g. silence, noise, repetition, and waveform substitution for lost audio packets) [38] can affect the quality.



(a) Correctly trainined (b) Example of an over-trained ANN

Figure 4.9: The problem of overtraining using ANN



(a) Using RNN (b) Using ANN

Figure 4.10: Performance of ANN and RNN to interpolate and extrapolate for different number of hidden neurons

- **Encoding parameters:** Parameters concerning the ability to encode or compress the original media-type signals to fit the output stream into certain bit rate. There are two types of compression (lossless and lossy), however, the only one that affects the quality is the lossy one. The majority of the existing codecs use both types. Hence some of the parameters due to the encoding impairments can be classified into the type of the codec used, the sampling rate, and the number of bits per sample, the number of layers in the case of layered codecs. For video, we have in addition the bit rate, the frame rate, the encoded frame types, etc.

- **Other parameters:** like the echo (which may occur due to the long end-to-end delay), crosstalk effect (when two or more people start talking at the same time), or the number of participating sources. These effects occur in bi-directional sessions or multi-party conferences. In addition,

there is the audio-video synchronization aspect for the case of multimedia applications [40].

There is also the aspect of the session types. Actually, session-types may evoke additional parameters to be taken into account. Two-way sessions present interactivity between the end-users. In the case of speech, echo, end-to-end delay, crosstalk effects may affect the quality. These factors do not affect the quality for one-way sessions. One-way sessions can be equivalent to bi-directional sessions under the following hypothesis: when there is a mechanism to control the echo or if we assume ideal echo suppression. The crosstalk effect can be solved for the case of half-duplex conversions. The delay jitter effect can be reduced by implementing a dejittering mechanism [37].

These parameters may not affect the quality for some other cases of the media-types. For example, for pure video transmission in real time, even for the two-way sessions as the videoconferencing applications, it is easy to map them into two one-way sessions. As there is no echo, nor crosstalk problems. However, the end-to-end delay should have to be parameterized (considered as a quality-affecting parameter).

In addition, if our method is used to assess the quality of multimedia (audio and video) applications, we recommend to use two separate modules to assess the quality for each one. An extra module should be used to assess the overall quality given the measures of the other two modules plus the value of the synchronization parameter. In this case, during the subjective quality tests, users are asked to assess the quality of each media type and the overall quality.

4.7 Expected Use of Our Method

Let us briefly discuss here on the benefits of having a tool to automatically measure (and, if useful, in real time) audio and video quality.

4.7.1 Possible Uses For Speech and Audio

- In IP-telephony applications, at both end-user sides, this tool can be used to monitor in real time the received audio quality. In this way, each user can know how the other user hears what he/she sends and receives.
- At the client side of the streaming audio applications (Internet Radios).
- Based on the quality measurement using this tool, the operators can use it as criteria for billing.
- As the technology of IP telephony improves in these days, constructors can implement hardware version of this tool and integrate it in the IP-telephone set. In fact, one of the advantages of using the NN is that once we get a working software model that can assess the audio quality in real time, the hardware version can be built easily.
- The applications that transmit the audio on packet network can use this tool to negotiate for the best configuration to give the best quality. For example, changing the bit rate, using another codec, changing the packetization interval, using some kind of FEC, changing the playback buffer size, etc. are possible decisions that can be taken to improve the quality or even to maintain a certain level of the quality.

4.7.2 Possible Uses for Video Applications

- In video conferencing and in the majority of video applications, at both end-user sides, such a tool can be used to monitor the received video quality in real time for control purposes. For example, changing the bit rate, using another codec, changing the frame rate, using some kind of FEC, changing the playback buffer size, etc. are possible decisions that can be taken to improve the quality or to maintain it at a certain level.

- Based on the ability to measure video quality in real time, operators could use quality as a criterion for billing.
- The applications that transmit video on packet networks can use this tool to negotiate the best configuration in order to achieve the best possible quality.
- It can also help in the encoding process, as quality in encoders is also a way to "fit" the stream into the available global channel bandwidth. In video codecs using temporal compression (ex: MPEG, H.261, H.263...), a quality factor parameter is usually used to reduce the output stream bandwidth and to improve, at the same time, the assessed quality (yet before any transmission). It would be even more interesting to have the history of all the important parameters (network and video) to compress the video signal rather than only the PSNR (Peak Signal to Noise Ratio) objective measure (which is poorer in quality and time consuming than our approach).

4.8 Operation Mode

Figure 4.11 depicts the schematic diagram showing the run-time mode of the implemented tool. Briefly, the original signal is impaired by the encoder and the network. The parameters' values collector gets the values that are fed to the trained NN. The NN evaluates the quality based on the parameters' values. As we see from this Figure, we do not need access to the original signal to evaluate the quality (which is the case for the traditional objective methods depicted by the block diagram in Figure 4.12). This point is very important, as the access to both original and distorted signals constitutes a major disadvantage of the existing objective quality measures which limits their use in real-time applications.

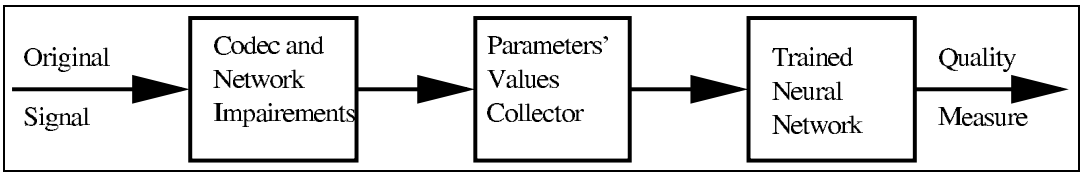


Figure 4.11: The run-time mode of our method.

When our method is intended to be used in user's applications to evaluate the quality of the transmitted multimedia in real time, the proposed architecture is as shown in Figure 4.13. At the sending side, the media source (video camera, microphone, or recorded material) generates a continuous media stream. That stream is encoded in real time by the appropriate encoder. In this stage the quality of the signal may be degraded due to the encoder's impairments. The output of the encoder is then packetized and sent to the packet network using a suitable transport protocol (RTP is the most common used one for real-time multimedia transportation in IP networks). Here also, the quality of the signal may be degraded due to network impairments.

At the receiving side, the sent stream (stream of packets) is reconstructed by RTP protocol. The encoded signal data is retrieved from the packets (depacketized). Then it is decoded by a suitable decoder. The decoded signal is sent to the appropriate output device (speakers or video card) to be played.

The interaction between our tool and the other elements is as follows. The parameters' collector part probes all the working parameters from the encoder, decoder, packet network and the transport protocol. Then the trained NN part evaluates the signal quality as a function of these parameters. In this way, the end-user at the receiver side can see the quality measure instantaneously. While at the sender side, if necessary, the quality can be sent by means of the transport protocol from time to time. (If RTCP protocol is used, it can be sent every 5 sec. However, if other dedicated protocol is used, the quality score can be fed to the sources each time we want.) This means that the frequency

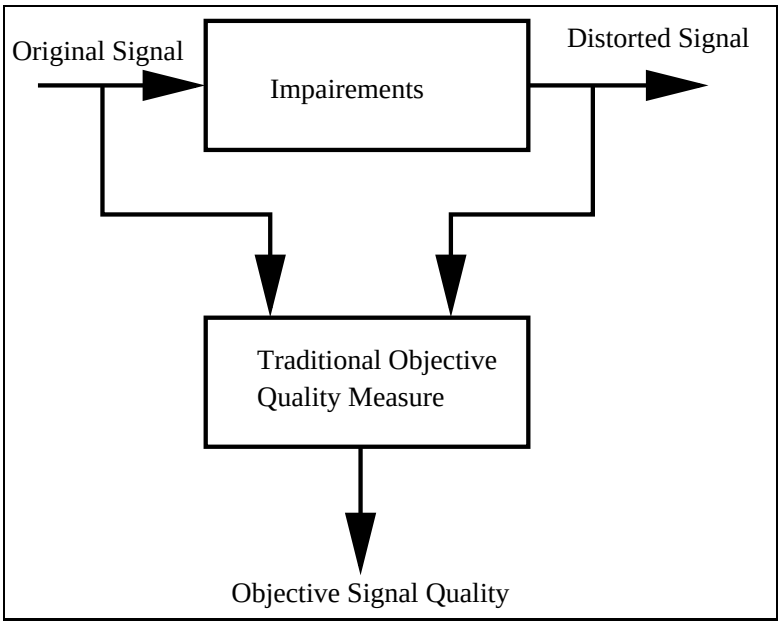


Figure 4.12: The current existing model of objective methods.

update of the parameters and hence the quality evaluation can be done at any time the user wants at the receiver side, while at the sender side the user can have feedback about the quality at least every 5 sec. (again, if RTCP is used).

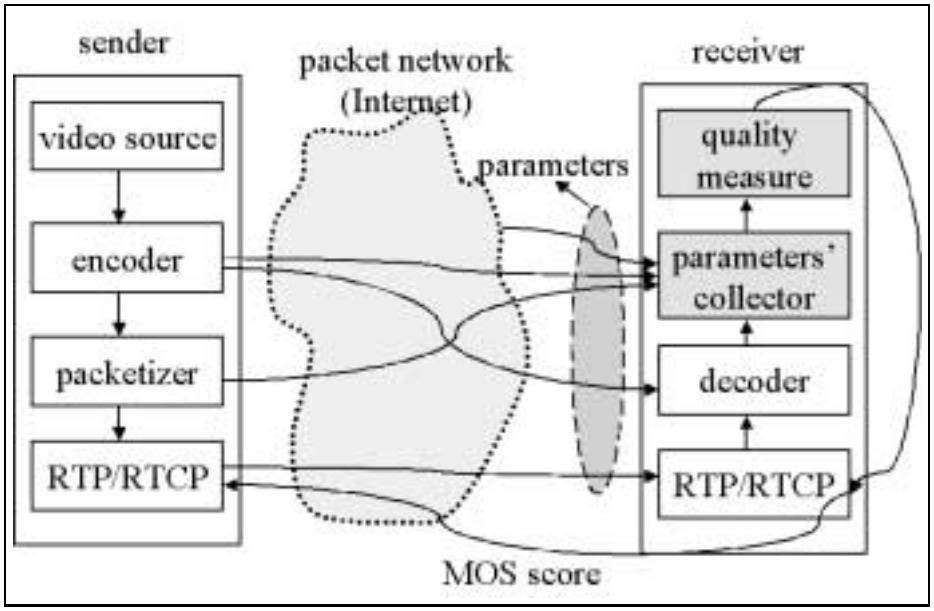


Figure 4.13: Operation mode for the tool in real-time video system.

4.9 Conclusions

This Chapter presented a detailed description of our new method which aims to assess real-time multimedia streams transmitted over packet networks in real time. We presented the methodology in

general regardless of the media type. Our method can be used for speech, audio, video or multimedia quality assessment. Session types also can be taken into account (one-way, interactive or multiparty conferences). The development of the method consists of the following steps. First, the quality-affecting parameters should be identified. Then a set of sequences distorted by wide range variation of these parameters should be constructed. After that, a suitable subjective quality test is performed on these sequences. The result is a database consisting of a set of parameters' values and the corresponding quality scores. A suitable neural network architecture should be identified, trained and tested by the resulting database. The result is a neural network tool that can be used to assess in real time the quality of the chosen media when transmitted in real time over packet networks.

One of the advantages of our method is that it takes into account the direct effect of network parameters (e.g. loss rate, jitter, etc.) and the codec parameters (e.g. bit rate, frame rate, etc.), as well as the other hybrid parameters (e.g. echo, crosstalk, etc.). In addition, it is a hybrid method between subjective quality measures (human perception of the quality) and objective measures (it can work in real time). This Chapter presented some of the advantages of using our method.

We also provided a guideline about how to carry out the suitable subjective quality tests, which constitutes the most difficult part in implementing that method. In addition, the interaction between our method and the existing real time network components in operation mode is provided.

Validation of the methodology is postponed to the next two Chapters in which, we verify the applicability of it to correctly evaluate in real time the quality of speech and video transmitted in real time over IP networks. Based on these results, we provide a study of the impact of the quality-affecting parameters on the quality in Chapter 7. Based on that study and that method, we give in Chapter 8 a new rate control protocol that takes into consideration not only the traditional network parameters measures for the rate control, but also the user's perception of the quality to dynamically change the sending parameters in order to guarantee the best possible quality.

Part II

Measuring Speech and Video Quality, and Applications

Chapter 5

Measuring Speech Quality in Real-Time

5.1 Introduction

The recent uses of Voice over IP [32], IP telephony [60], and Voice and Telephony over ATM [152] have set forth a great need to assess audio quality, in real time, when the audio is transmitted over any packet network. In addition, the number of network applications that require audio quality assessment increases rapidly. Despite the importance of this point, few methods are available; furthermore, the few contributions in this field are mainly concentrating on the differentiation of encoding algorithms without taking into account network parameters.

In the literature, there are some objective speech quality measures. The most commonly used one are Signal-to-Noise Ratio (SNR), Segmental SNR (SNRseg), Perceptual Speech Quality Measure (PSQM) [15], Measuring Normalizing Blocks (MNB) [139], ITU E-model [68], Enhanced Modified Bark Spectral Distortion (EMBSD) [155], Perceptual Analysis Measurement System (PAMS) [112] and PSQM+ [13]. The main purpose of these measures is to evaluate the quality of speech signals distorted by encoding impairments. Some of these measures work well for this case, but when using them to evaluate the quality of speech signals impaired by both encoding and network transmission, their performance degrades too much (the correlation with subjective measures becomes poor). In addition, the majority of these measures are not suitable to evaluate the quality in real time, as they are computationally intensive and they operate on both the original signal and the processed one.

Transmitting audio signals over any packet network can fall into one of the following categories: a) unidirectional session consisting of a sender that emits frames of audio signals and a receiver that playbacks these frames (e.g. audio streaming [103]); b) bi-directional sessions, when both the sender and the receiver can emit and playback speech frames, producing interactivity between the two ends; c) multi-party conference, when there are more than two ends contributing to the same session. In this Chapter, we are only concerned with category a).

The "Quality-Affecting" parameters that affect audio quality when transmitted over packet network can be classified as follows:

- **Encoding and Compression Parameters:** they are due to the impairments resulting from the encoding or compression of the original audio/speech signals (see Section 3.5 for details about speech and audio compression and codecs). This can be classified into the type of the codec used (ex. PCM, GSM, ADPCM, etc.), the sampling rate, the packetization interval of the signal and the number of bits per sample [28, 16, 43, 80, 86].

- **Network Parameters:** they are due to the transmission of speech streams over packet network. The most known parameters are the loss rate due to network congestion, the arrival

jitter or delay variation, the loss distribution, the and end-to-end delay. Furthermore, effects of error concealment techniques (e.g. silence, noise, repetition, and waveform substitution for lost packets) [28, 38, 80] can affect audio quality (see Section 3.3.2).

- **Other Parameters:** like echo (which may occur due to long end-to-end delay), crosstalk effect (when two or more people start talk at the same time), or the number of participating sources. These effects occur in bi-directional sessions or multi-party conferences [40].

In the previous Chapter, we presented a new method to evaluate the quality of multimedia streams transmitted in real time over packet networks in general, regardless of the media type. In this Chapter, we aim to validate our approach in the case of speech quality assessment. Like any problem to be solved by NN, the development of a tool to evaluate in real time speech quality consists of two steps: the first step is to prepare some examples (a set of input-output pairs), the second step is to train and test a suitable NN using these examples. For the first step, we have to identify the speech-quality-affecting parameters (representing the inputs to the system). Then, we have to build a database consisting of a set of impaired speech samples (the distortion in each one should correspond to selected values of the quality-affecting parameters). Then, the subjective quality test must be carried out to obtain the MOS of all the distorted speech signals (the quality level represents the desired output of the system). In another database, what we call the quality database, the values of the quality-affecting parameters and the corresponding quality score are to be stored. This constitutes the set of input-output pairs or the examples. For the second step, a suitable NN architecture must be selected. Then, the NN has to be trained and tested by the resulting quality database.

The goal is to use the NN to learn the way human subjects evaluate the quality of certain levels of both network and encoding distortions. In addition, as a well-trained NN can generalize well, it can be used to evaluate or predict the quality for wide-range variations of all the input parameters. Therefore, it can be used to evaluate the quality for new (i.e. not within the quality database) values of the quality-affecting parameters in real time.

This chapter is organized as follows: We describe the experiment we used to identify the values of network parameters and their ranges in Section 5.2. In Section 5.3, we present the subjective quality tests we did for different languages. We present the overall procedure used to generate the different speech databases for the MOS experiment, based on a specific network testbed. The results obtained and the validation of our methodology are given in Section 5.4. In Section 5.5, we analyze the performance of some of the existing objective speech quality mesures. Finally, in Section 5.6 we provide the conclusions of this Chapter.

5.2 Measuring Network Parameters in a Testbed

This Section presents the testbed that we used to identify the most relevant parameters and their ranges for real-time transmission of speech. In addition to the parameters mentioned in the introduction, we noticed that real-time audio in general and speech in particular are sensitive to other factors that, when combined with the network behavior, give a different resulting perceived quality. One of these parameters is the spoken language [79]. A series of tests with different languages showed in fact that they do not equally tolerate the losses. For these reasons, all our MOS experiments were done for three different languages (Arabic, Spanish and French).

A survey on IP-telephony users today showed that they generally fall into the following categories. The first one concerns IP-telephony nationwide. For that reason, we did three series of measurements between Rennes (ENST-B) and peers laying respectively in Rennes (Irisa) (2 Km distance), Brest (300 Km) and Sophia Antipolis (1300 Km). The second category is related to international phone calls. For that reason tests were accomplished between Rennes and Mexico City (Mexico). The number of hops, the minimum, the average, and the maximum one-way delay in ms between Rennes (ENST-B)

and the other sites are shown in Table 5.1. The third category of communications in a LAN was not considered because it was difficult to find a real practical situation where a 10Kb/s IP flow does not easily find its way through a 100Mb/s or even a 10Mb/s LAN.

Each session consisted of sending a 160-byte packet every 20 ms as real-time traffic carried by RTP protocol. The duration of the session was 10,000 packets. The receiver reported the total number of packets, the total loss rate, and percentage rate of each n-consecutively-lost packets. The destination considered any packet that arrived after the playback threshold as lost. This is to avoid the jittering problem. In fact, there are several algorithms to choose the best value of the playback threshold in order to minimize the percentage loss rate and to avoid the jittering problem [37].

Table 5.1: Number of hops and one-way delay statistics between the peers (Site) and Rennes (ENST-B) one.

Site	Hops	Minimum delay	Average delay	Maximum delay
Irisa	6	4.2	11	70
Brest	7	7.1	43.3	117
Sophia	12	24	35	60
Mexico	28	149	159	221

For each site, we repeated the tests 50 times in working-day hours and in different days. Then we selected the results that gave the maximum and the minimum percentage loss rate. By varying the playback time length, the percentage loss and the loss distribution change accordingly. Figure 5.1 depicts the minimum and the maximum percentage loss rate. As expected, the loss rates decrease when we increase the playback buffer length of the receiver and the number of hops. In Figure 5.2 and Figure 5.3 we plot the rates of consecutively lost (CL) packets against the buffer size. The first Figure shows minimum values and the second shows maximum ones. The two Figures show the frequency of n^{th}-consecutively lost packets where n varies from one to ten.

As it is clear from these two Figures, in national sites, the three consecutive loss pattern is considered the limit. In the international case, five consecutive losses is the limit. The same approach was taken in [17] (with almost the same paths for tests); however, the number of years separating the two experiments clearly shows an improvement in delay and loss bounds.

The tests described were only intended to give a realistic figure on the average values taken by the parameters that we used in the neural networks training. Therefore, the databases used to train the neural network are as close as possible to real network situations.

5.2.1 Speech-Quality-Affecting Parameters

Based on the statistics collected from the above experiment, we have selected values for the network and encoding parameters as follows:

1. **Loss Rate**: 0, 5, 10, 20 and 40 %. In fact, the loss rate depends on the bandwidth, network load, congestion and the choice of a strict playback time. For more details about packet loss dynamics, see [103]. Of course, one can use any mechanism of FEC [18, 19, 116] to reduce the amount of total loss but on the cost of extra delay, which in turn controls the loss rate if strict playback time mechanism is used [37]. Other techniques for error concealment are given in Section 3.3.2. In networks employing some kind of QoS mechanisms like IP DiffServ or ATM, one can control the amount of losses to achieve certain levels of audio/speech quality [11].

2. **Loss Distribution**: we have chosen the number of consecutively lost packets as the loss distribution, which varies from 1 up to 5 packets dropped at a time. Of course in some situations, there may be more than 5-consecutively-lost packets. However, for the sake of simplicity, we consider them as multiple of 5-consecutively-lost packets.

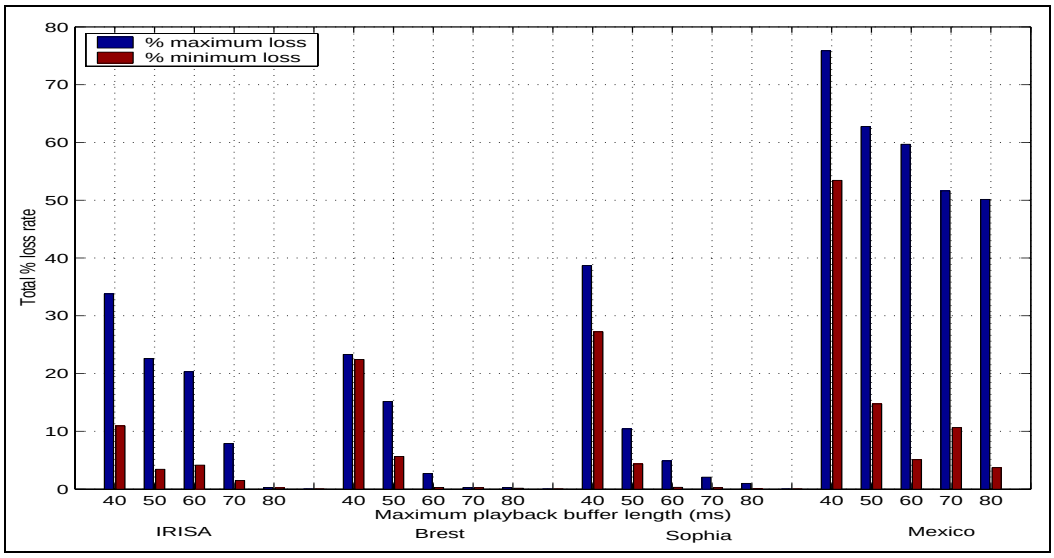


Figure 5.1: Maximum and minimum percentage loss rates as a function of the playback buffer length between Rennes and the other different sites.

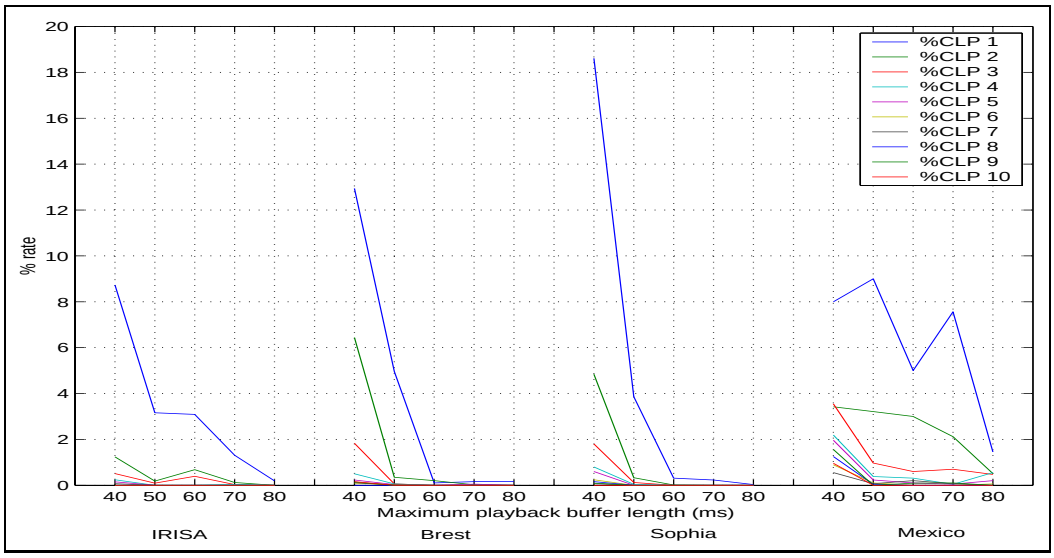


Figure 5.2: Minimum rates for one-to-ten consecutively lost packets as a function of the playback buffer length between Rennes and the other different sites.

3. **Packetization Interval**: We have chosen the values of the packetization intervals as 20, 40, 60 and 80 ms. In fact the majority of the existing real-time applications use one or more of these values. It should be mentioned that people prefer to use small values of packetization intervals to avoid the degradation of the quality due to the increase of the overall delay. On the other hand, using smaller values reduces the effective network utilization. For example, for 20-ms packetized GSM voice, the size of each packet is 33 bytes. When using the RTP/UDP/IP stack, the minimum length of the header of each packet is (12+8+20=40 bytes). In this case, the network utilization factor is bad (cf. Section 8.3.1, page 151).

4. **Encoding Algorithms**: Concerning the speech encoding algorithms, we have selected PCM (64kbps), G726 ADPCM (32kbps) and GSM-FR (13.2kbps). For French language experiment,

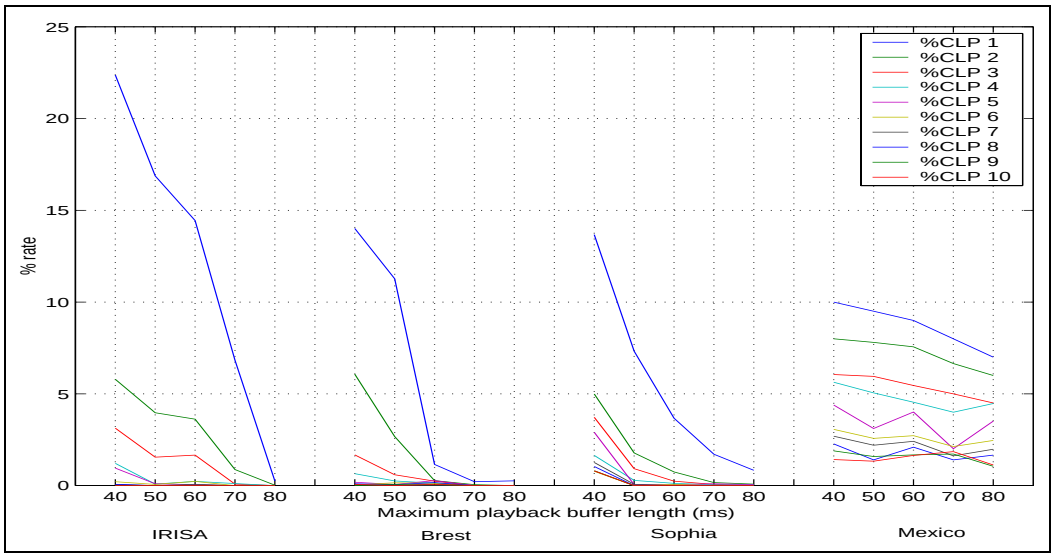


Figure 5.3: Maximum rates for one-to-ten consecutively lost packets as a function of the playback buffer length between Rennes and the other different sites.

we did not consider the ADPCM codec. The corresponding packet sizes are 160, 80, and 33 bytes for a 20ms packetization interval. From the literature [100], the corresponding subjective MOS quality rates are 4.4, 4.1 and 3.6 respectively for English language. These values correspond to an absolute score that evaluates the codec without other impairments. These codecs are the most widely used ones over the public Internet in real-time talks and chat applications.

5.2.2 Other Effects

The delay will map to loss if a strict playback time mechanism is used. Moreover, when a dejittering buffer [37] is implemented, the effect of jitter is masked and mapped to loss. In one-way sessions, there is no echo effect. However, when two-way sessions are used, one should use a mechanism to control the echo effect, like echo suppression or echo cancellation. For a complete discussion for packet loss, delay jitter and the recovering mechanisms, see [60].

5.2.3 Our Method and Session Types

For the case of bi-directional sessions or multi-party sessions, our method can be used the same way as it is employed in unidirectional sessions. The only exception is that an extra parameter should be used to qualify the amount of echo in each side. (Knowing that the end-to-end delay effect is included in the echo estimation, thus, there is no need to add another parameter for the absolute delay effect.) Therefore, only the echo estimator's measure is needed as an extra quality-affecting parameter before completing the distorted databases and training the NN. It should be noted that the quality at both sides may not have the same value because the speech streams do not follow necessarily the same paths. Hence network parameters' values may have different values. In addition, the echo may differ from one side to the other due to the different round-trip time (RTT) from one source to the other.

5.3 A Mean Opinion Score (MOS) Database for Different Languages

Here, we collected a database conformed by a set of samples of distorted speech signals, according to the quality-affecting parameters variations chosen as described in Section 5.2.1 and as shown in

Table 5.2.

Distorted speech signals were generated by using an IP network testbed, comprised by a sender, a router, and a receiver. The sender controlled the packetization interval and the selection of the encoding algorithm. The router controlled the loss rate, and the loss pattern distribution. Finally, the receiver stored the received signals, decoded them, substituted lost packets by silence periods, and calculated the loss rate and number of consecutively lost packets. Of course, one can substitute lost packets by comfort noise, waveform substitution, etc. (See Section 3.3.2, page 64 for more details). For simplicity we chose silence insertion.

Each sample consists of speech material sent from the sender to the receiver according to a given combination of network parameters values obtained in Section 5.2.1 and in data from the literature. In addition, it has been established that encoders' performance shows language dependency [79]. Thus, language type is another variable included in our database. We conducted MOS tests in three different languages: Spanish, Arabic and French. The quality scores were obtained through a series of MOS experiments, which were carried out following an ITU-recommended method [70] named ACR (See Section 4.3 for more details about how to carry out subjective quality tests). During the tests, groups of 15, 15 and 7 subjects (for Arabic, Spanish, and French langauges respectively) were asked to indicate the quality of the speech they heard on a 5-point quality scale for each sample (a total of 96, 96 and 65 for Arabic, Spanish and French languages respectively). Then, a specific statistical analysis was carried out to identify the subjects who were not able to provide reliable ratings as described in Section 4.4. As a result, we removed all the rates of 1 and 3 subjects from the Spanish and Arabic languages respectively. After that, the MOS for each sample were calculated. In this way, we generated the database of distorted speech signals and their corresponding quality scores. The results are shown in Table 5.2.

Table 5.2: The quality databases for both Arabic and Spanish languages to train and test the NN. We omitted the data of French language as the subjective test were carried out by only 7 subjects. In addition, we did not consider nor ADPCM codec nor the same combinations of the parameters as Arabic and Spanish languages.

Codec Name	Packetization Interval PI (ms)	Loss Rate LR (%)	# Consecutive Lost Packets CLP	MOS for Arabic		MOS for Spanish	
				Actual	Predicted	Actual	Predicted
GSM	40	10	1	2.92	2.72	3.13	2.79
GSM	20	10	2	3.17	2.80	2.40	2.75
PCM	40	20	1	2.08	2.40	2.53	2.51
GSM	80	40	2	1.33	1.41	1.40	1.46
PCM	80	10	2	3.08	2.79	3.33	3.20
ADPCM	40	10	5	3.00	2.97	3.40	3.15
PCM	60	5	2	3.50	3.69	3.60	3.80
PCM	60	10	2	3.28	3.02	3.33	3.19
PCM	20	20	2	2.50	2.58	2.53	2.49
ADPCM	80	10	3	3.08	2.86	3.00	3.03
ADPCM	20	10	1	2.25	2.53	2.13	2.70
PCM	40	10	5	3.00	2.90	3.73	3.67
PCM	40	10	3	3.42	3.20	3.00	3.34
PCM	60	5	3	3.58	3.67	3.87	3.99
PCM	40	20	2	2.50	2.59	2.40	2.64
GSM	80	5	3	3.67	3.50	3.67	3.64
PCM	40	40	2	2.00	1.88	1.80	1.67
PCM	40	10	4	3.17	3.08	3.47	3.50

Table to be continued in the next page(s).

Table is continued from the previous page(s).

Codec Name	Packetization Interval PI (ms)	Loss Rate LR (%)	# Consecutive Lost Packets CLP	MOS for Arabic		MOS for Spanish	
				Actual	Predicted	Actual	Predicted
ADPCM	80	5	3	3.33	3.56	3.27	3.67
PCM	40	20	3	2.25	2.54	2.40	2.78
PCM	40	5	3	3.75	3.75	4.00	3.93
PCM	80	5	2	3.58	3.52	3.93	3.85
PCM	20	5	3	3.67	3.63	3.67	3.85
PCM	40	40	2	1.75	1.88	1.73	1.67
GSM	60	5	3	3.29	3.54	3.40	3.59
ADPCM	80	10	5	2.58	2.80	2.93	3.25
PCM	40	40	1	1.42	1.56	1.67	1.61
PCM	40	20	1	2.58	2.40	2.60	2.51
ADPCM	60	5	3	3.34	3.70	3.47	3.62
PCM	40	5	2	3.42	3.71	4.13	3.75
PCM	20	40	2	1.67	1.67	1.40	1.70
GSM	40	10	2	3.00	2.88	2.80	2.84
ADPCM	60	10	3	3.00	3.11	3.13	2.98
ADPCM	60	10	5	2.75	2.82	3.13	3.21
PCM	60	20	2	2.67	2.30	2.60	2.65
ADPCM	-	0	-	4.25	4.23	4.07	4.04
ADPCM	40	5	3	3.67	3.67	3.60	3.55
GSM	60	40	2	1.50	1.47	1.40	1.38
GSM	60	20	2	2.50	2.39	2.20	2.24
ADPCM	40	40	1	1.75	1.67	1.27	1.44
GSM	40	20	2	2.08	2.49	2.13	2.23
ADPCM	40	10	1	2.92	2.83	2.73	2.77
ADPCM	80	10	4	2.67	2.87	2.93	3.13
ADPCM	80	10	2	2.75	2.76	3.07	2.93
GSM	80	20	2	2.08	2.12	2.33	2.23
GSM	20	5	2	3.50	3.15	3.20	3.34
ADPCM	40	10	1	2.67	2.83	2.93	2.77
GSM	80	5	2	3.67	3.39	3.27	3.58
GSM	40	40	2	1.67	1.66	1.53	1.40
GSM	20	40	2	1.50	1.60	1.27	1.41
PCM	40	5	3	3.92	3.75	3.87	3.93
ADPCM	20	10	5	3.28	2.98	3.27	3.08
ADPCM	60	10	4	3.00	3.02	3.13	3.09
ADPCM	60	10	2	2.92	3.01	3.00	2.89
ADPCM	20	10	3	3.08	3.16	2.80	2.87
PCM	40	5	2	3.67	3.71	3.73	3.75
PCM	40	40	5	1.50	1.58	1.80	1.92
PCM	20	5	2	3.33	3.48	3.53	3.68
PCM	80	5	3	3.58	3.59	3.93	4.03
GSM	-	0	-	3.75	3.90	3.80	3.94
ADPCM	40	20	1	2.58	2.41	2.53	2.24
ADPCM	40	10	2	3.17	3.10	2.80	2.85
PCM	-	0	-	4.67	4.49	4.60	4.45
GSM	40	5	3	3.58	3.48	3.87	3.52
PCM	40	5	1	3.42	3.47	3.67	3.59
GSM	40	5	1	3.00	3.08	3.13	3.39
ADPCM	20	10	4	3.00	3.13	2.73	2.97

Table to be continued in the next page(s).

Table is continued from the previous page(s).

Codec Name	Packetization Interval PI (ms)	Loss Rate LR (%)	# Consecutive Lost Packets CLP	MOS for Arabic		MOS for Spanish	
				Actual	Predicted	Actual	Predicted
PCM	40	20	4	2.17	2.39	3.27	2.92
ADPCM	20	10	2	3.00	2.98	2.67	2.78
PCM	40	40	3	2.08	1.91	2.13	1.74
PCM	40	20	5	2.17	2.21	2.80	3.07
PCM	40	5	4	3.92	3.67	4.27	4.12
PCM	40	40	4	1.83	1.76	1.93	1.83
PCM	40	5	1	3.58	3.47	3.60	3.59
ADPCM	40	5	1	2.75	3.25	3.13	3.34
PCM	80	20	2	2.25	2.13	2.53	2.62
ADPCM	40	10	3	2.92	3.22	3.00	2.94
PCM	80	40	2	1.83	1.84	1.53	1.65
ADPCM	40	10	4	3.17	3.17	3.33	3.04
PCM	20	10	2	3.42	3.05	3.27	3.15
GSM	40	20	1	2.17	2.38	2.30	2.19
PCM	40	10	2	3.33	3.20	3.33	3.19
GSM	60	5	2	3.50	3.34	3.73	3.52
PCM	40	5	5	3.75	3.53	4.00	4.31
GSM	80	10	2	2.57	2.81	3.20	2.98
GSM	60	10	2	2.83	2.89	2.60	2.91
PCM	40	10	1	2.75	2.97	2.73	3.05
PCM	60	40	2	1.83	1.90	1.93	1.62
ADPCM	60	10	1	2.75	2.88	2.73	2.80
GSM	20	5	3	3.33	3.35	3.20	3.41
ADPCM	20	5	3	3.25	3.53	3.17	3.45
GSM	20	20	2	2.08	2.38	2.13	2.10
ADPCM	80	10	1	2.92	2.77	2.73	2.85
PCM	40	10	2	3.25	3.20	3.13	3.19
GSM	40	40	1	1.58	1.76	1.27	1.39
GSM	40	5	2	3.17	3.27	3.53	3.45

5.4 Assessment of Speech Quality by Neural Networks

Based on the language dependency result obtained in Section 5.3, when finishing the MOS tests we ended up with one set of samples for each language. A three-layer feedforward neural network for each set of samples (language) consisting of 4 inputs (the quality-affecting parameters) in the input layer, 5 hidden neurons, and an output unit to generate a single output (the quality measure) was used.

We trained the neural network with the first 80 samples of the databases shown in Table 5.2 for each language. For French language, we trained the NN by the first 50 samples. The remaining samples were used to test the trained neural network. By comparing the actual quality scores against the NN's predictions, we show in Figures 5.4, 5.5, and 5.6 the predicted against the actual MOS values for the Arabic, Spanish and French training databases respectively. We show also scatter plots for these cases in Figures 5.8(a), 5.9(a) and 5.10(a). Similarly, we used the trained NN to evaluate the quality score for the testing databases (the last 16 samples in Table 5.2 for the Arabic and the Spanish languages and the last 15 samples for the French language). The samples are not within those in the training databases (thus non-seen by the neural network). We plot in Figures 5.7(a), 5.7(b), 5.7(c) the predicted against the actual MOS values for the Arabic, Spanish and French testing databases

respectively. We show also scatter plots for these cases in Figures 5.8(b), 5.9(b), and 5.10(b). From these figures, we can see two important results. First, the NN has the ability to learn very accurately the evaluation of the MOS for a given set of input parameters (Section 5.2.1). This is clearly shown in the Figures using the training databases. Second, it is able to have a very precise estimation of the MOS for any new values of the input parameters. This is shown in the Figures using the testing databases.

It can be observed from these Figures that speech quality scores generated by the NN model fits quite nicely with the nonlinear model "built" by the subjects participating in the MOS experiment. It can be established from these results that learning algorithms give neural networks the advantage of high adaptability, which allows them to self optimize their performance when functioning under a dynamical environment. The statistics are as shown in Table 5.3. As it can be observed, the results are very encouraging, the neural networks approach allowed us to get a very good model of a nonlinear mapping that resembles the way human subjects assess speech quality. The first experiment we did for the validation of our methodology was with the French langauge. We only used 7 subjects to do the subjective test. Therefore, the precision of the obtained subjective data is not as well as those of Arabic and Spanish cases. However, the NN learned quite well the problem as we can see from the obtained results for this case.

Table 5.3: Performance of the NN to learn the problem and react to non-seen examples.

Language	Training Database		Testing Database	
	Correlation coefficient	MSE	Correlation coefficient	MSE
Arabic	0.966	0.035	0.967	0.035
Spanish	0.969	0.035	0.961	0.045
French	0.965	0.048	0.957	0.055

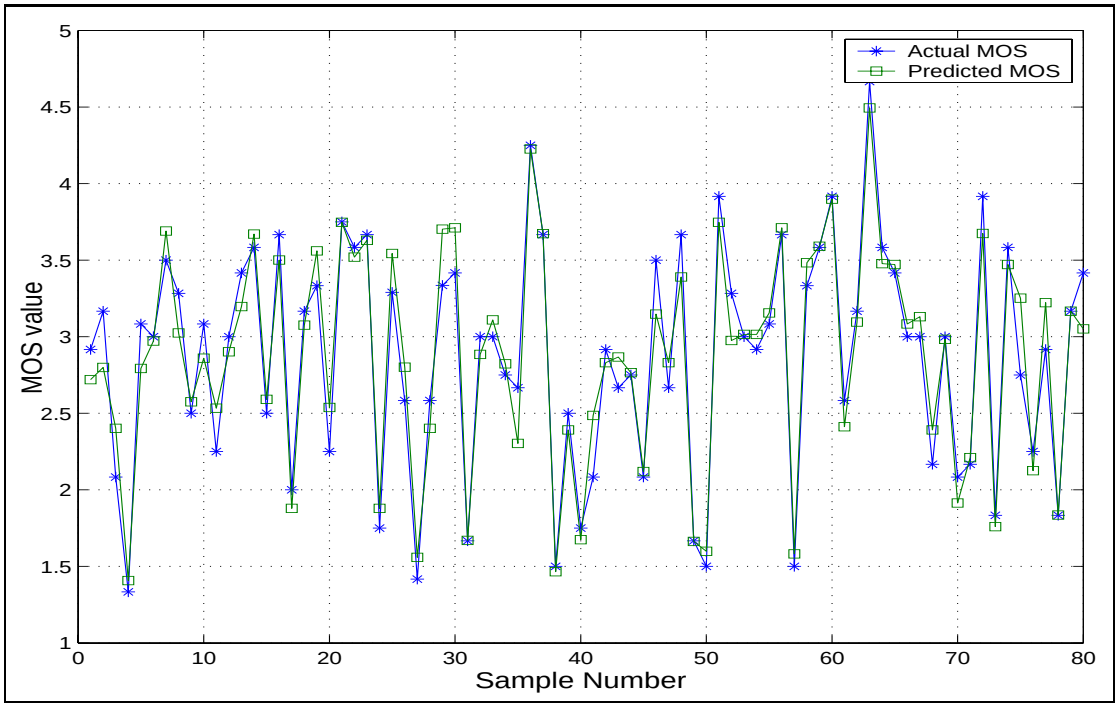


Figure 5.4: Actual vs. Predicted MOS values on the Arabic training database.

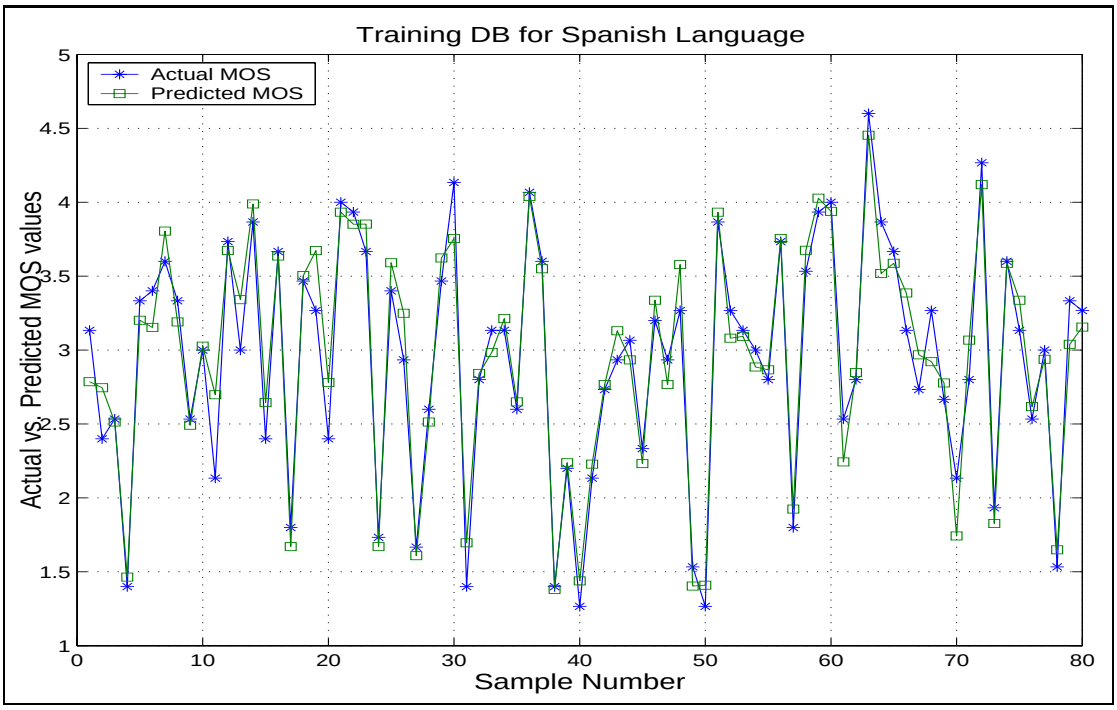


Figure 5.5: Actual vs. Predicted MOS values on the Spanish training database.

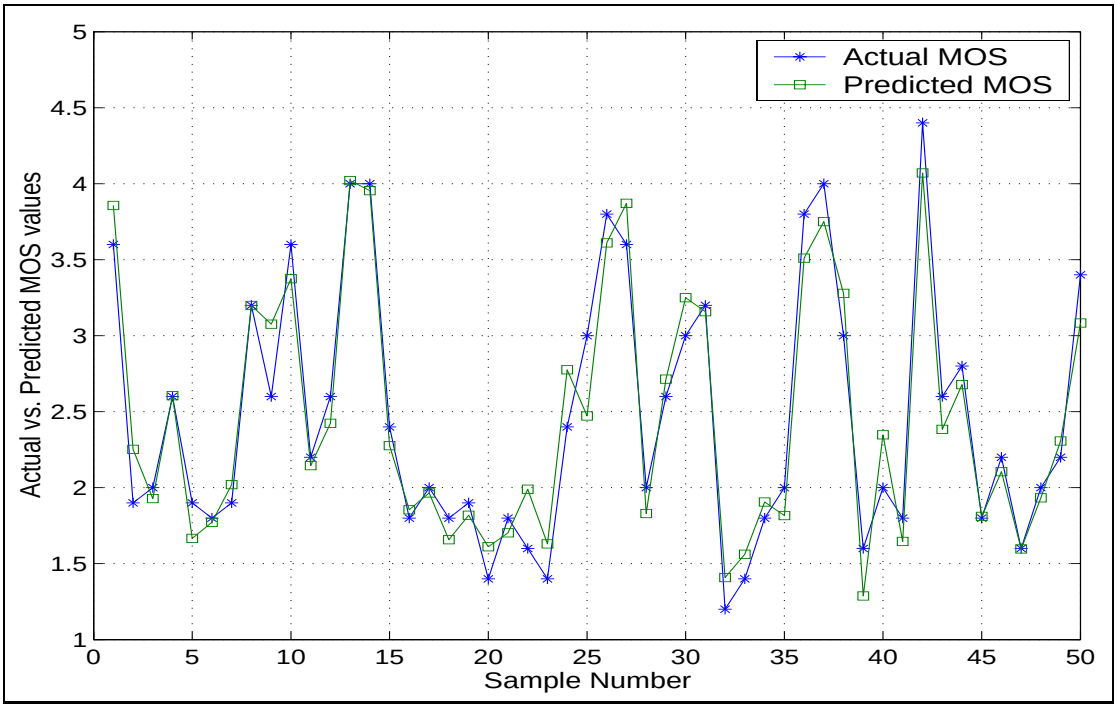
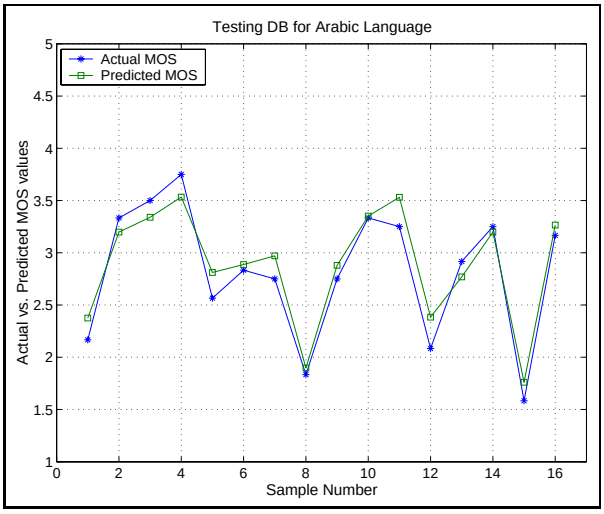
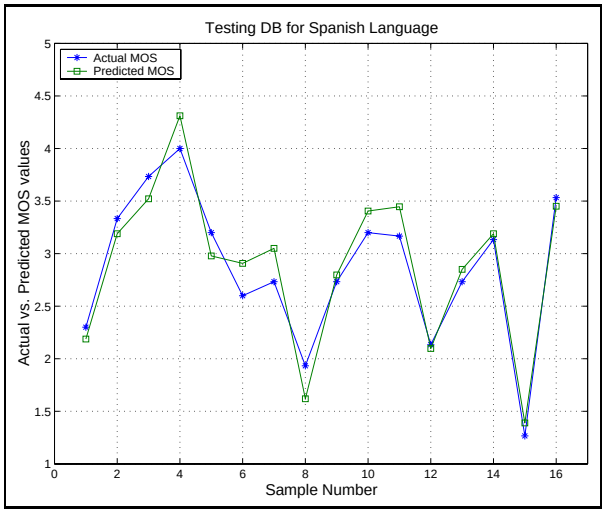


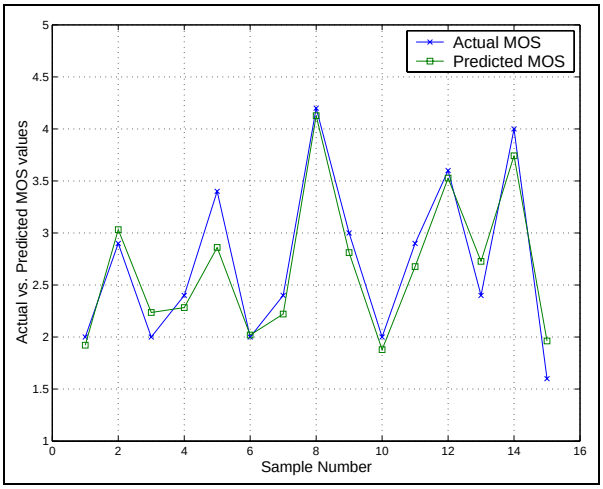
Figure 5.6: Actual vs. Predicted MOS values on the French training database.



(a) Arabic Language

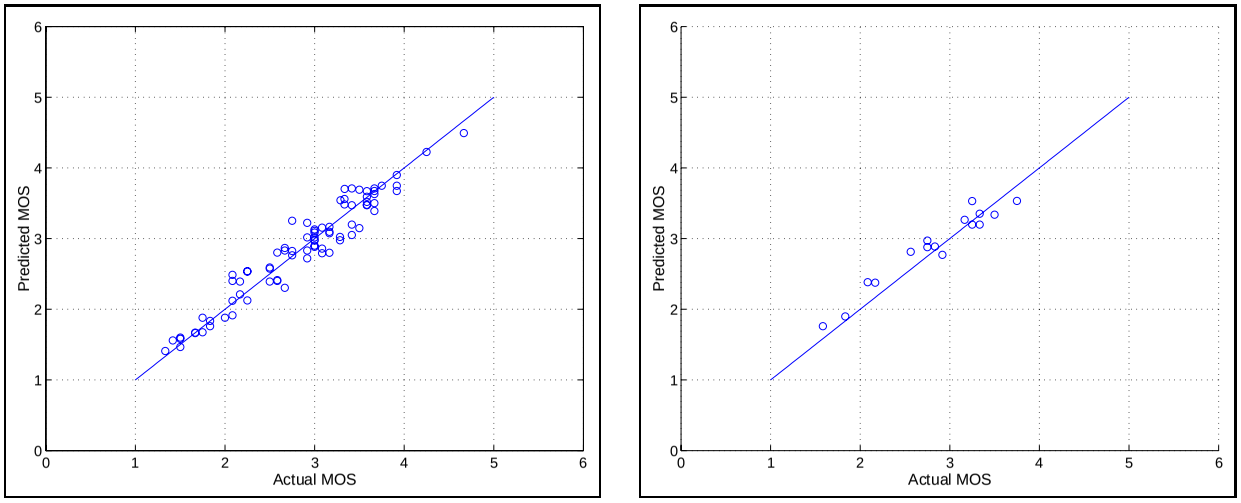


(b) Spanish Language



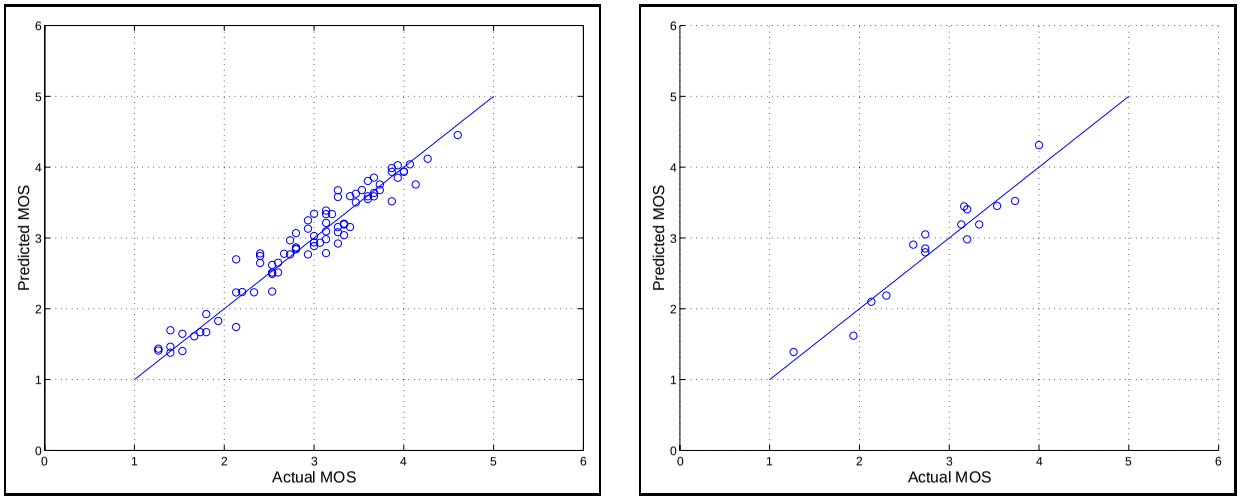
(c) French Language

Figure 5.7: Actual vs. Predicted MOS values on the testing databases.



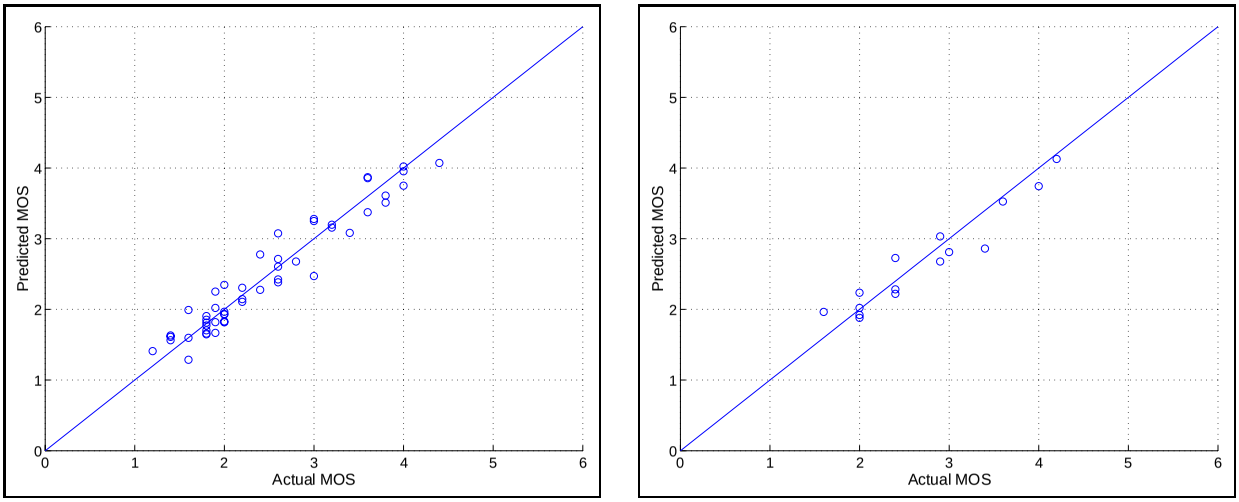
(a) Training DB (b) Testing DB

Figure 5.8: Scatter plots to show the correlation between Actual and Predicted MOS values (Arabic Language).



(a) Training DB (b) Testing DB

Figure 5.9: Scatter plots to show the correlation between Actual and Predicted MOS values (Spanish Language).



(a) Training DB

(b) Testing DB

Figure 5.10: Scatter plots to show the correlation between Actual and Predicted MOS values (French Language).

5.5 Performance of other Speech Quality Measures

Now, to have an idea about the performance of the known objective speech quality measures (namely the ones that have been introduced in Section 3.1), we have collected some data and Figures from the literature.

The reported results from the literature are presented in two parts: the first ones shows the performance of these metrics to evaluate speech quality when only encoding impairments are considered; the second one when they are used with both encoding and network impairments. Depending on the available data, the compared metrics are SNR, SNRseg, BSD, MBSD, EMBSD, PSQM, PSQM+, MNB(1,2), E-model and PAMS.

For the case when the existing metrics are used to evaluate only encoding impairments, from Table 5.4, as we see the simple metrics like SNR and SNRseg give poor correlation with subjective quality tests (ranging from 0.226 to 0.523 for SNR and from 0.221 to .521 for SNRseg). BSD, MBSD and EMBSD give better results than SNR or SNRseg. Regarding, PSQM and its enhanced version PSQM+, the performance is better and it is comparable to that of PAMS and MNB. The correlation coefficient can reach up to 0.98 for certain metrics. The variation of the correlation coefficient is due to the fact that some of these metrics are evaluated for different levels of distortions and others are only available for one kind of experiments.

On the other hand, for the other case when these metrics are used to evaluate the quality taking into account both network and encoding impairments, the performance degrades too much. The only available data comparing the majority of the existing metrics is found in [155, p. 106]. However, due to the project agreement, they did not specify the name of the compared metrics except the MBSD and the EMBSD. We can see that EMBSD gives correlation coefficient of 0.54 in this case against 0.87 for the encoding only case. The performance of MBSD is very bad, about 0.24 against 0.760 for the encoding only case. The best metric has correlation coefficient of 0.90 with MOS results.

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. Thus, it is the only available measure

which is computationally simple [57] and can be used in real-time applications. However, from the results reported in [57], the correlation coefficient, when using it for VoIP quality evaluation, is bad, about 0.70. In addition, the same work presented in [57] compared the performance of EMBSD and MNB (1 and 2) to evaluate the quality of speech signal distorted by network impairments (VoIP), the reported results show that the correlation coefficient of these measures as well as the E-model is bad with the subjective MOS as shown from Table 5.6 (cf. Table 5.3).

We show from that work two scatter plots of the MNB2 and the E-model quality evaluations with respect to MOS in Figure 5.11(a) and Figure 5.11(b) (cf. Figures 5.8 and 5.9). The expected results, for the perfect speech quality measure is that there should be a one-to-one relation between the MOS value and the measure's output. In another words, all the results should be drawn on the same line, with the minimum output of the measure corresponding to the MOS value of 1 and that of the maximum corresponding to the MOS value of 5. However, from the two Figures shown, we can see many variations. For the same value of MOS, there are many values of both measures. For example, for MOS=2.5, the MNB2 output varies from 0.1 to 0.7 (note that output range is from 0 to 1); and that of E-model varies from 20 to 80 (the range of output is 0-100). Similarly, for the same measures' output, there are many MOS values that can satisfy it. For example, for the MNB2's output of 0.4, the corresponding MOS value varies from 1.2 to 4.5; and for E-model's output of 80, the MOS value varies from 1.1 to 4.5 (the MOS scale is from 1 to 5). As we can see this reduces significantly the confidence of these measures to measure the quality of speech signals when distorted by variation of network

Table 5.4: Correlation coefficient of some existing objective speech quality measures with MOS. Results are taken from the literature. Only the encoding impairments are considered. Sources are [155, p. 103], [139, p. 84], [113, p. 1517] and [112, p. 10/7].

Objective Measures	Correlation with Subjective Measure (MOS)
SNR	0.226-0.523
SNRseg	0.221-521
BSD	0.367-0.919
PSQM	0.830-0.980
PSQM+	0.874-0.980
MNB2	0.740-0.98
PAMS	0.640-0.895
MBSD	0.760
EMBSD	0.870

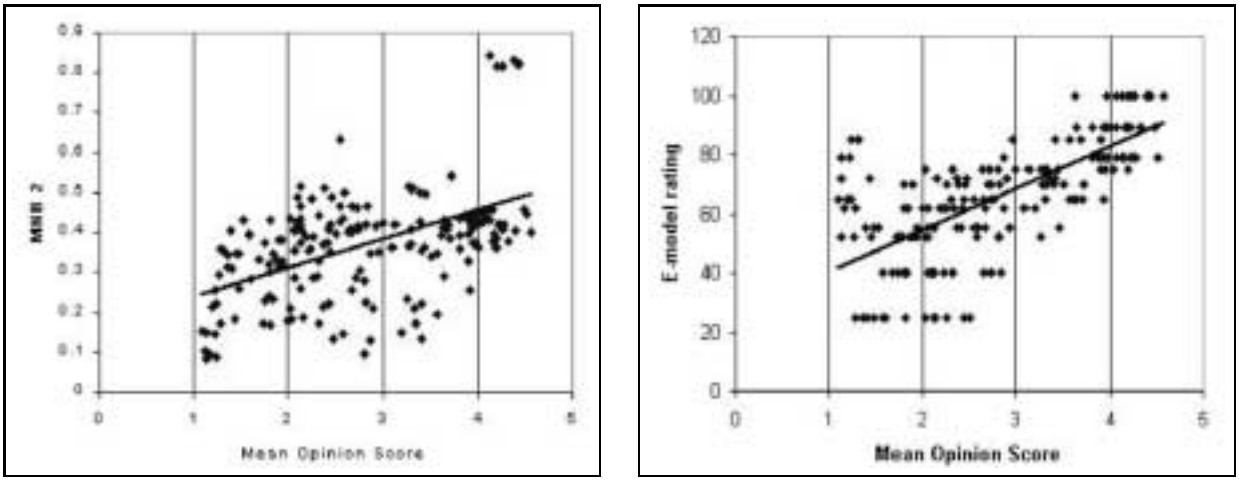
Table 5.5: Correlation coefficient of some existing objective speech quality measures with MOS for both network and encoding impairments. Results are taken from the literature. Source is [155, P. 106]. The letters from A to F represents the objective quality measures introduced in Sec. 3.1. The name of each measure has not been released because of the project agreement.

Objective Measures	Correlation with Subjective Measure (MOS)
A	0.87
B	0.85
C	0.56
D	0.86
E	0.90
F	0.86
MBSD	0.24
EMBSD	0.54

parameters. It should be noted that the E-model is designed and enhanced to take into account the effect of network distortion. In addition, these data is somewhat recent, from 2001. We can see from the results in both Tables 5.5 and 5.6, that there is no agreement in the common measures. This is because the measures in Table 5.5 did not tested by all the network parameters as those shown in Table 5.6.

Table 5.6: Correlation coefficient of EMBSD, MNB and E-model with MOS for VoIP impairments. Source is [57].

Objective Measures	Correlation with Subjective Measure (MOS)
EMBSD	0.38
MNB1	0.51
MNB2	0.54
E-model	0.70



(a) Performance of the MNB2

(b) Performance of the E-model

Figure 5.11: MNB2 and E-model results against the MOS subjective values in evaluating a set of speech samples distorted by both encoding and network impairments. Source is [57].

5.6 Conclusions

In this Chapter, we showed who to use neural networks to learn how humans evaluate the perceived quality of degraded speech signals transmitted over a packet network. We toke into consideration the effect of language, codec type, packetization interval, loss rate, and loss distribution.

We showed that the trained neural network reproduces accurately the subjective evaluations. It is also capable of evaluating subjective quality in presence of new sets of parameters' values. In this way, one can automatically measure in real time and with good confidence the subjective speech quality.

In order to build the database used for training and testing the neural network, we carried out a series of field tests. This helped us in the choice of the mentioned parameters and their ranges.

Chapter 6

Measuring Video Quality

6.1 Introduction

Many real-time video transmission applications over the Internet have appeared in the last few years. We find today: video phones, video conferencing, video streaming, tele-medical applications, distance learning, telepresence, video on demand, etc., with a different number of requirements in bandwidth and perceived quality. This gives rise to a need for assessing the quality of the transmitted video in real time. Traditionally, the quality is measured as a function of the encoding algorithms, without taking into account the effect of packet networks' parameters.

It is now easy to transfer audio in real time over the Internet with reasonable quality. On the other hand, it is not easy to do the same for video. This is because real-time video demands greater bandwidth and, in general, dedicated hardware encoders due to the high complexity of video compression algorithms. In addition, in the case of networks with differentiated services, when audio and video streams are to be transmitted over the same network, audio streams are privileged over video streams when there is congestion. Thus, to guarantee reasonable video quality a dedicated network with sufficiently high bandwidth and QoS levels must be used. This solution is not always feasible due to the high costs of these equipments and the per-service prices. Another factor that makes transferring video more difficult than audio over public networks is the sensitivity to errors. Public networks are usually error prone. However, encoded video streams do not tolerate errors as audio, as video can tolerate only up to 4 % loss without error concealment. While, in speech, with up to 20 %, we get reasonable quality and still understand the talk. Due to these problems, users tend to tradeoff the factor of quality in some cases to deliver real-time video with small resolutions (SQCIF or QCIF) over the low-cost public Internet.

The high complexity of video encoding and decoding processes with respect to those of speech and audio is proportional to the complexity of pure objective video quality measures. Due to the diversity of the encoding algorithms and their complexity, it is difficult to have objective quality measures that take into account all the possible coding and network impairments. In the literature, there are some objective quality measures that work well in some restricted situations, and fail in many others. In addition, as they operate pixel-by-pixel-wise and involve too much computations, the resulting latency and CPU uses limit their use in real-time applications, knowing that video encoding also demands very high computational power. Another weak point is that most of the available objective video quality measures require the access to both the original signal and the impaired one. This is impractical in the case of real-time transmission over packet networks.

Now, if we consider the parameters that affect the quality (quality-affecting parameters) of video transmission over packet networks, we can classify them as follows:

- **Encoding and compression parameters:** they control the amount of quality loss that happens during the encoding process; so they depend on the type of the encoding algorithm (MPEG,

H26x, etc.), the output bit rate, the frame rate (the number of frames per sec.), etc. [53, 82, 153]. It should be noted that there are certain codecs' specific parameters that may affect video quality. For example, using MPEG codecs some parameters are the temporal relation among frame kinds (i.e. I, B, P frames) and the length of the group of pictures; while for the H263 codec (see Section 6.2), the bit rate control strategy (variable or constant) and hence the associated quantization parameter, the Intra MB refresh rate (or any other bit error resiliency method used, see Section 3.3.2 for details) are some parameters. Refer to Section 3.4 for details about video compression and the most known video codec characteristics.

- **Network parameters:** they result from the packetization of the encoded video stream [111] and the transmission in real-time from the source to the destination. They include the following parameters: Packet loss rate resulting from dropping packets when the network is congested, Loss distribution (the burstiness of packet drop, or the number of consecutively lost packets), end-to-end delay resulting from the queuing of packets in the buffers of the traversed routers and the propagation delay and other factors, delay variation of the packets at the receiver(jitter), the type of FEC or any other method (if used) to protect against packet loss (see Section 3.3.2 in page 64 for details) [53, 147, 146], etc. [58, 23, 29].

- **Other parameters:** like the nature of the scene (e.g. amount of motion, color, contrast, image size, etc.), etc. In addition, the delay of the encoding and decoding in video application may degrade the quality if real time transmission is aimed.

Since we concentrate here only on pure video applications, we do not take into account parameters such as lip synchronization or other audio aspects. It is clear that quality is not linearly proportional to the variation of these parameters. The determination of the quality is a very complex problem, and there is no mathematical model that can take into account the effects of all these parameters. Hence, it is normal that, so far, there is no pure objective measure that can quantify the quality of all the effects of all these parameters altogether that can operate in all the situations.

In Chapter 4, we outlined a new methodology to evaluate in real time the quality of any media type transmitted over packet networks in real time. We validated the result for speech in Chapter 5. We would like to apply our method in this Chapter to evaluate its applicability to the case of video. Thus, we show how Neural Networks (NN) can be used to mimic the way by which a group of human subjects assess video quality when this video is distorted by certain quality-affecting parameters (e.g. packet loss rate, loss distribution, bit rate, frame rate, encoded frame type, etc.). In order to illustrate its applicability, we chose to assess the quality of video flows transmitted over IP networks and we carried out subjective quality tests for video distorted by variations of those parameters.

In this Section, we focus on how our approach can be applied to the case of pure video. (For the case when video is associated with other media types, speech or audio, see the end of Section 4.2 in page 79.) We give here a summary of the general description for the case of video. The overall procedure is summarized as follows. First, we have to identify the quality-affecting parameters, their ranges and values. Then we have to choose the original video sequences and to produce the distorted video database. After that, the subjective quality test is to be carried out for the distorted video database and the MOS scores (together with their statistical analysis) are to be calculated in order to form the video quality database. Then, we select the NN architecture and learning algorithm. Then the NN is to be trained and tested using the video quality database. Finally, we implement the final tool that consists of the two modules (parameters collection and MOS evaluation), in order to obtain a real-time quality evaluator.

In most cases, real-time video applications can be considered one-way sessions (i.e. they consist of a sender that produces the video and a receiver that consumes it). This behavior is different from that of audio applications. Indeed, in audio, the interactivity may imply some other parameters (e.g. echo, crosstalk effect, number of participating sources, etc.) that affect the overall quality [92].

The rest of this Chapter is organized as follows. In Section 6.2, we describe the simulator that we used to generate the video sequences required to identify the relation between the subjective quality measures and the parameters' values. A description of the quality-affecting parameters we considered in our experiment is given in Section 6.3. We outline the subjective quality test and the MOS calculation we performed in Section 6.4. The obtained results and the performance of the NN are given in Section 6.5. In Section 6.6, we analyze the performance of some of the existing objective video quality mesures. In Section 6.7, we present an application we implemented based on the obtained results to evaluate in real time the quality of any video sequence. Finally, we conclude our work in this Chapter in Section 6.8.

6.2 Simulator Description

In this Section, we describe the simulator that we used to generate the set of distorted video sequences required to train and test the NN. In addition we use it to build an application to evaluate video quality in real time (see Section 6.7). The simulator consists of an H263 encoder, decoder and network transport simulation.

6.2.1 H263 encoder

The encoder takes as inputs the original video sequence in YUV format and the resolution of the video, which can be any of the standard resolutions or a custom one. The start and the end of frames to be encoded may be given. The size of the search window to detect the motion vectors can also be controlled. The quantization parameter of P encoded frames may be fixed to control the output bit rate. However, the encoder supports several types of bit rate control methods: variable and constant bit rates. Thus, some of them override the quantization parameter option. It is known that the common frame sequence of H263 encoding is IPPPPP..., where the only frame encoded as I is the first one and all the others are encoded as P frames. The size of I frame may be five times large as that of P frame, hence to provide smoothing of the output bit rate, the encoder has the ability to vary the quantization parameter of the first I frame separately.

H263 behaves, by default, as a one layer codec, however, it supports an extra enhancement layer (with the ability of choosing its output bit rate and other options). The encoder takes as well the frame rate of the original video sequence as input. The output frame rate can be controlled by allowing the encoder to skip some images between each two encoded images. For example, if the input sequence is encoded at 30 frames/s, by skipping one image between each two encoded images, the output frame rate will be 15 frames/s.

ITU recommendation for H263 [69] defines some basic operations that all the implementations should support. In addition, some other functionalities are defined separately in annexes as options. The encoder that we deal with supports the following optional functionalities. These functionalities may be activated or not as needed. When activated some other options may be given to further fine tune the operation of the encoder. Technical specifications and details about these options can be found in the H263 standard [69].

- The use of unrestricted motion vector mode as defined in annex D.
- The use of syntax-based arithmetic coding as defined in annex E.
- The use of advanced prediction mode as defined in annex F.
- The use of PB-frames as defined in annex G. In this case, the ability of controlling the quantization parameter of the B frame is given.
- The use of improved PB-frames as defined in annex M.

- The use of advanced Intra coding mode as defined in annex I.
- The use of deblocking filter as defined in annex J.
- The ability of controlling the number of true B frames between P frames as defined in annex O. In this case, the quantization parameter of the true B frame may be varied.
- The ability to select SNR or spatial scalability mode as defined in annex O.
- When using the enhancement layer, in addition, the quantization parameter may be controlled as defined in annex O.
- The use of alternative inter vlc mode as defined in annex S.
- The use of modified quantization mode as defined in annex T.
- It is possible to make the picture sequence like that of MPEG codec, by forcing I frame every n Frames.

In page 191, we show a list of the command-line options of the encoder.

6.2.2 H263 decoder

The H263 decoder reads the streams encoded by the encoder as well as the control information generated to playback the encoded sequence. Like the encoder, it has several command-line options to fine-tune its operation. Instead of displaying the video on the screen (default operation), the decoder has the ability to store the output into a specified file in several formats. Regarding the frame rate, it is possible to precise any value instead of the default which is the rate of the encoded stream.

To simulate loss, there are two added functions. For both options, the decoder reads the packet limits to know the size of each one. In the first method, the lost packets are given in a file by their sequence number. In the second method, we can specify the loss rate and the loss burst size and the decoder will decide which packets to be dropped in a random way to meet the loss patterns. In addition, it is simple to define any other loss method in the code. The decoder should also read the information to know the distribution of the encoded macro-blocks (Intra and Inter) for each packet. In page 193, we show a list of the command-line options of the decoder.

6.2.3 Network Transport Simulation

Several functionalities are added to the codec to make it simulates the operation over packet network by TEMICS team at the IRISA. The encoder generates marks in the output bit streams to indicate the start and end of each packet. In addition, the encoder provides some ways to cope with packet loss effects (error resiliency) [84]. The ability of inserting a header at the beginning of each group of blocks (slice) is allowed to deal with the spatial encoding problem due to loss. In addition, it is possible to force Intra mode coding of macro-blocks in certain way to avoid the problem of loss due to the temporal encoding. The maximum packet size can be varied to suit the requirements of network supporting the transmission. The default size is set to 536 bytes to avoid the fragmentation of packets between routers if the transmission is over the Internet. The packetization process is made to be with conformance with IETF RFC 2429 [69]. The program can also generate packet loss patterns by Gilbert model. Another models of packet loss can be defined. In addition, the decoder has another way to deal with loss distribution by giving it the sequence numbers of packets to be dropped. We further modified the simulator to suite our need to generate the video sequences, to carry out subjective quality tests and to implement the video application.

6.3 The Quality-Affecting Parameters

To generate the distorted video sequences, we used a tool that encodes a real-time video stream over an IP network into H263 format [69], simulates the packetization of the video stream, decodes the received stream and allows us to handle the simulated lost model (See the previous Section).

We used a standard video sequence called *stefan* used to test the performance of H26x and MPEG4 codecs. It contains 300 frames encoded into 30 frames per sec., and lasts 10 sec. The encoded sequence format is CIF (352 lines x 288 pixels). The maximum allowed packet length is 536 bytes, in order to avoid the fragmentation of packets between routers.

We present here the quality-affecting parameters that we consider as having the highest impact on quality:

- The Bit Rate (BR): this is the rate of the actual encoder's output. It is chosen to take four values: 256, 512, 768 and 1024 Kbytes/sec. It is known that not all the scenes compress the same ratio, for the same video quality, depending on the amount of redundancy in the scene (spatial and temporal), as well as image dimensions. All video encoders use a mixture of lossless and lossy compression techniques. Lossless compression does not degrade the quality, as the process is reversible. Lossy compression degrades the quality depending on the compression ratio needed by changing the quantization parameter of the Discrete Cosine Transform (DCT). For details about video encoding, see Section 3.4. If the video is encoded only by lossless compression, the decoded video will have the same quality as the original one, provided that there is no other quality degradation. In our method, we normalize the encoder's output in the following way. If BRmax denotes the bit rate after the lossless compression process and BRout is the final encoder's output bit rate, we select the scaled parameter BR = BRout/BRmax. In our environments, we have BRmax = 1430 Kbyte/s.

- The Frame Rate (FR) or the number of frames per second: the original video sequence is encoded at 30 frames per sec. This parameter takes one of 4 values (5, 10, 15 and 30 fps). This is done in the encoder by dropping frames uniformly. A complete study of the effect of this parameter on quality is given in [53].

- The ratio of the encoded intra macro-blocs to inter macro-blocs (RA): this is done by the encoder, by changing the refresh rate of the intra macro-blocs in order to make the encoded sequence more or less sensitive to the packet loss [84]. This parameter takes values that vary between 0.053 and 0.4417 depending on the BR and the FR for the given sequence. We selected five values for it.

- The packet loss rate (LR): the simulator can drop packets randomly and uniformly to satisfy a given percentage loss. This parameter takes five values (0, 1, 2, 4, and 8 %). It is admitted that a loss rate higher than 8 % will drastically reduce video quality. In the networks where the LR is expected to be higher than this value, some kind of FEC [131] should be used to reduce the effect of losses. There are many studies analyzing the impact of this parameter on quality; see for example [4, 53, 58].

- The number of consecutively lost packets (CLP): we chose to drop packets in bursts of 1 to 5 packets. These values come from real measurements that we performed before [92]. A study of the effect of this parameter upon the quality is, for instance, in [58].

The delay and the delay variation are indirectly considered: they are included in the LR parameter. Indeed, it is known that if a dejittering mechanism with adaptive playback buffer size is used, then all the packets arriving after a predefined threshold are considered as lost. So, in this way, all delays and delay variations are mapped into loss. The receiver holds the first packet and some next packets

in the buffer for a while before playing them out. There are many studies to find the optimal values of the buffer sizes, the number of packets to be hold in the dejittering buffer and the time at which the first packet in the buffer is to be played out [37]. The amount of the hold time is a measure of the size of the jitter buffer.

In article [159], authors argue that the effect of delay and delay jitter can be reduced by the MPEG decoder buffer at the receiver. This is also valid for H263 decoder buffer. A de-jittering scheme for the transport of video over ATM is given in [122]. As stated, the author argues that the scheme can be easily adapted to IP networks. Other study about jitter is given in [162] where it is stated that the jitter affect the decoder buffer size and the loss ratio in a significant way.

If we choose to consider all the combinations of these parameters' values, we have to take into account 4x4x5x5x5 = 2000 different combinations. It is the role of the NN to interpolate the quality scores for the missing parts of this potentially large input space. We followed the procedure given in Section 4.3, in page 81, to find the minimum number of combinations to use. Hence, we chose to give default values and to compose different combinations by changing only two parameters at a time. This led to 94 combinations (as shown in Table 6.2). We will see from the next Chapter that this number is sufficient to train and test the NN.

6.4 Subjective Quality Test and MOS Experiment

We used the method named "Degradation Category Rating (DCR)" test. For details about subjective quality tests and other test procedures, see Section 4.3. We summarize here the used method. A pair of video sequences is presented to each observer, one after the other: subjects should see the first one, which is not distorted by any impairment, and then the second one, which is the original signal distorted by some configuration of the set of chosen quality-affecting parameters. Figure 4.6 (in page 84) shows the sequence and timing of presentations for this test. The time values come from the recommendation of the ITU-R [67].

As the observer is faced by two sequences, he/she is asked to assess the overall quality of the distorted sequence with respect to the non-distorted one (reference sequence). Figure 4.2 (in page 82) depicts the ITU-R nine-grade scale. The observers should give the test presentation a grade from one to nine corresponding to their mental measure of the quality associated with it. It should be noted that there exist several quality scales (see Section 4.3.5.1, in page 82). We chose this nine-grade one as a tradeoff between precision and dispersion of the subjective evaluations.

Following the ITU-R recommendations, overall subjective tests should be divided into multiple sessions and each session should not last more than 30 minutes. For every session, we should add several dummy sequences (about four or five). These sequences should not be taken into account in the calculation. Their aim is to be used as training samples for the observers to learn how to give meaningful rates.

We divided the test into two sessions, and added 5 distorted sequences to the first session and 4 to the second session. These nine sequences were not considered in the MOS calculation as their aim is to be used as a training phase for the human subjects. At the same time, they were used to verify how reliable is the person carrying out the test, as they were replicated from the real 94 samples. Thus the total number of video sequences to be evaluated is 103 samples.

We invited 20 people to perform the subjective tests. Each subject was given a scoring sheet (Table 6.1 depicts a portion of it) and the quality scale. The test procedure was explained to each one. A computer program was implemented to control the evaluation process, keeping in mind the timing aspect of the DCR test as given in Figure 4.6. Hence, the subject typed the number of sample, then the original sample was played back on the screen, with the whole background in gray (see Figure 6.1), as specified in the ITU recommendation. After that a complete gray period of 3 seconds was shown, then the distorted version was played back during 10 seconds. After that a phrase asking the subject to give the mental quality evaluation of the distorted sequence with respect to the original

Table 6.1: A portion of the scoring sheet distributed to each subject.

Session number 1	
Sample Number	Sample Score
1	
2	
3	
⋮	⋮
50	
51	
52	

Session number 2	
Sample Number	Sample Score
53	
54	
55	
⋮	⋮
102	
103	

was displayed. The subject had the ability to review any sequence to decide on the score he had to give if he/she had some doubt.

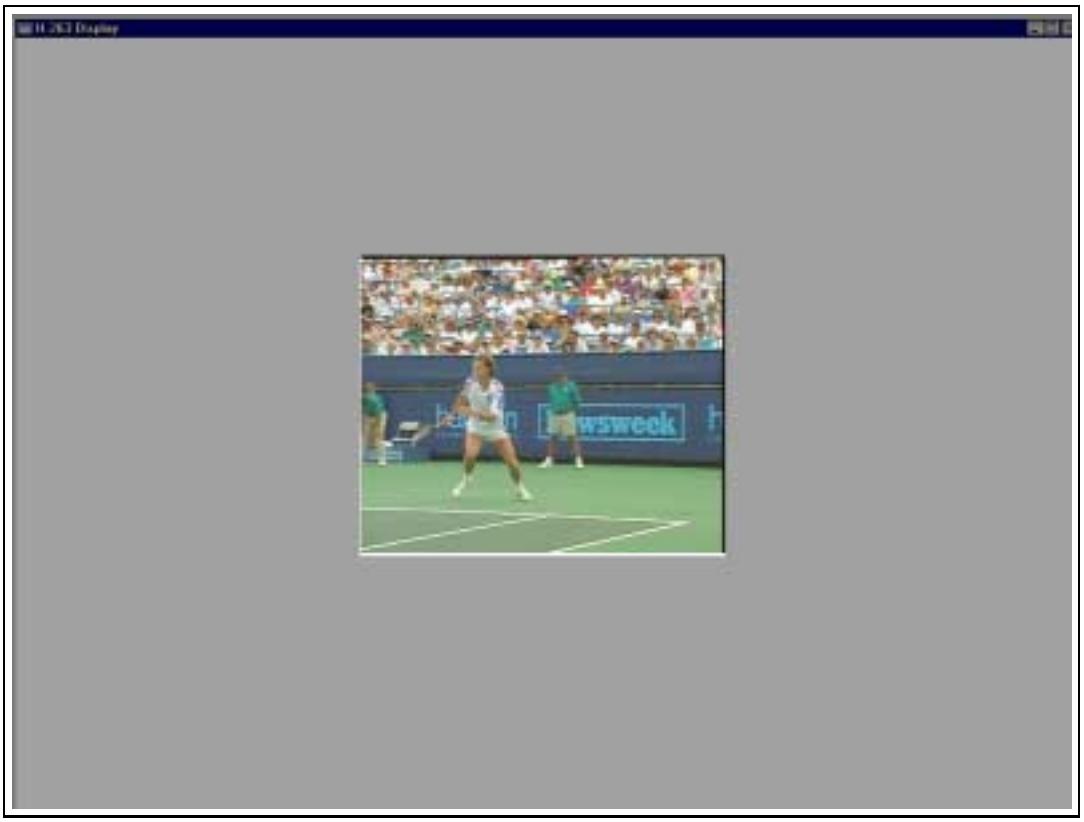


Figure 6.1: A screen dump showing an instance during the subjective quality test evaluation.

After that, a prescreening of the results was performed (see Section 4.4 for details); as a consequence, we discarded the notes of two subjects. The 95% confidence intervals after and before removing these two subjects are shown in Figure 6.2. As it is clear from the Figure, the removal of the rates of these two subjects significantly increased the precision of the overall MOS scores.

Table 6.2: The quality database to train and test the NN (before being normalized)

Bit Rate (BR)	Frame Rate (FR)	Consecutive Loss Packets (CLP)	Loss Rate (LR)	Ratio (RA)	MOS Value	
					Actual	Predicted
768	6	1	0.00	0.12	5.60	5.73
768	15	1	4.30	0.12	4.60	4.27
768	15	4	8.10	0.30	2.70	2.68
768	30	1	0.00	0.14	7.30	7.30
1024	30	1	0.00	0.44	8.60	8.25
256	6	1	0.00	0.14	2.80	3.34
768	15	1	7.50	0.09	4.00	3.53
768	6	1	2.30	0.24	3.60	4.19
768	30	1	0.00	0.40	7.80	7.34
768	15	1	8.60	0.30	2.40	2.27
768	30	1	1.10	0.40	6.30	7.00
768	30	1	1.80	0.40	7.30	6.75
768	30	1	4.10	0.40	5.30	5.06
512	15	1	0.00	0.09	5.40	4.86
256	15	1	0.00	0.18	4.00	3.72
768	15	5	7.60	0.30	3.20	2.97
1024	15	1	3.80	0.36	3.90	3.68
1024	15	1	0.00	0.11	7.60	7.59
768	30	1	8.30	0.40	1.40	2.05
1024	15	1	0.00	0.36	7.40	7.21
1024	15	1	1.20	0.36	6.10	6.02
1024	15	1	2.00	0.36	4.80	5.08
768	30	1	0.00	0.06	6.80	6.81
256	15	1	7.60	0.18	1.30	0.93
768	15	3	1.60	0.30	5.10	5.12
512	10	1	0.00	0.23	5.20	5.11
1024	15	1	8.10	0.36	1.90	2.21
768	15	1	8.60	0.16	2.70	3.09
1024	15	1	0.00	0.14	7.90	7.95
768	15	1	1.20	0.09	5.00	5.24
512	30	1	0.00	0.33	6.30	5.97
768	15	1	2.10	0.09	4.30	4.87
256	10	1	0.00	0.17	3.70	3.35
768	15	1	0.00	0.16	6.90	6.55
768	10	1	0.00	0.19	6.00	6.07
1024	10	1	0.00	0.32	6.30	6.05
768	10	1	2.10	0.29	4.10	4.71
768	15	1	0.00	0.30	7.30	7.74
768	15	2	1.10	0.30	5.80	5.97
768	15	5	0.90	0.30	6.00	6.10
768	15	1	1.20	0.30	5.70	6.41
768	15	1	1.00	0.12	5.20	5.42
256	15	1	0.00	0.10	3.00	3.29
768	15	4	1.30	0.30	5.80	5.72
768	15	1	2.00	0.16	5.10	4.93
256	15	1	1.90	0.18	2.40	2.07
768	10	1	0.00	0.14	6.10	5.93
768	15	3	1.60	0.30	5.00	5.12
1024	15	1	0.00	0.20	8.10	8.24

Table to be continued in the next page(s).

Table is continued from the previous page(s).

Bit Rate (BR)	Frame Rate (FR)	Consecutive Loss Packets (CLP)	Loss Rate (LR)	Ratio (RA)	MOS Value	
					Actual	Predicted
768	15	1	1.20	0.16	5.40	5.40
256	15	1	0.00	0.07	2.40	3.18
768	15	3	4.00	0.30	3.60	3.33
512	15	1	0.00	0.07	5.30	4.74
768	15	2	1.90	0.30	5.60	5.05
768	15	1	2.00	0.30	5.40	5.56
256	15	1	1.90	0.18	2.30	2.07
768	15	5	2.50	0.30	4.80	5.28
768	15	2	3.70	0.30	3.00	3.63
768	15	5	3.90	0.30	4.30	4.06
768	15	1	2.00	0.12	4.80	4.91
768	15	1	0.00	0.09	6.20	6.14
768	15	1	4.20	0.30	4.80	4.25
768	10	1	4.50	0.29	3.80	3.63
768	30	1	0.00	0.09	6.70	6.83
768	15	4	4.30	0.30	4.00	3.50
768	15	1	7.50	0.12	3.00	3.48
512	15	1	3.90	0.24	2.20	2.65
768	15	5	7.60	0.30	3.00	2.97
768	15	2	7.90	0.30	2.70	2.33
768	15	1	4.40	0.09	4.40	4.28
768	10	1	7.50	0.29	1.70	1.63
768	10	1	0.00	0.11	6.00	5.85
768	15	3	8.10	0.30	2.10	2.58
768	10	1	0.00	0.29	6.90	6.64
768	10	1	1.30	0.29	5.00	5.26
768	6	1	7.80	0.24	2.30	2.15
768	15	1	0.00	0.12	6.30	6.30
768	15	1	4.40	0.16	4.70	4.18
1024	6	1	0.00	0.25	5.70	5.68
256	15	1	0.00	0.06	2.80	3.12
512	15	1	0.00	0.24	5.80	5.61
256	30	1	0.00	0.19	4.50	4.14
768	15	4	1.80	0.30	5.60	5.39
512	15	1	1.30	0.24	4.10	4.14
768	6	1	0.00	0.15	5.80	5.72
512	15	1	2.40	0.24	3.40	3.28
512	6	1	0.00	0.21	4.80	4.80
512	15	1	7.90	0.24	1.50	1.61
768	6	1	1.20	0.24	4.30	4.75
768	6	1	3.70	0.24	3.20	3.71
768	6	1	0.00	0.24	6.00	6.11
768	6	1	0.00	0.19	5.50	5.66
512	15	1	0.00	0.13	5.60	5.06
256	15	1	4.60	0.18	1.50	1.34

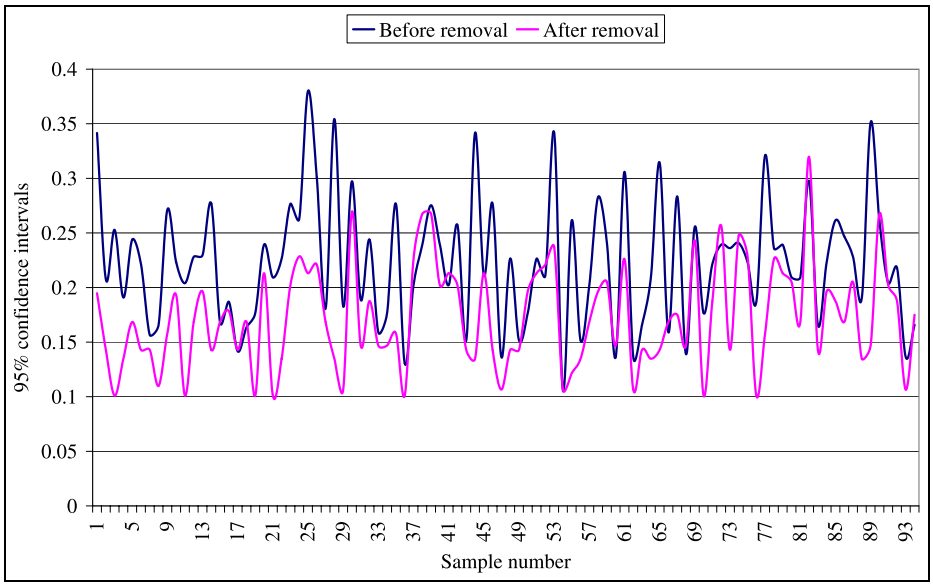


Figure 6.2: The 95% confidence intervals before and after removing the rate of two unreliable subjects. We can see how the 95% confidence interval decrease after the removal of the rates of these two persons.

6.5 Results

Based on the results obtained from the previous Sections (the subjective quality test and the Quality Database), we present in this Section the results of training and testing the NN.

6.5.1 Training the NN

We used a three-layer feedforward architecture. The number of input neurons in the input layer of NN is equal to the number of selected parameters (five). There is only one output of the NN, which is the corresponding MOS score. All the values (inputs and output are normalized to achieve better performance)

The training algorithm for ANN is the well-known Backpropagation algorithm. For RNN, we used Gelenbe's algorithm [47]. In addition we used the set of training algorithms that we developed for RNN. The precision is the same, but using our algorithms, the required iterations to achieve the same error conditions is very small compared to Gelenbe's algorithm (see Section 10.5.4 for numerical data).

The number of hidden neurons in the hidden layer is variable as it depends on the complexity of the problem (inputs, outputs, training set, and the global precision needed). There is a trade-off between the number of hidden neurons and the ability of the NN to generalize (i.e. to produce good results for inputs not seen during the training phase). Hence, the number of hidden neurons should not be too large. In our case we chose 5 neurons for RNN and 4 for ANN.

After carrying out the MOS experiment for the 94 samples, we divided the database into two parts: one to train the NN containing 80 samples (the first in Table 6.2), and the other to test the performance of the NN, containing other 14 samples. After training the NN using the first database and comparing the training set against the values predicted by the NN, we got a correlation coefficient of 0.9801, and a mean square error of 0.108; that is, the NN model fits quite well the way in which humans rated video quality. The result is shown in Figures 6.3 and 6.5(a).

It should be noted that for the optimal number of hidden neurons for both ANN and RNN, the obtained performances are almost the same. However, ANN can be overtrained easily if care is not taken. In this case, the performance on the training phase is very good but it cannot generalize well for the non-seen samples during the training. As previously mentioned in Chapter 4, we varied the

number of hidden neurons for both ANN and RNN. We used them to evaluate the quality for each case when varying the Bit Rate and fixing the other parameters. As we can see from Figure 4.10(a) and Figure 4.10(b) that for the optimal number of hidden neurons, they give the same performance. But ANN cannot generalize for the other values. However, RNN almost give the same results for different values of hidden neurons. In addition, for the maximum value of Bit Rate, RNN gives an MOS score of almost 9 (as expected).

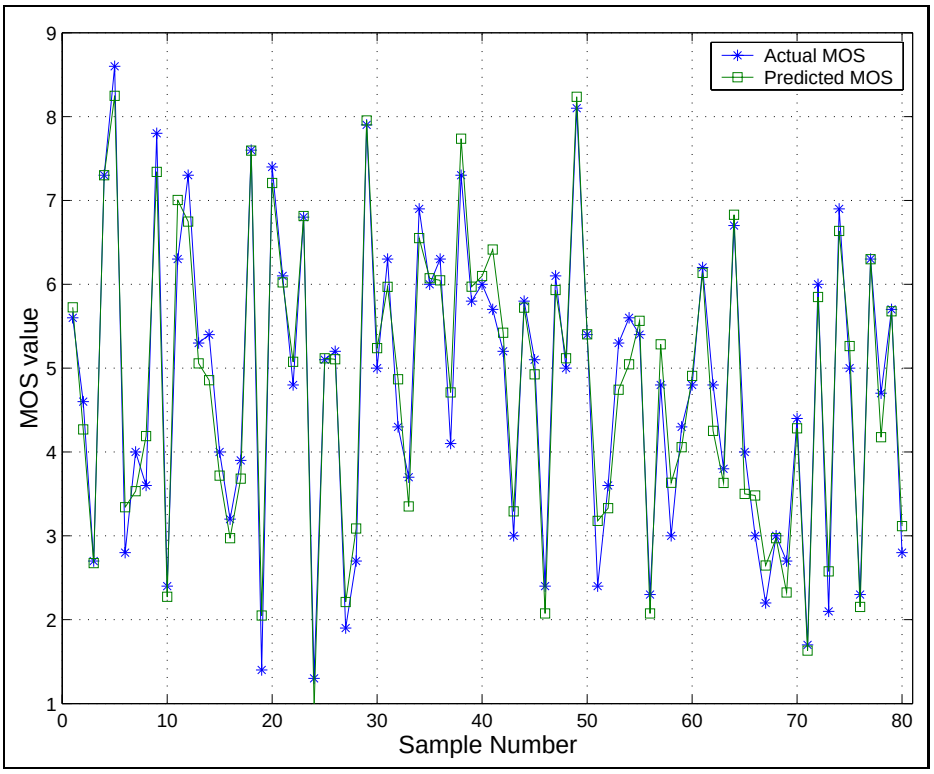


Figure 6.3: Actual and Predicted MOS scores for the training database. The NN learned the way by which the group of human subjects rated video quality for the video samples with very good precision.

6.5.2 How Well does the NN Perform?

In order to address the question "How well does the NN perform?", the NN was applied to the testing set (which contains samples that never have been seen during the training process). The results were correlation coefficient of 0.9821 and a mean square error of 0.07. Once again the performance of the NN was excellent, as can be observed in Figure 6.4. We show also a scatter plot for this case in Figure 6.5(b).

From Figures 6.3, 6.4, and 6.5, it can be observed that video quality scores generated by the NN fits nicely the nonlinear model built by the subjects participating in the MOS experiment. Also, by looking at Figure 6.4, it can be established that learning algorithms give NN the advantage of high adaptability, which allows them to self optimize their performance when functioning under a dynamical environment (that is, reacting to inputs never seen during the training phase).

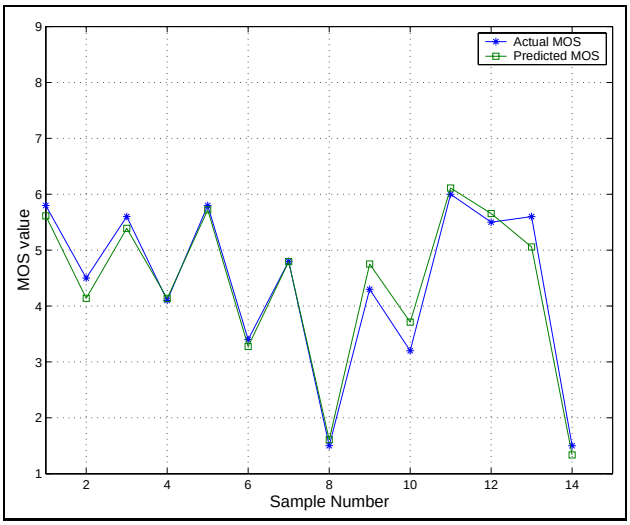
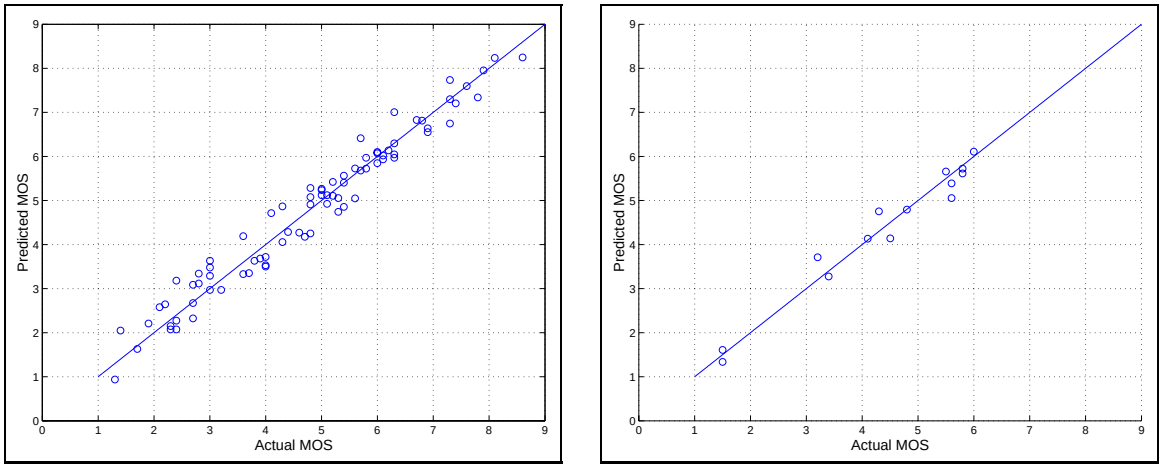


Figure 6.4: Actual and Predicted MOS scores for the testing database. The trained NN is able to evaluate video quality for new samples that never have been seen during the training, with good correlation with the results obtained by human subjects



(a) Training DB (b) Testing DB

Figure 6.5: Scatter plots showing the correlation between Actual and Predicted MOS scores.

6.6 Performance of other Video Quality Measures

As mentioned before in Section 3.2 the most known measures to evaluate video quality are the ITS and EPFL metrics. The correlation coefficient obtained when testing it for some video scenes (see Figure 6.6) without network impairments is about 94% as reported in [143]. However, when this metric is used to evaluate the quality when varying only the bit rate, the performance becomes very bad namely for low bit rate. This can be seen from Figure 6.9 and Figure 6.13. In addition, this measure was designed basically to evaluate the quality of video impaired by encoding distortions. It is not able to evaluate the quality when varying the most basic encoding variable, namely the bit rate. Furthermore, as other models, it requires the access to both the original and the processed signals. This makes it not suitable to real-time application. More details about this measure is given

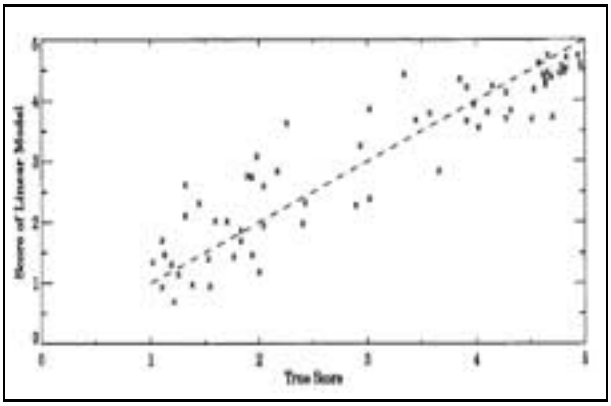


Figure 6.6: The performance of the ITS metric to evaluate video quality. This Figure is taken from [143, p. 508].

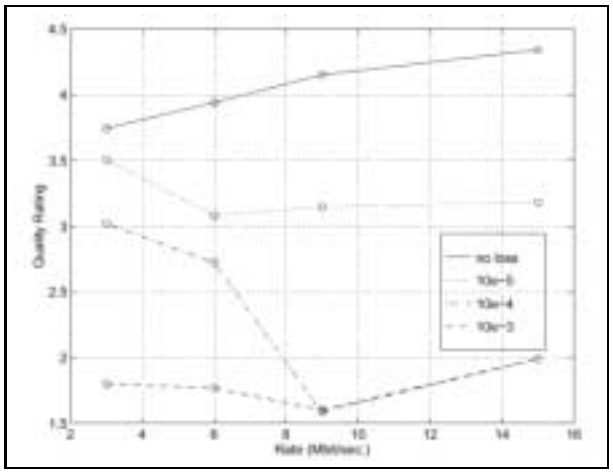


Figure 6.7: Quality assessment by MPQM as a function of the bitrate and the loss rate. This Figure is taken from [135, p. 88].

in [138, 142, 151].

The other ones are the EPFL metrics, namely, the Moving Pictures Quality Metric (MPQM), the Color MPQM (CMPQM), and the Normalization Video Fidelity Metric (NVFM) [135]. All these metrics are designed for high quality video broadcasting evaluation as stated in [135, p. 90] it is concluded that *"Eventually, it is to be noted that such metrics operate best on high quality video sequences and are thus best suited for quality assessment of broadcasted sequences rather than very low bit rates scenes. This is due to the fact that the vision model is essentially a threshold model and thus operates better when the stimulus to deal with, the distortion in this case, is close to threshold, i.e. for high quality scenes."* From the obtained results we can see from Figure 6.7 as well as the next Figures that this is not always true. As shown, the metric cannot evaluate the quality in the presence of loss. Logically, it is expected that the quality becomes better when the bit rate increases (reducing the encoding artifacts). But once, a very small amount of loss is applied, the metric output does the contrary, the quality decreases with the increase of the bite rate.

Another drawback for these metrics is that the computational complexity is too high, and so it may limit their use on real-time multimedia evaluation in the Internet. This is conformed by what is quoted from [135, p. 88] *"Due to computational complexity and memory management, the MPQM could only be applied to 32 frames of the sequences. It has been chosen to always use the first 32 frames of the video stream."*. Note that for a video sequence encoded at 30 frames/sec., there are 300 frames in only 10 sec. In addition, we quoted from [135, p. 135] *"... From an implementation point of view, some efforts have to be put in reducing the computational load of such vision models. Powerful architectures are still needed to obtain fast results with the models [128]..."* What is referred to by "Powerful architectures" are supercomputing ones as the citation 128 indicates. Note that MPQM metric is simpler and less accurate than CMPQM metric and that 32 frames correspond to 1 sec. of video.

Similar to ITS metric, MPQM and CMPQM cannot operate well for low bit rates, this is shown in Figures 6.8, 6.10, 6.12 and 6.13. As we can see, MPQM and CMPQM give a score of 3 instead of 1 on the 5-point quality scale (as it should be at very low bit rate). NVFM gives better results for some sequences, but fails for others, as shown in Figure 6.11 it gives a score of 2 instead of 1 for the "Mobile & Calander" video sequence.

Regarding the correlation with subjective quality tests, it is not very good as we can see from Figure 6.13, the vertical solid lines. Unfortunately, there are no available numerical values for the correlation coefficient with subjective measures, but we can see it is not expected to be good.

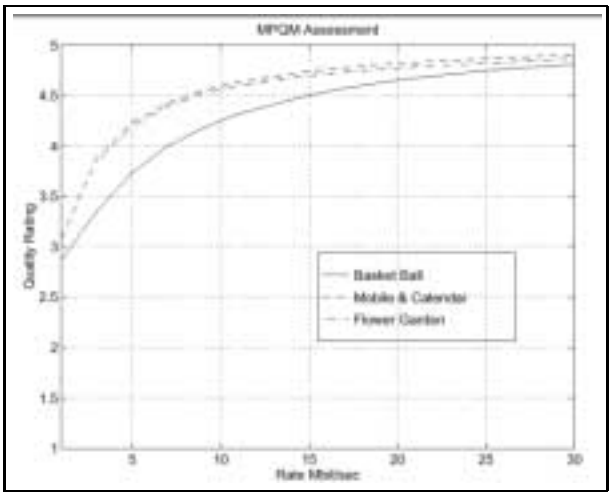


Figure 6.8: MPQM quality assessment of MPEG-2 video as a function of the bit rate. This Figure is taken from [135, p. 82].

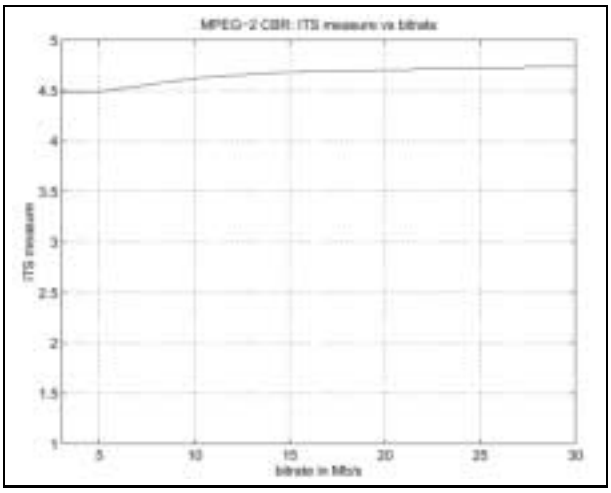


Figure 6.9: The quality assessment by the ITS model of MPEG-2 video as a function of the bit rate. This Figure is taken from [135, p. 83].

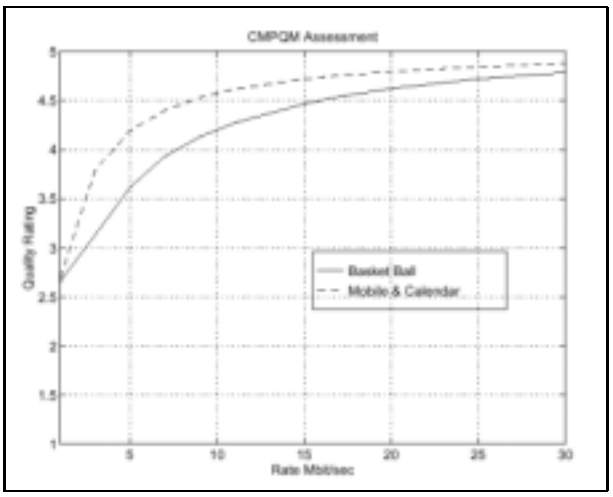


Figure 6.10: CMPQM quality assessment of MPEG-2 video as a function of the bit rate. This Figure is taken from [135, p. 83].

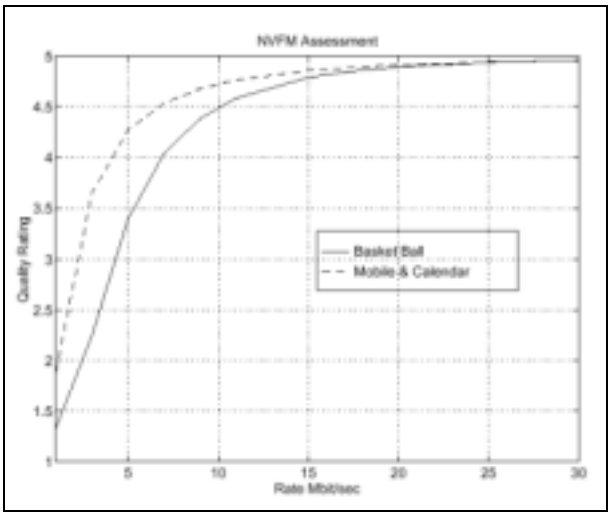


Figure 6.11: NVFM quality assessment of MPEG-2 video as a function of the bit rate. This Figure is taken from [135, p. 83].

6.7 A Demonstration Application

We used the results of this Chapter to implement an application to evaluate video quality in real time for all the ranges of the chosen quality-affecting parameters (namely: LR, FR, CLP, BR, and RA). We have extended the simulator described in Section 6.2 with a graphical user interface to control the variation of the input parameters and to visualize the video under test. (Note that the simulator already supports the packetization and network transmission simulation.)

The application incorporates two operating modes: manual and automatic modes as depicted in Figures 6.14 and 6.17 respectively. In the manual mode, the five parameters can be changed manually. The network parameters (LR and CLP) can be varied smoothly during video playback. However, the codec's parameters (BR, FR, and RA) have to be fixed before starting playback of the video sequence under test. This is because the encoder does not support dynamic variation of the encoding parameters. In addition, we have only considered CBR video transmission. The parameters collector

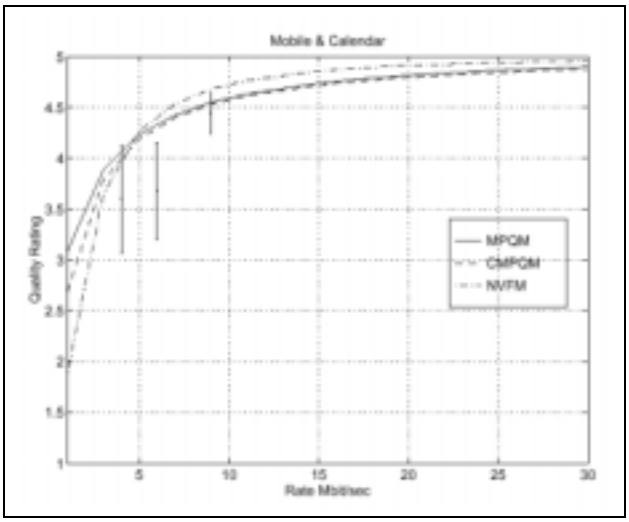


Figure 6.12: Comparison of the subjective data (vertical solid lines) against MPQM, CMPQM and NVFM metrics for the video sequence "Mobile & Calendar". This Figure is taken from [135, p. 84].

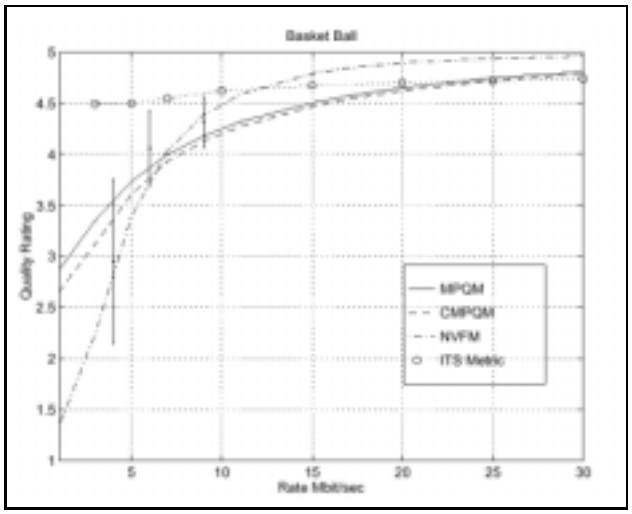


Figure 6.13: Comparison of the subjective data (vertical solid lines) against MPQM, CMPQM, NVFM and ITS metrics for the video sequence "Basket Ball". This Figure is taken from [135, p. 84].

module probe the values of the parameters regularly, every predefined interval (currently fixed to 300 ms). The trained NN module (currently RNN is used) measures the quality based on the values of the parameters.

In the automatic mode, the application plays back the video sequences and distorts them from a predefined sets of parameters' values. The goal of this mode is to study the variation of the quality with the parameters (varying one of them and keeping the others fixed). In the current implementation, we integrated three standard video sequences (namely Stefan, Children, and Foreman).. Each of these sequences has different video characteristics and nature. "Stefan" is characterized by fast motion, while the "Children" sequence is characterized by slow motion, more spatial redundancy and more colors saturation. The "Foreman" sequence is a moderate one between "Children" and "Stefan" sequences. It should be noted that the BR of the best quality of the encoded versions are 1345, 1200, 1000 KByte/s for Stefan, Foreman, and Children respectively. By using the algorithm described in Section 6.3, our application work quite well for all the three video sequences regardless of the variability of the BR from sequence to another. We show the manual mode with the tested video sequences for Stefan, Children, and Foreman in Figures 6.14, 6.15, and 6.16 respectively.

6.8 Conclusions

In this Chapter, it has been described how NN can be used to create a nonlinear mapping between non-subjective video signals measures (i.e., packet loss rate, loss distribution, bit rate, frame rate, encoded frame type, etc.), and a subjective (i.e., MOS) measure of video quality. This mapping mimics the way in which human subjects perceive video quality at a destination point in a communication network.

We have validated our approach by building a NN to assess in real time the video quality transmitted over the Internet, taking into account the previously mentioned parameters. We have shown that the NN performs quite well in measuring video quality in real time. Using the results of this Chapter, we present in the next Chapter a study of the impact of the mentioned parameters on video quality.

We have implemented an application to measure in real time video quality in packet networks. This

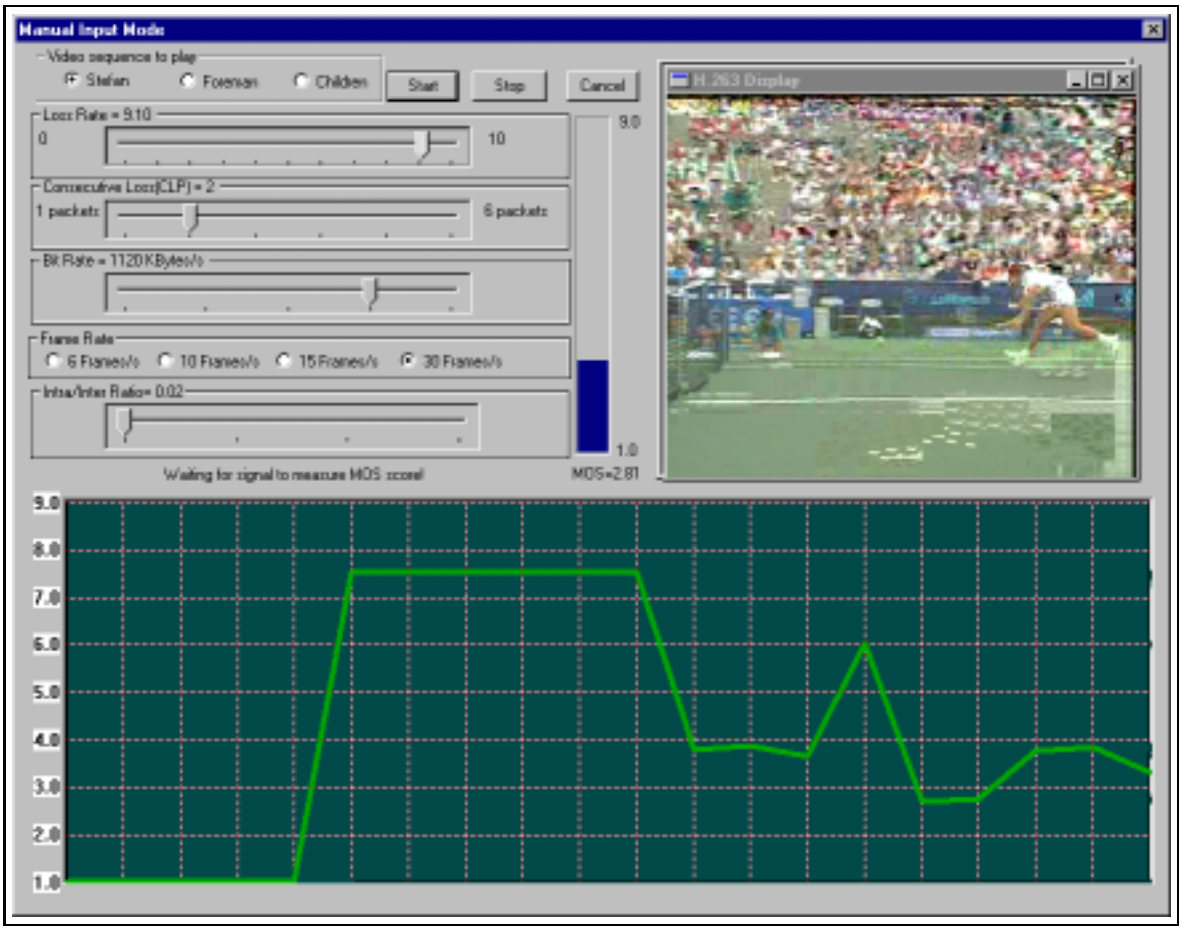


Figure 6.14: A screen dump showing manual mode for Stefan

application incorporate an H263 encoder, a decoder and a network transmission simulator. With the advances in current processors and network bandwidths, it is possible to use software video codecs in real time (traditionally, the encoder required dedicated hardware to run in real time). Our application can be modified to implement a software application supporting real-time encoding, transmission, decoding, and measurement of video quality. One attractive use is video teleconferencing over packet networks.

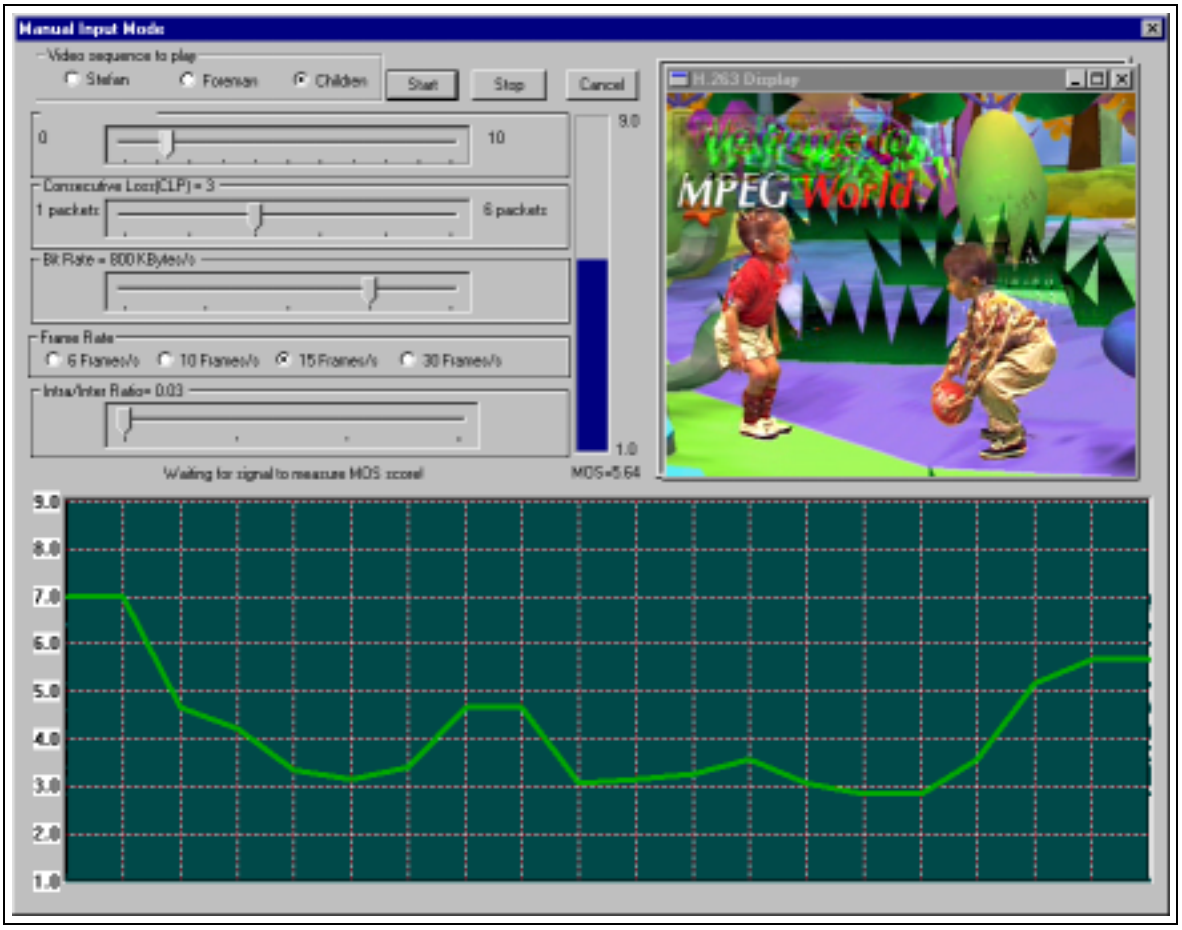


Figure 6.15: A screen dump showing manual mode for Children

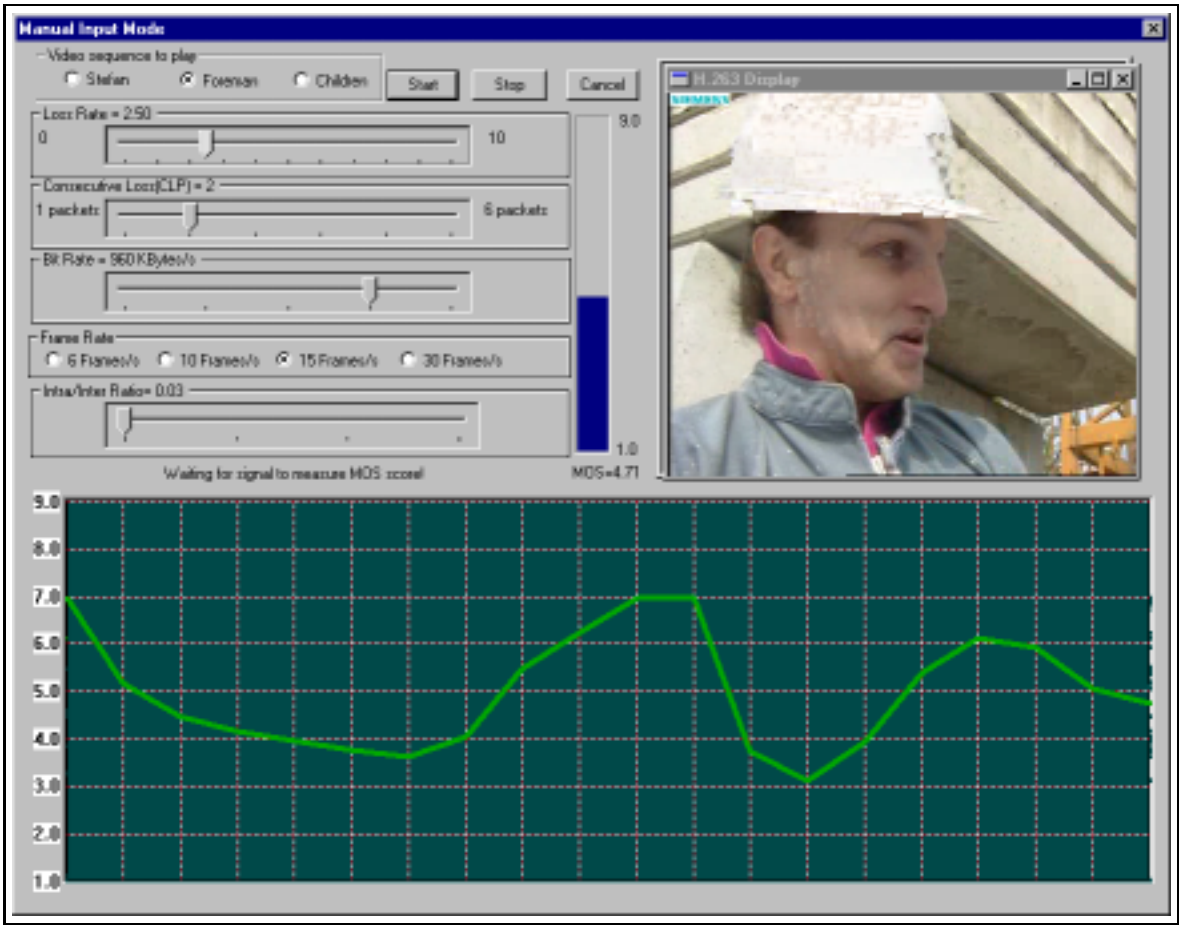


Figure 6.16: A screen dump showing manual mode for Foreman

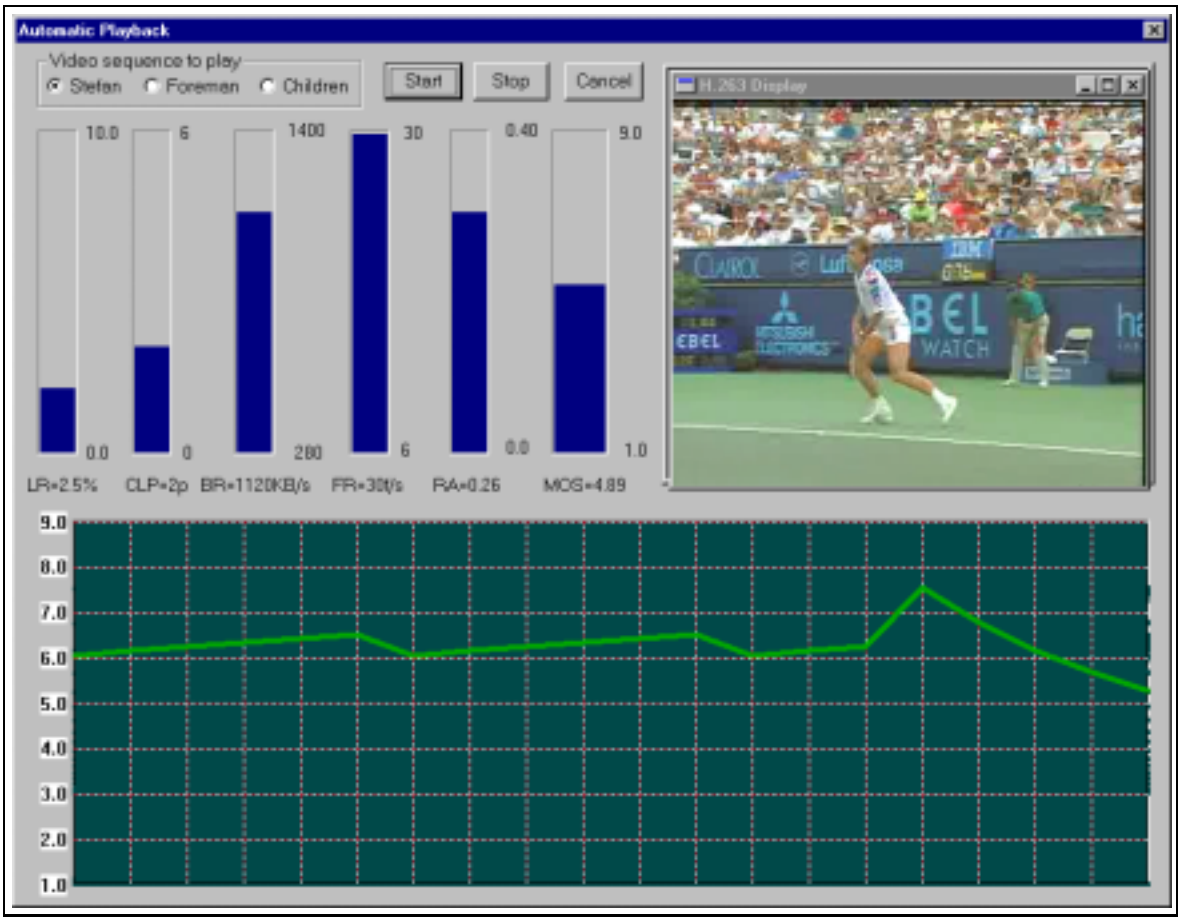


Figure 6.17: A screen dump showing Automatic Mode for Stefan

Chapter 7

Study of Parameters Effects on the Quality

7.1 Introduction

In Chapters 5 and 6, we presented tools to evaluate the quality of speech and video quality when distorted by certain parameters. In this Chapter, our aim is to study and analyze the impact of the quality-affecting parameters for both kinds of media signals. Namely, for speech, the considered parameters are the speech codec used, the loss rate (LR), the number of Consecutively Lost Packets (CLP), and the packetization interval (PI). Similarly, for video study, we consider, the Bit Rate (BR), the Frame Rate (FR), the RAtio of Intra to Inter macroblocks (RA), the LR, and the CLP. For a description of these parameters as well as their ranges and values see Section 5.2.1 and Section 6.3 respectively. We used the results of the previous Chapters to study the combined impact of the mentioned parameters on real-time speech and video quality for wide range variations of these parameters.

Previous studies either concentrated on the effect of network parameters without paying attention to encoding parameters, or the contrary. Papers that consider network parameters and use subjective tests for the evaluation restrict the study only to one or two of the most important ones. For example, the LR and CLP are studied in [58], while [153] studies mainly the effect of BR and [53] works only on the effect of FR, *etc.* In [61], the authors present a study of the impact of both LR and PI on speech quality. Other examples are [29, 75, 130, 119, 28, 61, 86]. (see Section 7.2 for more details.) The main reason for this is the fact that subjective quality tests are expensive to carry out. Moreover, to analyze the combined effect, for instance, of three or four parameters, we need to build a very large set of human evaluations in order to reach a minimal precision level.

In this Chapter, we use RNN to measure speech and video quality as a function of the quality-affecting parameters. Concerning the approach followed in this Chapter, based on the RNN model, it is important to mention that RNN are used in several related problems. For example, they are used in video compression with compression ratio that goes from 500:1 to 1000:1 for moving gray-scale images and full-color video sequences respectively [31]. They are also used as decoders for error correcting codes in noisy communication channels, which reduces the error probability to zero [2]. Furthermore, they are used in a variety of image processing techniques which go from image enlargement and fusion to image segmentation [50]. A survey of RNN applications is given in [12] and a comparison between ANN and RNN is given in Section 4.5.1.

The organization of the rest of this Chapter is as follows. An overview of the related works is given in Section 7.2. In Section 7.3, we present our study and analysis of the impact of the parameters on real-time speech quality. We follow the same approach in Section 7.4, we present our study and analysis of the impact of the parameters on real-time video quality. Finally, in Section 7.5 we provide

the conclusions for this Chapter.

7.2 Related Works

In [29], the authors study the effect of both loss and jitter on the perceptual quality of video. They argue that, if there is no mechanism to mask the effect of jitter, the perceived quality degrades in the same way as it degrades with losses. In [134, 58, 75, 146, 147, 130], the effect of audio synchronization on the perceived video quality is analyzed (for instance, by quantifying the benefits of audio synchronization on the overall quality).

The main goal of [53] is to study the effect of the FR for different standard video sequences on the overall perceived quality. A related work is [119] where the effect of FEC and FR on the quality is the subject of the study. The work presented in [23] is a study of LR effects on MPEG video streams. The authors consider the effect of LR on the different types of MPEG frames. In [153] and [102], a study of the effect of BR on the objective quality metrics (PSNR, NVFM, and MPQM) is presented. The effect of CLP on video quality is analyzed in [58]. The authors of [82] study the effect of packet size and the distribution of I-frames in the layered video transmission over IP networks. In [24] the analysis goes deeper: the authors present a study of the effect of motion on the perceived video quality.

In [61], the effect of both LR and PI on speech quality is presented. The study is based on an automatic speech recognition system which is based on hidden Markov models instead of the usual subjective quality tests. The effect of LR on three speech codecs is evaluated in [28]. The codecs under consideration are Adaptive Differential Pulse Coded Modulation (ADPCM), Sub-Band Coding (SBC), and Adaptive Predictive Coding (APC). The authors concluded that there are two effects due to lost packets. First missing of fragments of the speech. Second, the adaptation logic is disrupted after the lost segment (in the decoding part). Pure delay effects on speech quality in telecommunications are evoked in [80]. It is mainly for conversation sessions. They argue that round-trip delay in the range of 500ms is annoying. In [146] the authors studied the variation of speech quality for different ways of error concealment (silence, waveform, or LPC repair). Additionally, they evaluated subjective speech quality against loss and they compared the result for redundancy and no redundancy cases. The impact of cell delay varition on speech quality in ATM networks is given in [86].

7.3 Parameters Impact on Speech Quality

To illustrate the impact of the considered parameters (Codec, LR, CLP and PI) on speech quality, we have drawn a set of 3D Figures and a bar-graph. To generate the data necessary to draw these Figures, we used an RNN trained and tested by the "Spanish" quality databases described and evaluated in Chapter 5. We did the same tests using the "Arabic" databases, but the results were approximately the same, therefore, we omitted the resulting Figures as they will not provide further useful information concerning our study here.

For all the figures shown in Figure7.1, on the left-hand side, we show the variation of the quality as a function of LR and CLP for the different codecs and for PI=20ms. In this Figure but on the right-hand side, we show the variation of the quality as a function of LR and PI for the different codecs and CLP=1. In Figure 7.2 on the left-hand side, we show the variation of the quality as a function of CLP and PI at LR=5 %. On the right-hand side we do the same but for LR=10 %. Note that the Z-axis scale is the same for each pair of Figures. We show also in Figure 7.3 the variation of the quality as a function of LR and the different codecs for PI=20ms and CLP=2. In the next subsections, we study the effect of each parameter to the quality.

7.3.1 Loss Rate (LR)

As shown in all the Figures, the most dominant effect on the quality is due to the LR. Speech quality degrades severely when LR increases for constant values of the other parameters. As shown in Figure 7.1(a), for PCM codec at PI=20ms, the quality degrades from 3.7 to 1.5 (about 2.2 points) and from 4.1 to 2.2 (about 1.9 points) when LR increases from 5 to 40 % for CLP=1 and 5 respectively. The same results is qualitatively the same for the other codecs, knowing that PCM gives the best quality for the same values of the other parameters. From Figure 7.2, we can see that the effect of CLP and PI is more important in the situation where LR=10 % than that where LR=5 %. In addition, we can get the same quality by changing these two parameters if the LR changes. For example, for CLP=2 using ADPCM codec, the quality when LR=5 % and PI=80ms is the same as LR=10% and PI=20ms (about 3 points). The same observation is for CLP. In addition, if we have poor network conditions, by changing the codec, the quality can be improved. A difference of 0.5 point is generally observed when changing from ADPCM to PCM.

When there are lost packets in the speech flows there are two effects that degrade speech quality. The first is that lost parts of some words occur and parts of speech become incomprehensible. The other effect is due to the encoding and decoding algorithms. Some encoding algorithms use the information of the past speech frames to encode the current. If one frame is lost many frames cannot be decoded and the effect of loss becomes more annoying than for other codecs that do not use temporal compression. The effect of LR has been studied in some related works in the literature, namely in [28, 61].

7.3.2 Consecutive Lost Packets (CLP)

As shown in Figure 7.2 and Figure 7.1(a), 7.1(c) and 7.1(e) the effect of increasing CLP is benefic to speech quality when keeping the other parameters constant. An absolute improvement of 0.5, 0.7, and 0.5 when CLP increases from 1 to 5 in the Figures is shown on the left-hand side in Figure 7.2, where LR=5 %. While, we can see an improvement of 0.8, 0.7, 0.6 when LR=10 %. However, as we can see from Figure 7.2, there is no improvement on the quality when PI is high (about 60ms). Thus, the improvement becomes more important for smaller values of PI.

The explanation of this phenomena is as follows. When fixing all the other parameters (including the LR) and increasing CLP, the number of loss occurrences decreases. Therefore, the number of distorted words in the speech decreases when increasing CLP.

7.3.3 Packetization Interval (PI)

The PI represents directly how many speech frames there are in each packet. Similar to CLP, the overall effect of increasing PI tends to improve speech quality. This is clearly shown from Figures 7.1 and 7.2. However, for smaller value of LR, there is no noticeable improvement when changing PI. Its effect becomes more important for higher LR and smaller CLP. For higher value of CLP, there is no effect of PI on speech quality. We can see that, the effect of PI depends on the used codec and the value of LR. For example, using PCM at 5 % LR, only an improvement of 0.2 against 0.4 for LR=10 %. However, when using GSM, the improvements are 0.3 and 0.6. In addition, when the network conditions are bad, we can improve the quality by increasing the PI. For example, using ADPCM codec and when CLP=1, the quality is the same (3 points) when (LR=5 % and PI=20ms) and when (LR=10 % and PI=40ms). These observations are important, because unlike the network parameters (LR and CLP), which could not be modified except by using FEC or improving the network infrastructure, this parameter can be modified to improve the quality. Another advantage of increasing PI is to improve the network utilization as explained in Section 8.3.1. Our result we get here regarding this parameter is with agreement with that in [61].

7.3.4 Speech Codec

It is known that speech codecs give different values of subjective speech quality, depending on the compression rates and the encoding algorithm employed (as discussed in Section 3.5). For no network impairments (LR=0), the subjective quality obtained is 4.6, 4.1 and 3.7 in the Arabic language when using PCM, ADPCM and GSM codecs respectively. In Spanish language, the obtained scores are 4.7, 4.2 and 3.8. (Refer to Figure 7.3.) These results show that the subjective quality tests we have carried out and explained in Chapter 5 is in agreement with what can be found in the literature, as it is reported that the respective scores for these codecs are 4.4, 4.1 and 3.6 for the English language [60, 79].

We can also see from Figure 7.2 that the codecs considered do not tolerate the loss equally. Thus, for the same network conditions, by changing the speech codec, we can get better quality. For example, there is an improvement of 0.6 when changing from GSM to PCM at both LR=5 % and 10 %.

7.4 Parameters Impact on Video Quality

In this Section, we study the effect of the five parameters (BR, LR, CLP, FR and RA) on video quality. As it is impossible to visualize the variation of the quality as a function of all the five parameters, we chose to visualize on a set of 3D Figures in which we varied two parameters and kept the other three fixed. The MOS value is computed as a function of the values of all the five parameters by using a RNN trained with the quality databases described and evaluated in Chapter 6. It should be noted that the axis of these Figures are rotated in such a way to give the best visualization of each Figure. This means that the orientation of some axis can vary from Figure to Figure. In addition, the scale of quality axis (or Z-axis) is variable from one Figure to another to show some useful features of the given Figure. In the following subsections, we analyze the effect of each parameter on video quality and we also discuss their main combined effects.

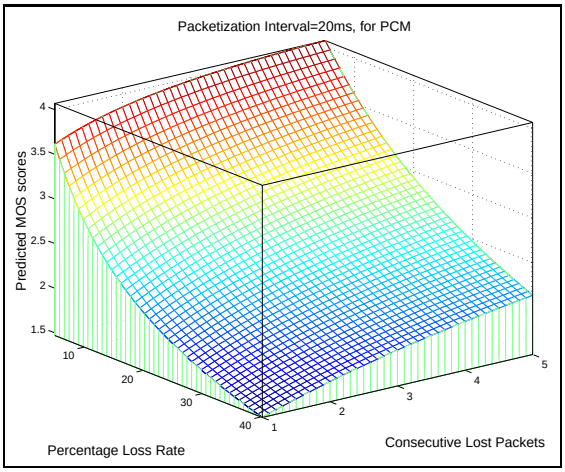
7.4.1 Bit Rate (BR)

As shown in Figure 7.4 through Figure 7.7, the parameter BR exhibits a great influence on the quality. From Figure 7.4, MOS values vary from 3 to 6.7 for (FR=6 fps, LR=0 %, CLP=2, and RA=0.1) and from 3.7 to 7.7 when the FR goes up to 30 fps. This is for a variation of the BR from 0.15 to 0.8. Its effect is comparable to the effect of LR as shown in Figure 7.5. The improvement of the quality is significant for lower LR and decreases for higher LR values. The quality goes from 2.8 to 7.8 for zero loss, but it only presents about 1.3 of absolute improvement for a 10 % LR. The fact that in Figure 7.6 there is no effect of CLP on the quality comes simply from the setting LR = 0.0 %. This shows how well the NN learned the problem of evaluating the quality. From Figure 7.7, for higher values of the BR, there is no effect of the variations of the RA parameter on the quality; however, its effect increases and becomes benefic for lower values of BR.

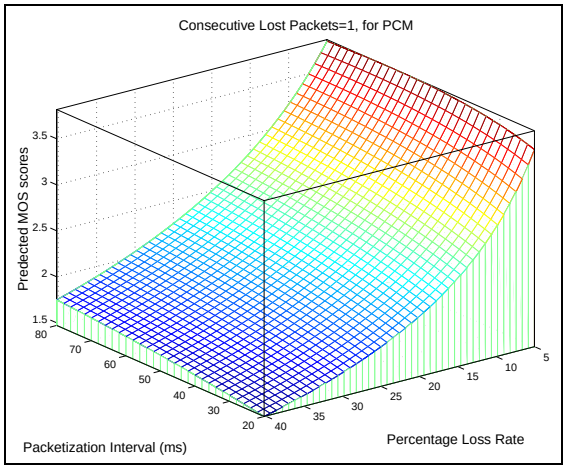
When the encoder has to decrease the BR of the given stream, it increases the quantization parameter in order to compress further the original image. This process increases the artifacts of the output stream, and hence increases the distortion which becomes more noticeable by humans as the BR decreases. This parameter is studied in [153, 102]; however, the interaction with other parameters is not shown.

7.4.2 Frame Rate (FR)

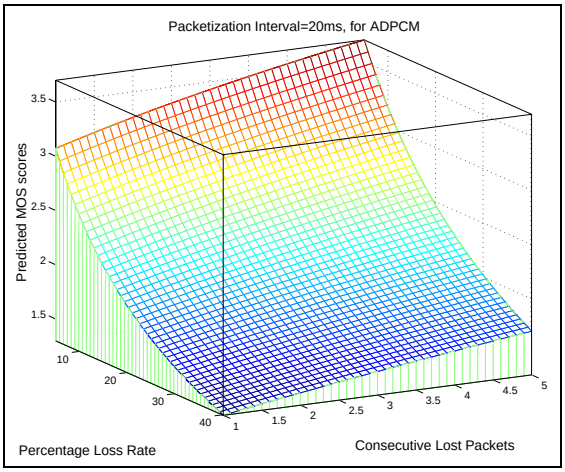
The effect of this parameter on the quality is not as significant as in the BR or LR (see below) cases. This is clearly shown in Figs. 7.4, 7.8, 7.9 and 7.10. For (BR=0.8, LR=0 %, CLP=2, and RA=0.1), an improvement from 6.6 up to 7.6 is achieved when the FR varies from 6 to 30 fps. Similarly, there is an absolute change of 0.8 for BR=0.15, as depicted in Figure 7.4. We can also note that the enhancement



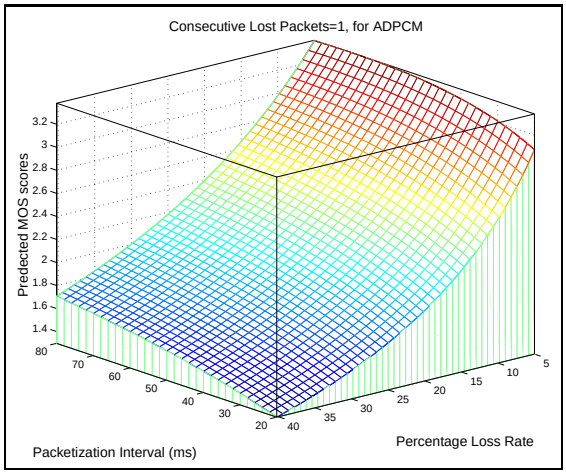
(a)



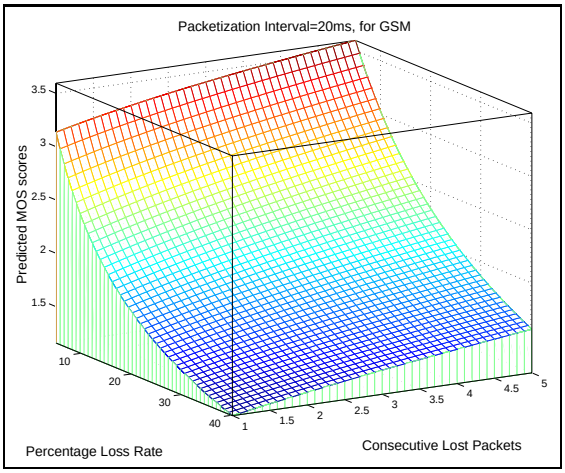
(b)



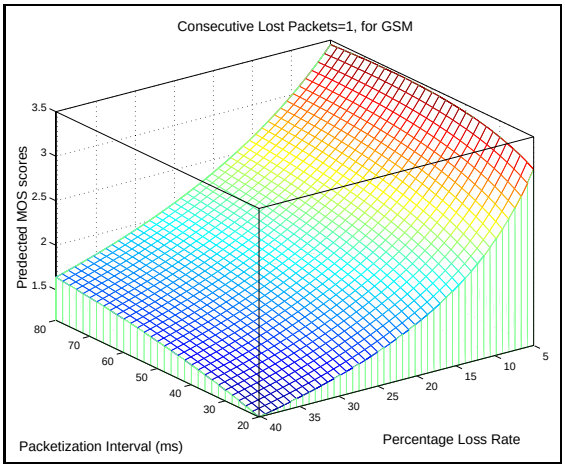
(c)



(d)

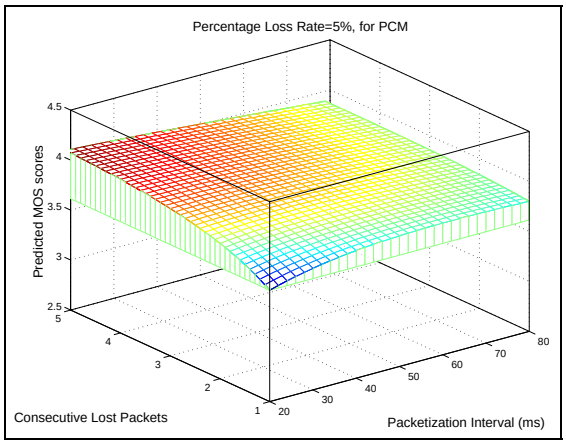


(e)

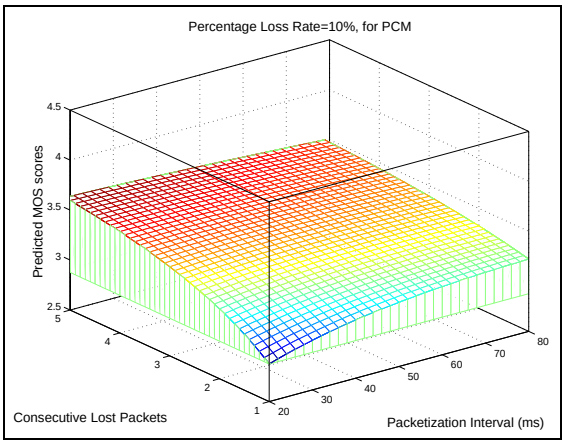


(f)

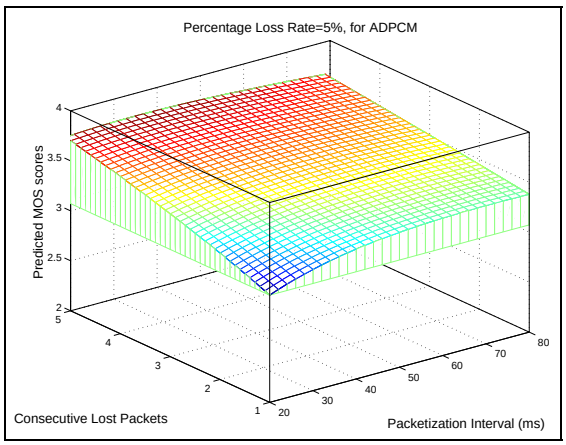
Figure 7.1: On the left, we show the impact of LR and CLP on speech quality for the different codecs and PI=20 ms. On the right we show the effect of LR and PI on speech quality for CLP=1.



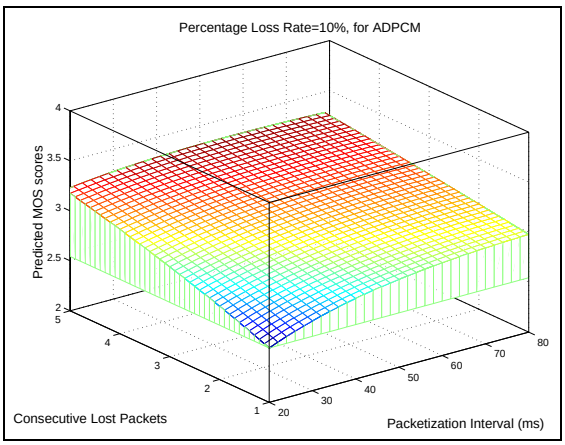
(a)



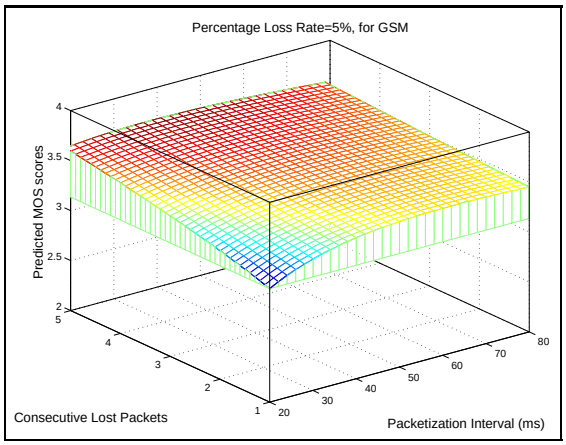
(b)



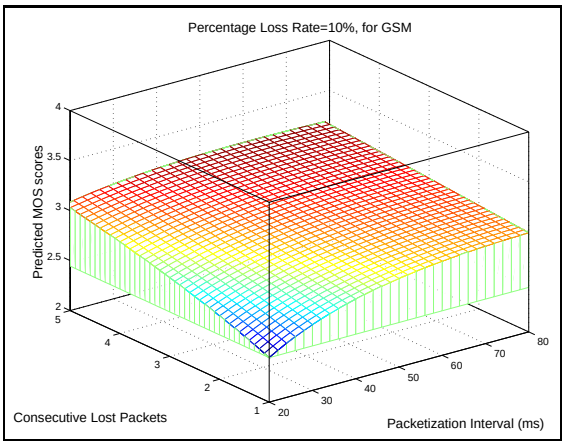
(c)



(d)



(e)



(f)

Figure 7.2: The impact of CLP and LR on speech quality when LR=5 % (left) and when LR=10 %(right) for PCM, ADPCM and GSM codecs.

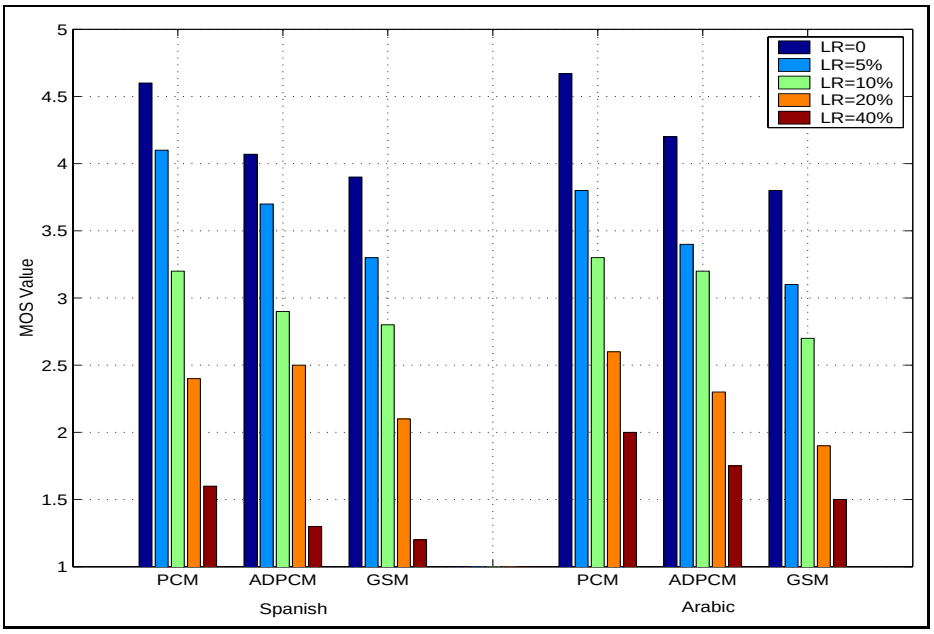


Figure 7.3: The variations of the quality as a function of the LR and the employed speech codec in both languages for PI=20 ms and CLP=2.

of the quality for FR greater than 16 is negligible, as already observed in previous works [119]. As the BR decreases, the effect of FR becomes smaller, see Figure 7.14.

The improvement of the quality when the LR changes from zero to 10 % (MOS value varies from 6.5 up to 7.5 for FR=30 fps) decreases until it becomes negligible as shown in Figure 7.8. From Figure 7.10, we can see that we can get constant relative improvements of the quality whatever the value of RA (from 6.0 to 7.0 for RA=0.05 and from 6.4 to 7.3 for RA=0.5 – this is for BR=0.6, CLP=2 and LR=0 %).

These results may seem surprising, but they have been observed in different previous studies. For example, experimental results showed that for FR larger than 16 fps, the viewer is not so sensitive to

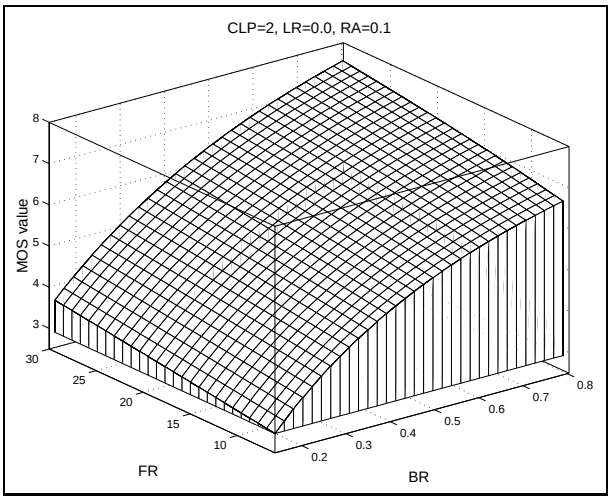


Figure 7.4: The impact of BR and FR on video quality.

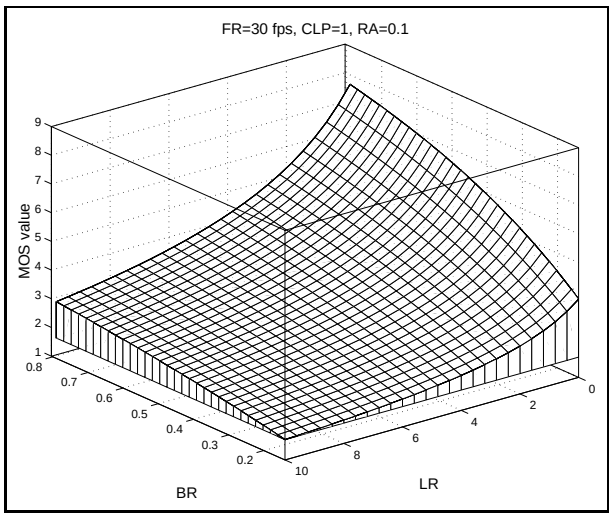


Figure 7.5: The impact of BR and LR on video quality.

changes in FR values [119]. The work done in [53] clearly shows that the enhancement of the quality for a wide range variation from 6 to 25 fps is really small. Both studies were based on subjective quality tests.

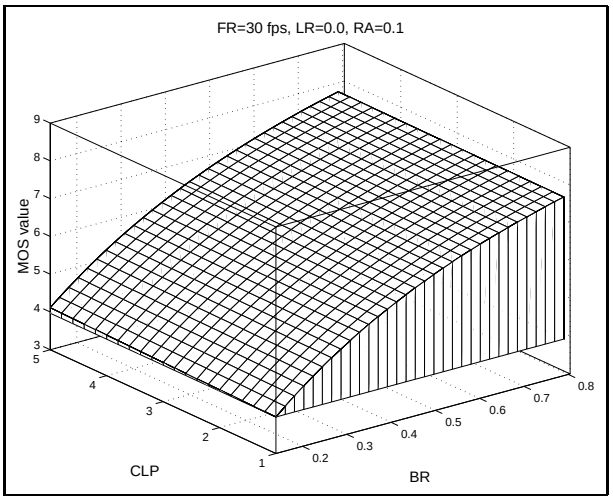


Figure 7.6: The impact of BR and CLP on video quality.

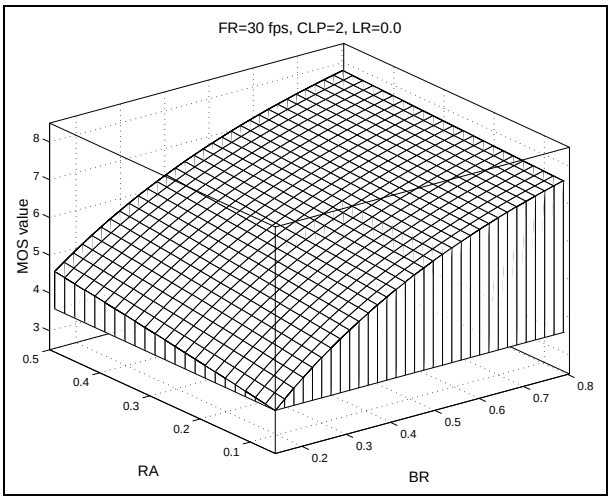


Figure 7.7: The impact of BR and RA on video quality.

7.4.3 Loss Rate (LR)

The effect of LR on video quality was the main goal of several previous studies as it is an important parameter [23, 53, 58]. As shown in Figure 7.5, the quality drastically decreases when the LR increases from 0 up to 10 %. But the absolute decrease of the quality depends on the other parameters' values: about 1.3 for BR=0.15, while 5.0 for BR=0.8. As depicted in Figure 7.8, the absolute deterioration is about 3.8 for 6 fps, but 4.6 for 30 fps. The variation of the quality is about 4.4 for RA=0.5 and 3.8 for RA=0.05, when BR=0.35, FR=30 fps and CLP=2, as shown in Figure 7.12.

Again, the fact that for LR set to zero there is no change on quality when CLP varies (Figs. 7.6, 7.9 and 7.11), shows that our RNN captures with great precision and reliability the characteristics of the quality as a function of the parameters. It may be thought that this is due to large values of BR and FR, but we varied all these parameters and we got the same behavior. In the case of using ANN, it is not easy to get this result (see Figure 4.9(b)).

7.4.4 The number of Consecutively Lost Packet (CLP)

As the result obtained for this parameter may seem strange, let us clarify its effects. When keeping all other parameters constant and increasing the value of CLP, the distance between any two consecutive loss occurrences increases. This has two consequences: loss occurrences decrease and consequently the deterioration of frames becomes smaller (i.e. smaller number of frames partially distorted by loss). As it is well known and also shown in our previous analysis of FR, the eye is less sensitive to higher values of FR. Moreover, for each lost packet the past macro-blocs (a portion of the image) are still shown on the screen as a kind of error resilience. Hence, larger values of CLP may introduce deterioration in smaller frames. This is equivalent to smaller LR values but slightly lower FR values. As it was previously shown that the effect of LR is much greater than that of FR, the effect of higher values of CLP is benefic to the quality. This result is in agreement with that obtained in [58], although the authors did not explain the reasons why this happens.

Starting by Figure 7.11, and for zero LR, there is no visible effect of CLP on the quality, but as the LR increases the effect of CLP increases up to 0.5 for LR=10.0 %. From Figure 7.13, we can

see that there is an improvement of 0.5 of the quality when CLP changes from 1 to 5 for BR=0.7, FR=30 fps and LR=1.0 %. In conclusion, CLP effect is important and comparable to FR's. When CLP increases, quality increases too, particularly in the case of poor conditions (i.e. lower values of BR or higher values of LR), while increasing FR improves quality especially for good conditions (high BR and low LR).

7.4.5 Intra-to-Inter Ratio (RA)

RA is the ratio of the encoded intra macro-blocs to those encoded as inter, hence it is expected that this parameter has benefic effect when it increases. This is clearly shown in Figs. 7.7, 7.10, 7.12 and 7.13. Its effect is more important than FR's, and more interestingly, for lower values of the BR parameter. This is shown in Figure 7.7, where for BR=0.15 the quality increased from 3.4 up to 4.6. This improvement is relatively important when the network bandwidth is small, but for larger BR, the effect is negligible.

The two parameters LR and RA are related, as shown in Figure 7.12. For smaller LR, there is an improvement on the quality for the increase in RA. But this improvement vanishes when the LR increases. This means that we can get better video quality when the available channel bandwidth (BR) is small and for smaller values of LR by increasing the value of RA. In such a case, the effect of RA on the quality is more benefic than that of FR, as shown in Figure 7.14.

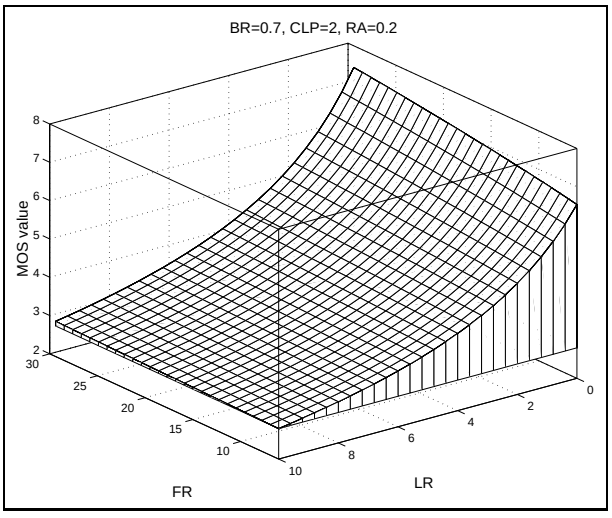


Figure 7.8: The impact of FR and LR on video quality.

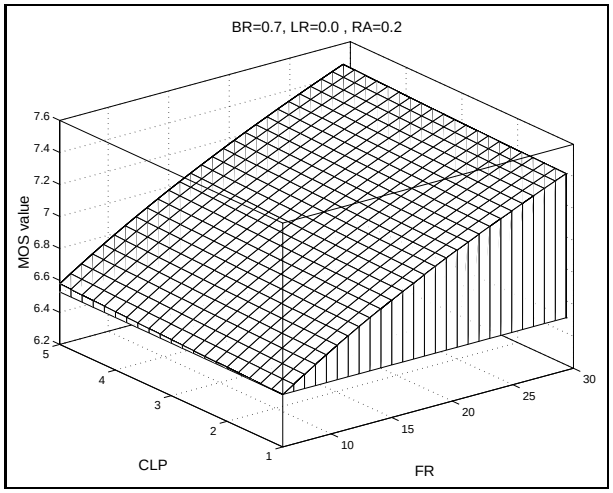


Figure 7.9: The impact of FR and CLP on video quality.

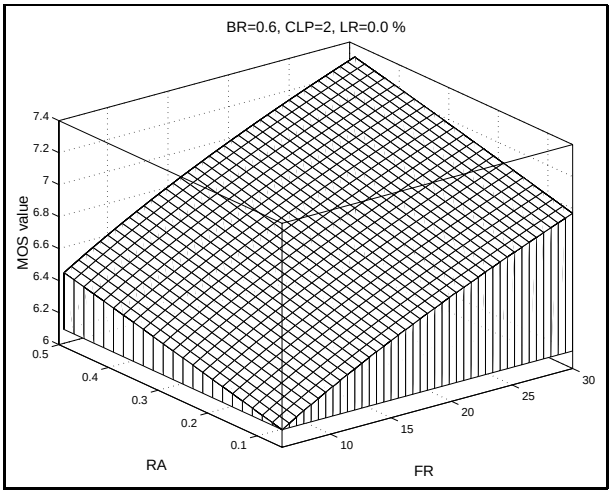


Figure 7.10: The impact of FR and RA on video quality.

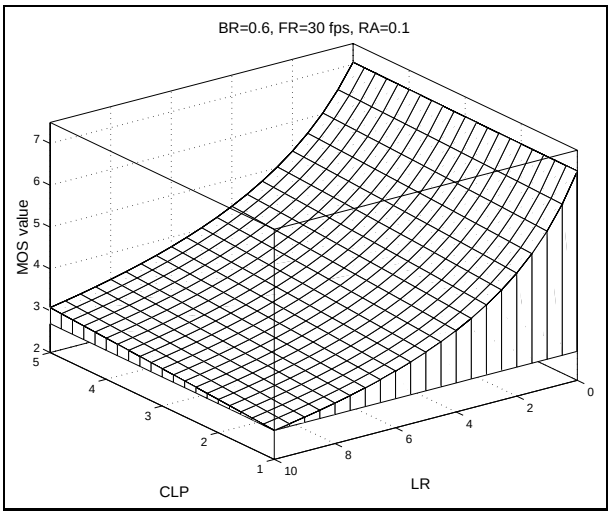


Figure 7.11: The impact of LR and CLP on video quality.

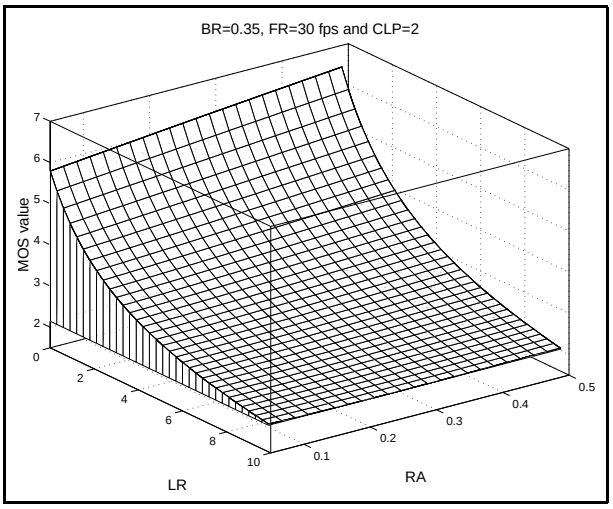


Figure 7.12: The impact of LR and RA on video quality.

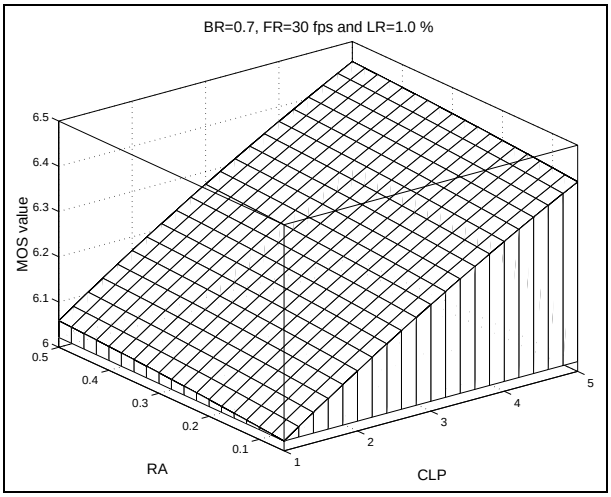


Figure 7.13: The impact of CLP and RA on video quality.

7.5 Conclusions

In this Chapter, we have presented a study of the impact on video quality of some of the most important parameters supposed to have an impact on it, namely, the stream bit rate, the scene frame rate, the network loss rate, the burst loss size and the ratio of the encoded intra to inter macro-blocs for H263 codecs. Similarly, we have presented in this Chapter a study of the impact of the most important parameters on speech quality, namely the codec used (PCM, GSM and ADPCM), the loss rate, the loss distribution and the packetization interval.

As far as we know, there is no previous study of the combined effect of several parameters on neither speech nor video quality. There are several studies in the literature, but they either concentrate on only one or two parameters, and/or they based they study on objective measurements like PSNR or SNR (which do not correlate well with subjective results). The goal of this analysis is to help in the understanding of the behavior of real-time media streams transmitted over best-effort networks. This may be used, for instance, in developing control mechanisms allowing the delivery of the best possible video quality given the current network state.

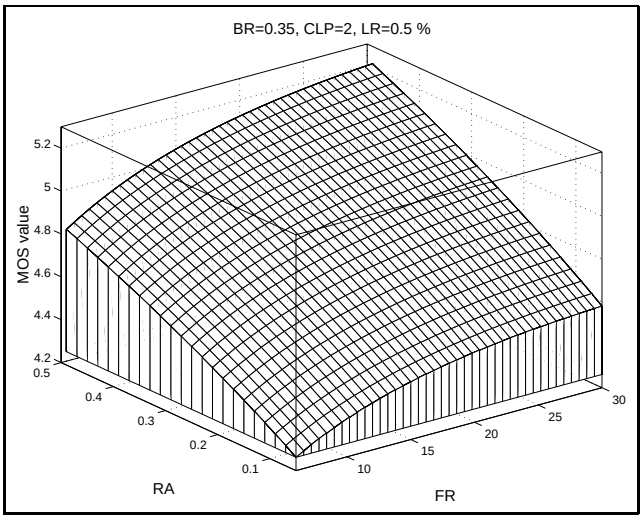


Figure 7.14: RA is more benefic than FR for lower values of BR.

These studies is based on our methodology, presented in Chapter 4 and the results of Chapters 5 and 6. We used the RNN for all our studies given in this Chapter because it behaves better than the standard ANN model.

Chapter 8

A New Rate Control Mechanism

8.1 Introduction

Recent studies on real-time multimedia applications [60, 81, 103, 152] involving the transmission over packet switched networks have emphasized the difficulties of having a best-effort Internet service model. Best effort introduces variable delays and loss distribution patterns that greatly decrease multimedia quality. This best-effort delivery policy will not change for a long time; thus, in order to get acceptable Quality of Service (QoS) levels, it has become extremely important to develop control mechanisms that eliminate, or at least minimize, the negative effects of some specific network parameters on the quality of multimedia signals as perceived by users at destination points [18]. In fact, the quality perceived by end-users may define the scope of applicability [81], or, furthermore, the final acceptance of real-time multimedia services [147].

Several approaches have been developed to address this issue, including:

a) forward error correction (FEC) techniques, which have been developed to minimize the effect of packet loss, by sending additional information to aid in packet recovery [18, 103, 116];

b) control mechanisms working at destination points introduced to minimize the effect of delay variations between successive packets [19, 37];

c) adapting scalable bit-rate codec and prioritized transmission algorithms, at the network layer, used to get a smooth degradation of quality during network congestion periods [11];

d) TCP-friendly rate control protocols to avoid congesting the network [22, 26, 41, 42, 44, 77, 107, 108].

One of the goals behind the use of TCP in the Internet is to avoid the collapse of the network. TCP has the property of being fair to other TCP-like flows. This is to avoid congesting the network and to provide the maximum possible utilization of the network. However, as mentioned in Chapter 3, TCP is not suitable for the transport of real-time multimedia flows. The most appropriate protocol for these flows is UDP, in conjunction with other protocols (RTCP, RTP, etc.). Using UDP, which is non-reliable in the sense that the sender cannot guarantee that packets will arrive to the source in the correct order or completely get lost, the source sends packets at the rate it wants without taking care if this will congest the network or not.

Allowing the sources to send at the rate they need can result in a severe network congestion. Thus, the idea of using rate control protocols instead of the open loop UDP, with the aim of regulating the sending rate of all the UDP sources, has the following goals:

- to avoid congesting the network due to limited resources (bandwidth and routers performance).
- to provide fairness to competing TCP flows. In response to losses in the network, TCP connections sharing the same congested links with UDP flows reduce their transmission rates. However, without any rate reduction on behalf of the non-congestion-controlled traffic, the competing TCP

connections would starve and receive a smaller bandwidth share than the UDP flows. Therefore, UDP sources need to be enhanced with adaptive control mechanisms that not only aim at reducing loss ratios and improve bandwidth utilization but also at behaving in a more fair way towards competing TCP connections (such a protocol is thus called TCP-*friendly*). TCP-friendliness indicates that, if a TCP and an adaptive UDP flows with similar transmission behavior and round trip delays face the same loss values both flows should receive similar bandwidth shares [108].

- lastly, rate control mechanisms are used to maximize the network's resources utilization by all the sources.

The majority of the existing rate control protocols aim to reach all these goals. Thus, they are generally referred as TCP-Friendly Rate Control (TFRC for short). These protocols try to use the same existing mechanisms of TCP to regulate the sending rate of UDP sources. It is clear form this point that the performance of these protocols may lead, in some situations, to the reasons for which TCP is not used to transport real-time multimedia flows (see Section 3.3 for details).

Thus, using TCP-Friendly-like mechanisms to adapt the sending rate is a way to behave as a TCP connection and hence to be less aggressive than UDP open loop flows. A TCP-Friendly sender is able to adapt its bandwidth consumption according to network conditions. This means that it will send less data than a normal open loop UDP application. It means also that adapting the flow to the current measured network conditions will be benefic to the global Internet but may reduce dramatically the perceived quality [129]. It is stated in [126] that *"All of those approaches assume that the sender can adjust its transmission rate in accordance with the values determined by the adaptation scheme and in accordance with the dynamics of the TCP-friendly algorithm with arbitrary granularity and with no boundaries as to the maximum or minimum bandwidth values. However, in reality there are certain maximum and minimum bandwidth requirements for multimedia contents above and below which the sent data is meaningless for the receiver. Additionally, the used compression style for a multimedia content might dictate that the changes in the transmission rate of a video stream for example can only be realized in a granular way. Also, there might be some restrictions arising from the user himself with regard to the acceptable variations in the perceived QoS over a time interval. Such restrictions depend primarily on the transferred content, the used compression style, the communication scenario and the user himself."* Therefore, once the sender is obliged to decrease its bandwidth, it has to make a decision on how to decrease it in order to maximize the quality as perceived by the end-users. As we will see, this is the goal of the control mechanism that we propose in this Chapter.

The diversity of the existing encoding algorithms (and hence the encoders) makes it possible to have one encoder that gives better quality than the others for the same conditions (network state, output bit rate, etc.) as we showed in the previous Chapter. In addition, for the same encoder, we can considerably reduce the network utilization (bit rate) by changing some of its parameters (for example, changing the quantization parameter or the frame rate in video encoder), of course at the cost of having some impact on the perceived quality. An example of such a variable, is the frame rate of video streams. You can reduce it from 30 to 15 frames/sec. without loosing a significant amount of the perceived quality. This is because experiments [119] (confirmed in our study) show that human eyes (HVS) are not too sensitive to the changes of frame rates greater than 16 frames/sec (see Chapter 7 for similar conclusions).

These two observations, after a thorough analysis and study of both speech and video streams distorted by several network and encoder parameters' values, led us to think about designing a new protocol to guarantee the delivery of the best possible quality of the stream while maintaining the TCP-friendliness of the stream sp as to avoid network collapse during congestion.

In this Chapter, we want to integrate user perception and network parameters for rate control instead of basing the rate control only on network passive measurements (loss, delay, etc.) as the traditional protocols. The neural network approach that we have previously described in Chapter 4 and validated in Chapters 5 and 6 can measure the quality of multimedia flows when affected by both

network and encoding impairments. As we have seen, the results obtained correlate well with those obtained by human subjects for wide range values of both network and encoding distortion.

An additional motivation of our work is to lay the basis for designing some kind of network protocols that take into account the end-user perception of the quality instead of basing these protocols on passive network measurements. So far, it has not been possible, because there was no mechanism that could measure the quality in real time without having the access to the original signal, nor gave results that correlate well with human perception, nor quantified the direct influence of each of the quality-affecting parameters while being computationally simple. However, our tool satisfies all these requirements to a certain extent.

This Chapter is organized as follows: in Section 8.2, we provide an overview of the existing rate control mechanisms found in the literature. We describe our proposed rate control scheme and we provide a list of the possible controlling parameters in Section 8.3. We validate our protocol and show the obtained results in Section 8.4. A general discussion of some points related to the proposed protocol is given in Section 8.5. Finally, Section 8.6 concludes this part of our work.

8.2 Related Works

There are two kinds of approaches for end-to-end TCP-friendly congestion control: window-based [26] and rate-based [41] approaches. The window-based approach is derived from the AIMD (additive increase multiplicative decrease) rule by adjusting the window size in response to the acknowledgments or to packet loss detection. The rate-based approach attempts to handle the transmission rate directly under the AIMD rule, or sometimes based on analytical equations [41]. The basic method used in AIMD schemes is based on increasing the transmission rate of a sender by an additive value during underload situations. During overload situations the rate is reduced multiplicatively. The adaptation decision is taken from the observations, based on the network state, made during a specific time interval [124].

There are many mechanisms to implement the increase and decrease phases of both kinds of adaptation approaches. The authors of [124, 125, 126] investigate various mechanisms for realizing an additive increase and multiplicative decrease adaptation scheme based on RTP. They also identified the optimal approaches for increasing and decreasing the transmission rate. Other work in this area can be found in [22, 26, 41, 42, 44, 77, 107, 108].

In [77], an end-to-end congestion control mechanism for Internet video is presented. The end system adjusts the sending rate depending on the perceived network status, to satisfy the bandwidth and packet loss requirements while exhibiting a TCP-friendly behavior. Based on that mechanism, a TCP-friendly Internet video streaming employing variable frame-rate encoding is presented in [127, 78, 76]. It consists of TFRC and on available bandwidth estimator. The video is compressed according to the available bit rate and the end-to-end estimated delay. The goal is to enhance the network adaptability to mitigate the packet loss and bandwidth fluctuation.

Another congestion control mechanism that uses the frame rate as a control parameter is presented in [42]. The congestion is detected by measuring the buffer occupancy and loss rate. The available channel bandwidth is estimated by measuring the incoming data rate at the receiver. The frame rate is adjusted accordingly when congestion is detected. In [132], a real-time Internet video congestion control mechanism using error resilient scalable compression and TFRC is presented. Compressed video is packetized into individually decodable packets of equal expected *visual importance*. The packets can be truncated to instantaneously meet the time varying bandwidth imposed by a TFRC. The idea of changing the frame rate, such as the work given in [77, 127, 78, 76], is much better than this one. This is because dropping packets will degrade the quality too much, but reducing the sending rate by dropping complete frames will reduce considerably the sending rate, do not lose packets and the degradation on the quality will not be as harm as dropping packets randomly from several frames (see Section 7.4 for more details).

A TFRC protocol for real-time video applications is presented in [144]. In addition to the classical rate estimation (based on delay and loss), the authors use the perceived video quality (based on the very simple SNR measure) as a way to adjust the transmission rate, and to provide fairness with a competing TCP flow. However, as we have shown in Section 6.6, PSNR gives very poor correlation with subjective quality measurements. Knowing that PSNR is better than SNR for video quality evaluation (in other words, SNR is not a reliable metric to measure the perceived video quality), it is expected that the performance will be consequently poor. In addition, the SNR is measured at the sender side, thus it takes only into account the encoding impairments and not the network impairments. Another comment about this work is that the protocol used to evaluate the network state is RTCP (see next for explanations about its limitation in rate control).

A proposal of TFRC called Time-lined TCP which is targeted at streaming media (applications that tolerate both loss and delay) is presented in [97]. The protocol consists of associating a deadline with the data, trying to re-send non-acknowledged packets the same way as TCP, but the process is given up when the deadline is expired. The packet is considered lost only if their deadline has expired before acknowledgment. The idea seems interesting, but it will not work well for real-time applications that cannot tolerate too much delay or loss such as audio or video conferencing.

An adaptation scheme called rate adaptation protocol (RAP) is presented in [108]. Just as with TCP, sent packets are acknowledged by the receivers with losses indicated either by gaps in the sequence numbers of the acknowledged packets or timeouts. Using the acknowledged packets, the sender can estimate the round trip delay. If no losses were detected, the sender can periodically increase its transmission rate additively as a function of the estimated round trip delay. After detecting a loss the rate is reduced by half in a similar manner to TCP. However, this approach does not consider the cases of severe losses that might lead to long recovery periods for TCP connections. Hence, the fairness of such an approach is not always guaranteed. In addition, similar to other many TFRC protocols, reducing the sending rate by half in response to a single packet drop may produce a severe degradation of the perceived multimedia quality [126].

Based on that protocol, the authors present in [106, 107, 109, 110] a mechanism for using layered video in the context of unicast congestion control. This quality adaptation mechanism adds and drops layers of the video stream to perform long-term coarse-gained adaptation, while using a TFRC to react to congestion in short timescales. The mismatches between the two timescales are absorbed using buffers at the receiver.

Another TFRC proposal is given in [124, 125]. It is based on loss-delay adaptation for regulating the transmission behavior of multimedia senders. The goal is to improve the network utilization, avoid congestion, and provide fairness toward competing TCP connections. It uses RTCP to estimate loss and delay. The disadvantage of this proposal is due to the use of RTCP to estimate the loss and delay. This will not react to the rapid changes in the network (see next for the limitation of RTCP).

8.2.1 Limitation of RTCP in Rate Control

In the majority of the existing rate control schemes, RTP is used to implement the required feedback in order to establish a closed loop between the sender and the receiver. In addition, using RTP as the transport protocol, the associated control protocol RTCP can be used to give feedback the loss and delay statistics from the receiver to the sender in order to establish a decision basis for the adaptation algorithm. The shortest time between two successive RTCP messages is about 5 seconds. This limits the use of RTCP in rate control adaptation (namely the short-term reaction). In order to react quickly, adaptation schemes proposed in the literature use feedback information arriving on short time scale of only a few milliseconds or even every sent packet. Hence, the sender can react to rapid changes in network. The frequency of RTCP feedback messages indicates that an RTP sender can not react fast enough to rapid changes in the network congestion. Therefore, the goal of RTCP-based adaptation is to adjust the sender's transmission rate to the average available bandwidth and not to react to rapid

changes in network conditions [124]. Also, as the RTCP messages usually acknowledge the reception of a large number of packets, the sender would be allowed to insert a large burst of packets into the network, which might lead to losses or increase them [124].

8.2.2 Equation-based TFRC

The motivation behind our selection of the Equation-based TFRC (EB-TFRC) [41] among the other AIMD rate control protocols is that EB-TFRC provides smooth rate variation and not too many oscillations. This is because EB-TFRC does not reduce the sending rate by half in reaction to a single packet drop like other protocols and still provides fairness to other competing TCP connections. The primary goal of EB-TFRC is not to aggressively find and use the available bandwidth, but to maintain a relatively steady sending rate while still being responsive to congestion. The main features of EB-TFRC protocol as provided in [41] are:

- Do not aggressively seek out available bandwidth. That is, increase the sending rate slowly in response to a decrease in the loss event rate.
- Do not reduce the sending rate by half in response to a single loss event. However, do reduce the sending rate by half in response to several successive loss events.
- The receiver should report feedback to the sender at least once per round-trip time if it has received any packets in that interval.

Therefore, EB-TFRC best suits multimedia applications because of the overall end-user quality perception. It has been found that users prefer to see video with constant frame rate rather than changing it rapidly depending on the available bit rate [108, 124, 126].

Like many other AIMD TFRC protocols, EB-TFRC uses a control equation that explicitly gives the maximum acceptable sending rate as a function of the recent loss event rates (that means there is an upper bound the sender should not exceed). The sender adapts its sending rate, guided by this control equation, in response to feedback from the receiver. For traffic that competes in the best-effort Internet with TCP, the appropriate control equation for equation-based congestion control is the TCP response function characterizing the steady-state sending rate of TCP as a function of the round-trip time and steady-state loss event rate, which is given by [41]:

$$T = \frac{s}{R\sqrt{\frac{2p}{3}} + t_{RTO}(3\sqrt{\frac{2p}{8}})p(1+32p^2)}. \tag{8.1}$$

Here T is the upper bound on the sending rate in bytes/sec, s is the packet size, R is the round-trip time, p is the steady-state loss event rate, and t_{RTO} is the TCP retransmit timeout value in the congestion state. For more details about EB-TFRC, see [41] and the software implementation available in [5].

8.3 Our Proposed Rate Control Protocol

It is likely to see a progressive adaptation of most of the multimedia RTP/UDP/IP based protocols to one or some of the proposed TCP-Friendly protocols. Although this is required to avoid the congestion and collapse of the global Internet, it can drastically reduce user-perceived quality, as explained before. In [41], a TFRC mechanism is proposed to control the rate of real-time applications such as audio and video conferencing. See Section 8.2.2 for more details.

We propose to use the neural networks that we presented in the previous Chapters in order to decide on the best adaptation that a sender could use when it receives congestion control information from the network. We show that the usage of the neural networks helps in improving MOS results

under the same network conditions. As shown in Chapter 7, the same MOS scores can be obtained by several combinations of input parameters. For example, for a given bandwidth, suggested by TFRC, one can choose the codec that gives the best quality. On the other hand, for a given MOS score one can choose the best-input parameters to reduce the required bandwidth. By changing the packetization interval (and hence changing the packet rate sent to the network), one may improve the loss ratio. Reducing the sending rate by choosing a different codec, adding or removing FEC, changing the packetization interval, are decisions that can change the MOS in the receiver under the same congestion circumstances and may reduce the bandwidth required by the application to deliver the real-time speech signals from the sender to the receiver (see bellow).

The new control protocol (shaded parts in Figure 8.1) is composed of three parts:

1. The first part is a TFRC that periodically calculates the suggested sending rate. The periodicity of the calculation can depend on RTCP recommendations (5 seconds and not more than 5% of the data traffic) if long-term adaptation is needed or can behave like the TFRC implementation (every Round Trip Time) if shorter-term adaptation is needed. See Sections 8.2.1 and 8.5 for further information.

2. The second part is the trained neural network at the receiver side, to measure in real time the MOS based on the network conditions and the encoding parameters. It sends feedback information to the sender, which contains the network statistics as well as the MOS result evaluated at the receiver. See the description of the approach in Chapter 4 and the validation of the method for speech and video in Chapters 5 and 6.

3. The third part takes these results and based on internal rules, decides on the new parameters to be used. In this module, a set of controlling parameters should be defined (see bellow for a list of the possible parameters). In addition, the impact of these parameters on the quality should be known. (See Chapter 7 for a guideline.) The goal of this module is to select the best possible values of the controlling parameters to maximize the quality as if it were perceived by the end user (represented by the NN module) based on the network state as measured by the first part. A compromise between parameters stability and bandwidth utilization should be established. A too slow change frequency of the parameters would result in bad quality during congestion and unfairness toward the competing TCP connections. Similarly, too high change frequency, if possible, could result in too many fluctuations in the perceived quality and this is not suitable for the end users, as mentioned before.

8.3.1 The Possible Controlling Parameters

The following parameters can participate in the rate control protocol (refer to Chapter 7 for more details):

The bit rate (BR). The sender can change dynamically the BR, when a variable BR (VBR) encoder is used, using multilayer encoding, and/or using another encoding algorithm. Multilayer encoding as a rate control mechanism is one of the most successful mechanisms in the literature [106, 107, 109, 110].

Changing the codec. Eventually, the codec type or some of the encoder's specific parameters (sampling rate, sample size, quantization parameters, etc.) can be changed to fit the needed bit rate. As described in Sections 3.4 and 3.5, there are lossless, lossy and hybrid encoding techniques. For the same bit rate and the network state, you can select the codec (or change the parameters of the current one) to get the best possible perceived quality.

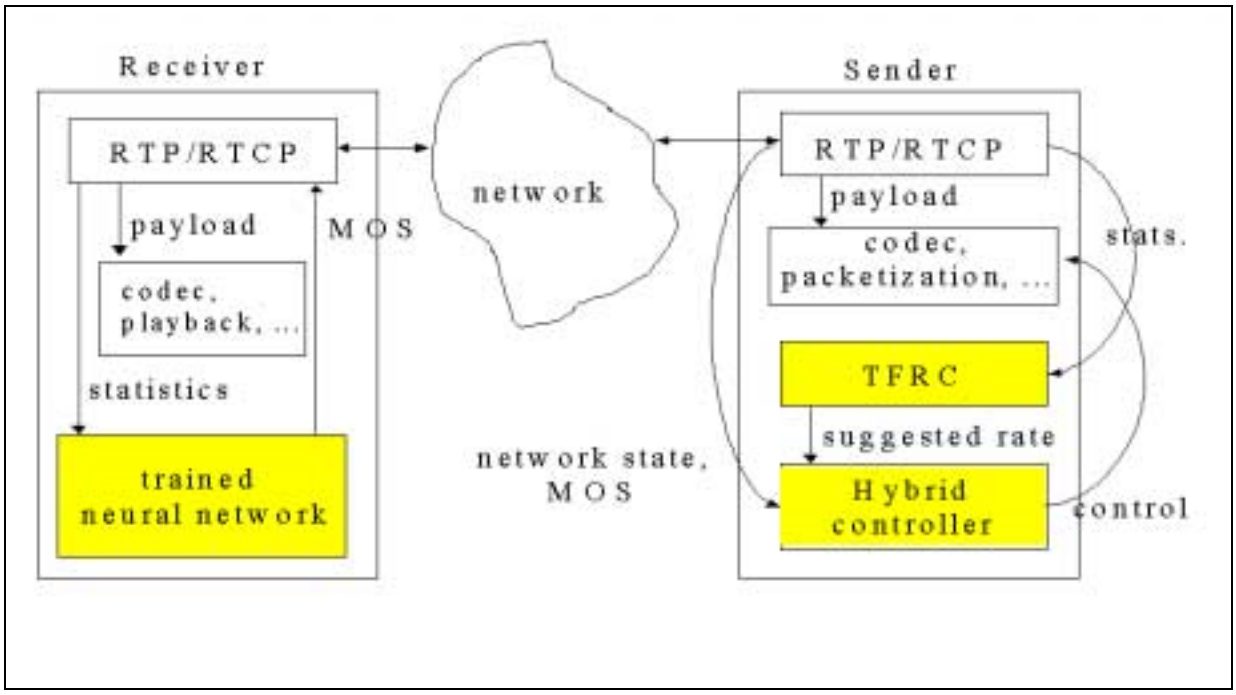


Figure 8.1: Architecture of the proposed control mechanism.

The frame rate (FR). In the case of video as experiments and study on HVS showed, the human eye is not very sensitive to variation of FR higher that 16 frames/s. Thus, by skipping some frames, the output BR can fit the value suggested by TFRC and not degrade the quality too much. This case is extensively studied in the literature as it is a very effective way of the rate control [42, 127, 78, 76].

Forward Error Correction (FEC). FEC can be used to protect against loss. Thus, by choosing the amount and type of redundant data [20, 18, 19, 21, 22, 116], the effect of loss can be reduced and hence the quality can be improved. But too much FEC increases the overall BR and delay. Thus, a compromise should be made.

Buffer size. Changing the buffer size in the case of real-time streaming multimedia flows, the client fetches some data from the source before starting playing the media [116]. After that, the client fetches the data whenever the size of the data in the buffer reaches some predefined value or when there is loss. By changing the size of this threshold dynamically as a function of the network state and the perceived quality, it is possible to keep some data always available to the decoder. By using our control mechanism, we believe that the quality can be enhanced working on this parameter.

Packetization interval. In the case of speech, changing the packetization interval (and hence the packet size), the possible sending rate as suggested by the EB-TFRC will change (as it is directly proportional to the packet size from Eqn. 8.1). But, on the other hand, a too large packet size will increase the overall delay, which is not tolerable in the majority of real-time multimedia applications. In addition, it will cause the problem of fragmentation (packets greater than predefined length are divided into several packets to be fit in this length) between routers. Fragmentation increases the probability of loss because if one fragment of the packet is lost, all the fragments of that packet become useless and are hence deliberately dropped. On the other hand, larger packet size may improve the network utilization (see Section 5.2.1).

8.4 Simulation Results

In this Section, we are going to validate our approach for both speech and video. As we proceed differently in each case, we split this Section into two subsections.

8.4.1 Speech Case

In order to test this control mechanism, we used an RTP/RTCP stack that we upgraded with the EB-TFRC implementation [5]. The network topology prepared for the tests is composed of a sender implementing TCP-Friendly, a router, and receiver.

We used only one parameter (the codec type) to illustrate the interest of our control mechanism. Adding other parameters is of course possible; we think that the relevance will generally depend on the nature of the application. A local development (*"bypass"* software developed at INRIA Sophia Antipolis) in the router simulates a bottleneck that happens after a minute of normal operation. In addition to the bottleneck, a delay is applied to packets going through the router.

In Figure 8.2, we plot the behavior of the calculated bandwidth as suggested by TFRC and the saved bandwidth when the receiver decides to change from PCM (64Kbps) to GSM (13.2Kbps) codec based on a decision taken by the sender to maintain a good quality. We can see that when the bottleneck is detected, the sender adapts its rate to even less than the available bandwidth.

In Figure 8.3, we compare the two measured MOS values at the receiver side for the case when the sender follows our control mechanism by changing the codec and the case when the sender blindly follows what TFRC suggests and hence continues using the PCM codec in the bottleneck periods. Clearly, the MOS is improved even in presence of congestion. The decision to change the codec, that is usually taken manually by an expert user (he/she has to understand all "advanced features" of the conferencing software) when he/she starts to suffer from the network conditions, is here taken automatically by our controller. Moreover, our control mechanism may also be helpful in the absence of congestion by deciding the best combination of the parameters by which the end-user will receive the best quality.

Finally, by using this control mechanism, and to maintain a certain level of quality, the sender can decide to use the exact bandwidth required by the application in the absence of congestion and not to give only an upper bound. This is shown in the first part in Figure 8.2.

8.4.2 Video Case

In the case of video transmission, to show the effect of changing some of the controlling parameters, we proceed as follows. Based on the experimental and simulation studies of EB-TFRC given in [41], and based on the governing equation 8.1, we suppose for simplicity that for different network states, the sending rate as suggested by EB-TFRC is as depicted by the dot-dashed line in Figure 8.4. The fluctuations in the rates are due to the effect of the different variables in that equation. This is to take into account the variations of the end-to-end delay, the packet size, the loss event, the number of competing connections, etc.

As we previously mentioned, the sender should not (in some cases cannot) follow the fluctuations of TFRC, to avoid degrading the quality. In addition, for the purpose of clarifying the idea, we also suppose that the sender changes the sending rate according to the solid line depicted in the same Figure. We have selected two controlling parameters in order to meet the suggested rates, namely the quantization parameter (QP) and the frame rate (FR).

By changing the QP, the encoder can compress the original signal to the level we want, and hence produce the desired bit rate (BR). (See Sections 3.4 and 6.2 for more details). By skipping some frames from the original video signal, and hence changing the FR, also the BR can be changed to meet the required sending rate.

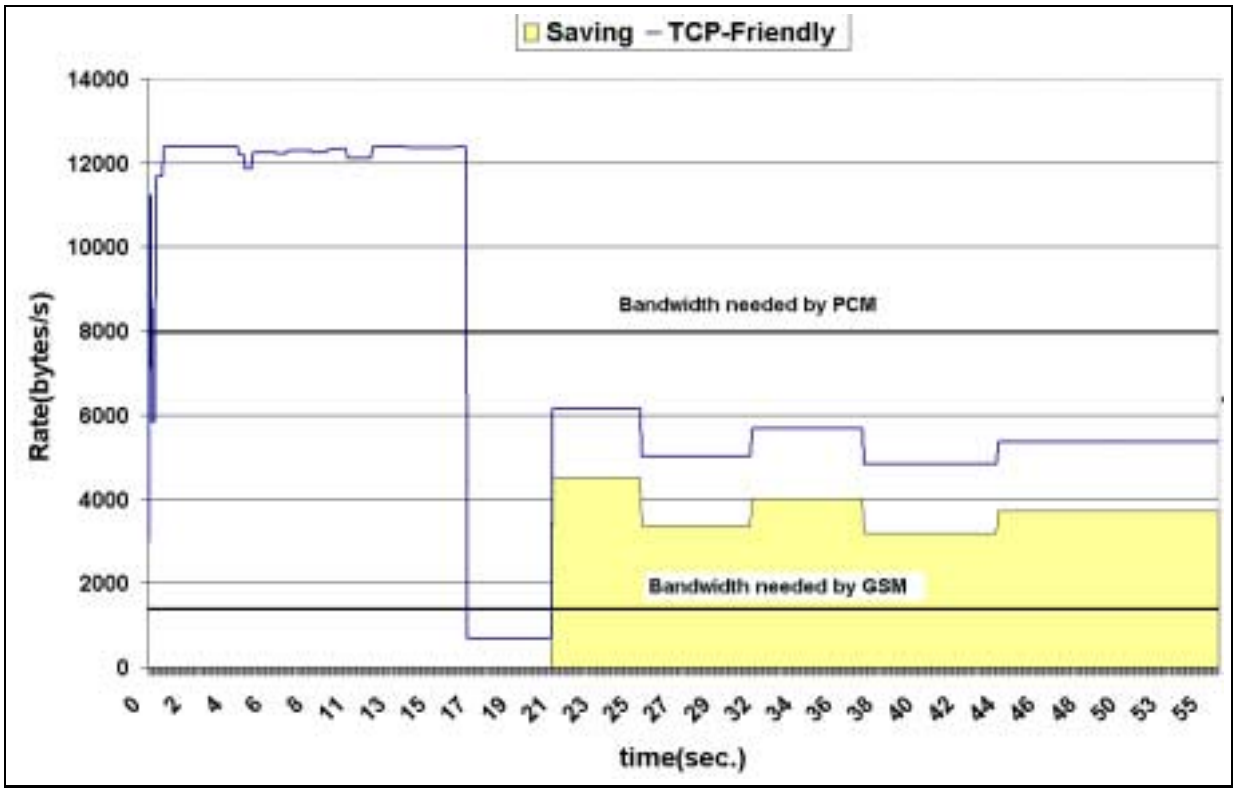


Figure 8.2: Rates suggested by TCP-friendly and the saving using control rules when changing the codec in the case of Speech

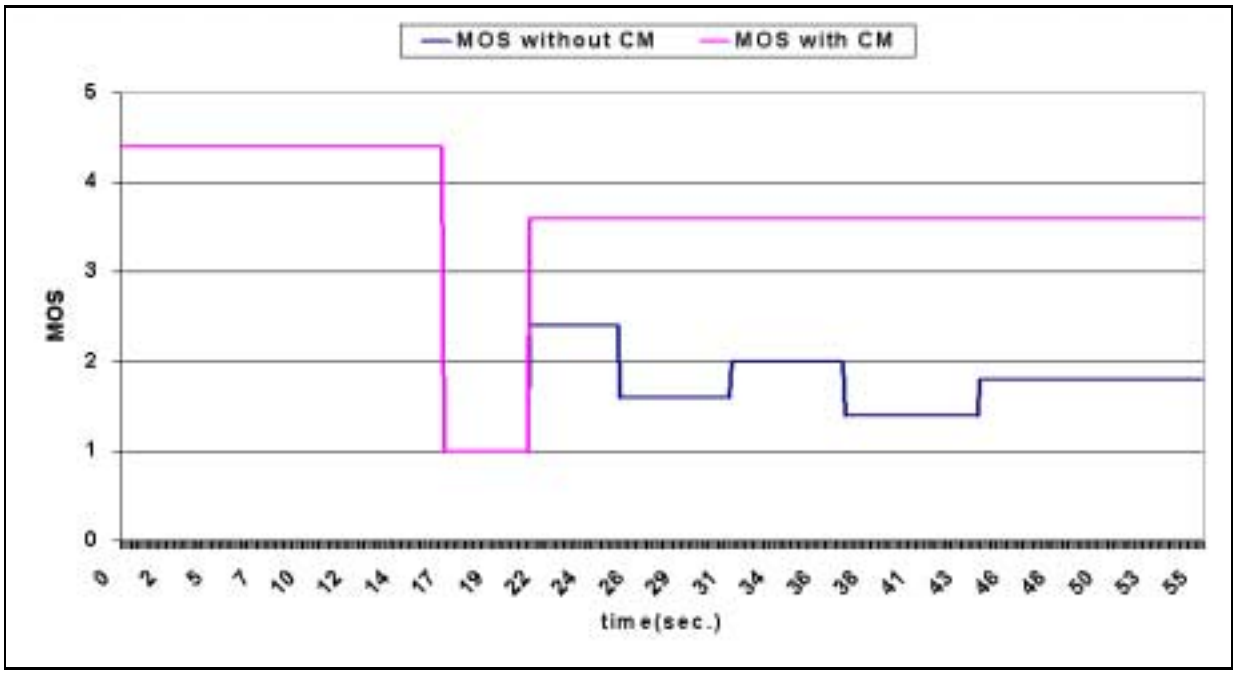


Figure 8.3: MOS values with and without our control when changing the codec in the case of Speech (CM stands for Control Mechanism)

For the case of using only the QP as controlling variable, we fixed FR to 30 frames/s and RA to 0.05. In the congested periods, we suppose that there is 1% loss with CLP of 1. Similarly, for the case of FR, we fixed all the parameters as before except BR and FR. The QP in this case is the one that gives the best quality (the maximum BR). We used the trained RNN (see Chapters 6 and 7) to measure the quality for each case. We show in Figure 8.5 the quality evaluation for both cases. As we can see, the case of changing the FR gives better quality than that when allowing only the QP to change. The improvement of the quality is more important in the case when the sending rate has to be very small. The explanation of this fact has already been mentioned before. Briefly, by fixing the FR to 30 frames/s and changing only the QP to reduce the required BR, the impairments of the encoded signal increase due to the encoding artifacts. On the other hand, by fixing the QP to its best value, and reducing only the FR, the impairment level is not as annoying from the user's perception as the other case. This is because human eyes are less sensitive to the change in FR rather than luminance and chrominance distortions (see Section 7.4.2).

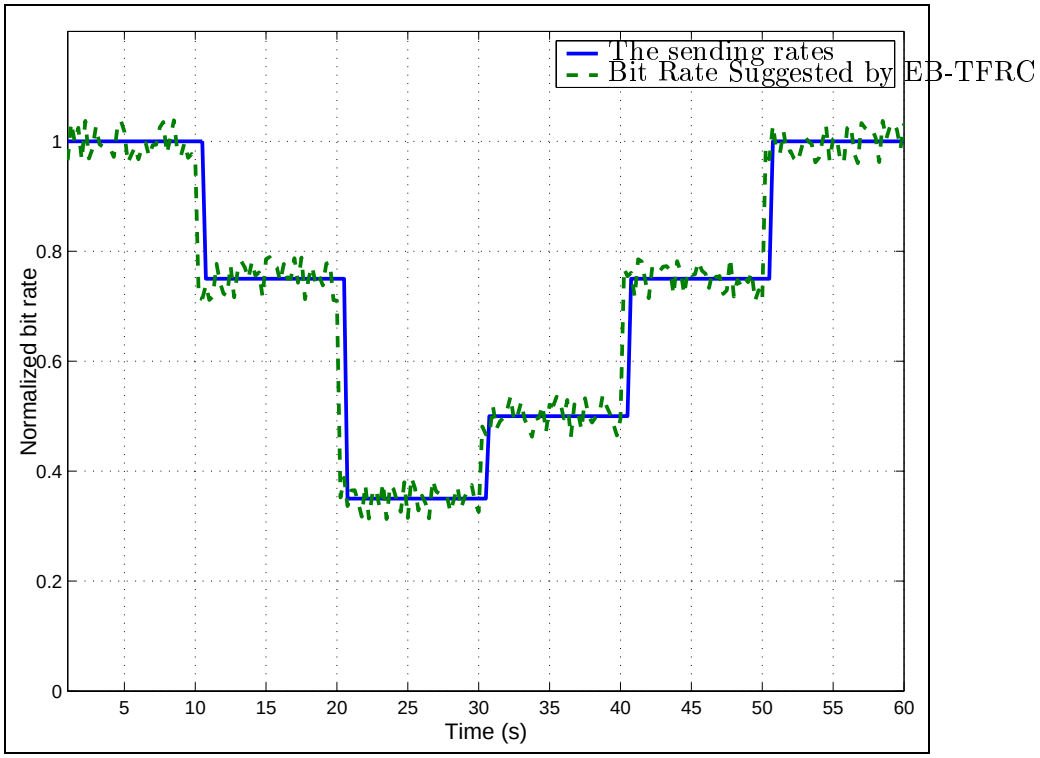


Figure 8.4: The supposed rates suggested by TCP-friendly and that of the sender

8.5 Discussion

As we can see from the results, some times the needed BR is less than that suggested by TFRC. In this case, where the encoder gives less BR than TCPF suggestion, the additional unused BR can be used to add some enhancement layers, in the case of multilayer encoding. Another option is to prefetch a part of the stream data, if possible for the streaming applications, so that it can be used when there are loss or congestion. Another possibility is to use FEC to increase the quality by protecting against loss.

As shown in Figure 8.1, we place the NN module in the receiver. This choice is to avoid the need to use other protocols to give feedback on the network statistics (e.g. loss, etc.) to the sender. Obviously, in two-way sessions, it must be implemented in both sides. In other words, each of the sender and the

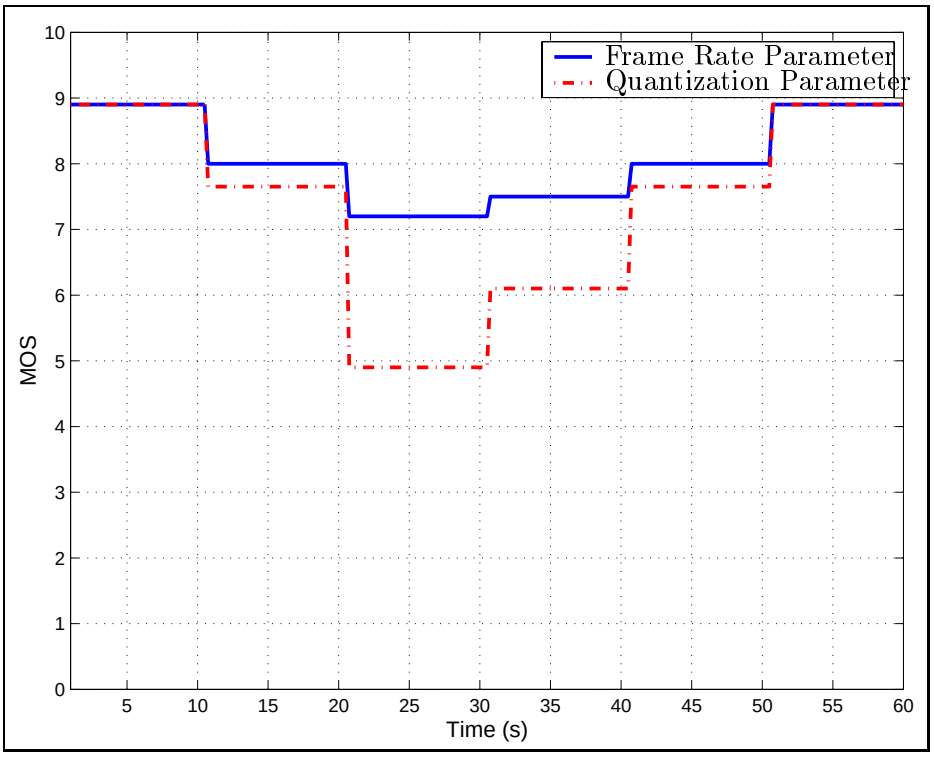


Figure 8.5: MOS values when changing frame rate and those when changing the quantization parameter to meet the bit rates shown in Figure 8.4

receiver must has its own module of all the previously mentioned ones.

All the feedback information (from the sender to the receiver or vice-versa) should be sent using the TFRC. This is because, unlike RTCP which sends the information back every 5 sec., TFRC sends the acknowledgment one time every RTT estimated period of time. In this case, the sender can change its sending rate rapidly based on this information. However, as suggested in the literature [108, 124, 126], changing the sending rate too fast can lead to an unstability problem. In addition, changing the sending rate with high frequency will influence the human perception considerably, as it degrades the overall quality. It should be also mentioned that one of the drawbacks of using the equation-based TFRC is the large amount of feedback information needed, but this is a common problem of all the rate control protocols aimed to react rapidly to the changes in the network state.

8.6 Conclusions

In this Chapter, we have presented a new rate control mechanism that combines an automated real-time subjective multimedia quality assessment with a TCP-Friendly rate controller. It helps in delivering the best multimedia quality and in saving any superfluous bandwidth for a given network situation (by determining the exact sending rate to be used, instead of just giving an upper bound). The controller decides, based on the subjective quality (measured by the trained neural network according to the approach described in Chapter 4) and on the network conditions (TCP-Friendly rate controller suggestions), which parameters should be modified to achieve this task.

We used, with our rate controller, the equation-based TCP-Friendly congestion control protocol that has some advantages for real-time multimedia applications. We provided a list of the possible controlling parameters to be used with our proposal. We showed the efficiency of the proposed approach in the cases of speech and video transmission. The perceived quality improved considerably in our

experience when using the new controller. By using different parameters, we believe that the quality can be improved. Studying the effects of the other parameters and building a complete controller are some of the future research directions.

Part III

On the Neural Networks Tools

Chapter 9

Using Neural Networks for Traffic Prediction: a New Method

9.1 Introduction

Many perspectives appear when observing the good performance of NN (optimization issues, tuning of parameters, ...). We are interested here in predicting the behavior of network and the applications running over it. One of the hot research topics in traffic engineering is traffic prediction. Traffic prediction is an important subject because there are many applications in networking that can be efficiently implemented based on a good traffic predictor. Traffic prediction is crucial for successful congestion control, congestion avoidance, dynamic bandwidth allocation, dynamic contract negotiation, traffic shaping and engineering, etc.

However, there are some difficulties that prevent providing so far a good traffic predictor that can work in real situations. This is because networks traffic is, in general, a complex, time-variant, nonlinear, and non-stationary process (see for instance [27] for the case of high speed networks). Furthermore, this process is governed by parameters that are difficult to measure [56]. Sometimes, there are completely non-predictable portions of this time-variant process (The so-called "spiky" samples). Therefore, a precise model of this process becomes difficult as its complexity increases.

Despite these problems, traffic prediction is possible because, as the measurements and previous studies have shown, there coexist both long-range and short-range dependencies in network traffics [54, 85, 27]. For example, the amount of traffic differs from the week-end to that in the week-days. However, it is statistically similar for all the week-ends. Also during the same day, in the morning, and at night, at some specific time of the day, etc.

The traffic prediction problem has been tackled by several techniques; least mean square filter or similar methods [85], regression and statistical models (AR, ARIMA, FARIMA, etc.) [27, 121, 85], and fuzzy logic with regression [120, 27]. Recently, Neural Networks (NN) have been found to be a powerful method to efficiently describe a real, complex, and unknown process, with non-linear and time-varying properties. There are several proposals using the NN to model the traffic process [156, 157, 39, 88, 133]. The first one seems to be [133], in 1993. However, the existing proposals concentrate only on the short-range dependencies and neglect the long-range ones. Thus, when testing these models on real traces lasting for several minutes or hours, they give good performance. However, when employing them for predicting the traffic for days or weeks, their performance degrades significantly [56].

Our aim in this study is to present and evaluate a new model for traffic prediction that can work well for IP and ATM networks. However, we have tested it only on IP real traces. The advantage of our model is that it makes use of the long-range information as well as the traditional short-range information in the past history, to predict efficiently the future arrivals of packets or cells.

We used the NN as a tool to predict the traffic after training them with the information from

the past history. It is not a problem to use either ANN or RNN, because both give similar results. However, RNN as stated in Section 4.5 is faster with respect to ANN, which makes it a good candidate when a light-weight application is targeted (to be used for example in a critical router).

For our model to be efficient and light, the NN can be trained either on-line, or periodically off-line. This is to take into account the system variation with time. (For example, in the case of the Internet, it is known that the traffic increases continuously.) Thus, we can track this variation without rebuilding a new predictor.

We validated our model by using real data and, at the same time, explored its efficiency in a real situation. It should be noted that many of the existing models were validated using data produced by simulation [88], or with artificially generated data (by simple equations) [157, 156], or considering "easy-to-predict" traffic (MPEG traces that have some kind of periodicity due to the frame types) [39, 133, 156]. These models work quite well for their associated data, but fail once they are applied to real ones. This limits their applicability to real applications [56]. On the other hand, other studies that have been validated by real traces concern short time periods (i.e. two hours). When these models are used to predict longer time periods or when the size of the time step[1] becomes larger, the performance degrades [27, 120].

The outline of this Chapter is as follows. In Section 9.2, we describe our model. Then in Section 9.3, we evaluate it by means of several experiments. Some possible applications that can benefit from our technique are discussed in Section 9.5. A comparison between our model and some existing ones is the subject of Section 9.4. Finally the conclusions are given in Section 9.6.

9.2 Our Method

The structure of our predictor is illustrated is as shown in Figure 9.1. To build such a predictor, three steps need to be carried out. The first is to form a database collected from real traffic traces for a sufficiently long period and to format this database in a specific manner (as described bellow). The second is to identify a suitable NN architecture and training algorithm as described in Sec. 9.3.3. Generally, the three-layer feedforward network and any efficient training algorithm like the Levenberg-Marquardt or the Conjugate Gradient algorithms may be a good choice. The third step is to train and test this NN and to select a retraining strategy (either periodically off line whenever the performance degrades bellow certain predefined threshold, or on line), see Sec. 9.3.6 for experimental results regarding this point. Concerning the real traces, they should be collected from the network whose traffic we want to predict. The accuracy of the method will greatly depend on the network traffic used in this learning phase.

Each of the training and testing databases should consist of 5 parts. The first is the "now window" which consists of $F(T) \dots F(T-n)$, where $F(T)$ is the current value of the traffic and $F(T-n)$ represents the value of the n^{th} previous step. The second part is the "yesterday window", which contains $F_y(T) \dots F_y(T-y)$, where $F_y(T)$ is the value of the traffic yesterday at the same time as now, and $F_y(T-y)$ is that of y^{th} previous step. Similarly, the third part is the "week window", which contains $F_w(T) \dots F_y(T-w)$, where $F_w(T)$ is the value of the traffic in the previous week, in the same day and at the same time as now. $F_w(T-w)$ is that of w^{th} previous step. Then come the "day" and "date", where the "day" represents the day of the week from 0 to 6, and the "time" represents the time of the day. The "time" variable takes discrete values depending on the "step size". For example, if the "step size" is one minute, the "time" takes values that vary from 0 to 60×24. The final part is the next value of the traffic, $F(T+1)$, that is, the output of the NN. All these data should be normalized to the range from 0 to 1. This is to achieve better performance, see Sec. 9.3.2 for more details. The "now window" represents the short-term information of the traffic process. Both the "day window"

[1] It is the time unit during which the amount of traffic is measured (the average, the total number of packets or cells, or any similar statistical measure).

and "week window" as well as the "day" and "date" represent the long-term information.

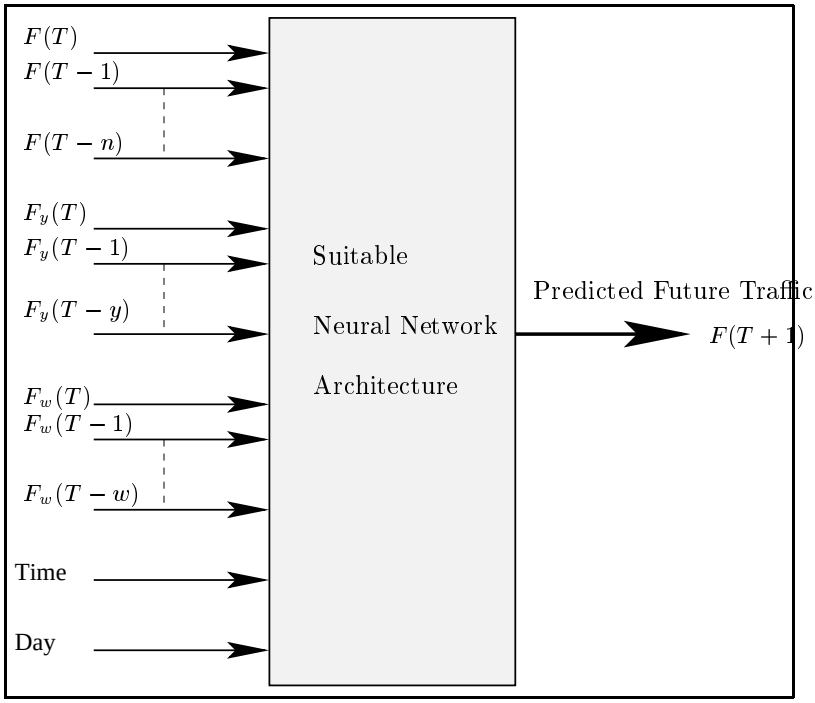


Figure 9.1: A black-box representation of our tool to predict in real time the future traffic, where $T \in [0, \infty]$ is the time step number, $F(T)$ is the current value of the traffic, $F(T-n)$ is that at the previous n^{th} step, $F_y(T)$ is that in the same instant but yesterday, $F_w(T)$ is that in the previous week, "Time" is the discrete time of the day, and "Day" is the day of the week

9.3 Experimental Results and Evaluation of the New Model

This Section is divided into several subsections to emphasize on several points.

9.3.1 Real Traces for Training

For all our experiments, we used real network traces representing the total traffic collected from the ingoing and outgoing flows at the "ENST de Bretagne (ENSTB)", at Rennes, France during 6 months (starting from October 1999 to March 2001). The traces are sampled and averaged every 5 minutes. We show in Table 9.1 a portion of this data.

As previously mentioned, other proposals used artificially generated traces by means of mathematical models or simulations which are easier to predict. Others used special kinds of traffic (MPEG) which has periodicity by nature due to the frame patterns IBBPBBPI..., which is easy to fetch for the NN. All these proposals works quite well for these special traces, but fail when used to test real traffics as shown in Section 9.4.

9.3.2 Training Database Description

To convert from the trace format to databases suitable for training and testing, we used a script that concatenates each day of the month. Then, based on the windows lengths, the script generates a database like the one partially shown in Table 9.2. While scanning the month, we report the maximum value of each traffic flow type. All the traffic data is then normalized by this value. The

Table 9.1: Some entries from the ENSTB trace.

Sample number	Incoming traffic	Outgoing traffic	Total traffic
1	1580	1073	2653
2	2353	841	3194
3	3636	1523	5159
4	1600	1329	2929
5	303	965	1268
6	217	555	772
7	250	639	889
8	254	1188	1442
9	260	4434	4694
10	263	888	1151
...	...	...	...

Time variable is simply a counter incremented at each time step. It represents the time of the day and it is normalized by the total number of samples per day. Similarly the *Day* represent the day of the week. It is normalized by 7. For the following experiments we used the total amount of traffic (the sum of the incoming and outgoing traffic). See Section 9.3.7 for details about the possible uses of the separate flows.

Table 9.2: Some entries from the training database after being normalized between 0 ... 1.

Step	Day	Time	F(T)	F(T-1)	F_y(T)	F_y(T-1)	F_w(T)	F_w(T-1)	F(T+1)
...	...	...	...	...	...	...	...	...	...
21	0.000	0.037	0.406	0.500	0.024	0.029	0.002	0.001	0.442
22	0.000	0.038	0.442	0.406	0.029	0.024	0.002	0.002	0.372
23	0.000	0.040	0.372	0.442	0.028	0.029	0.003	0.002	0.229
24	0.000	0.042	0.229	0.372	0.031	0.028	0.009	0.003	0.166
25	0.000	0.043	0.166	0.229	0.032	0.031	0.009	0.009	0.212
...	...	...	...	...	...	...	...	...	...
743	0.143	0.240	0.104	0.173	0.298	0.311	0.002	0.002	0.087
744	0.143	0.242	0.087	0.104	0.280	0.298	0.003	0.002	0.093
745	0.143	0.243	0.093	0.087	0.314	0.280	0.003	0.003	0.281
746	0.143	0.245	0.281	0.093	0.256	0.314	0.003	0.003	0.411
747	0.143	0.247	0.411	0.281	0.238	0.256	0.003	0.003	0.049
...	...	...	...	...	...	...	...	...	...
1950	0.429	0.252	0.277	0.280	0.045	0.035	0.420	0.671	0.206
1951	0.429	0.253	0.206	0.277	0.053	0.045	0.165	0.420	0.221
1952	0.429	0.255	0.221	0.206	0.076	0.053	0.258	0.165	0.521
1953	0.429	0.257	0.521	0.221	0.134	0.076	0.360	0.258	0.101
1954	0.429	0.258	0.101	0.521	0.135	0.134	0.223	0.360	0.143
...	...	...	...	...	...	...	...	...	...

9.3.3 Experimental Tests to Identify the Best Length of Each Window

To identify the best architecture (in term of each window size, and the number of hidden neurons) for the neural network, we carried out the following experiment. From the ENSTB trace, we created a set of training and testing databases. Each database is characterized by the length of the now-, yesterday-, and week-window sizes. We varied the now window from 2 to 5, the yesterday window from 0 to 3, and the week window from 0 to 3.

For the neural network architecture, we varied the number of hidden neurons from 1 to 10. We trained and tested each case of the 36 networks with the different training and testing databases, and repeated that for the different architectures obtained by varying the number of hidden neurons. As it is known that the initial values of the neural network weights (generally chosen randomly) affect the overall performance of the neural network, we initialized the random seed generator to the same value every time we created a new network.

For each trained neural network, we used it to predict all the samples of the next two weeks. We reported the number of hidden neurons, the size of each window and the MSE value. The minimum number of the hidden neurons that gives the best performance is 2. The optimal values of the window size is that 3 for the "now", 2 for "yesterday", and 2 for the "week" windows. For these optimal values we got MSE=0.0041.

These values illustrate the effectiveness of our model. The yesterday and the last-week windows as well as the date and day, when used, improved the performance of the NN to predict and to learn the traffic process. After choosing the best windows sizes, we show in Figure 9.2 the selected NN architecture for our model.

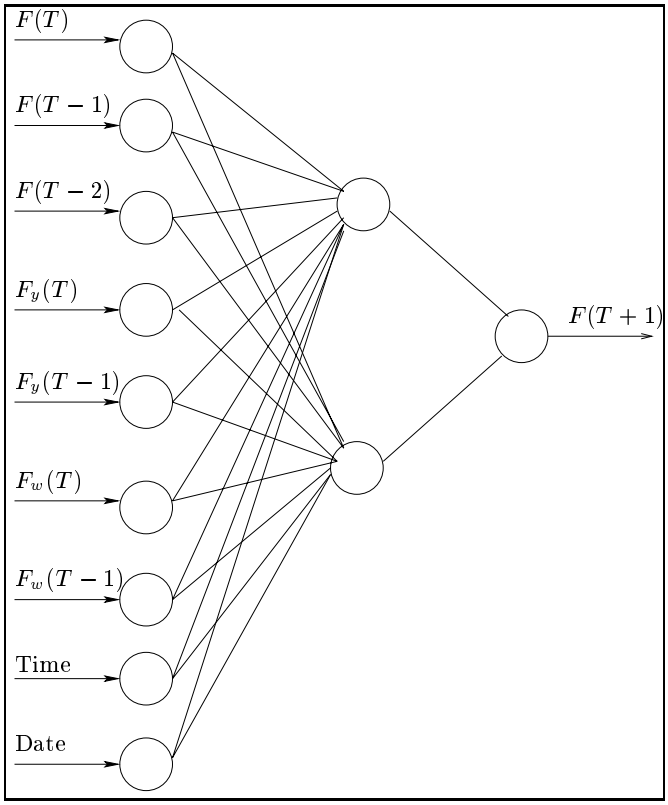


Figure 9.2: Our best architecture employing both short- and long-range dependencies in traffic prediction for the ENSTB Network.

9.3.4 Performance of the NN to Predict Traffic

The trained NN was used to predict the whole traffic of the next two weeks. In Figure 9.3 we show the actual and the predicted traffic against the time samples. (Note that we selected sequentially 1 sample every 4 samples for drawing to make the Figure somewhat clear.) We also show in Figure 9.4 the actual and the predicted traffic for the third and fourth days in the predicted third week. For this prediction we got MSE = 0.0046.

From these Figures, we can see that the NN managed to predict the whole next two weeks with very good precision. To visualize the performance of the NN, we depict in Figure 9.5 the difference between the actual and the predicted traffic for this experiment. Furthermore, a histogram for the percentage distribution of the samples from -1 to 1 with a step of 0.1 is shown in Figure 9.6. From these Figures we can see that about 78% of the samples are predicted with a precision between $\pm$ 0.05. About 8% of the samples are predicted with a precision between 0.05 and 0.15, where the actual traffic is higher than the prediction. On the other hand, about 10% of the samples are predicted with a precision between 0.05 and 0.15, where the actual traffic is lower than the prediction.

From Figure 9.5, we can see that a few samples are predicted with a precision between 0.2 and 0.4. This is due to the phenomena known as the *"spikes"*. A spiky sample occurs when the difference between the future and current sample is large and there is no information in the past samples that could indicate the possible occurrence of the spike. This is a well known phenomena in the traffic process. By definition, spikes are non-predictable (at least, with the same precision as the rest of the process). Furthermore, we recommend that the spikes should be removed from the training database. The idea is to avoid introducing spurious information during the learning process. Fortunately, their occurrence in real traffic is not frequent as seen from Figures 9.5 and 9.6.

In addition, if the spikes are not taken into account for the calculation of the MSE, the performance improves significantly. For example, by removing 124 spiky samples out of the 4200 samples of the predicted weeks, and considering simply that a sample is spiky if it differs by greater than 0.3 from the past sample, the MSE improved from 0.0046 to 0.00163. Figure 9.7 shows the difference between the actual and the predicted traffic after removing these spikes. It is clear that the performance is much better than in the case of Figure 9.5.

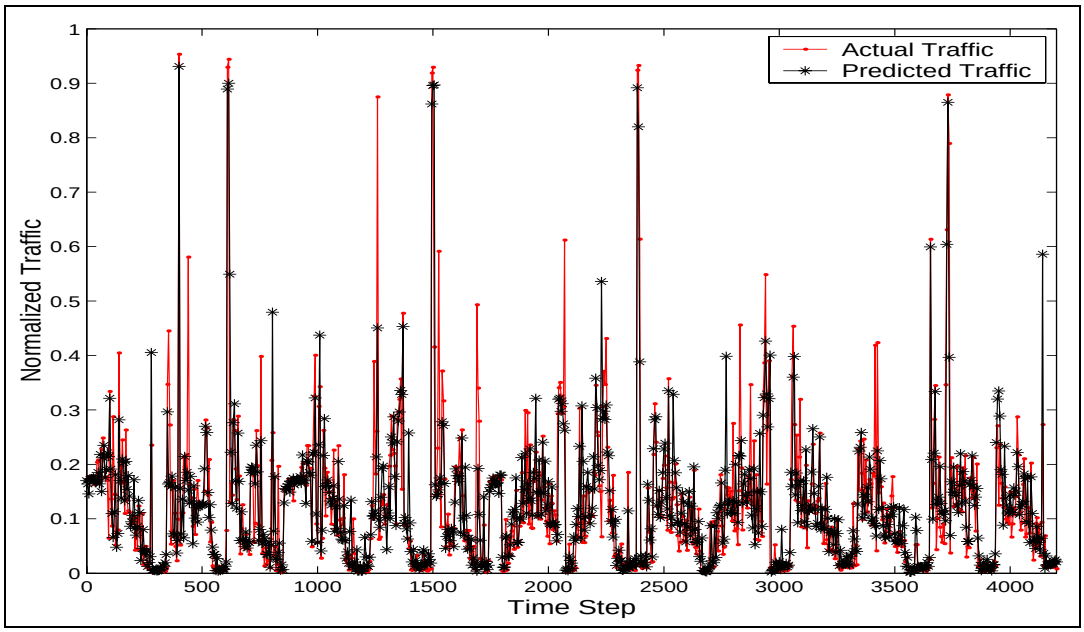


Figure 9.3: The actual traffic against the predicted one for the whole complete next two weeks.

9.3.5 More Than One Step in the Future

In the previous analysis, we used the NN to predict only one future step. However, it is possible to use our model to predict more than one step in the future. There are two ways of doing that. The first way is to train a NN having m outputs (number of future steps) and the same past inputs as our model (time, day, windows). This model is very naive and its performance is very bad.

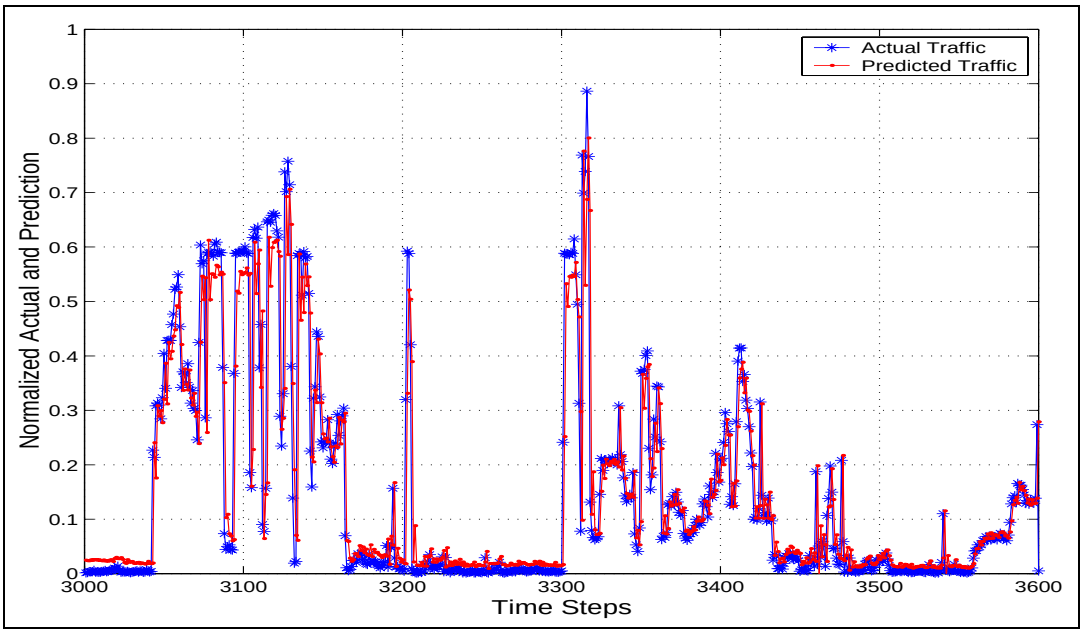


Figure 9.4: The Normalized actual against that predicted for the third and fourth days from the third week.

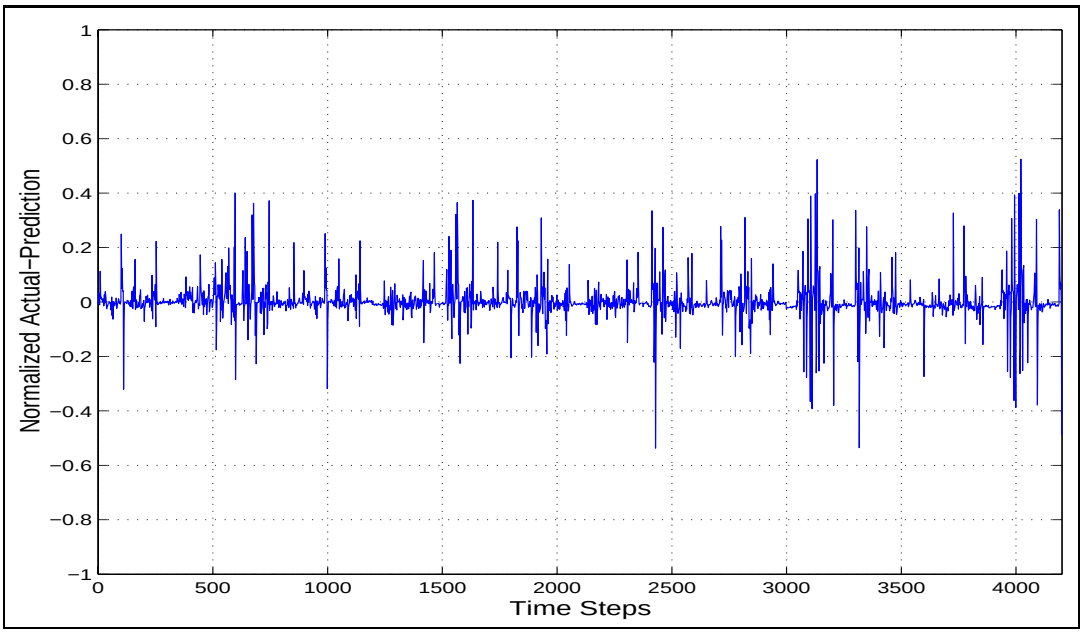


Figure 9.5: The difference between the actual and the predicted traffic for the complete next two weeks.

We propose a more sophisticated way to predict the future steps of the traffic by proceeding as follows. The same NN model as we described in Section 9.2 is to be used to predict $F(T+1)$ as before. The same trained NN is then to be used to predict $F(T+2)$ by setting $F(T-i) = F(T-i+1)$, $n \geq i \geq 0$, $F_y(T-j) = F_y(T-j+1)$, $y \geq j \geq 0$, and $F_w(T-k) = F_w(T-k+1)$, $w \geq k \geq 0$. We can continue to predict more than two steps in the future in the same way.

We used our model to predict the second step in the future for the whole next two weeks from the

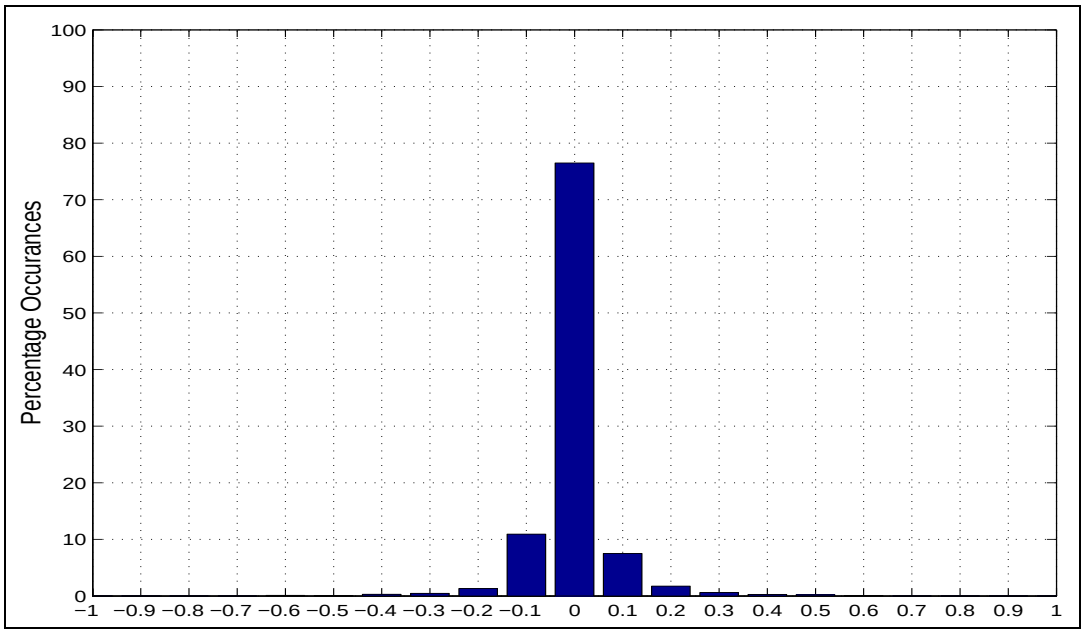


Figure 9.6: The histogram of the distribution of difference between actual and prediction with a step of 0.1.

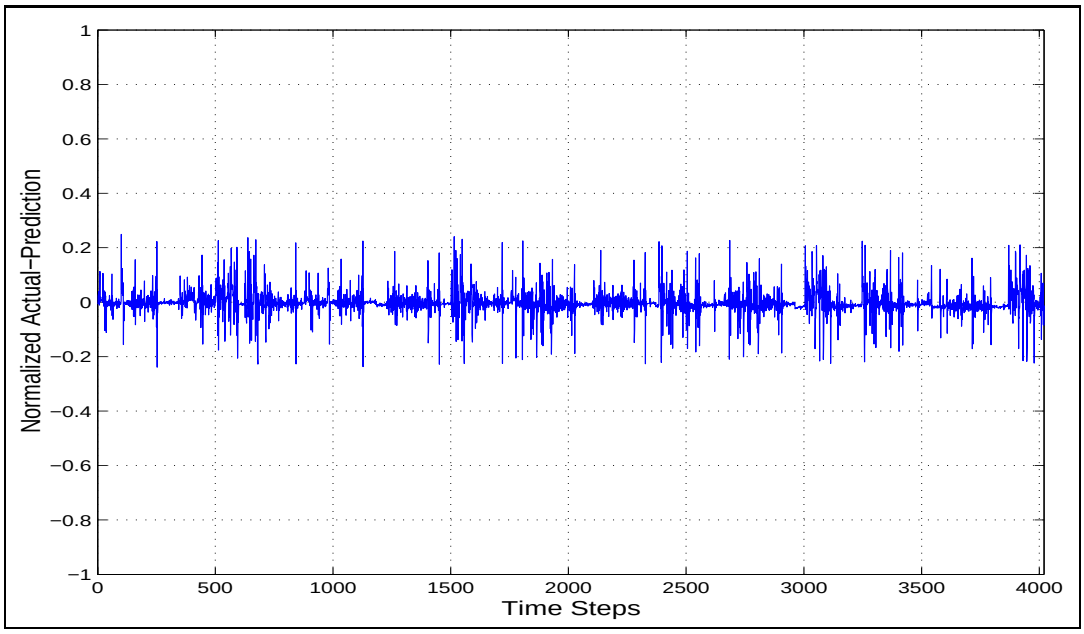


Figure 9.7: The difference between the actual and the predicted traffic for the complete next two weeks once the spiky samples are removed.

past values using the same trained NN described in Section 9.3.4, and obtained a MSE of 0.006. We show in Figure 9.8, the difference between the actual and the predicted traffic for the complete next two weeks for the second step. As we can see the performance of our model is good enough to predict even more than one step in the future.

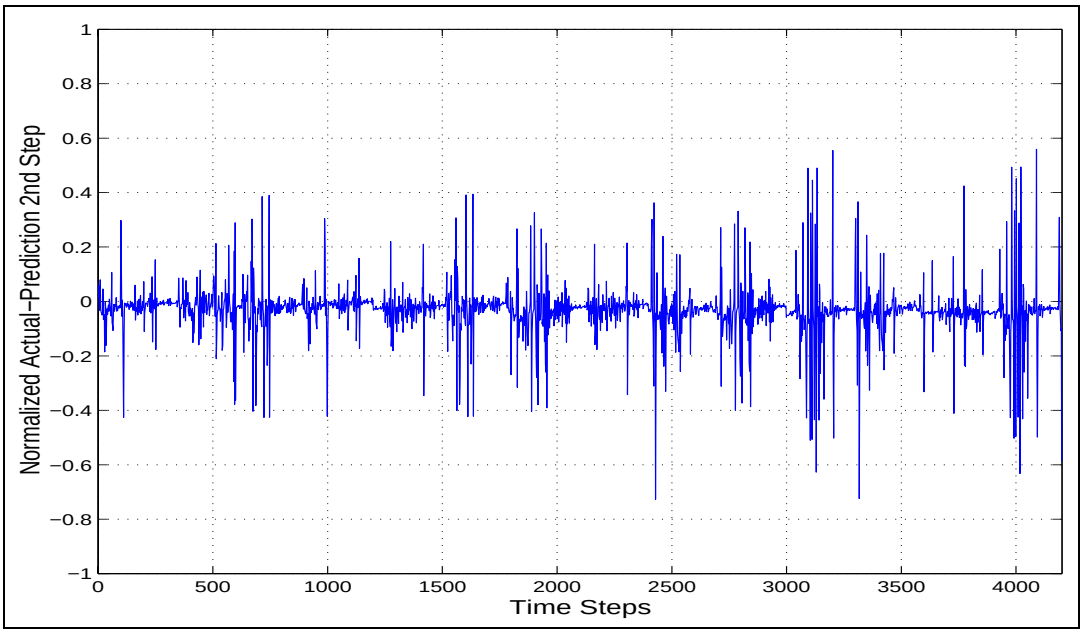


Figure 9.8: Predicting 2nd step ahead: the difference between the actual traffic and the predicted one for the complete next two weeks, including the spikes.

9.3.6 Conditions to Retrain the NN

As previously mentioned, the performance of the model may degrade when there are changes in the system. In this case, the NN should take this into account. This can be done either by on-line training or off-line training (in this case, retraining the working model when the MSE increases over some threshold).

For our model, trained by the first week, we tested it using samples of the next complete week as given in Section 9.3.4 which gave for the MSE=0.0046. However, when we used the same trained NN to predict the sixth week, we got MSE=0.012. This may be noticed from Figures 9.5 and 9.8, where the accuracy at the end is not the same as at the beginning. That means that the performance degraded and the NN should be retrained. Retraining the trained NN blindly will not improve the performance too much. This process is a critical one, so, it should be carried out carefully.

The retraining samples should be selected as follows. Removing all the spiky samples is a must to get better performance. We have to choose some samples that give good prediction of the last tested period. We have also to choose some samples that are mis-predicted. The choice of the mis-predicted samples is to let the NN learn the changes of the traffic process. The choice of the well-predicted samples is important to keep the NN informed about the past history of the traffic process and to avoid spoiling the trained NN.

We have carried out two experiments to retrain the NN. In the first one, we trained it blindly with a database containing spikes. We got MSE=0.009 on the testing database. Once we removed the spikes from the training database in the second experiment, the MSE improved to 0.0054 on the testing database.

It is important to mention that in [56], the authors argued that neither the on-line nor the off-line training improves the performance. But this is due to the fact that the spiky samples do not help. As we have shown, by removing them and choosing a fair number of samples from the well- and mis-predicted samples, the performance improved.

9.3.7 Traffic Flow Type

As stated before, the traces we used in these experiments constitute the total external (incoming and outgoing flows). However, we have tested our model for each flow separately. We have got approximately the same success rate for incoming only and outgoing only than for the total flow. That means that our model is suitable for the three cases, and everyone can implement it to suite his/her requirements. It would be redundant to provide further details or show some Figures about each flow separately, so we will omit them.

9.4 Comparison with Other Models

The majority of the existing works in this domain employ the 5-5-1 NN (as depicted in Figure 9.9) to predict the future traffic. To compare this architecture with our model, the traditional one has a "five-now-window" without day nor week windows, nor time nor date, and the NN has 5 hidden neurons.

We run the same experiment as we did for our model but using the traditional 5-5-1 model, and we have found that the performance of our model is much better than the traditional one. With the latter, we have got for the MSE 0.0092 instead of 0.0046 with our model (a factor of 1.9 improvement when using our model).

It is important to say that the traditional model were tested either on simulated traces or on traces that last only several hours sampled every second (Bellcore). It is not surprising to see in the literature that this simple model gives good results. However, once it is employed in long-term prediction, it does not give the same success rate as we do using our proposed methodology.

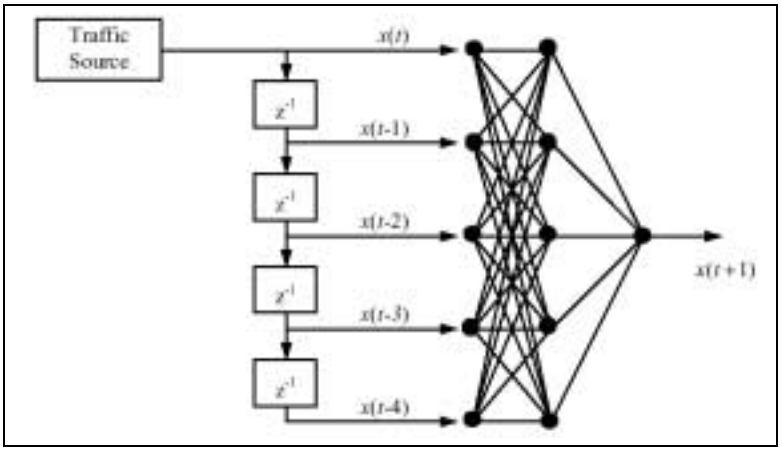


Figure 9.9: The traditional NN model that has been widely used to predict network traffic. This Figure is taken from [56, p. 115], where z^{-1} represents a unit-step delay function.

From [56] and the work presented in [157], we show in Figure 9.10 which is taken from [56], that the NN can capture the traffic with very good performance. However, the authors obtained these results by generating the data to train and test the NN using the following equation.

$$X(t) = 4 \times X(t-1) \times (1 - X(t-1)) \tag{9.1}$$

with $X(0) \in]0.0, 1.0[$ except 0.5. It is clear that this traffic model is very simple and very easy to be learned by any classical NN architecture. However, this kind of data (claimed in [56] to simulate real network traffic) is far from being like real network traces as argued.

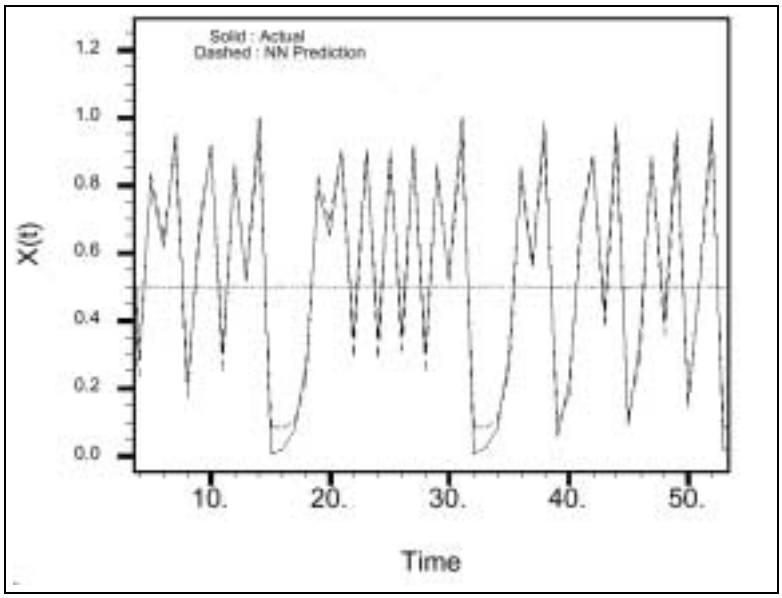


Figure 9.10: The actual against the NN prediction when training and testing it by data generated by Eqn. 9.1. This Figure is taken from [56].

9.5 Possible Uses of our Model

Our model can be used mainly for:

- Dynamic bandwidth allocation and dynamic contract negotiation: once a good traffic predictor is built, the BW can be allocated dynamically, instead of statically as it is currently done. In this case, an enhanced version of our proposed model can be used to dynamically renegotiate the contract with the service provider such that the clients can use only what they will consume.

- Dynamic traffic shaping: other direct use of our technique is within some large institutions or enterprises structured in groups where there a given protion of the BW is allocated for each group. For example in the university, one can find an organization where there is a given BW for the students, employees, staff, research, ..., which is statically allocated. Our model can be used so that each group can have roughly what it needs. The rest can be redistributed to the others who may starve for small BW. For example, suppose that the total BW is 100Mbps, the dean has fixed part of 10Mbps, and the students have 30Mbps. Not all the time the dean is working in his office, while the students' network may be congested from time to time, our model can be used to detect automatically that the dean is not working now and decide that the unused BW could be given to the students. Of course in such a scenario, a separate NN should be used for each group's local network.

- Anticipating the best bit rate and encoding parameters the source has to use to deliver the best real-time multimedia quality. Traditionally, the current congestion reacting protocols (with the aim of maximizing the QoS) operate based on the current network state (congestion level) and the receiver asks the sender to modify the sending parameters (bit rate in general) accordingly. Obviously, there is a certain time needed to react. During this time, the quality can remain poor. By using a traffic predictor coupled with the real-time quality evaluation we have presented in the previous chapters, and the rate control protocol we have presented in Chapter 8, the source can anticipate the state of the network and modify the parameters, if needed, in advance before getting the network congested and degrading the quality. This will improve both the quality of

the multimedia flows and the network availability.

9.6 Conclusion and Discussion

We have proposed in this chapter a new model for traffic prediction. The novelty of this model with respect to the existing literature, is that it makes use of both the long-range and short-range periodicity of the traffic process to provide a more accurate estimation. Good candidates for solving this kind of problems are neural networks, used here to learn from the past history to predict the future arrivals of the traffic process.

As other models, for short-range periodicity, we use the information in the past few time steps ("now" window). However, to make use of the long-range information, we added other inputs: the current time of the day, the current day of the week, past information about the traffic in the previous day at the same time, and past information about the traffic in the same day of the previous week.

We validated our model by using real traffic traces collected from a large institution (the ENSTB) over a long period (6 months). We did not use traces generated by artificial simulations or mathematical equations (generally the case in the existing models) as they are easy to model by NN and do not characterize well real traffic processes. In addition, we did not use any special kind of traffic (like MPEG) which are also easy to learn by NN as they are periodical by nature. The existing models based on these kinds of traces work well for the special type of flow, but their performances degrade significantly when applied to traces corresponding to general traffic, as we show in this Chapter.

We run some experiments to identify the best architecture of our model. The best candidate appears to be: 3, 2, and 2 for now, yesterday and last week window sizes respectively. This architecture may be the best for the ENSTB network, and should not be taken as a reference. We provided the guidelines to show how to identify the optimal values corresponding to a given network. We also derived a methodology to predict more than one step in the future and we evaluated it. We have found that when the retraining samples are selected carefully, the performance improves considerably.

We showed through different experiments that our technique outperforms existing ones, and our explanation is that our method uses long-range information while classical attempts to solve the same problem don't.

We discussed how different applications can take benefit from our traffic prediction. Dynamic bandwidth allocation, dynamic long-term contract negotiation, pricing, traffic shaping and engineering are some of them.

Chapter 10

New Random Neural Network Training Algorithms

10.1 Introduction

We have shown that the Random Neural Network (RNN) [45, 51, 46] performs well for multimedia quality evaluation. A possible problem with RNN for large applications (large number of neurons and large size of training examples) is that it can take a significant amount of time to be trained. The training algorithm in the available software associated with this tool is the gradient descent one proposed by the inventor of RNN, Erol Gelenbe [47]. This algorithm can be slow and may require a high number of iterations to reach the desired performance. In addition, it suffers from the zigzag behavior (the error decreases to a local minimum, then increases again, then decreases, and so on).

This problem oriented our research in order to find new training algorithms for RNN, inspired by the fact that for ANN, there are many training methods with different characteristics. We present in this Chapter two new training algorithms for RNN. The first one is inspired from the *Levenberg-Marquardt* (LM) training algorithm for ANN [55]. The second one is inspired from a recently proposed training algorithm for ANN referred as *LM with adaptive momentum* [8]. This algorithm aims to overcome some of the drawbacks of the traditional LM method for feedforward neural networks.

We start by describing the gradient descent algorithm and then we present our training algorithms for RNN. We then evaluate them through a comparative study of their performances. Finally, we give some conclusions.

10.2 Gradient Descent Training Algorithm for RNN

Gelenbe's RNN training algorithm [47] proceeds as follows. Recall that w_{ij}^+ and w_{ij}^- represent the rates at which neuron i sends respectively excitation or inhibition spikes to neuron j. Let n denote the total number of neurons in the network. The adjustable network parameters are the two matrices $\mathbf{W}^+ = \{w_{ij}^+\}$ and $\mathbf{W}^- = \{w_{ij}^-\}$, both of which have $n \times n$ elements[1]. These parameters should be determined by the training algorithm from the training examples (a set of K input-output pairs). The set of inputs is denoted by $\mathbf{X} = (x^{(1)}, \ldots, x^{(K)})$. Each of the inputs $x^{(k)}$, $k \in [1 \ldots, K]$, consists of a set of excitation-inhibition pairs $x^{(k)} = (\lambda^{+(k)}, \lambda^{-(k)})$ representing the signal flow entering each neuron from outside the network. Thus, for each set of inputs, we have $\lambda^{+(k)} = (\lambda_1^{+(k)}, \ldots, \lambda_n^{+(k)})$ and $\lambda^{-(k)} = (\lambda_1^{-(k)}, \ldots, \lambda_n^{-(k)})$. The set of outputs is denoted by $\mathbf{Y} = (y^{(1)}, \ldots, y^{(K)})$, where $y^{(k)} = (y_1^{(k)}, \ldots, y_n^{(k)})$; the elements $y_i^{(k)} \in [0, 1]$ represent the desired output of each neuron.

[1] This is actually true in the general case, including for instance optimization applications of neural networks. In our work, where the topology is feedforward (as usual in learning applications), we actually have only half of this number of variables.

The tranining technique must adjust the parameters $\mathbf{W}^{+/-}$ in order to minimize a cost function $E^{(k)}$, given by

$$E^{(k)} = \tfrac{1}{2}\sum_{i=1}^{n} a_i(\varrho_i^{(k)} - y_i^{(k)})^2, \quad a_i \geq 0, \tag{10.1}$$

where $\varrho_i^{(k)}$ and $y_i^{(k)}$ are the neuron's i actual and desired output for the input-output pair $k \in \{1,\ldots,K\}$ respectively. The constant a is set to zero for the neurons that are not connected to the outputs of the network. At each successive input-output pair k, the adjustable network parameters $\mathbf{W}^{+(k)} = \{w_{i,j}^{+(k)}\}$ and $\mathbf{W}^{-(k)} = \{w_{i,j}^{-(k)}\}$, where $i,j = 1,\ldots,n$ need to be updated. The algorithm used by Gelenbe is the gradient descent one. Observe that the values in the matrices $\mathbf{W}^+$ and $\mathbf{W}^-$ must not be negative (since their elements are transition rates).

The rule for the weights update is as follows:

$$\begin{aligned} w_{u,v}^{+(k)} &= w_{u,v}^{+(k-1)} - \eta\sum_{i=1}^{n} a_i(\varrho_i^{(k)} - y_i^{(k)})[\partial\varrho_i/\partial w_{u,v}^{+}]^{(k)}, \\ w_{u,v}^{-(k)} &= w_{u,v}^{-(k-1)} - \eta\sum_{i=1}^{n} a_i(\varrho_i^{(k)} - y_i^{(k)})[\partial\varrho_i/\partial w_{u,v}^{-}]^{(k)}. \end{aligned} \tag{10.2}$$

Here, $\eta > 0$ (it is called the learning rate), $\{u,v\} = 1,\ldots,n$ and ϱ_i is the output of neuron i calculated from the k^{th} input and from the equations by setting $w_{u,v}^{*} = w_{u,v}^{*(k-1)}$, where $*$ can be either $+$ or $-$.

Recall that

$$\varrho_i = \frac{T_i^+}{r_i + T_i^-}, \tag{10.3}$$

where

$$T_i^+ = \lambda_i^+ + \sum_{j=1}^{n}\varrho_j w_{j,i}^+, \quad T_i^- = \lambda_i^- + \sum_{j=1}^{n}\varrho_j w_{j,i}^-, \tag{10.4}$$

and

$$r_i = \sum_{j=1}^{n}[w_{j,i}^+ + w_{j,i}^-]. \tag{10.5}$$

Using Eqn. 10.3 to compute $[\partial\varrho_i/\partial w_{u,v}^{*}]^{(k)}$, we have:

$$\partial\varrho_i/\partial w_{u,v}^{+} = \sum_j \Big[\partial\varrho_j/\partial w_{u,v}^{+}[w_{i,j}^{+} - w_{i,j}^{-}]/D_i\Big] - 1[u\equiv i]\varrho_i/D_i + \varrho_u/D_i, \tag{10.6}$$

$$\partial\varrho_i/\partial w_{u,v}^{-} = \sum_j \Big[\partial\varrho_j/\partial w_{u,v}^{-}[w_{i,j}^{+} - w_{i,j}^{-}]/D_i\Big] - 1[u\equiv i]\varrho_i/D_i - \varrho_u\varrho_i/D_i, \tag{10.7}$$

where $D_i = r_i + \lambda_i^- + \sum_j \varrho_j w_{j,i}$ and $1[x] = \begin{cases} 1 & \text{if } x \text{ is true} \\ 0 & \text{otherwise} \end{cases}$.

Let $\varrho = (\varrho_1,\ldots,\varrho_n)$, and define the $n\times n$ matrix:

$$\mathbf{W} = \{[w_{i,j}^{+} - w_{i,j}^{-}\varrho_j]/D_j\} \quad i,j = 1,\ldots,n.$$

We can now write

$$\begin{aligned} \partial\varrho/\partial w_{u,v}^{+} &= \partial\varrho/\partial w_{u,v}^{+}\mathbf{W} + \gamma_{u,v}^{+}\varrho_u, \\ \partial\varrho/\partial w_{u,v}^{-} &= \partial\varrho/\partial w_{u,v}^{-}\mathbf{W} + \gamma_{u,v}^{-}\varrho_u, \end{aligned} \tag{10.8}$$

where the elements of the n-vectors $\gamma_{u,v}^{+} = [\gamma_{u,v}^{+(1)},\ldots,\gamma_{u,v}^{+(n)}]$, $\gamma_{u,v}^{-} = [\gamma_{u,v}^{-(1)},\ldots,\gamma_{u,v}^{-(n)}]$ are given by:

$$\gamma_{u,v}^{+(n)} = \begin{cases} -1/D_i & \text{if } u = i, \quad v \neq i, \\ +1/D_i & \text{if } u \neq i, \quad v = i, \\ 0 & \text{otherwise;} \end{cases}$$

$$\gamma_{u,v}^{-(n)} = \begin{cases} -(1+\varrho_i)/D_i & \text{if } u = i, \quad v = i, \\ -1/D_i & \text{if } u = i, \quad v \neq i, \\ -\varrho_i/D_i & \text{if } u \neq i, \quad v = i, \\ 0 & \text{otherwise.} \end{cases}$$

It can be observed that Eqn. 10.8 can be rewritten as follows:

$$\begin{aligned} \partial\varrho/\partial w_{u,v}^{+} &= \gamma_{u,v}^{+}\varrho_u[\mathbf{I} - \mathbf{W}]^{-1}, \\ \partial\varrho/\partial w_{u,v}^{-} &= \gamma_{u,v}^{-}\varrho_u[\mathbf{I} - \mathbf{W}]^{-1}, \end{aligned} \tag{10.9}$$

where $\mathbf{I}$ is the $n \times n$ identity matrix.

The training algorithm can be started by first initializing the two matrices $\mathbf{W}^{+(0)}$ and $\mathbf{W}^{-(0)}$ randomly. Then, we have to choose the value of η. For each successive value k, starting from $k = 1$, one must proceed as follows:

1. Set the input values to $(\lambda^{+(k)}, \lambda^{-(k)})$.
2. Solve the system of nonlinear equations 10.3 and 10.4.
3. Solve the system of linear equations 10.9 with the results of step 2.
4. Using Eqn. 10.2 and the results of Eqn. 10.3 and Eqn. 10.4, update the matrices $\mathbf{W}^{+(k)}$ and $\mathbf{W}^{-(k)}$. It should be noted that the values in the two matrices $\mathbf{W}^{+(k)}$ and $\mathbf{W}^{-(k)}$ should not be negative, as stated before. Thus, in any step k of the algorithm, if the iteration yields a negative value of a term, we consider two possibilities:
   (a) Set the negative value to zero and stop the iteration for this term at this step k. In the next step $k+1$, iterate on this term with the same rule starting from its current zero value.
   (b) Go back to the previous value of the term and iterate with a smaller value of η.

After the training phase, the network can be used in normal operation (the only needed computations to obtain the outputs are those given by equations 10.3 and 10.4).

10.3 New Levenberg-Marquardt Training Algorithms for RNN

In the gradient descent algorithm, the error function to be minimized should be continuous and differentiable with respect to its parameters. Thus, the gradient of the error with respect to any of the parameters can be analytically computed. Once the gradients have been specified, the most straightforward approach for error minimization is gradient descent, where at each point the update direction of the parameter vector is the opposite of the gradient at this point. This approach, due to its simplicity, has many drawbacks:

- the zig-zag behavior,
- the difficulty in choosing the value of the learning rate parameter η,
- the fact that it is slow, usually requiring a large number of training steps.

To alleviate from these problems, we tried to apply the LM method, which is an approximation to Newton methods, on RNN. The LM technique is known to be the best algorithm for optimization problems applied to ANN [55]. In this Section we are going to introduce the LM method as well as some other versions of it. Recently, there has been an improvement of the traditional LM method, which is referred as the adaptive momentum LM for ANN. We will also present the same algorithm adapted to RNN.

10.3.1 Analytical Formulation of the Algorithm

Following Gelenbe's notations, let us write

$$N_i = \lambda_i^+ + \sum_j \varrho_j w_{i,j}^+, \tag{10.10}$$

$$D_i = r_i + \lambda_i^- + \sum_j \varrho_j w_{i,j}^-, \tag{10.11}$$

$$\varrho_i = N_i / D_i, \tag{10.12}$$

where $i \in 1, \ldots, n$ represents the neuron index, w^-, w^+ the weights, λ_i^+, λ_i^- the excitation and inhibition external signals for neuron i, and ϱ_i the output of the neuron i. Define

$$E = \sum_{k=1}^{K} E^{(k)}, \tag{10.13}$$

recalling that

$$E^{(k)} = \tfrac{1}{2} \sum_{i=1}^{n} a_i (\varrho_i^{(k)} - y_i^{(k)})^2, \quad a_i \geq 0. \tag{10.14}$$

The mathematical formulation of LM method applied to RNN is as follows:

- We define a generic vector p, of M elements, containing the adjustable parameters $w_{i,j}^+$ and $w_{i,j}^-$; $p = (p_1, \ldots, p_M)$; $p^{(k)}$ is the parameter vector at step (k) of the training process.
- We define also $g = (g_1, \ldots, g_M)$ the gradient vector, where $g_l = \partial E / \partial p_l$ for $l = 1, \ldots, M$. Denote by $g^{(k)}$ the gradient vector at point $p^{(k)}$.
- The weights update based on Newton method is as follows:

$$p^{(k+1)} = p^{(k)} + s^{(k)}, \tag{10.15}$$

where $s^{(k)}$ is the Newton's direction obtained by solving the system

$$\mathbf{H}^{(k)} s^{(k)} = -g^{(k)}, \tag{10.16}$$

$\mathbf{H}^{(k)}$ being the Hessian matrix at step k.

For LM, the Hesssian matrix $\mathbf{H}^{(k)}$ is approximated by:

$$\mathbf{H}^{(k)} = \mathbf{J}^{(k)^T} \mathbf{J}^{(k)} + \mu^{(k)} \mathbf{I}. \tag{10.17}$$

Here, $\mu^{(k)} > 0$ and $J^{(k)}$ is the Jacobian matrix at step k, given by

$$\mathbf{J} = \begin{bmatrix} \frac{\partial e_1}{\partial p_1} & \cdots & \frac{\partial e_1}{\partial p_M} \\ \vdots & \ddots & \vdots \\ \frac{\partial e_N}{\partial p_1} & \cdots & \frac{\partial e_N}{\partial p_M} \end{bmatrix}, \tag{10.18}$$

where $e_i = y_i - \varrho_i$ is the prediction error of neuron i, ϱ_i is the output of neuron i at the output layer, y_i is the desired output of neuron i at the output layer, and N is the number of outputs multiplied by the number of training examples K.

Each element of matrix J is computed using the following equations:

$$\mathbf{J}_{l,m} = \partial e_l / \partial p_m = -\partial \varrho_l / \partial p_m, \qquad \text{where } l = 1, \ldots, N \text{ and } m = 1, \ldots, M, \tag{10.19}$$

$$\partial \varrho / \partial p_m = \begin{cases} T^+_{u,v} \varrho_u [\mathbf{I} - \mathbf{W}]^{-1} & \text{if } \; p_m = w^+_{u,v}, \\ T^-_{u,v} \varrho_u [\mathbf{I} - \mathbf{W}]^{-1} & \text{if } \; p_m = w^-_{u,v}, \end{cases} . \tag{10.20}$$

From Eq. 10.16, we obtain:

$$s^{(k)} = -[\mathbf{H}^{(k)}]^{-1} g^{(k)}. \tag{10.21}$$

Equations 10.17 and 10.21 give:

$$s^{(k)} = -[\mathbf{J}^{(k)^T} \mathbf{J}^{(k)} + \mu^{(k)} \mathbf{I}]^{-1} g^{(k)}. \tag{10.22}$$

By grouping Equations 10.15 and 10.22 we obtain:

$$p^{(k+1)} = p^{(k)} - [\mathbf{J}^{(k)^T} \mathbf{J}^{(k)} + \mu^{(k)} \mathbf{I}]^{-1} g^{(k)}. \tag{10.23}$$

To compute vector g, we use:

$$g = \mathbf{J}^T e, \tag{10.24}$$

where $e = (e_1, \ldots, e_N)$.

10.3.2 Different Variants of the LM Training Algorithm

There are several methods to update the $\mu^{(k)}$ parameter at each step k; the main idea is to choose a value for which the error function is minimized. After initializing the weights randomly, the training algorithm using the LM method in general is as follows:

1. Present all inputs to the network and compute the corresponding network outputs and errors from Eqns. 10.12 and 10.13, and the sum of the squares of errors over all the inputs using Eqn. 10.14.

2. Compute the Jacobian matrix based on Eqn. 10.18. The Hesssian matrix can be obtained from Eqn. 10.17; then the gradient vector is to be calculated from Eqn. 10.24.

3. The weights adjustments Δp are obtained from Eqn. 10.21.

4. Recompute the sum of squares using $p + \Delta p$. If this new sum of squares is smaller than the one computed in step 1, then reduce μ by β, let $p = p + \Delta p$, and go back to step 1. If the sum of squares is not reduced, increase μ by β and go back to step 3.

5. The algorithm is assumed to converge when certain criteria are satisfied. The stop criteria may include: the μ value is greater or less than a predefined threshold, there is no adjustment on the weights, or the elements of the gradient vector g become zeros, a maximum number of iterations is reached, etc.

The parameter μ is initialized to a positive value (for example 0.01), β is a constant and has a positive value (for example 10), p is the vector that contains the adjustable parameters, and Δp is equivalent to the search direction s.

As described in [55], the Leveberg-Marquardt algorithm is a modification of the backpropagation algorithm to train ANN models. The only changes to this algorithm in order to apply it to RNN are the equations used to compute the outputs, errors, sum of squares of errors and the Jacobian matrix, given by equations 10.10 through 10.24.

There is another variant of the above algorithm, which differs only in the way the value of μ is updated. The idea is to keep it constant "in the center":

if $E - E_{new} > 0.75 * L$
then $\mu = \mu/\beta$
else if $E - E_{new} < 0.25 * L$
then $\mu = \mu * \beta$
else keep μ as it is

where $L = s \times gp + s \times s^T \times \mu$. The weights are updated only if E_{new} is less than the previous E. We use this algorithm in all the results presented in the remaining of this Chapter.

One of the drawbacks of the LM method is that it requires a large amount of memory. In nowadays computers, this is not, in general a big issue. However, it must be observed that there is a slight modification to the above algorithm in order to use less memory space (referred as "the reduced memory LM") but it is slower. The difference is in the method used to compute the search direction. The Cholesky factorization is used to calculate it from the Hesssian matrix and the gradient vector.

10.4 New LM with Adaptive Momentum for RNN

Even though LM is one of the most powerful algorithms to train feedforward networks, it still has some drawbacks. One of them is the need to compute the Jacobian matrix and to invert the Hessian matrix with dimensions equal to the number of weights of the network. However, this problem is compensated by the increased rate of convergence of the algorithm, which becomes quadratic as we approach to a solution. Another disadvantage is that it is not guaranteed to converge to the global minimum of the cost function. When the algorithm converges to a local minimum, it is not possible to escape. Thus, in [8], the authors proposed an algorithm that handles local minima with increased robustness and still maintains its fast convergence rate. The results of the simulation tests they made have revealed the power of this method to obtain solutions in difficult problems whereas other techniques including LM fail to converge.

The main idea to help LM to overshoot to a local minima is to include a momentum term that inserts second order derivative information in the training process and provides iterations whose form is similar to that in the Conjugate Gradient (CG) method. The major difference with the CG method is that the coefficients regulating the weighting between the gradient and the momentum term are heuristically determined whereas in the CG they are adaptively determined.

10.4.1 The Algorithm with Adaptive Momentum

The goal is to choose minimization directions, which are not interfering and linearly independent. This can be achieved by the selection of conjugate directions which forms the basis of the CG method. Two vectors are non-interfering and linearly or mutually conjugate with respect to $\nabla^2 E(w_t)$ when

$$dw_t^T \nabla^2 E(w_t) dw_{t-1} = 0 \tag{10.25}$$

where $E(w) = \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{n}(o_i^{(k)} - y_i^{(k)})^2$, $\nabla^2 E(w_t)$ is the Hessian matrix $\mathbf{H}_t = \mathbf{J}^{(k)^T}\mathbf{J}^{(k)} + \mu^{(k)}\mathbf{I}$, with K is the number of training patterns, n is the number of outputs, w_t the weights vector, and dw_t is the optimal step (or the search direction). The objective is to reach a minimum of the cost function with respect to w_t and to simultaneously maximize $\Phi_t = dw_t^T \nabla^2 E(w_t) dw_{t-1}$ without compromising the need for a decrease of the cost function. At each iteration of the learning process, the weight vector w_t will be incremented by dw_t, so that:

$$dw_t^T \nabla^2 E(w_t) dw_t = (\delta P)^2 \tag{10.26}$$

where δP is a constant and the change $dE(w_t)$ in $E(w_t)$ is equal to a predetermined quantity $\delta Q_t < 0$:

$$dE(w_t) = \delta Q_t. \tag{10.27}$$

This is a constrained optimization problem which can be analytically solved by introducing two Lagrange multipliers λ_1 and λ_2. Then function ϕ_t is introduced to evaluate the differentials involved in the right hand side and to substitute the function Φ_t:

$$\phi_t = \Phi_t + \lambda_1(\delta Q_t - dE(w_t)) + \lambda_2[(\delta P)^2 - dw_t^T \nabla^2 E(w_t) dw_t]. \tag{10.28}$$

By replacing Φ_t by its value, we obtain:

$$\phi_t = dw_t^T \nabla^2 E(w_t) dw_{t-1} + \lambda_1(\delta Q_t - \nabla E(w_t)^T dw_t) + \lambda_2[(\delta P)^2 - dw_t^T \nabla^2 E(w_t) dw_t]. \tag{10.29}$$

To maximize ϕ at each iteration, the following equations must be satisfied:

$$d\phi_t = d^2 w_t^T (\nabla^2 E(w_t) dw_{t-1} - \lambda_1 \nabla E(w_t) - 2\lambda_2 \nabla^2 E(w_t) dw_t) = 0, \tag{10.30}$$

and

$$d^2\phi_t = -2\lambda_2[d^2 w_t^T \nabla^2 E(w_t) d^2 w_t] < 0. \tag{10.31}$$

Hence from 10.30 we obtain:

$$dw_t = -\frac{\lambda_1}{2\lambda_2}[\nabla^2 E(w_t)]^{-1} \nabla E(w_t) + \frac{1}{2\lambda_2} dw_{t-1}. \tag{10.32}$$

From equations 10.27 and 10.32 we obtain:

$$\delta Q_t = \frac{1}{2\lambda_2}(I_{GF} - \lambda_1 I_{GG}), \tag{10.33}$$

where

$$I_{GG} = \nabla E(w_t)^T [\nabla^2 E(w_t)]^{-1} \nabla E(w_t), \tag{10.34}$$

$$I_{GF} = \nabla E(w_t)^T dw_{t-1}. \tag{10.35}$$

From 10.33 we obtain λ_1:

$$\lambda_1 = \frac{-2\lambda_2 \delta Q_t + I_{GF}}{I_{GG}}. \tag{10.36}$$

It remains to obtain λ_2. This can be done by substituting Eqn. 10.30 in Eqn. 10.26 to obtain:

$$4\lambda_2^2(\delta P)^2 = I_{FF} + \lambda_1^2 I_{GG} - 2\lambda_1 I_{GF}, \tag{10.37}$$

where

$$I_{FF} = dw_{t-1}^T \nabla^2 E(w_t) dw_{t-1}. \tag{10.38}$$

Finally, Eqn. 10.36 is substituted into Eqn. 10.37 and solve for λ_2 to obtain:

$$\lambda_2 = \frac{1}{2}\left[\frac{I_{GG}(\delta P)^2 - (\delta Q_t)^2}{I_{FF}I_{GG} - I_{GF}^2}\right]^{-\frac{1}{2}}. \tag{10.39}$$

The positive square root value has been chosen for λ_2 in order to satisfy Eqn. 10.31. Note the bound $|\delta Q_t| \leq \delta P\sqrt{I_{GG}}$ set on the value of δQ_t by Eqn. 10.39. The value chosen for δQ_t is always: $\delta Q_t = -\zeta \delta P\sqrt{I_{GG}}$ where ζ is a constant between 0 and 1. Therefore the free parameters are ζ and δP $(0 < \delta P < 1)$. The authors recorded during their simulations that the best results are given for $0.85 \leq \zeta \leq 0.95$ and $0.1 \leq \delta P \leq 0.6$.

10.5 Performance Evaluation of the Proposed Algorithms

In this Section, we provide some experimental results that we obtained in evaluating the proposed algorithms. We compare these results with those obtained for the gradient descent algorithm. For this purpose, we used two problems that have been already used in [1] for validation purposes. In addition, the XOR problem is also considered. The description of these problems is as follows:

- **Pattern classification**: in this problem, the goal is to classify two patterns [-1 1 -1 -1 1 1 1] and [1 -1 1 1 -1 -1 -1]. The desired outputs for these patterns are [1 0] and [0 1] respectively. As stated in [1], even though this problem may seem trivial, it is used in applications in decoding data coded with special sequences as in [2]. For this problem, we used the three-layer feedforward network consisting of 7 neurons in the input layer, 5 neurons in the output layer and 2 neurons in the output layer as shown in Figure 10.1. It should be mentioned that this problem has been used to show that RNN performs better than ANN. Even if ANN can converge in less time and iterations to the solution, RNN is the only one that can generalize well when a perturbation of the input is applied (inversion of one bit or week signal amplitude). For more details, refer to [1].

- **Fully recurrent network**: this problem concerns a 9-neuron single-layer fully recurrent network as shown in Figure 10.2. The goal is to use the NN to store the analog pattern [0.1 0.3 0.9 0.1 0.3 0.5 0.9 0.1] as a fixed attractor for the network. One of the possible applications of such a problem is the recognition of the normalized gray levels of some texture images. Hopfield recurrent training algorithm cannot be used in this case as the patterns are not binary. Further analysis of this problem by RNN is given in [1].

- **The XOR problem**: this problem is widely used in the literature by many researchers to evaluate the performance of the proposed training algorithms. This is because the binary-XOR problem is very hard for NN to learn. Here, the NN architecture we use for this problem is shown in Figure 10.3.

We have implemented the proposed algorithms in MATLAB. During their evaluation, we have noticed that when we allow negative values for the weights, we obtain better results. Thus, to deal with this point, we have considered three different procedures when a negative weight is produced. In the first one, whenever there is a negative value of the weight, we simply put it to zero. The second one is as proposed in [87]. To ensure that the weights are always positive, the relation $p^2 = w_{i,j}$ is used. In this case, the above equations are slightly modified to take into account the derivatives with

respect to the weights. The third case is to allow negative weights. We provide a comparative study for these cases. In the following analysis, Figures and Tables, we use the acronyms given bellow:

GD: the basic gradient descent training algorithm, as proposed in [1].

LM: the Levenberg-Marquardt training algorithm; in this case, we allow negative weights to take place during the training.

LM1: the Levenberg-Marquardt training algorithm, but in this case, no negative weights are allowed: whenever a negative value of any weight is produced, we simply put it to zero.

LM2: the Levenberg-Marquardt training algorithm, no negative weights allowed, by using $p^2 = w_{i,j}$ and by modifying the derivatives of the equations.

AM-LM: the Levenberg-Marquardt with adaptive momentum training algorithm. We allow negative weights to occur.

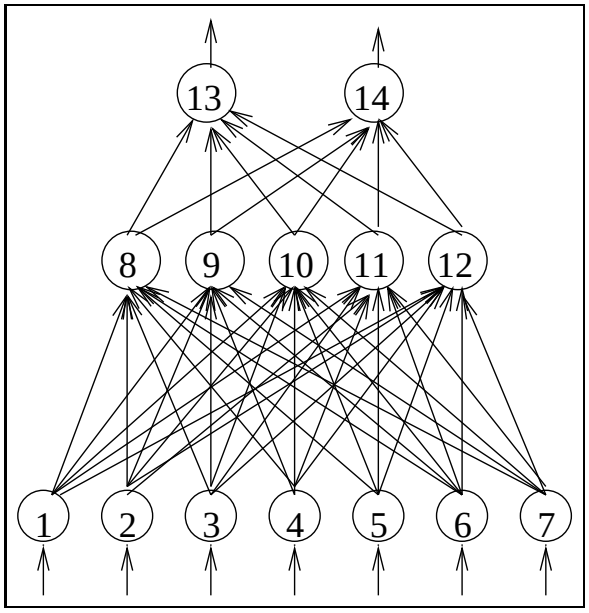


Figure 10.1: The 7-5-2 feedforward RNN network architecture.

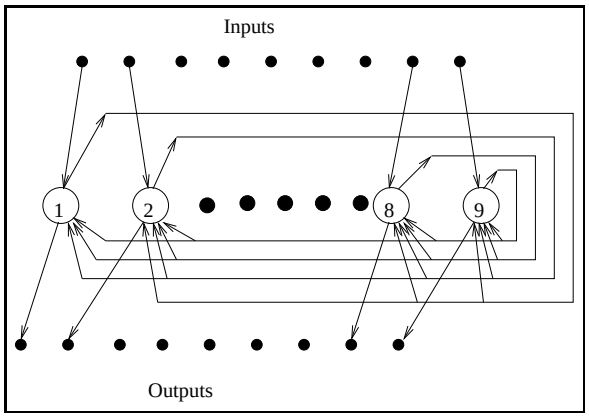


Figure 10.2: The fully-connected recurrent RNN network architecture.

10.5.1 ζ and dP Parameters for AM-LM

In the GD and other learning algorithms, the value of the learning rate η has to be chosen. The convergence time and the accuracy depend on this choice. The optimal value of this parameter varies from one problem to another and from network architecture to another; however, most of the implementations set its value to 0.1.

The AM-LM has two parameters (ζ and dP) to be determined, instead of one. In [87], the authors studied this problem and they determined through simulations "ideal" ranges for these parameters: $\zeta \in [0.85\,, 0.95]$ and $dP \in [0.1\,, 0.6]$. We will show the performance of this algorithm when adapted to RNN for the first problem and for these ranges. To do that, for each pair of values, we counted the number of times the algorithm converges to a predefined mean square error MSE value (5×10^{-5}). For each case, the total number of tests is fixed to 100. We show in Figure 10.4 the results for the first problem. As we can see from these Figures, the values of these parameters effectively affect the

performance of the algorithm. The best range for dP was [0.4, 0.6]. However, the impact of ζ appears to be random and not easy to characterize. In addition, we can note that the impact of dP is more significant than that of ζ. Thus, we propose that this parameter must be fixed to 0.90 and that only dP must be changed.

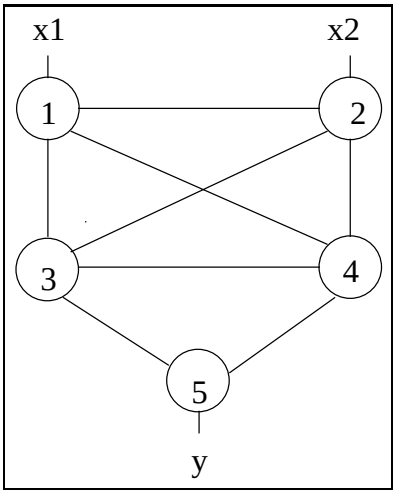


Figure 10.3: The RNN network architecture used to solve the XOR problem.

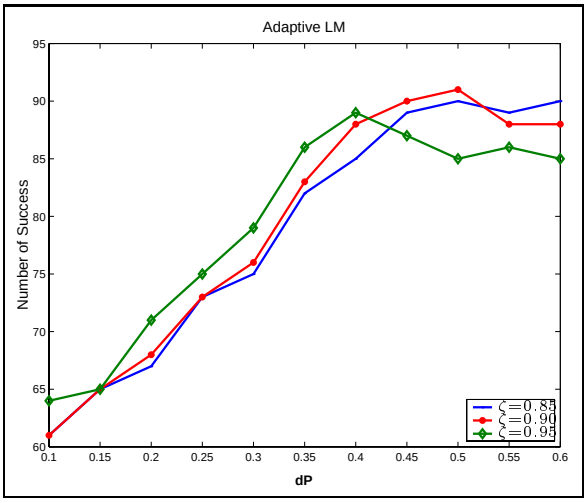


Figure 10.4: The impact of the two variables ζ and dP on the performance of the adaptive momentum LM training algorithm for RNN. The results for the first problem.

10.5.2 Algorithms' Performance Comparison

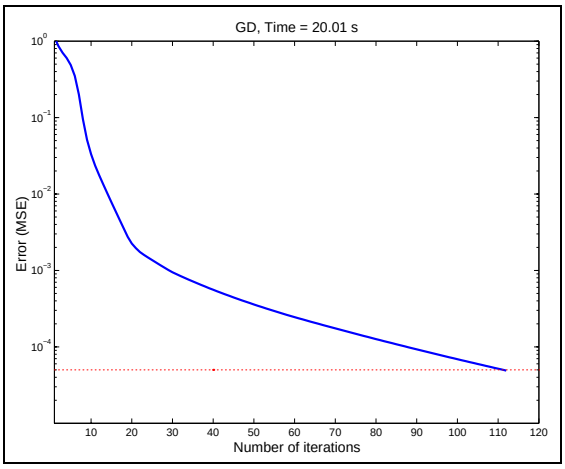
We compare here the performance of the different training algorithms (GD, LM, LM1, LM2 and AM-LM) for RNN. For the two previously defined problems, we test the convergence time and the number of iterations needed to reach some fixed MSE value. The MSE goals are 5×10^{-5} and 4×10^{-7} for the first and second problems respectively. For the GD algorithm, we set $\eta = 0.1$. For the LM algorithm, we fixed β to 2 and the initial value of μ to 0.1. For AM-LM, the values of ζ and dP are fixed to 0.90 and 0.5 respectively. We show in the set of Figures 10.5 and in that of Figures 10.6 the variation of MSE against the number of iterations as well as the total time taken to converge to the MSE goal for the first and second problems respectively.

From Figures 10.5(a) and 10.5(b), we can see that LM considerably outperforms GD. The latter takes 20.01 sec. in 112 iterations against 1.2 sec. in 9 iterations for the former to reach the same error level. Thus, LM requires about 8 times less to converge. For the second problem, as shown from Figures 10.6(a) and 10.6(b), GD takes 38 sec. and 270 iterations against 0.87 sec. and 7 iterations for LM. In this problem LM is 43 times faster than GD.

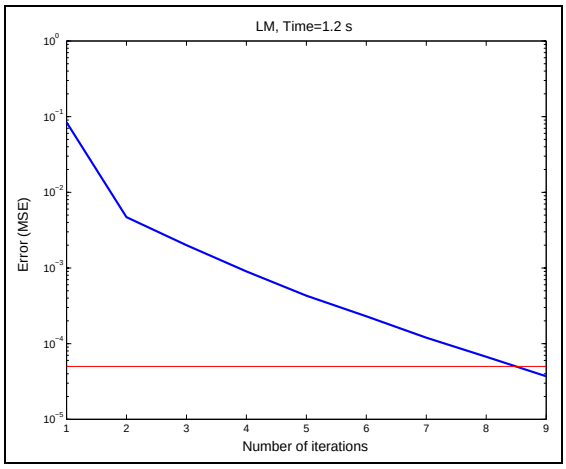
Keeping the weights non-negative degrades the performance of the training algorithms for RNN. This can be shown from Figures 10.5(b), 10.5(c) for the first problem, and Figures 10.6(b), 10.6(c), 10.6(d) for the second problem. As we can see, LM performs better than both LM1 and LM2. In addition, we can see that LM1 performs better than LM2. For the first problem, LM1 does not converge. This is because most of the weights are zeroed and hence the Hessian matrix becomes singular. However, in general LM and LM2 perform better than GD. It should be mentioned that RNNs (as ANNs) are to be used as black boxes once they are correctly trained. The most important point is that the black box behaves correctly and efficiently instead of looking into the internal implementation. Hence, we suggest that the non-negative weights constraint should not be used as it greatly degrades the performance of the training algorithms. Of course, the price to pay is that the nice probabilistic interpretation is lost.

From Figure 10.5(d), we can see that AM-LM outperforms all the other training algorithms. It is about 3 times faster than LM and 52 times faster than GD for the first problem. In addition, we can see that in three iterations, it converged to 10^{-7} against 5×10^{-5} for the other training algorithms for the first problem. As we will see next for the first problem, LM can converge in only 2 steps and takes only 0.22 sec. Moreover, AM-LM can converge in only one step and takes 0.14 sec. In addition, in some cases the two algorithms may reach absolute zero error.

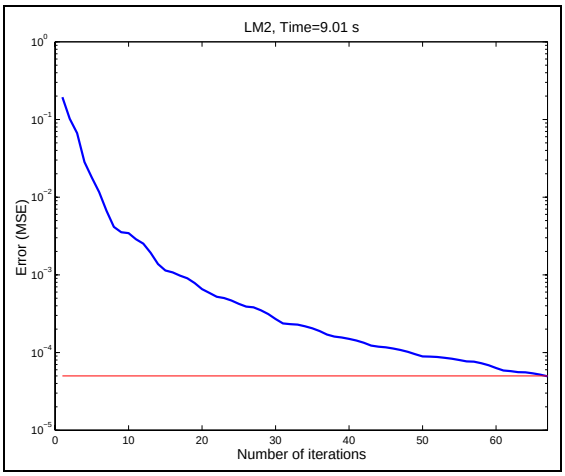
The XOR problem is solved by the AM-LM algorithm. Depending on the initialization of the weights, it takes from 1 to 8 iterations to converge. In some cases, it converges to zero error. Using GD, it takes 2140 iterations to reach 5×10^{-5} MSE [87].



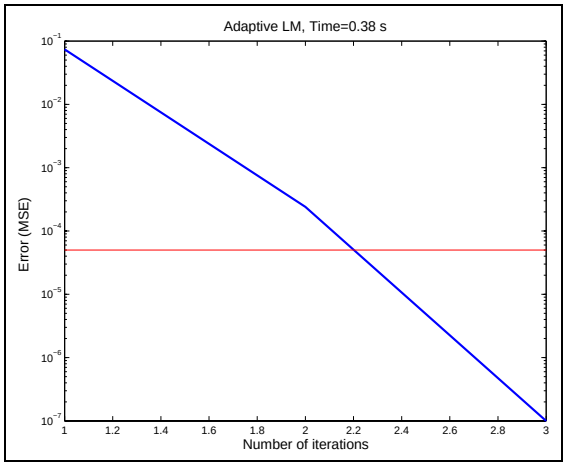
(a) GD training algorithm



(b) LM training algorithm



(c) LM2 training algorithm

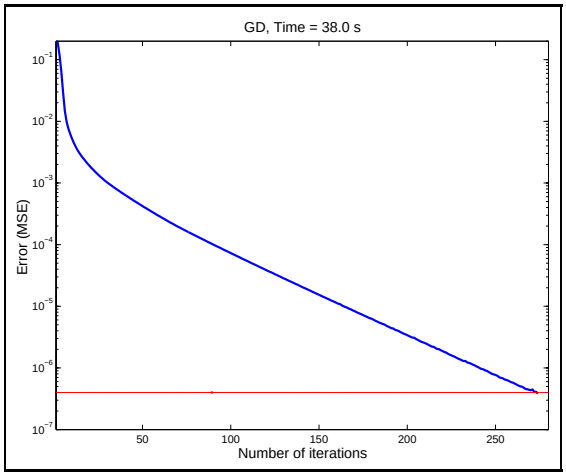


(d) AM-LM training algorithm

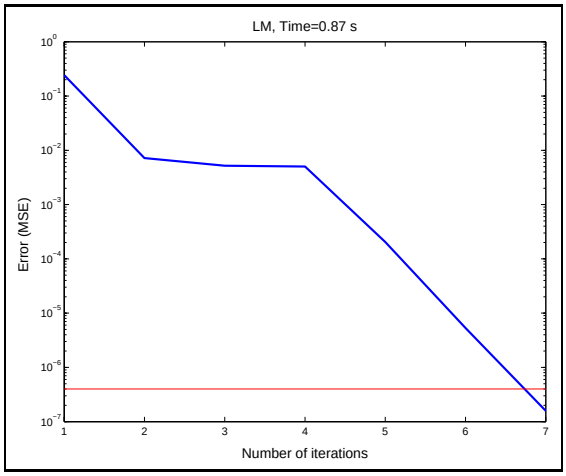
Figure 10.5: The performance of the GD, LM, LM2 and AM-LM training algorithms on the first problem.

10.5.3 Testing the Success Rate and the Performance

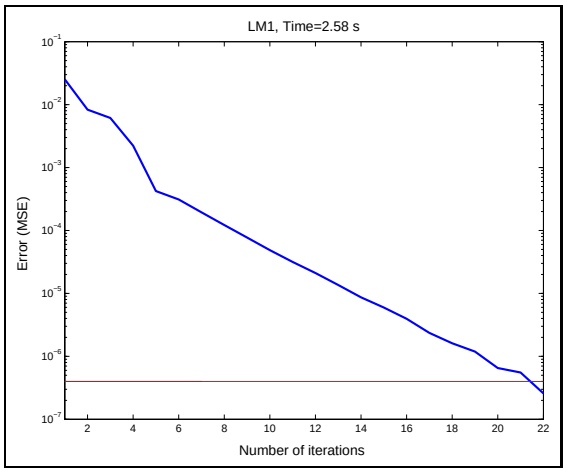
We have carried out two experiments to test the success rate and the performance of the training algorithms. To do that, for each training algorithm, we have fixed a MSE goal of 5×10^{-5} and



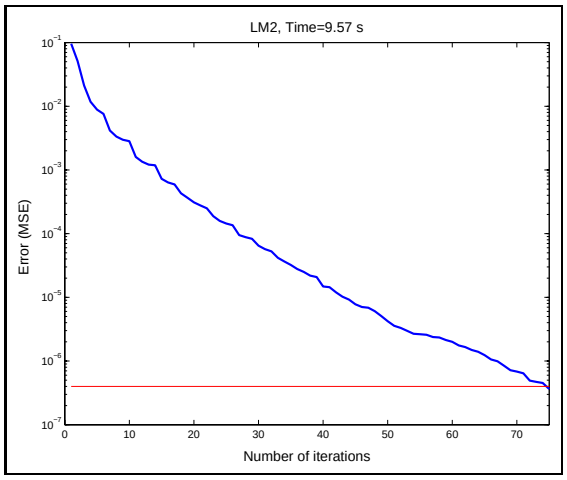
(a) GD training algorithm



(b) LM training algorithm



(c) LM1 training algorithm



(d) LM2 training algorithm

Figure 10.6: The performance of the GD, LM, LM1, and LM2 training algorithms on the second problem.

4×10^{-7} for the first and second problems respectively. It is known that the convergence of the NN training algorithms depends on the initial values of the weights, which are typically initialized randomly. We have thus repeated the same experiment for each problem and each training algorithm 100 times. We have limited the maximum number of iterations to 100. For each case, the random seed generator is fixed to the index of the test (from 1 to 100). This is to limit the biasing of the initialization on the comparison. The weights are initialized in the range $0 \ldots 0.1$. Each time the algorithm converges to the specified error, the delay and current error are stored. For each algorithm in the first and second problems, the success rate, the number of iterations statistics, the required time statistics, and the reached MSE statistics (the minimum, the maximum, the mean and the standard deviation) are calculated and reported. The results are shown in Tables 10.1 and 10.2 for the first and second problems respectively. For the AM-LM algorithm, we set $dP = 0.6$ and $\zeta = 0.90$.

As we can see from Table 10.1, LM and GD give approximately the same success rate, but, LM converges in less iterations and time. The minimum and maximum time needed for LM are 0.33 and 3.58 with a mean of 1.15 sec. against 2.92 and 13.94 with a mean of 8.63 sec. for GD. However,

when applying the non-negative weights constraint on LM, as shown in column LM2, the performance degrades too much. With respect to the AM-LM algorithm, it gives the best results among all the other algorithms. The success rate is 99. In addition, the required time and iterations are the best. The mean time needed is 0.67 against 8.63, 1.15, 14.25 sec. for GD, LM and LM2 respectively. We can note also that AM-LM can converge in only one iteration against 21, 2 and 48 iterations for GD, LM and LM2 respectively. Regarding the error, some times LM and AM-LM can converge to 0 error.

It should be mentioned that AM-LM does not work well for the fully connected RNN network (as the case of the second problem). Thus, we do not provide the results in this case. We can note from Table 10.2 that LM converges all the times (100 successes). The convergence for GD is very poor, only 22 success rate. In addition, for LM without non-negative weights constraints, the results are better than all the other cases (LM1 and LM2). However, the performance is better than that of GD. LM can converge in only 5 iterations against 64, 9 and 38 iterations for GD, LM1 and LM2 respectively. Moreover, LM takes less time to converge (about 1.05 sec. in average against 9.56. 3.78 and 9.40 sec. for GD, LM1 and LM2 respectively). Similarly, the error statistics are minimal for LM.

We can notice that both LM and AM-LM, when they converge, they do not take too much iterations (6 and 9 iterations for the first and second problems respectively). This can be used to improve the success rate and hence the performance by restarting the training using different weights initialization.

Table 10.1: Comparison between GD, LM, LM2 and AM-LM for the first problem.

	GD	LM	LM2	AM-LM
Success	81	83	50	99
Min(iterations)	21	2	48	1
Max(iterations)	97	24	100	6
Mean(iterations)	60.27	7.14	82.52	2.83
Std(iterations)	21.26	4.85	12.01	0.95
Min(time)(s)	2.92	0.33	8.4	0.22
Max(time)(s)	13.94	3.58	17.57	1.70
Mean(time)(s)	8.63	1.15	14.25	0.67
Std(time)	3.06	0.72	3.06	0.31
Min(error)	3.0537×10^{-5}	0	4.7736×10^{-5}	0
Max(error)	4.9987×10^{-5}	4.9293×10^{-5}	4.9991×10^{-5}	3.8730×10^{-5}
Mean(error)	4.8643×10^{-5}	2.4402×10^{-5}	4.9153×10^{-5}	1.0993×10^{-6}
Std(error)	2.2703×10^{-6}	1.7641×10^{-5}	5.6558×10^{-7}	5.0233×10^{-5}

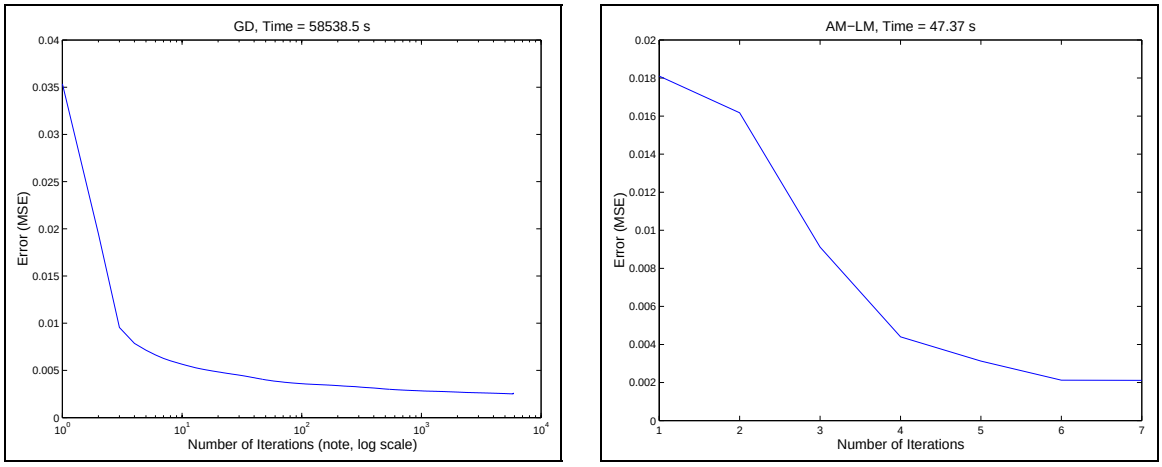
Table 10.2: Comparison between GD, LM, LM1 and LM2 for the second problem.

	GD	LM	LM1	LM2
Success	22	100	95	84
Min(iterations)	64	5	9	38
Max(iterations)	100	9	34	78
Mean(iterations)	86.45	5.48	18.51	56.32
Std(iterations)	11.76	0.88	4.93	9.27
Min(time)(s)	5.84	0.54	0.94	4.35
Max(time)(s)	9.56	1.05	3.78	9.40
Mean(time)(s)	8.14	0.67	2.05	6.69
Std(time)	1.12	0.10	0.53	1.17
Min(error)	3.3503×10^{-7}	4.6590×10^{-9}	5.3220×10^{-8}	2.8499×10^{-7}
Max(error)	3.9810×10^{-7}	3.8989×10^{-7}	3.9886×10^{-7}	3.9001×10^{-7}
Mean(error)	3.7638×10^{-7}	1.1062×10^{-7}	3.1342×10^{-7}	3.6046×10^{-7}
Std(error)	1.8052×10^{-8}	8.2365×10^{-8}	6.5777×10^{-8}	2.5932×10^{-8}

10.5.4 Learning Performance in the Video Quality Problem

Here, we give the results of the GD and AM-LM to learn the problem of video quality assessment presented in Chapter 6. The training database contains 80 samples (each is composed of 5 inputs and one output). The topology of the RNN is the three-layer feedforward architecture having 5 neurons in the input layer, 5 hidden neurons and one output neuron. We trained this network using the two algorithms; the stop criterion was MSE=0.0025. For GD, we used $\eta = 0.1$ and for AM-LM, we used $dP = 0.7$ and $\zeta = 0.90$.

We depict in Figure 10.7 the variation of the error with respect to the number of iterations. As we can see, AM-LM gives better performance than GD in terms of speed: it takes only 7 iterations in 47.37 sec., while GD reaches the same error after 5870 iterations in 58538.5 sec.



(a) GD training algorithm

(b) AM-LM training algorithm

Figure 10.7: Comparsion between the performance of GD and that of AM-LM on the video quality database presented in Chapter 6.

10.6 Conclusions

In this Chapter, we have proposed two new training algorithms for RNN. The first one is based on the Levenberg-Marquardt method for ANN. This algorithm is one of the most powerful training algorithms in ANN in terms of performance (time required to converge, precision, number of iteration, etc.) for a given problem. The second one is an improvement of the Levenberg-Marquardt method. It introduces the adaptive momentum to further accelerate and enhance the training process by adding the information of the second order derivatives to the cost function. We have studied these two algorithms and we have provided the steps and mathematical derivation for them to be used with RNN.

We have evaluated the performance of these algorithms against the most widely available one for RNN, namely the gradient descent one. The experimental results clearly show the improvement and efficiency obtained by the new techniques.

Our work in this area has been motivated by two reasons. First, as explained in Chapter 6, RNN behaves better in our problems (including the multimedia quality assessment) than ANN. Second, the available training algorithm for RNN, namely the gradient descent which is very general, can, in some cases, converge very slowly to the solution. In addition, it suffers from the zigzag problem. ANNs have been studied intensively and there are many training algorithms available for them. Naturally,

we have chosen the most powerful algorithms, to the best of our knowledge, for ANN and we adapted them to RNN.

However, it is known that Levenberg-Marquardt methods can converge to local minima in some cases depending on the initialization of the weights. One simple solution to this problem is to restart the training process whenever a local minimum convergence has taken place but by initializing the weights differently. As shown by our results, when the proposed algorithms converge to the global minimum, they converge very fast. For example, it takes only one or two iterations to reach 0 error in the XOR problem. More sophisticated solutions can be studied in the future, for instance combining the Levenberg-Marquardt methods with the "simulating annealing" technique which is based on a well-studied mathematical model. Another point that deserves attention in future research work is the study of the best ranges of the learning parameters (namely, μ, β, ζ and dP), etc.

General Conclusions of this Dissertation

Summary of the Main Contributions

The work presented in this dissertation can be summarized as follows. We have proposed a methodology to evaluate automatically the quality of real-time multimedia streams[2], taking into account the fact that there are many factors affecting the quality, including encoding impairments and the distortion due to the transmission over packet networks. The goal of this method is to overcome the limitations of the available quality measuring techniques in the literature. The advantages of our method are: (i) the results obtained correlate well with human perception as our procedure is partially built using subjective data; (ii) it is not computationally intensive, as a simple neural network (NN) is used to measure the quality; (iii) there is no need to access the original multimedia signals (before encoding and transmission), which serve only during the development phase to identify some objective factors that are used to allow the NN to learn the relation between the original and processed signals based on these objective factors; (iv) many factors that cannot be taken into account by the traditional methods can be easily included (some examples, among many others, are the frame rate and the use of FEC).

We used our method to evaluate real-time speech quality by taking into account the following parameters: packet loss rate, loss distortion, packetization interval, several encoding speech codecs and the spoken language effect. We showed that, once trained, the NN can evaluate accurately the quality of other cases (not considered during the training phase). We have shown that the NN can learn quite well the way a group of human subjects, participating in subjective quality tests, evaluates speech quality. In this way, the trained NN reacts similarly as human subjects for other samples. The results obtained correlate very well with those obtained by subjective tests.

We followed the same approach to evaluate real-time video quality. The considered parameters are: packet loss rate, loss distribution, bit rate, frame rate, and encoding type for the H.263 codec. The application of our technique to video streams led to similar results as in the case of speech.

The aforementioned characteristics of our methodology make it possible to use it in real time without significant processing overhead. The fact that the quality can be measured, with great confidence, without having access to the original signals, makes it possible to develop new communication protocols, which could make use of the automatic evaluation of the end-user's perception of multimedia quality as a function of objective measurable parameters like the loss rate, the frame rate, etc.

As an example of such protocols, we designed a new rate control mechanism. This mechanism combines an automated real-time multimedia quality assessment with a TCP-Friendly rate controller. It helps in delivering the best multimedia quality and in saving superfluous bandwidth for a given network situation (by determining the exact sending rate to be used, instead of just giving an upper bound). The controller decides, based on the quality (measured by the trained neural network) and on the network conditions (TCP-Friendly rate controller suggestions), which parameters should be

[2]Our technique can also be used as a batch tool, of course.

modified and how to achieve this task.

Before implementing such a protocol, it is necessary to understand the impact of the parameters on multimedia quality. The requirements are that the quality must correlate well with human perception and that the analysis must include the combined effect of all the considered quality-affecting parameters at a time. There are two possible solutions to satisfy them: either using subjective quality tests or using a good objective measure that takes into account the direct effect of a wide range variation of these parameters. Unfortunately, the first solution is not practical as it is very expensive to carry out and highly time consuming. To reach a minimum precision level, a huge number of evaluations is necessary (for example, in the video case, to consider 5 parameters, about 2000 evaluations need to be done). Regarding the second solution, there is no previously objective measure that satisfies both requirements.

Our method can be used to solve this two-part problem. Thus, always for video problem, to study the impact of the 5 selected parameters with our tool, we obtained good results using 80 subjective evaluations and another 14 to cross-validate the NN. The trained NN allows us to evaluate the quality for wide range variations of the considered parameters. We followed this approach to study and analyze the quality for speech and video. The considered parameters are those previously mentioned for both cases.

In our work, we used two types of NN, namely Artificial NN (ANN) and Random NN (RNN). We strongly recommend the use of RNN as they offer some advantages in these applications. This claim is based on the numerous experiments we did in order to compare the performance of these tools. Among these advantages we can mention the following: (i) RNN can capture more precisely the nonlinear relation between the quality and the affecting parameters; (ii) RNN do not suffer too much from the problem of overtraining; (iii) in the run-time phase, RNN are much faster than ANN for a given architecture.

However, there is one drawback of RNN, which is the available training algorithm (gradient descent), which can be quite slow to converge to the solution or to a minimal precision. For example, in the case of video, to reach a mean square error of 0.0025, a typical learning phase takes about 5000 iterations in 58538.5 sec. (about 16 hours in a standard workstation).

This problem motivated us to explore new learning techniques for RNN. This led us to propose two new training algorithms. The first one is based on the Levenberg-Marquardt method for ANN. This algorithm is one of the most powerful for ANN in terms of performance (time required to converge, precision, number of iterations, etc.). The second one is an improvement over the general Levenberg-Marquardt method for the case of feedforward networks. It introduces the adaptive momentum to further accelerate and enhance the training process by adding the information of the second order derivatives in the cost function. We have studied these two algorithms and we have provided the steps and mathematical derivations necessary to use them with RNN. We have shown that these two algorithms outperform the available gradient descent one. For the video problem discussed in this document and the same architecture, the second proposed algorithm reaches the error goal in only 7 iterations and takes only 47.37 sec. We have evaluated their performances for other problems and also provided some variants of both techniques.

We have also presented a new model for traffic prediction that makes use of both the long-range and short-range periodicity of the traffic process to provide more accurate results. The existing models concentrate only on the short-range information while neglecting the long-range information, which is very important to characterize the traffic process more precisely.

Possible Extensions

We describe here some of the possible extensions of the work presented in this dissertation.

- **Multimedia Quality Assessment:** As multimedia quality is affected (when encoded and

transmitted in real time using packet networks) by a large number of parameters, one of the directions in future research is to build a robust database, by conducting a series of MOS experiments taking into account different combinations of these parameters and to build a generic tool that may be used in different applications. A generic tool can be built to evaluate the quality of multimedia (audio + video) taking into account several codecs and parameters.

- **Understanding the Parameters' Impact:** Some future research directions for this work include the study of other codecs like MPEG2/4 and the analysis of the effect of audio and video synchronization. The combined effects of other parameters can be studied by following the approach presented in this dissertation. Network loss is one of the most annoying factors on multimedia quality, and there are several techniques proposed to tackle this problem. A general solution is the use of Forward Error Correction (FEC). There are also some error concealment techniques depending on the media type (for example, waveform substitution for speech or linear interpolation for video). It is interesting to study the impact of these different techniques on human perception.

- **Network Technologies:** Our work focused on IP networks, but it can also be applied to other network technologies including ATM and wireless networks. ATM networks can guarantee some levels of QoS (e.g. low loss and better bandwidth availability). Wireless networks technology evolves and spreads rapidly in these days, with emphasis on real-time multimedia applications, including videophone teleconferencing and video streaming. Thus, it will be interesting to further investigate the possibilities of adapting our methods to these networks in the future. This concerns the following points: the real-time multimedia quality assessment, the study of the impact of the quality-affecting parameters on the quality, the rate control protocol, and the traffic prediction methodology.

- **Encoder Design:** A method called Perception Based Spectrum Distortion (PBSD) is proposed in [161]. A very important issue in that work is that the authors used their metric instead of MSE to redesign the ADPCM speech encoder in order to set the quantization level. This led to an improvement in the subjective quality over the normal ADPCM codec. Similarly, there is ongoing research in video encoding, focusing on finding other metrics that give better results than PSNR to be used in encoding design. A preliminary result [98] shows good improvements. But as we have shown, most of the available objective measures have several limitations (mainly, their high computational cost and the need to access the original signal). Thus, they cannot be used in real time to improve the encoding based on human perception. We believe that our method is a good candidate to do such work. In addition, it can take into account the impact of network distortion.

- **Rate Control:** We used, with our rate control, the equation-based TCP-Friendly congestion control protocol that has some good properties for real-time multimedia applications. It is important to investigate the use of other protocols jointly with ours. We provided a list of the possible controlling parameters to be used with our proposal. By using these parameters, we believe that the quality can be improved. The effect of the other parameters and building a complete controller are some possible future research directions.

- **Traffic Prediction Model:** Based on this model, many applications can be implemented. Dynamic bandwidth allocation, dynamic long-term contract negotiation, pricing, traffic shaping and engineering, are some examples.

- **Random Neural Network Training:** It is known that Levenberg-Marquardt methods can converge to local minima in some cases depending on the initialization of the weights. One simple solution to this problem is to restart the training process whenever a local minimum convergence

has taken place, but by initializing the weights differently. As shown by our results, when the proposed algorithms converge to the global minimum, they converge very fast. For example, it takes only one or two iterations to reach 0 error in the XOR problem. More sophisticated solutions can be studied in the future, for instance combining the Levenberg-Marquardt methods with the "simulating annealing" technique which is based on a well-studied mathematical model. Another point that deserves attention in future research work is the study of the best ranges of the learning parameters (namely, μ, β, ζ and dP).

Appendix

We show in the following list all the available commandline options for the H.263 encoder, which we used to generate the distorted video sequences.

```
TMN (H.263) coder version 3.0, University of British Columbia, CANADA,
based on Telenor's coder version 2.0, Copyright (C) 1995, 1996 Telenor
R&D, Norway

Encoding format: QCIF (176x144)

Usage:  mn [options] -i <filename> [more options]
Options:
   -i <filename> original sequence [required parameter]
   -o <filename> reconstructed frames [./out.raw]
   -B <filename> filename for bitstream [./stream.263]
   -a <n> image to start at [0]
   -b <n> image to stop at [0]
   -x <n> (<pels> <lines>) coding format [2]
      n=1: SQCIF n=2: QCIF n=3: CIF n=4: 4CIF n=5: 16CIF
      n=6: Custom (12:11 PAR)
          128x96   176x144    352x288   704x576   1408x1152
          pels x lines
   -s <n> (0..15) integer pel search window [15]
   -q <n> (1..31) quantization parameter QP [13]
   -A <n> (1..31) QP for first frame [13]
   -r <n> target bitrate in bits/s, default is variable bitrate
   -ren <n> target bitrate in bits/s, in the enhancement layer
   -rfi RATE_FILE target bitrates contained in the file RATE_FILE
   -C <n> Rate control method [3]
   -k <n> frames to skip between each encoded frame [2]
   -Z <n> reference frame rate (25 or 30 fps) [30.0]
   -l <n> frames skipped in original compared to reference
      frame rate [0]
   -e <n> original sequence has n bytes header [0]
   -g <n> insert sync after each n GOB (slice) [0]
      zero above means no extra syncs inserted
   -w write difference image to file "./diff.raw" [OFF]
   -m write repeated reconstructed frames to disk [OFF]
   -t write trace to tracefile trace.intra/trace [OFF]
   -j <n> force an Intra MB refresh rate every <n> macroblocks [0]
   -D <n> use unrestricted motion vector mode (annex D) [OFF]
      n=1: H.263 n=2: H.263+ n=3: H.263+ unlimited range
   -E use syntax-based arithmetic coding (annex E) [OFF]
   -F use advanced prediction mode (annex F) [OFF]
   -G use PB-frames (annex G) [OFF]
   -U <n> (0..3) BQUANT parameter [2]
   -M use improved PB-frames (annex M) [OFF]
   -I use advanced intra coding mode (annex I) [OFF]
   -J use deblocking filter (annex J) [OFF]
```

```
-c <n> frames to select number of true B pictures
   between P pictures (annex O) [0]
-d <n> to set QP for true B pictures (annex O) [13]
-i <filename> enhancement layer sequence
-u <n> to select SNR or spatial scalability mode (annex O) [OFF]
   n=1: SNR n=3: SPATIAL(horiz) n=5: SPATIAL(vert) n=7: SPATIAL(both)
-v <n> to set QP for enhancement layer (annex O) [13]
-S use alternative inter vlc mode (annex S) [OFF]
-T use modified quantization mode (annex T) [OFF]
-p <n> a I picture every n frames
-CR <n> Conditionnal Replenishment With treshold: n
-MS Mode and QP Selection (base layer)
-MSUI Mode and QP Selection (base layer): only INTRA&UNCODED Modes
-MSEP Mode and QP Selection (enhancement layer)
-FOR FORWARD Mode forced (with -MSEP)
-FUP UPWARD Mode forced (with -MSEP)
-PG parameter P of the Gilbert Model
-QG parameter Q of the Gilbert Model
-ARQ <filename>  Prise en compte des infos de Feedback
-PL <filename>  Sortie des limites de paquets dans
   un fichier ASCII
-PLEL <filename>  Sortie des limites de paquets SNR_EL
      dans un fichier ASCII
-RF <filename>  Sortie des debits instantannes utilisés
-EL <filename>  Sortie de la layer de raffinement SNR
    dans le fichier filename
-vi Utilisation de l'algo de Viterbi pour l'optimisation R-D
-L <nb_loops> Bouclage sur la sequence par nb_loops aller-retours
-sk [filename] Write the number of the skipped frames in the
    file filename (default : skip.txt)
-aa <filename> Write the results of intra MB of each packet
    in a file [Mos_file]
-ab n the default packet size in byte <536>
-h Prints help

 Default filenames and other options in square brackets
 are chosen in config.h
```

The commandline options of the H.263 decoder, which we used during the subjective video quality test and in the video demonstration application:

```
Usage:   tmndecode options bitstream outputfilename

Options:
   -vn  verbose output (n: level)
   -on  output format
      n=0 : YUV
      n=1 : SIF
      n=2 : TGA
      n=3 : PPM
      n=4 : X11 Display
      n=5 : YUV concatenated
      You have to choose one output format!
   -q disable warnings to stderr
   -r use double precision reference IDCT
   -t enable low level tracing
   -s <filename>  output reconstructed frame to filename
      (YUV concatenated)
   -p enable tmn-8 post filter
   -c enable error concealment
   -x interpolate pictures to double size before display
   -fn  frame rate
      n=0 : as fast as possible
      n=99: read frame rate from bitstream (default)
   -l loop sequence
   -TR enable Absolute Temporal References trace
   -TREP enable Absolute Temporal References trace
   -E enhancement_file: decode SNR-ENH layer in file
      enhancement_file
   -EF enh_recon_sequence: reconstruct SNR-ENH layer in file
       enh_recon_sequence
   -PL base.lim: specifie le fichier de limites de paquets de
       la couche de base
   -PLEL enh.lim: fichier de limites de paquets de la couche
         de raffinement
   -A arq.txt: fichier de numeros de paquets perdus
      (couche de base)
   -AE enh_arq.txt: fichier de numeros de paquets perdus
       (couche de raffienement)
  -AM <Mos_file> is a must file name that contains the
      #intra and inter MBs in each packet
  -Znn % loss rate <00>
  -Ynn no. of consecutively lost packets <1>
```

Bibliography

[1] H. Abdelbaki. Random neural network simulator. `ftp://ftp.mathworks.com/pub/contrib/v5/nnet/rnnsimv2/`.

[2] H. Abdelbaki, E. Gelenbe, and S.E. El-Khamy. Random neural network decoder for error correcting codes. In *International Joint Conference on Neural Networks – IJCNN*, volume 5, pages 3241–3245, 1999.

[3] H. Abdelbaki, E. Gelenbe, and S.E. EL-Khamy. Analog hardware implementation of the random neural network model. In *Proceedings of International Joint Conference on Neural Networks - IJCNN*, volume 4, pages 197–201, 2000.

[4] H. Abdelbaki, E. Gelenbe, T. Kocak, and Said E. El-Khamy. Random neural network filter for land mine detection. In *Proceedings of Sixteenth National Radio Science Conference – NRSC*, 1999.

[5] ACIRI. Equation-based congestion control. `http://www.aciri.org/tfrc/code`.

[6] J. Aguilar and A. Colmenares. Recognition algorithm using evolutionary learning on the random neural networks. In *International Conference on Neural Networks*, volume 2, pages 1023–1028, 1997.

[7] J. Aguilar and V. Rossell. Multiple classes random neural network model and color pattern recognition problems. In *Proceedings of International Joint Conference on Neural Networks - IJCNN*, volume 5, pages 503–508, 2000.

[8] N. Ampazis and S. J. Perantonis. Levenberg-marquardt algorithm with adaptive momentum for efficient training for feed forward networks. In *International Joint Conference on Neural Networks – IJCNN*, Italy, 2000.

[9] I. Anttila. Transferring real-time video on the Internet. In *Proceedings of HUT Internetworking Seminar*, May 1997.

[10] A. Aussem. Call admission control in ATM networks with random neural network. In *IEEE International Conference on Neural Networks*, pages 2482–2487, 1994.

[11] F. Babich and M. Vitez. A novel wide-bande audio transmission scheme over the Internet with a smooth quality degradation. *Proceedings of ACM SIGCOMM Computer Communication Review*, 30(1), Jan. 2000.

[12] H. Bakircioglu and T. Kocak. Survey of random neural network applications. *European Journal of Operational Research 126*, 2000.

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

[14] J. Beerends and J. Stemerdink. A perceptual audio quality measure based on a psychoacoustic sound representation. *Journal of Audio Eng. Soc.*, 40:963–978, September 1992.

[15] 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.

[16] J-C. Bolot. Characterizing end-to-end packet delay and loss in the Internet. *Journal of High-Speed Networks*, 2(3), December 1993.

[17] J-C. Bolot. End-to-end packet delay and loss behavior in the Internet. In *Proceedings of ACM SIGCOMM*, pages 289–298, San Fransisco, 1993.

[18] J-C. Bolot, S. Fosse-Parisis, and D. Towsley. Adaptive FEC-based error control for Internet telephony. In *IEEE INFOCOM*, New York, March 1999.

[19] J-C. Bolot and A.V. Garcia. The case for FEC-based error control for packet audio in the Internet. In *ACM Multimedia Systems*, 1996.

[20] J-C. Bolot and T. Turletti. Adaptive error control for packet video in the Internet. In *Proc. ICIP*, Lausanne, Sept. 1996.

[21] J-C. Bolot and T. Turletti. Experience with rate control mechanisms for packet video in the Internet. *Computer Communication Review*, Jan. 1998.

[22] J-C. Bolot and A. Vega Garcia. Control mechanisms for packet audio in the Internet. In *IEEE INFOCOM*, San Fransisco, CA, March 1996.

[23] J.M. Boyce and R.D. Gaglianello. Packet loss effects on MPEG video sent over the public Internet. In *Proceedings of ACM Multimedia*, 1998.

[24] M. Caramma, R. Lancini, and M. Marconi. Subjective quality evaluation of video sequences by using motion information. In *Proceedings of International Conference on Image Processing: ICIP*, Kobe, Japan, Oct. 1999.

[25] C. Cerkez, I. Aybay, and U. Halici. A digital neuron realization for the random neural network model. In *International Conference on Neural Networks*, pages 1000–1004, 1997.

[26] S. Chang, H. Choa, and X. Guo. TCP-friendly window congestion control with dynamic grouping for reliable multicast. In *IEEE GLOBECOM*, pages 538–547, 2000.

[27] B-S. Chen, S-C. Peng, and K-C. Wang. Traffic modeling, prediction, and congestion control for high-speed networks: A fuzzy AR approach. *IEEE Transactions on Fuzzy Systems*, 8(5):491–507, 2000.

[28] 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, 1989.

[29] M. Claypool and J. Tanner. The effects of jitter on the perceptual quality of video. In *Proceedings of ACM Multimedia Conference*, 1999.

[30] C. Cramer, E. Gelenbe, and H. Bakircioglu. Low bit rate video compression with neural networks and temporal subsampling. In *Proceedings of IEEE Special issue on Neural Networks*, volume 84, pages 1529–1543, October 1996.

[31] C. Cramer, E. Gelenbe, and P. Gelenbe. Image and video compression. *IEEE Potentials*, 1998.

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

[33] P. Cuenca, L. Orozco-Barbosa, A. Garrido, and F. Quiles. Study of video quality metrics for MPEG-2 based video communications. In *IEEE Pacific Rim Conference on Communications Computers and Signal Processing*, pages 280–283, 1999.

[34] P. Cuenca, L. Orozco-Barbosa, F.J. Quiles, and A. Garrido. Loss-resilient ATM protocol architecture for MPEG-2 video communications. *IEEE Journal on Selected Areas in Communications*, 18(6):1075–1086, June 2000.

[35] R. De Alengar Lotufo, W.D.F. Da Silva, A.X. Falcao, and A.C.F. Pessoa. Morphological image segmentation applied to video quality assessment. In *Proceedings of International Symposium on Computer Graphics, Image Processing, and Vision - SIBGRAPI*, pages 468–475, 1998.

[36] D. De Vleeschauwer, J. Janssen, and G. H. Petit. Delay bounds for low bit rate voice transport over IP networks. In *Proceedings of SPIE Conference on Performance and Control of Network Systems III*, volume 3841, pages 40–48, Sept. 1999.

[37] 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 IFIP-TC6/European Commission International Conference on Networking 2000*, pages 374–385, Paris, 2000.

[38] S. Dimolitsas. Objective speech distortion measures and their relevance to speech quality assessments. *IEEE Proceedings*, 136(5), Oct. 1989.

[39] A.D. Doulamis, N.D. Doulamis, and S.D. Kollias. Traffic prediction and network resources estimation of VBR MPEG-2 sources using adaptively trained neural networks. In 10^{th} *Mediterranean Electrotechnical Conference, MEleCon*, volume II, pages 717–720, 2000.

[40] 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, 1999.

[41] S. Floyd, J. Handly, M. Padhye, and J. Widmer. Equation-based congestion control for unicast applications. In *Proceedings of ACM SIGCOMM 2000*, 2000.

[42] C.W. Fung and S.C. Liew. End-to-end frame-rate adaptive streaming of video data. In *IEEE International Conference on Multimedia Computing and Systems*, pages 67–71, 1999.

[43] R. Gareiss. Voice over IP services: The sound decision. *Data Communications International*, 27(4):74–85, 1998.

[44] L. Gautier, C. Diot, and J. Kurose. End-to-end transmission control mechanisms for multiparty interactive applications on the Internet. In *IEEE INFOCOM*, pages 1470–1479, 1999.

[45] E. Gelenbe. Random neural netowrks with negative and positive signals and product form solution. *Neural Computation*, 1(5):501–511, 1989.

[46] E. Gelenbe. Stability of the random neural network model. In *Neural Computation*, volume 2, pages 239–247, 1990.

[47] E. Gelenbe. Learning in the recurrent random neural network. *Neural Computation*, 5(2):154–164, 1993.

[48] E. Gelenbe. Hopfield energy of the random neural netowrk. In *IEEE International Conference on Neural Networks*, pages 4681–4686, June 1994.

[49] E. Gelenbe. The spiked random neural network: nonlinearity, learning and approximation. In *Fifth IEEE International Workshop on Cellular Neural Networks and Their Applications Proceedings*, 1998.

[50] E. Gelenbe, H. Bakircioglu, and T. Kocak. Image processing with the random neural network (RNN). In *13th International Conference on Digital Signal Processing Proceedings - DSP*, volume 1, pages 243–248, 1997.

[51] E. Gelenbe and A. Stafylopatis. Global behaviour of homogeneous random neural systems. In *Applied Mathematics Modeling*, volume 15, pages 534–541, 1991.

[52] M. Ghanbari and V. Seferidis. Cell-loss concealment in ATM video codecs. *IEEE Transactions on Circuits and Systems for Video Technology*, 3(3):238–247, 1993.

[53] G. Ghinea and J.P. Thomas. QoS impact on user perception and understanding of multimedia video clips. In *Proceedings of ACM Multimedia 98*, 1998.

[54] M. Grossglauser and J-C. Bolot. On the relevance of long-range dependence in network traffic. *IEEE/ACM Transactions on Networking*, 7(5):629–640, 1999.

[55] M.T. Hagan and M.B. Menhaj. Training feedforward networks with the Marquardt algorithm. *IEEE transactions on Neural Networks*, 5(6), November 1994.

[56] J. Hall and P. Mars. The limitations of artificial neural networks for traffic prediction. *IEEE Proceedings on Communications*, 147(2), April 2000.

[57] T. A. Hall. Objective speech quality measures for Internet telephony. In *Voice over IP (VoIP) Technology, Proceedings of SPIE*, volume 4522, 2001.

[58] 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*, Germany, 1999.

[59] J.H. Hansen and B.L. Pellom. Speech enhancement and quality assessment: A survey. In *submitted to IEEE Sig. Proc. Mag.*, Nov. 1998.

[60] M. Hassan, A. Nayandoro, and M. Atiquzzaman. Internet telephony: Services, technical challenges, and products. *IEEE Communications Magazine*, pages 96–103, Apr. 2000.

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

[62] Y. Inazumi, Y. Horita, K. Kotani, and T. Murai. Quality evaluation method considering time transition of coded video quality. In *Proceedings of International Conference on Image Processing - ICIP*, volume 4, pages 338–342, 1999.

[63] ITU-R Recommendation BS.1116-1. Methods for the subjective assessment of small impairments in audio systems including multichannel sound systems. `http://www.itu.int/`.

[64] ITU-R Recommendation BS.1283. Subjective assessment of sound quality: A guide to existing recommendations. `http://www.itu.int/`.

[65] ITU-R Recommendation BS.1284. Methods for the subjective assessment of sound quality – general requirements. `http://www.itu.int/`.

[66] ITU-R Recommendation BS.1286. Methods for subjective assessment of audio systems with accompanying pictures. `http://www.itu.int/`.

[67] ITU-R Recommendation BT.500-10. Methodology for the subjective assessment of the quality of television pictures. `http://www.itu.int/`.

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

[69] ITU-T Recommendation H.263. Video coding for low bit rate communication. `http://www.itu.int/`.

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

[71] ITU-T Recommendation P.861. Objective quality assessment of telephone-band (300-3400 hz) speech codecs. `http://www.itu.int/`.

[72] ITU-T Recommendation P.910. Subjective video quality assessment methods for multimedia applications. `http://www.itu.int/`.

[73] ITU-T Recommendation P.911. Subjective audiovisual quality assessment methods for multimedia applications. `http://www.itu.int/`.

[74] ITU-T Recommendation P.920. Interactive test methods for audiovisual communications. `http://www.itu.int/`.

[75] C. Jones and D.J. Atkinson. Development of opinion-based audiovisual quality models for desktop video-teleconferencing. In *International Workshop on Quality of Service*, 1998.

[76] J. Kim, Y. Kim, H. Song, T. Kuo, Y. Chung, and C. Kuo. TCP-friendly Internet video streaming employing variable frame-rate encoding and interpolation. *IEEE Transactions on Circuits and Systems for Video Technology*, 10(7), Oct. 2000.

[77] Y. Kim, Chung., Y., J. Kim, and C. Kuo. An end-to-end congestion control mechanism for Internet video. In *IEEE 3rd Workshop on Multimedia Signal Processing*, pages 21–26, 1999.

[78] Y. Kim, J. kim, and C. Kuo. Network-aware error control using smooth and fast rate adaptation mechanisms for TCP-friendly Internet video. In *Proceedings of Ninth International Conference on Computer Communications and Networks*, pages 320–325, 2000.

[79] 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*, Nov. 1996.

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

[81] J. Kostas, M. Borella, I. Sadhu, G. Schuster, J. Grabiec, and J. Mahler. Real-time voice over packet switched networks. *IEEE Network Journal*, 12(1), 1998.

[82] S. Lavington, N. Dewhurst, , and M. Ghanbari. The performance of layered video over an IP network. In *Packet Video*, 2000.

[83] S. Lavington, H. Hagras, and N. Dewhurst. Using a MLP to predict packet loss during real-time video transmission. In *Univ. Of Essex, UK, Internal Report CSM329*, 1999.

[84] F. Le Leannec and C. Guillemot. Packet loss resilient H.263+ compliant video coding. In *Proceedings of International Conference on Image Processing*, 2000.

[85] W.E. Leland, M.S. Taqq, W. Willinger, and D.V. Wilson. On the self-similar nature of Ethernet traffic. In *ACM SIGCOMM*, pages 183–193, September 1993.

[86] 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, 1998.

[87] A. Likas and A. Stafylopatis. Training the random neural network using quasi-newton methods. *European Journal of Operations Research*, 126(2):331–339, 200.

[88] W. Lobejko. VBR traffic prediction in ATM system. In *MILCOM Proceedings*, pages 1585–1588, 1997.

[89] H. Miyajima and S. Yatsuki. Dynamics of distance between patterns for higher order random neural networks. In *International Conference on Neural Networks*, volume 2, pages 1029–1034, 1997.

[90] S. Mohamed, F. Cervantes, and H. Afifi. Real-time audio quality assessment in packet networks. *Network and Information Systems Journal*, 3(3-4):595–609, 2000.

[91] S. Mohamed, F. Cervantes, and H. Afifi. Une méthode inter-subjective pour l'évaluation de la qualité sonore de voix sur ip. In *4ème édition des Journées Doctorales Informatique et Réseaux (JDIR)*, Paris, France, Nov. 2000.

[92] S. Mohamed, F. Cervantes, and H. Afifi. Audio quality assessment in packet networks: an inter-subjective neural network model. In *Proceedings of IEEE 15th International Conference on Information Networks: ICOIN-15*, Japan, Jan. 2001.

[93] S. Mohamed, F. Cervantes, and H. Afifi. Integrating networks measurements and speech quality subjective scores for control purposes. In *IEEE INFOCOM*, Alaska, USA, Apr. 2001.

[94] 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), December 2002.

[95] S. Mohamed, G. Rubino, F. Cervantes, and H. Afifi. Real-time video quality assessment in packet networks: A neural network model. In *Proceedings of International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA)*, Las Vegas, Nevada, USA, June 2001.

[96] S. Mohamed, G. Rubino, and M. Varela. The impact of encoding and network impairments on real-time speech quality. In *Submitted to IEEE Infocom*, 2003.

[97] B. Mukherjee and T. Brecht. Time-lined TCP for the TCP-friendly delivery of streaming media. In *Proceedings of International Conference on Network Protocols*, pages 165–176, 2000.

[98] Marcus Nadenau. *Integration of Human Color Vision Models Into High Quality Image Compression*. PhD thesis, EPFL, Lausanne, Swiss, 2000.

[99] F. Nino, F. Hernandez, and A. Parra. Financial time series modeling with evolutionary trained random iterated neural networks. In *Proceedings of IEEE/IAFE/INFORMS Conference on Computational Intelligence for Financial Engineering – CIFEr*, pages 178–181, 2000.

[100] P. Ojala, H. Toukomaa, T. Moriya, and O. Kunz. Report on the MPEG-4 speech codec verification tests. In *ISO/IEC JTC1/SC29/WG11 MPEG98/N2424*, Oct. 1998.

[101] M. Olivier and S. Manuel. Dynamics of large random recurrent neural networks: oscillations of 2-population model. In *International Joint Conference on Neural Networks – IJCNN*, pages 341–344, 1999.

[102] S. Olsson, M. Stroppiana, and J. Baina. Objective methods for assessment of video quality: State of the art. *IEEE Transactions on Broadcasting*, 43(4):487–495, 1997.

[103] C. Perkins, O. Hodson, and V. Hardman. A survey of packet-loss recovery for streaming audio. *IEEE Network*, 12(15):40–48, 1998.

[104] A. Pessoa, A. Falcao, R. Nishihara, A. Silva, and R. Lotufo. Video quality assessment using objective parameters based on image segmentation. *SMPTE Journal*, Dec. 1999.

[105] S.R. Quackenbush, T.P. Barnwell, and M.A. Clements. *Objective measures of speech quality.* Prentice Hall, New Jersey, 1988.

[106] R. Rejaie. On design of Internet multimedia streaming applications: An architectural perspective. In *Proceedings of IEEE International Conference on Multimedia and Expo*, New York, Aug. 2000.

[107] R. Rejaie and M. Handley. Quality adaptation for congetion controlled video playback over the Internet. *IEEE Journal on Selected Areas in Communications*, 18(12), 2000.

[108] R. Rejaie, M. Handley, and D. Estrin. RAP: and end-to-end rate-based congestion control mechanism for realtime streams in the Internet. In *IEEE INFOCOM*, pages 1337–1345, 1999.

[109] R. Rejaie, M. Handley, and D. Estrin. Architectural considerations for playback of quality adaptive video over the Internet. In *Proceedings of IEEE International Conference on Networks*, pages 204–209, 2000.

[110] R. Rejaie, M. Handley, and D. Estrin. Layered quality adaptation for Internet video streaming. *IEEE Journal on Selected Areas in Communications*, 18(12):2530–2543, 2000.

[111] RFC 2429. RTP payload format for the 1998 version of ITU-T rec. H.263 video (H.263+). IETF, 1998.

[112] A. Rix. Advances in objective quality assessment of speech over analogue and packet-based networks. In *the IEE Data Compression colloquium*, London, Nov. 1999.

[113] A. Rix and M. Hollier. The perceptual analysis measurement system for robust end-to-end speech assessment. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing – ICASSP*, pages 1515–1518, 2000.

[114] A. Rix, M. Hollier, and P. Gray. Predicting speech quality of telecommunication systems in quality differentiated market. In *IEE Conference on Telecommunications*, pages 156–160, 1998.

[115] A. Rix, R. Reynolds, and M. Hollier. Robust perceptual assessment of end-to-end audio quality. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 39–42, 1999.

[116] J. Rosenberg, L. Qiu, and H. Schulzrinne. Integrating packet FEC into adaptive voice playout buffer algorithms on the Internet. In *IEEE INFOCOM*, 2000.

[117] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. *Learning internal representations by error propagation*, volume 1. Parallel Distributed Processing, MIT Press, Cambridge, Massachusetts, 1986.

[118] D.E. Rumelhart, B. Widrow, and M.A. Lehr. The basic ideas in neural networks. *Communications of the ACM*, 37(3):87–92, March 1994.

[119] R. Schaphorst and D. Bodson. Subjective and objective testing of video teleconferencing/videophone systems. In *IEEE GLOBECOM*, 1991.

[120] M.F. Scheffer, J.P. Beneke, and J.S. Kunicki. Fuzzy modeling and prediction of network traffic flactuations. In *COMSIG*, pages 41–45, 1994.

[121] Y. Shu, Z. Jin, L. Zhang, L. Wang, and O.W. Yang. Traffic prediction using FARIMA models. In *IEEE International Conference on Communications*, pages 891–895, 1999.

[122] K. Shuaib, T. Saadawi, M. Lee, and B. Basch. A de-jittering scheme for the transport of MPEG-4 and MPEG-2 video over ATM. In *Proceedings of IEEE Military Communications Conference - MILCOM*, pages 1211–1215, 2000.

[123] S.A. Shumsky. Phase portrait characteristics of random neural networks. In *RNNS/IEEE Symposium on Neuroinformatics and Neurocomputers*, pages 145–156, 1992.

[124] D. Sisalem and A. Wolisz. Towards TCP-friendly adaptive multimedia applications based on RTP. In *Proceedings of IEEE International Symposium on Computers and Communications*, pages 166–172, 1999.

[125] D. Sisalem and A. Wolisz. LDA+: A TCP-friendly adaptation scheme for multimedia communication. In *IEEE International Conference on Multimedia and Expo - ICME*, pages 1619–1622, 2000.

[126] D. Sisalem and A. Wolisz. Constrained TCP-friendly congestion control for multimedia communication. *Lecture Notes in Computer Science*, 2156, 2001.

[127] H. Song, J. Kim, and J. Kuo. Real-time H.263+ frame rate control for low bit rate VBR video. In *Proceedings of 1999 IEEE International Symposium on Circuits and Systems - ISCAS*, pages 307–310, 1999.

[128] A. Stafylopatis and A. Likas. Pictorial information retrieval using the random neural network. *IEEE Transactions on Software Engineering*, 18(7):590–600, 1992.

[129] D. Tan and Zakhor A. Real-time Internet video using error resilient scalable compression and TCP-friendly transport protocol. *IEEE Transactions on Multimedia*, May 1999.

[130] K.T. Tan and M. Ghanbari. A combinational automated MPEG video quality assessment model. In *the Seventh International Conference on Image Processing And Its Applications*, pages 188–195, 1999.

[131] K.T. Tan and M. Ghanbari. A multi-metric objective picture-quality measurement model for MPEG video. *IEEE Transactions on Circuits and Systems for Video Technology*, 10(7):1208–1213, 2000.

[132] W. Tan and A. Zakhor. Real-time Internet video using error resilient scalable compression and TCP-friendly transport protocol. *IEEE Transactions on Multimedia*, 1(2):172–186, 1999.

[133] A.A. Tarraf, I.W. Habib, and T.N. Saadawi. Neural networks for ATM multimedia traffic prediction. In *International Workshop on Applications of Neural Networks to Telecommunications*, pages 85–91, 1993.

[134] A. Vahedian, M. Frater, and A. Arnold. Impact of audio on subjective assessment of video quality. In *Proceedings of International Conference on Image Processing - ICIP*, pages 367–370, 1999.

[135] C.J. van den Branden Lambrecht. *Perceptual Models and Architectures for Video Coding Applications*. PhD thesis, EPFL, Lausanne, Swiss, 1996.

[136] C.J. van den Branden Lambrecht, D.M. Costantini, G.L. Sicuranza, and M. Kunt. Quality assessment of motion rendition in video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(5):766–782, 1999.

[137] C.J. van den Branden Lambrecht, O. Verscheure, J. Urbain, and F. Tassin. Quality assessment of image features in video coding. In *Proceedings of International Conference on Image Processing*, pages 302–305, 1997.

[138] S. Voran. The development of objective video quality measures that emulate human perception. In *IEEE GLOBECOM*, pages 1776–1781, 1991.

[139] S. Voran. Estimation of perceived speech quality using measuring normalizing blocks. In *IEEE Workshop on Speech Coding For Telecommunications Proceeding*, pages 83–84, 1997.

[140] S. Voran. Objective estimation of perceived speech quality — Part I: development of the measuring normalizing block technique. *IEEE Transactions on Speech and Audio Processing*, 7(4), 1999.

[141] S. Voran. Objective estimation of perceived speech quality — Part II: development of the measuring normalizing block technique. *IEEE Transactions on Speech and Audio Processing*, 7(4), 1999.

[142] S. Voran and S. Wolf. The development and correlation of objective and subjective video quality measures. In *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, pages 483–485, 1991.

[143] S. Voran and S. Wolf. The development and evaluation of an objective video quality assessment system that emulates human viewing panels. In *International Broadcasting Convention - IBC*, pages 504–509, 1992.

[144] N. Walkamiya, M. Murata, and H. Miyahara. On TCP-friendly video transfer with consideration of application-level QoS. In *IEEE International Conference on Multimedia and Expo - ICME*, pages 843–846, 2000.

[145] S. Wang, A. Skey, and A. Gersho. An objective measure for predicting subjective quality of speech coders. *IEEE Journal on Selected Areas in Communications*, 10(5), 1992.

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

[147] A. Watson and M.A. Sasse. Measuring perceived quality of speech and video in multimedia conferencing applications. In *Proceedings of ACM Multimedi*, pages 55–60, 1998.

[148] L.R. Welch, A.D. Stoyenko, and S. Chen. Assigning ADT modules with random neural networks. In *Proceeding of the Twenty-Sixth Hawaii International Conference on System Sciences*, pages 546–555, 1993.

[149] B. Widrow, D. Rumelhart, and M. Lehr. Neural networks: Applications in industry, business and science. *Communications of the ACM*, 37(3), 1994.

[150] B. Widrow and S. Strearns. *Adaptive Signal Processing*. Prentice-Hall, Englewood Cliffs, N.J., 1985.

[151] S. Wolf, M.H. Pinson, S.D. Voran, and A.A. Webster. Objective quality assessment of digitally transmitted video. In *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, pages 477–483, 1991.

[152] D.J. Wright. Voice over ATM: An evaluation of implementation alternatives. *IEEE Communications Magazine*, 34(5):72–81, 1996.

[153] H.R. Wu, T. Ferguson, and B. Qiu. Digital video quality evaluation using quantitative quality metrics. In *Proceedings of 4th International Conference on Signal Processing*, 1998.

[154] H.R. Wu, C. Lambrecht, M. Yuen, and B. Qiu. Quantitative quality and impairment metrics for digitally coded images and image sequences. In *Proceedings of Australian Telecommunication Networks & Applications Conference*, Dec. 1996.

[155] 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.

[156] S.A. Youssef, I.W. Habib, and T.N. Saadawi. A neurocomputing controller for bandwidth allocation in ATM networks. *IEEE Journal on Selected Areas in Communications*, 15(2):191–199, Febuary 1997.

[157] E.S. Yu and C.Y. Roger Chen. Traffic prediction using neural networks. In *IEEE GLOBECOM*, pages 991–995, 1993.

[158] J. Zhang, J.F. Arnold, and M.R. Frater. A cell-loss concealement technique for MPEG-2 coded video. *IEEE Transactions on Circuits and Systems for Video Technology*, 10(4):659–665, June 2000.

[159] R. Zhang, S.L. Regunathan, and K. Rose. Video coding with optimal inter/intra-mode switching for packet loss resilience. *IEEE Journal on Selected Areas in Communications*, 18(6):966–976, 2000.

[160] T. Zhang and Y. Xu. Unequal packet loss protection for layered video transmission. *IEEE Transactions on Broadcasting*, 45(2):243–248, 1999.

[161] Y. Zhen and L. Canwei. A new speech quality assessment method based on human perception & its applivations. In *Proceedings of IEEE International Conference on Acoustics Speech, and Signal Processing – ICASSP*, pages 1515–1518, 2000.

[162] W. Zhu, Y.T. Hou, and Y. Wang. Modeling and simulation of MPEG-2 video transport over ATM networks considering the jitter effect. In *IEEE Workshop on Multimedia Signal Processing*, 1997.

Index

Absolute Category Rating (ACR), 82
Adaptive Differential PCM, 71
Audio Compression, 69

Bark Spectral Distortion, 54
Bit Rate (BR), 117, 136, 150

Coding
    Block, 66
    H.261 Video, 67
    H.263 Video, 67
    Hybrid, 66, 70
    Lossless, 66
    Lossy, 66
    MPEG-1/2/4, 68
    Source, 70
    Temporal, 66
    Waveform, 69
Color Moving Pictures Quality Metric, 60
Consecutive Lost Packets (CLP), 135, 140

Degradation Category Rating (DCR), 82

Encoding Algorithms, 100
Encoding Parameters, 89, 97, 113
Enhanced Modified BSD, 55
EPFL Measures, 59

Forward Error Correction, 64
Forward Error Correction (FEC), 151
Frame Rate (FR), 117, 136, 151

Gradient Descent, 171
GSM Codec, 72

H.263
    Decoder, 116
    Encoder, 115

Internet Protocol, 62
ITS Measure, 58
ITU, 81
    5-point quality scale, 82
    9-point quality scale, 82
ITU E-model, 56

Jitter, 63

Levenberg-Marquardt, 171, 173
Loss Distribution, 99
Loss Rate (LR), 99, 117, 135, 140

Mean Opinion Score (MOS), 84, 101, 118
Mean Square Error, 57
Measuring Normalizing Blocks, 56
Moving Pictures Quality Metric, 59

Network Parameters, 88, 97, 98, 114
Network Transport Simulation, 116
Neural Networks, 72, 86, 104, 122, 159
    Artificial, 72, 87
    Random, 74, 87, 171
Normalization Video Fidelity Metric, 60

Objective Speech Quality Assessment, 53
Objective Video Quality Assessment, 57, 61

Packet Delay, 63
Packet Loss, 62, 99
Packetization Interval, 151
Packetization Interval (PI), 100, 135
Peak Signal to Noise Ratio, 57
Perceptual Analysis Measurement System, 56
Perceptual Speech Quality Measure, 55
Pulse Code Modulation, 71

Quality-Affecting Parameters, 78, 79, 88, 117
    Speech, 99
    Study of, 133

Rate Control, 145, 149
    Equation-based TCP-Friendly, 149
    TCP-Friendly, 146
Real Time Control Protocol, 62
Real time Transport Protocols, 62

Segmental Signal-to-Noise Ratio, 54
Signal-to-Noise Ratio, 53
Speech Quality Assessment, 97, 104, 109

Spikes, 164
Standard Audio Codecs, 71
Standard Video Codecs, 67
Subjective Quality Assessment, 81
Subjective Quality Test, 118

TCP-Friendly, 146
Traffic Prediction, 159
Training Algorithms, 171
Transport Control Protocol, 62

User Datagram Protocol, 62

Video Compression, 65
Video Quality Assessment, 113, 124

Abstract

The Internet is an avenue receiving each day many new important applications, each of which has its requirements and offers services to a specific category of end-users. Among these applications, there are many on multimedia, which its traffic constitutes an important part of the total Internet traffic. These applications increase the need for multimedia quality assessment. This hot research topic is difficult for many reasons, including the fact that many different factors affect the quality; among them, the degradation due to the encoding of the original signal and the distortion due to the transmission over best effort networks.

This work includes four main contributions. The first one is about automatic multimedia quality evaluation. We present a new methodology that provides several advantages over the existing objective measures: the obtained results correlate well with human perception (the reference in such applications), it is computationally simple (we use neural networks) and there is no need to access the original signal in the operation phase (this is one of the major drawbacks of the existing measures). These points make it possible to integrate our method in multimedia applications and to run it in real time. We use this method to develop tools in order to evaluate in real time speech and video quality after traversing the network.

The second one deals with rate control. It has become extremely important to develop control mechanisms that eliminate, or at least minimize, the negative effects of some specific network characteristics on the quality of multimedia signals as perceived by the end-users. Such protocols should provide fairness with other competing TCP flows (controlling a large part of Internet traffic), maximize the QoS (as perceived by end-users) and optimize the network's resources utilization. We show how to develop one of such protocols that can react based on the network state (evaluated by a TCP-friendly mechanism) and the end-user's perception (evaluated by a trained neural network following the method mentioned before) to satisfy these goals. A part of this work consists of studying and understanding the combined effect of several important factors on multimedia quality.

The third contribution is about traffic prediction. We present a new model for traffic prediction that makes use of both the long-range and short-range periodicity of the traffic process to provide a more realistic scheme.

Finally, the fourth one deals with new training algorithms for a specific class of neural networks called random neural networks. Most of the work presented in this dissertation is based on these tools. We have found that they give better performance than the standard neural networks, but that the available training algorithm suffers from some problems including the speed of convergence. That is why we have proposed two new training algorithms to overcome these problems.

Résumé

L'Internet est une "avenue" recevant chaque jour de nouvelles applications importantes, dont chacune a ses conditions et offre des services à une catégorie spécifique d'utilisateurs. Parmi ces applications, on en trouve de multiples variantes autour du multimédia, où leurs trafics constituent une partie importante de tout le trafic de l'Internet. Ces applications augmentent le besoin de l'évaluation de la qualité. Ce point actif de la recherche est difficile pour plusieurs raisons, entre autre le fait que de nombreux facteurs affectent la qualité, notamment la dégradation due au codage et à la transmission sur un réseau best-effort.

Ce travail comprend quatre contributions principales. La première contribution est l'évaluation automatique de la qualité des flux multimédias. Nous présentons une nouvelle méthodologie qui offre plusieurs avantages par rapport aux mesures dites objectives existantes : les résultats obtenus se corrèlent bien avec la perception humaine (la référence dans de telles applications), elle est rapide et simple en terme de temps de calcul (nous utilisons les réseaux de neurones) et il n'y a aucun besoin d'accéder au signal original dans la phase opérationnelle (c'est l'un des inconvénients principaux des mesures existantes). Ces points permettent d'intégrer notre méthode dans des applications fonctionnant en temps réel. Nous employons cette méthode pour développer des outils d'évaluation de la qualité, en temps réel, séparément pour la parole et pour la vidéo, après traversée du réseau.

La seconde contribution est sur le contrôle de débit. Il est devenu extrêmement important de développer des mécanismes de contrôle qui éliminent ou au moins réduisent au minimum les effets négatifs de quelques phénomènes de réseau spécifiques, sur la qualité perçue des signaux multimédias. De tels protocoles devraient fournir l'équité pour l'ensemble des flux concurrents de TCP (qui contrôle une bonne partie du trafic de l'Internet), optimiser la QoS (comme perçue par les utilisateurs) et maximiser l'utilisation des ressources du réseau. Nous montrons comment développer de tels protocoles pouvant se baser (et réagir) sur l'état du réseau (évalué par un protocole TCP-friendly) et la perception de l'utilisateur (évaluée par un réseau de neurones qualifié en suivant la méthode mentionnée précédemment) pour satisfaire ces buts. Une partie de ce travail est d'étudier et comprendre l'effet combiné de plusieurs facteurs importants sur la qualité.

La troisième contribution concerne la prévision du trafic. Nous avons présenté un nouveau modèle pour la prévision du trafic qui se sert de la périodicité à longue portée et à courte portée de ce processus, pour fournir une méthode de prédiction plus efficace.

Enfin, la quatrième contribution traite de nouveaux algorithmes d'apprentissage pour un type spécifique de réseaux de neurones appelés réseaux de neurones aléatoires. La majorité du travail présenté dans cette thèse est basée sur ces outils. Nous avons constaté qu'ils donnent une meilleure performance que les réseaux de neurones standards, mais que l'algorithme disponible d'apprentissage souffre de quelques problèmes comprenant la vitesse de la convergence. C'est pourquoi, nous avons proposé deux nouveaux algorithmes pour les surmonter.