Université du Québec
Institut national de la recherche scientifique (INRS)
Centre Énergie Matériaux Télécommunications (EMT)

MODÈLES D'APPRENTISSAGE AUTOMATIQUE D'ESTIMATION DE QUALITÉ PERÇUE DANS LES COMMUNICATIONS EN TEMPS RÉEL

MACHINE LEARNING BASED PERCEIVED QUALITY ESTIMATION MODELS IN REALTIME COMMUNICATIONS

Par

Edip Demirbilek

Thèse présentée pour l'obtention du grade de
Philosophiae doctor, (Ph.D)
en Télécommunications

Jury d'évaluation

Examinateur externe	Omneya Issa CRC Canada, Innovation, Science and Economic Development
Examinateur externe	Fabrice Labeau Université McGill, Electrical and Computer Engineering
Examinateur interne	Tiago H. Falk Université du Québec, INRS, EMT
Directeur de recherche	Jean-Charles Grégoire Université du Québec, INRS, EMT

Acknowledgements

I would like to express my gratitude to my supervisor, Prof. Jean-Charles Grégoire, for his understanding and patience, and for having and showing confidence in my methods during my research.

I am grateful to my sibling, Leyla Mutlu, and rest of my family for their moral and emotional support through my graduate studies.

A special gratitude goes to my friend Brian Vasey for his support, encouragement and editing assistance.

I recognize that this research would not have been possible without and am grateful for the financial assistance of the Natural Sciences and Engineering Research Council of Canada (NSERC) and Summit-Tech Multimedia Communications Inc. under the Collaborative Research and Development (CRD) grant program.

Résumé

L'objectif de notre travail est de développer des modèles d'apprentissage automatique qui prédisent la qualité audiovisuelle perçue. La prédiction se fait à partir d'un ensemble de paramètres corrélés dérivés d'un ensemble de données extraits de la cible. Afin d'atteindre cet objectif, nous avons tout d'abord développé, avec VLC, un banc d'essai de la VsD (Vidéo sur Demande) et avons généré un ensemble de données préliminaires de la qualité audiovisuelle. Le but était d'étudier divers algorithmes d'apprentissage automatique. Ces premières expérimentations nous ont encouragé à développer un banc d'essai plus robuste, basé sur le framework multimédia GStreamer. Nous avons généré, avec ce nouveau banc d'essai, un ensemble de données de qualité audiovisuelle, propre à notre contexte. Ces données reflètent les configurations contemporaines des communications interactives pour le taux d'image par seconde, la quantification vidéo, les paramètres de réduction du bruit et le taux de perte des paquets du réseau. Nous avons ensuite utilisé cet ensemble de données afin de développer divers modèles, reposant soit sur l'information média (« paramétriques »), soit sur les données réseau (« bitstream »), d'estimation de la qualité perçue. Ces modèles sont basés sur les méthodes des forêts d'arbres décisionnels, des techniques dites de démarrage (« bootstrap »), de l'apprentissage profond et de la programmation génétique.

Pour les modèles paramétriques, les quatre méthodes ont atteint une précision élevée en terme de corrélation RMSE et de Pearson. Les modèles basés sur les forêts d'arbres décisionnels et les techniques de bootstrap montrent un petit avantage par rapport à l'apprentissage profond quant à la précision qu'ils ont atteint. Les modèles basés sur la programmation génétique sont moins performants même si leur précision est impressionnante. Nous avons également obtenu une précision élevée en utilisant les autres ensembles de données sur la qualité visuelle, accessibles au public. Les métriques de performance que nous avons calculées sont comparables aux modèles existants formés et testés sur ces ensembles de données.

Pour les modèles bitstream, les méthodes de forêts d'arbres décisionnels ainsi que les techniques de bootstrap ont surpassé les modèles basés sur l'apprentissage profond et la programmation génétique ainsi que tous les modèles paramétriques. Cependant, les modèles bitstream réalisés en programmation génétique et en apprentissage profond ont moins bien performé que les modèles paramétriques à cause d'une augmentation significative du nombre de caractéristiques dans l'ensemble de données bitstream. Dans l'ensemble, nous concluons que le calcul de l'information bitstream mérite l'effort fourni pour la générer. Ce calcul aide à construire des modèles plus précis mais demeure utile uniquement pour le déploiement de bons algorithmes.

Sur la base de nos résultats, nous concluons que les algorithmes basés sur l'arbre de décision conviennent aux modèles paramétriques ainsi qu'aux modèles bitstream. De plus, nous savons que l'extraction de données corrélées supplémentaires de l'ensemble de données nous aide à générer des modèles plus précis lorsque des algorithmes d'apprentissage automatique appropriés sont déployés.

L'ensemble des données, les outils et les codes d'apprentissage automatique qui ont été développés au cours de cette recherche sont gracieusement offerts à la communauté pour des fins de recherche et de développement.

Mots-clé: Banc d'essai de la communication multimédia; Données de mesure de la qualité audiovisuelle; Modélisation de la qualité perçue; Apprentissage automatique; Forêts d'arbres décisionnels; Techniques de bootstrap; Apprentissage profond; Programmation génétique.

Abstract

This research has started with the initial objective to build machine learning based models that predict the perceived audiovisual quality directly from a set of correlated parameters that are extracted from a target quality dataset. To reach that goal, we have first created a VideoLAN Video-on-Demand based testbed and generated a preliminary audiovisual quality dataset that let us experiment with various machine learning algorithms. These early experiments encouraged us to create a more robust testbed based on the GStreamer multimedia framework. With this new testbed, we have generated the INRS audiovisual quality dataset that reflects contemporary real-time configurations for video frame rate, video quantization, noise reduction parameters and network packet loss rate. Then we have utilized this INRS dataset to build several machine learning based parametric and bitstream perceived quality estimation models based on Random Forests, Bagging, Deep Learning and Genetic Programming methods.

For the parametric models, all four methods have achieved high accuracy in terms of RMSE and Pearson correlation with subjective ratings. Random Forests and Bagging based models show a small edge over Deep Learning with respect to the accuracy they have achieved. Genetic Programming based models fell behind even though their accuracy is impressive as well. We have also obtained high accuracy on other publicly available audiovisual quality datasets and the performance metrics we have computed are comparable to the existing models trained and tested on these datasets.

For the bitstream models, both the Random Forests and Bagging based bitstream models have outperformed the Deep Learning and Genetic Programming based bitstream models as well as all of the parametric models. However, both the Genetic Programming and Deep Learning based bitstream models fell behind the parametric models due to a significant increase in the number of features in the bitstream dataset. Overall we conclude that computing the bitstream information is worth the effort and helps to build more accurate models. However, it is useful only for the deployment of the right algorithms.

In light of our results, we conclude that the Decision Trees based algorithms are well suited to the parametric models as well as to the bitstream models. Moreover, we know that extracting additional correlated data from the dataset helps us to generate more accurate models when suitable machine learning algorithms are deployed.

The dataset, tools and machine learning codes that have been generated during this research are publicly available for research and development purposes.

Keywords Multimedia Communication Testbed; Audiovisual Quality Dataset; Perceived Quality Modeling; Machine Learning; Random Forests; Bagging; Deep Learning; Genetic Programming;

Synopsis

1 Introduction

La qualité de service (QoS) est l'une des méthodes standards d'évaluation de la qualité des communications multimédia. Elle consiste en la mesure de plusieurs paramètres tels que le débit, le délai, la gigue et le taux de perte de paquets (Perkis, 2016). Il est nécessaire d'avoir une compréhension claire des éventuelles variations de ces paramètres lors de l'analyse des caractéristiques d'une communication multimédia en temps réel. Ces variations se produisent dans les couches qui permettent les communications de bout en bout. Ceci est particulièrement important dans des environnements où les flux multimédia traversent des ressources dont la capacité de la bande passante est limitée et fluctuante, tels que les réseaux sans fil.

L'une des approches standards utilisées pour faire face à des situations changeantes durant une interaction multimédia multipartite est de mesurer periodiquement les différents paramètres de la QdS et les renvoyer au point final de transmission. Le but est de contrôler le taux et les caractéristiques du flux multimédia. Un tel modèle d'adaptation améliore généralement la qualité des communications individuelles. Les mesures de la QdS peuvent également être signalées pour des besoins de suivi de la qualité.

Plusieurs recherches de grande importance sur le contrôle de la QdS dans les deux couches d'application et de réseau existent. Diverses approches et outils de gestion ont été développé (Bordetsky *et al.*, 2001). Cependant, au cours de ces dernières années, la convergence de l'industrie du média numérique et celle des technologies de l'information et des communications (TIC) a abouti à un changement de paradigme de la QdS vers la qualité multimédia perçue (Perkis, 2016).

Alors que la QdS mesure principalement la précision de la transmission des données dans le réseau, d'autres facteurs additionnels affectent la perception par l'usager de la qualité de présentation du multimédia (Perkis, 2016). Lors d'une adaptation multimédia de la qualité perçue et même si une meilleure QdS offre souvent une meilleure qualité de présentation, des cas peuvent exister où les compromis de QdS sont nécessaires pour améliorer la qualité perçue globale (Hansen *et al.*, 2013). Ces compromis peuvent être effectués statistiquement ou dynamiquement pour une communication en temps réel.

L'approche classique de la modélisation de la qualité audiovisuelle consiste à développer des fonctions permettant de prédire la qualité audio et vidéo de manière indépendante, et ensuite de les combiner en utilisant une autre fonction de prédiction de la qualité audiovisuelle perçue globale. La méthode alternative est de construire des modèles qui prédisent la qualité audiovisuelle directement sans passer par des fonctions intermédiaires. Les approches de modélisation d'apprentissage automatique ont été appliquées avec succès à l'estimation de la qualité perçue (Maki *et al.*, 2013). Dans

ce travail, nous avons adopté la deuxième approche et avons construit des modèles d'apprentissage automatique qui prédisent la qualité audiovisuelle globale dans une seule fonction. Dans ce but, nous avons généré un ensemble de données de la qualité audiovisuelle propre à notre contexte, incluant à la fois les séquences audiovisuelles ainsi que l'estimation de la qualité correspondante. Cet ensemble de données de la qualité audiovisuelle consiste en deux instances; une version paramétrique où les données d'apprentissage et de test sont obtenues à partir des couches d'application et de réseau. Aucune information relative aux signaux originaux n'est utilisée. La deuxième version est bitstream où des données additionnelles de la couche bitstream ainsi qu'une quantité réduite de l'information du signal original sont utilisées.

Nous avons utilisé cet ensemble de données durant les phases d'apprentissage, de validation et de test de plusieurs modèles d'estimation de la qualité perçue, en utilisant l'apprentissage automatique. Nous avons construit des modèles d'estimation de la qualité perçue paramétriques sans référence ainsi que des modèles bitstream à référence réduite. Pour ce faire, nous nous sommes basés sur les méthodes des forêts d'arbres décisionnels, des techniques bootstrap, de l'apprentissage profond et de la programmation génétique.

Pour les modèles paramétriques, nous avons utilisé la version paramétrique de l'ensemble de données que nous avons généré. Toutes les méthodes mentionnées auparavant ont atteint une précision élevée en terme de corrélation RMSE et de Pearson. Les modèles basés sur les forêts d'arbres décisionnels et les techniques de bootstrap montrent un petit avantage par rapport à l'apprentissage profond quant à la précision qu'ils fournissent à l'ensemble de données utilisé. Les modèles basés sur la programmation génétique sont moins performants même si leur précision est impressionnante. Nous avons également obtenu une précision élevée en utilisant les autres ensembles de données sur la qualité visuelle, accessibles au public. Les métriques de performance sont comparables aux modèles existants formés et testés sur ces ensembles de données.

Pour les modèles bitstream, nous avons utilisé la version bitstream de l'ensemble de données. Les modèles basés sur les forêts d'arbres décisionnels et les techniques de bootstrap ont surpassé les modèles basés sur l'apprentissage profond et la programmation génétique, quant à la précision qu'ils fournissent sur l'ensemble de données étendu que nous avons utilisé. De plus, ces modèles d'arbre de décision ont eu de meilleurs résultats, comparés aux modèles paramétriques. Toutefois, les modèles bitstream réalisés en programmation génétique et en apprentissage profond ont moins bien performé que les modèles paramétriques en raison d'une augmentation significative du nombre de caractéristiques dans l'ensemble de données bitstream. Dans l'ensemble, nous concluons que le calcul de l'information bitstream mérite l'effort fourni pour la générer. Ce calcul aide à construire des modèles plus précis mais demeure utile uniquement pour le déploiement de bons algorithmes.

À partir de nos observations des deux modèles paramétriques et bitstream, nous concluons que les algorithmes basés sur l'arbre de décision conviennent aux modèles paramétriques sans référence ainsi qu'aux modèles bitstream avec référence réduite.

Sur la base de ces résultats, nous pouvons résumer nos principales contributions comme suit:

- la génération d'un ensemble de données de la qualité audiovisuelle, composé d'une version paramétrique et d'une autre bitstream. Cet ensemble de données comprend des vidéos de référence, des vidéos transmises, des scores de qualité détaillés et consolidés ainsi que tous les autres paramètres;
- le développement de modèles d'estimation de la qualité perçue, basés sur l'apprentissage automatique. Nous nous sommes principalement interessé aux modèles paramétrique sans référence ainsi qu'aux modèles bitstream avec référence réduite;

– le développement de ressources Open Source: des bancs d'essai basés sur VLC et GStreamer, un lecteur subjectif d'évaluation de la qualité, des scripts de configuration ainsi que des codes d'apprentissage automatique pour les modèles experimentaux, paramétriques et bitstream.

Dans la section qui suit, nous formulons quelques recommandations pertinentes pour des travaux similaires à la recherche que nous avons menée. Nous expliquons brièvement diverses approches de modélisation de la qualité perçue, les métriques statistiques utilisées pour l'évaluation la performance du modèle ainsi que les meilleures pratiques utilisées dans la modélisation lors de la mesure de la précision des tests sur un ensemble relativement restreint de données. Dans la section 3, nous expliquons les modèles standardisés de prédiction de la qualité audiovisuelle. Nous décrivons comment ces modèles diffèrent de l'approche que nous adoptons. Cette section est également très utile pour comprendre les configurations de l'ensemble de données de la qualité audiovisuelle que nous présentons dans la section 6. Dans la Section 4, nous décrivons brièvement les algorithmes d'apprentissage automatique que nous utilisons dans ce travail. La section 5 explique notre recherche préliminaire en vue de générer un ensemble de données de la qualité audiovisuelle et de construire des modèles d'estimation de la qualité perçue. La section 6 explique en détails la version paramétrique de l'ensemble de données de la qualité audivisuelle, géneré pour ce travail de recherche. Dans la section 7, nous introduisons les modèles paramétriques que nous avons construits et partageons les résultats obtenus pour les méthodes des forêts d'arbres décisionnels, des techniques de bootstrap, de l'apprentissage profond et de la programmation génétique. La section 8 portera sur la version bitstream de l'ensemble de données de la qualité audiovisuelle. Dans la section 9, nous étudierons les modèles bitstream que nous avons développés en utilisant les extensions correspondantes. Nous verrons une discussion à la fin de la section 7 et la section 9 où nous réaffirmons la motivation des modèles, discutons de divers aspects des modèles que nous avons présentés et montrons les différences entre ces modèles et les modèles standardisés existants. Nous résumerons le travail et partagerons notre perspective sur les travaux futurs dans la dernière section.

2 Contexte du travail

Nous discutons brièvement, dans cette section, des pratiques standard pour mener des tests de la qualité audiovisuelle, des différentes approches pour réaliser des modèles de qualité audiovisuelle, des métriques statistiques utilisées pour évaluer les performances du modèle ainsi que des meilleures pratiques utilisées pour valider les modèles sous un ensemble restreint de données.

L'union internationale des télécommunications propose diverses recommandations sur la manière dont les tests sur la qualité multimédia devraient être menés. Les recommandations communément utilisées pour les testes de la qualité audiovisuelle sont ITU-T P.913 (1998), ITU-T P.920 (1996) et ITU-T P.1401 (2012). Ces recommandations sont en effet un guide de méthodes de test, du materiel à utiliser ainsi que de l'environnement de test. Elles spécifient le nombre de sujets et leur éventuel processus de sélection.

Les méthodologies d'estimation subjective de la qualité ainsi que le choix de l'échelle de notation sont discutées en détail dans (Huynh-Thu *et al.*, 2011) et (ITU-T P.910, 1999). Dans Huynh-Thu *et al.* (2011), les auteurs montrent qu'il n'existe pas de différences statistiques globales entre les différentes échelles utilisées. Ils ont aussi montré que la présentation du stimulus unique fournit des résultats hautement reproduisibles même si différentes échelles sont utilisées. Les méthodes du seul stimulus, incluant l'évaluation de la catégorie absolue (Absolute Category Rating, ACR), sont souvent utilisées pour effectuer des experiences subjectives sur la qualité comme elles sont faciles

et rapides à mettre en oeuvre. La méthode ACR permet l'évaluation efficace de plusieurs fichiers d'une session; elle est bien adaptée aux tests de qualification. Les deux échelles ACR les plus utilisées sont les échelles à 5 points et celles à 11 points. L'échelle de catégorie 5 points est communément utilisée dans les télécommunications. Les étiquettes "excellent", "bon", "acceptable", "médiocre", et "mauvais" réfèrent aux valeurs 5, 4, 3, 2 et 1 lors du calcul du MOS (ITU-T P.910, 1999).

La qualité audiovisuelle dépend de la qualité audio et vidéo ainsi que de leur interaction. La qualité audiovisuelle globale peut être estimée directement à partir d'une fonction, et ceci indépendemment de la façon dont les dégradations affectent la qualité individuelle de l'audio et de la vidéo. Une approche alternative consite à prévoir, de manière individuelle, la qualité intermédiaire de l'audio et de la vidéo et les intégrer par la suite dans une qualité audiovisuelle globale. Dans Raake *et al.* (2011), les auteurs suggèrent qu'un modèle complexe, utilisant les fonctions intermédiaires audio et vidéo, générerait des prédictions plus précises.

Différentes approches de développement de modèles de qualité audiovisuelle existent. Raake *et al.* (2011) a catégorisé les modèles selon le type de données utilisées.

- Les modèles paramétriques prédisent l'impact des configurations de l'encodage et des altérations du réseau sur la qualité multimédia. Ils utilisent généralement l'information extraite des entêtes des paquets et n'ont pas accès aux données du paquet. Ces méthodes conviennent aux cas où les données sont chiffrées (Dubin *et al.*, 2016).
- Les modèles de planification sont semblables aux modèles paramétriques; la différence est d'où l'information d'entrée sera acquise. Ces modèles sont basés sur l'information de service disponible durant la phase de planification, alors que les modèles paramétriques prennent les informations d'entrée d'un service existant.
- Les modèles basés sur le média ou sur le signal incluent des aspects de la perception humaine et évaluent les caractéristiques physiques du signal expédié. Ils utilisent le signal décodé comme entrée pour calculer la valeur de la qualité.
- Les modèles bitstream exploitent les informations du flux élémentaire. Ces modèles traitent en général les entêtes et le payload du flux binaire vidéo. Ils traitent l'entête du flux binaire pour extraire des informations de transport telles que le flux de transport (Transport stream, TS) et/ou les champs timestamps et les numéros de séquence du protocole Real-time Transport Protocol (RTP). Le but est de détecter la perte de paquets. Ces modèles traitent le payload du flux binaire vidéo afin d'extraire un certain nombre de caractéristiques telles que le type d'image, le nombre de tranches, le paramètre de quantification (Quantization Parameter, QP), le vecteur de mouvement, le type de chaque macrobloc (MB) et ses partitions ainsi que les coéfficients de transformation du résidu de prédiction.
- Les modèles hybrides d'évaluation de la qualité exploitent les informations des entêtes de paquets, du flux élémentaire et des images reconstruites. L'information sur les images reconstruites est obtenue à partir de la séquence vidéo traitée, générée par un décodeur externe plutôt que par un décodeur interne du modèle.

Les modèles de qualité peuvent aussi être regroupés selon le type d'informations supplémentaires qu'ils traitent. Les modèles avec référence (Full reference, FR) traitent généralement la séquence source originale, alors que les modèles avec référence réduite (Reduced-Reference, RR) utilisent seulement une quantité limitée de l'information dérivée de la séquence source. Les modèles sans référence (No-Reference, NR) utilisent des séquences transmises sans utiliser aucune information du signal original.

Un aspect important de la modélisation de la qualité perçue est qu'un modèle objectif ne devrait pas prédire une opinion moyenne subjective de manière plus précise qu'un sujet de test moyen. L'incertitude des votes subjectifs est calculée par l'écart-type et l'intervalle de confiance (IC) correspondant. Ces paramètres statistiques visent à déterminer l'incertitude des sujets par fichier, ou par condition de test (ITU-T P.1401, 2012).

Traditionnellement, la performance d'un modèle est évaluée via trois métriques statistiques, utilisées pour informer de la précision du modèle, de sa consistance et de sa linéarité/monotonie (ITU-T P.1401, 2012).

La précision d'un modèle est habituellement déterminée par une interprétation statistique de la différence entre les valeurs MOS du test subjectif et sa prédiction sur une échelle généralisée. Un modèle précis a pour but de prédire la qualité avec l'erreur la plus faible en terme de RMSE lors des tests subjectifs (ITU-T P.1401, 2012). ITU-T Rec. P.863 (Beerends *et al.*, 2013) recommande de convertir cette valeur en "epsilon-modified RMSE" pour comparer les résultats selon différentes échelles (Garcia, 2014).

Les prévisions de qualité perçue doivent avoir constamment des faibles marges d'erreur sur la gamme des sujets d'essai. La cohérence du modèle est signalée en calculant soit la distribution des erreurs résiduelles, soit le rapport des valeurs aberrantes (outlier ratio). Pour calculer ce dernier, il faut trouver les valeurs aberrantes qui sont déterminées comme étant les points pour lesquels l'erreur de prédiction dépasse 95% de l'intervalle de confiance (ITU-T P.1401, 2012; Garcia, 2014).

Dans la littérature, deux métriques couramment utilisées pour le calcul de la linéarité d'un modèle existent: le coefficient de rang de Spearman et le coefficient de corrélation de Pearson. Le coefficient de corrélation de Pearson est utilisé chaque fois que les données échantillonnées ont une distribution presque normale. Dans d'autres cas, le coefficient de rang de Spearman est utilisé pour qualifier la linéarité entre les scores de qualité subjective prédits et réels (ITU-T P.1401, 2012; Garcia, 2014).

Les prévisions du modèle de qualité sont comparées aux scores de qualité réels pour évaluer la performance des modèles. Toutefois, dans le cas où la quantité de données d'apprentissage et de tests est limitée, on utilise K-fold ou "leave-one-out cross-validation" pour rapporter la performance du modèle de qualité. Dans l'approche K-Folds, les données disponibles sont divisées en K échantillons. À chaque étape, K-1 échantillons sont utilisés pour former les données et le seul échantillon restant est utilisé pour mesurer la précision du modèle. Cette procédure est répétée K fois en utilisant une partie différente des données disponibles comme données d'essai. Le fractionnement des données peut être fait au hasard ainsi que la stratification des échantillons. Selon les K échantillons sélectionnés, le résultat du modèle peut varier. Afin de rendre les prédictions plus robustes et indépendantes des K échantillons sélectionnés, il est recommandé de répéter la procédure à plusieurs reprises et en prendre la moyenne pour chacune des métriques. La pratique courante est d'utiliser pour 10 fois la validation croisée stratifiée (Garcia, 2014; Maki *et al.*, 2013).

3 Modèles standardisés de prévision de la qualité audiovisuelle

Certaines des méthodes normalisées ont été créées par compétition en sélectionnant les modèles qui ont atteint la plus haute précision de prédiction. Dans cette section, trois modèles de prévision de la qualité audiovisuelle des services de diffusion et des applications de téléphonie vidéo sont brièvement expliqués.

Le modèle ITU-T P.1201 (2012) est destiné à l'estimation de la qualité audiovisuelle des services de diffusion. C'est un modèle non intrusif d'information d'entête de paquet visant la surveillance de service et le benchmarking du streaming UDP. Le modèle prend en charge aussi bien les applications à basse résolution telles que la télévision mobile ainsi que les applications à plus haute résolution telles que l'IPTV. Le modèle utilise les informations extraites de l'entête du paquet ainsi que les informations fournies hors bande. Il fournit des prédictions distinctes de la qualité audio, vidéo et audiovisuelle sous forme de résultat en terme du MOS à 5 points. Le modèle a été validé pour la compression, la perte de paquets ainsi que le buffering des altérations de l'audio et de la vidéo avec des débits différents. Le contenu vidéo des différentes complexités spatiotemporelles avec différentes images clés (keyframes), cadences d'images (frame rates) et résolutions vidéo a été sélectionné. Le modèle ITU-T Rec. P.1201 a testé plus de 1166 échantillons à des résolutions inférieures et a testé plus de 3190 échantillons à des résolutions plus élevées. Les valeurs de corrélation RMSE et de Pearson (Garcia, 2014) pour la modélisation audiovisuelle ont été évaluées respectivement à 0,470 et 0,852 pour les applications à résolution inférieure et à 0,435 et 0,911 pour les applications à plus haute résolution. Les statistiques détaillées de la performance sont présentées dans (ITU-T P.1201, 2012).

Le modèle ITU-T G.1071 (2015) est recommandé pour la planification réseau des services de diffusion audio et vidéo. Cette recommandation concerne les domaines d'application à plus haute résolution (HR) tels que l'IPTV et les domaines d'application de résolution inférieure (LR) comme la TV mobile. L'application des modèles est limitée à la planification de la qualité d'expérience (QoE)/qualité de service (QoS). Le benchmarking et le suivi de la qualité ne font pas partie du cadre de cette recommandation. Le modèle prend en entrée les hypothèses de planification de réseau telles que la résolution vidéo, les types et profils de codecs audio et vidéo, les débits audio et vidéo et le taux de perte de paquets. Il fournit en sortie des prédictions distinctes de la qualité audio, vidéo et audiovisuelle définies sur l'échelle MOS à 5 points. Notons que des cas d'utilisation tels que la dégradation audio et vidéo au cours du buffering, les situations de transcodage, les effets du bruit et du délai sur l'audio, la diffusion audiovisuelle avec une adaptation significative du débit ne font pas partie du modèle. À noter aussi que des tests ont montré que, pour les applications à basse résolution utilisant les bases de données d'apprentissage et le test ITU-T P.1201.1, les coefficients d'estimation de la qualité audiovisuelle RMSE et de la corrélation de Pearson atteignent respectivement 0.5 et 0.83. Pour les applications à haute résolution et pour des bases de données d'apprentissage et de validation ITU-T P.1201.2, les coefficients d'estimation de la qualité audiovisuelle RMSE et Pearson atteignent 0,51 et 0,87 respectivement.

ITU-T G.1070 (2012) représente un modèle de planification recommandé pour la téléphonie vidéo. Dans ce modèle, la qualité multimédia globale est calculée en employant des paramètres de réseau et d'application ainsi que des paramètres sur le terminal. Il propose un algorithme qui estime la qualité du vidéophone pour la qualité de l'expérience et la qualité des planificateurs de services. Le modèle fournit aussi des estimations sur la qualité multimédia qui tiennent compte de l'interactivité pour permettre aux planificateurs d'éviter de sous-dimensionner le service. Le modèle contient trois fonctions principales d'évaluation de la qualité de la parole, de la qualité vidéo et de la qualité multimédia globale. La fonction d'estimation de la qualité de la parole est similaire à l'outil E-model (ITU-T G.107, 2003) et prend comme paramètres d'entrée le type de codec vocal, le taux de perte de paquets, le débit binaire et le niveau sonore d'écho de la parole. La fonction vidéo est générée pour le contenu «head-and-shoulders» et prend comme paramètres d'entrée le format vidéo, la taille d'affichage, le type du codec, le taux de perte de paquets, le débit binaire, l'intervalle d'images clés ainsi que le taux d'images. La fonction multimédia intègre séparément la qualité audio et la qualité vidéo en incluant l'asynchronisme audiovisuel (audiovisual asynchrony) et le délai de

bout en bout. Sur des ensembles de données précis, la précision du modèle d'évaluation de la qualité des communications multimédias en terme de corrélation de Pearson est de 0,83 pour QVGA et de 0,91 pour la résolution QQVGA . L'application du modèle est limitée à la planification de la QoE et de la QoS. Des applications telles que le benchmarking et le suivi de la qualité ne sont pas couvertes par la recommandation.

Les modèles ITU-T P.1201 (2012), ITU-T G.1070 (2012), et ITU-T G.1071 (2015) ont obtenu une précision de prédiction élevée par rapport aux ensembles de données de test fournis. Cependant, ces méthodes ont par définition des domaines d'application limités et couvrent des technologies de codage limitées. Par conséquent, les chercheurs ont essayé d'améliorer ces modèles (Garcia, 2014; Belmudez, 2015).

Pour les modèles normalisés, les prédictions audio, vidéo et audiovisuelles sont accomplies par leurs fonctions respectives dont la sortie est ensuite transmise à la fonction audiovisuelle pour prédire la qualité audiovisuelle. Une autre approche consiste à mettre en œuvre la fonction audiovisuelle de manière à ne pas nécessiter de prédictions intermédiaires pour la qualité audio et vidéo et encore pour être en mesure de saisir toutes les interrelations complexes entre les facteurs d'influence. Les techniques basées sur l'apprentissage automatique ont été appliquées avec succès dans la mise en oeuvre de ces fonctions (Gastaldo *et al.*, 2013; Maki *et al.*, 2013). Grâce aux techniques d'apprentissage automatique, nous pouvons, avec moins d'efforts, construire des modèles de prédiction adaptés à des cas d'utilisation spécifiques tout en obtenant une précision élevée. Historiquement, les approches fondées sur les réseaux de neurones ont été largement utilisées. Au cours de cette recherche, en plus des modèles de l'apprentissage profond, nous évaluons l'ensemble des méthodes basées sur l'arbre de décision et la programmation génétique pour mettre en œuvre la fonction de qualité audiovisuelle permettant de prédire la qualité perçue directement à partir des paramètres extraits des couches application et réseau. Les modèles basés sur l'apprentissage automatique capturent les relations complexes entre les facteurs d'influence, peu importe si l'ensemble de données est généré pour les services IPTV ou pour la vidéo-téléphonie.

4 Méthodes d'apprentissage automatique

Le monde contemporain de l'apprentissage automatique consiste en un nombre incalculable d'algorithmes ainsi que de leurs implémentations dans diverses bibliothèques. Certaines de ces méthodes sont destinées uniquement aux problèmes de classification. Cependant, plusieurs algorithmes sont adaptés à des problèmes de classification aussi bien que de régression. Nous décrivons brièvement les méthodes d'apprentissage automatique que nous avons étudiées en détail et avons adaptées afin d'obtenir les meilleures performances. La raison principale derrière le choix de ces méthodes est présentée dans la Section 7. Même si nous utilisons ces méthodes pour la régression, dans le résumé ci-dessous, nous ne nous sommes pas limités à leur utilisation unique en régression.

4.1 Méthodes d'ensemble basées sur l'arbre de décision

Les arbres de décision (Decision Trees, DT) sont des structures de données hiérarchiques qui peuvent être utilisées pour des problèmes de classification et de régression en utilisant efficacement la stratégie de «Diviser-et-conquérir» (divide-and-conquer). Un arbre de décision est composé de nœuds de décision internes où un test est appliqué à une entrée donnée. Il se compose aussi de branches conduisant à des valeurs d'attribut. Une valeur de classification ou de régression est attri-

buée par les noeuds feuilles. Le processus d'estimation provient du noeud racine, traverse les noeuds de décision jusqu'à ce qu'un noeud soit atteint (Alpaydin, 2014; Mushtaq *et al.*, 2012).

La structure arborescente permet une découverte rapide des nœuds qui couvrent une entrée. Dans un arbre binaire, la traversée de chaque nœud de décision exclut la moitié des cas. En raison de la convergence rapide et de la facilité d'interprétation, les arbres binaires sont parfois préférés à des méthodes plus précises (Alpaydin, 2014).

L'estimation peut être calculée que le modèle soit paramétrique ou non paramétrique. Pour l'estimation paramétrique, le modèle est construit sur l'ensemble de l'espace d'entrée à partir de données d'apprentissage; une structure arborescente statique est formée. Le même modèle est ensuite utilisé pour faire des estimations dès que les données de test sont disponibles. Pour l'approche non paramétrique, la structure arborescente n'est pas statique et, au cours du processus d'apprentissage, elle pousse au fur et à mesure que les branches et les feuilles sont ajoutées (Alpaydin, 2014).

Les arbres de décision présentent un biais (bias) faible et une variance très élevée, ce qui entraîne des problèmes d'ajustement lorsqu'ils se développent très profondément. Pour réduire la variance, les méthodes d'ensemble basées sur l'arbre de décision ont été construites. Les forêts d'arbres décisionnels sont en effet un ensemble de méthodes d'apprentissage pour la classification et la régression utilisant différents modèles d'arbres de décision pour obtenir une meilleure performance de prédiction. Durant la phase d'apprentissage, un tableau d'arbres de décision est formé et un sous-ensemble de données d'apprentissage, choisi au hasard, est utilisé pour former chaque arbre. Dans un problème de classification, les entrées sont soumises à chaque arbre dans la forêt d'arbres décisionnels et ceci afin d'obtenir un vote pour une classe. Un modèle de forêt d'arbres décisionnels recueille tous les votes, puis choisit la classe avec le plus grand nombre de votes. Ce comportement réduit les problèmes de variance élevée dont nous avons parlé plus haut. Cependant, comme il existe un compromis entre le biais et la variance, la classification RF introduit une légère augmentation du biais tout en réduisant la variance. Dans l'ensemble, ce modèle fournit des améliorations significatives en terme de précision de classification (Breiman, 2001; Mushtaq *et al.*, 2012).

Les forêts d'arbres décisionnels n'ont que deux paramètres à régler pendant la phase d'apprentissage. Ces paramètres sont le nombre d'arbres dans la forêt et le nombre de variables dans le sous-ensemble aléatoire à chaque noeud (Liaw & Wiener, 2002).

Au lieu de rechercher un seul modèle supérieur, les chercheurs ont remarqué que le fait de combiner de nombreuses variations produit de meilleurs résultats avec un peu d'effort supplémentaire. Comme nous pouvons le remarquer, pour les forêts d'arbres décisionnels, les modèles d'apprentissage génèrent de nombreux classificateurs et combinent leurs résultats. Cette approche a récemment suscité un grand intérêt. Les deux méthodes d'apprentissage les plus connues sont le boosting et les techniques de bootstrap. Pour ces deux méthodes, l'algorithme d'apprentissage combine les prédictions de plusieurs modèles de base (Liaw & Wiener, 2002; Oza, 2005; Domingos, 2012).

Dans la construction de modèles basés sur les arbres de décision avec des méthodes utilisant les techniques de bootstrap, chaque arbre est construit avec une variation aléatoire de l'ensemble de données d'apprentissage. La prévision est réalisée par un simple vote majoritaire. Le but est d'améliorer la stabilité et la précision. Cette approche réduit considérablement la variance et contribue à éviter les problèmes d'ajustement, mais elle augmente légèrement le biais. Bien qu'elle soit généralement appliquée aux arbres de décision, cette approche peut être aussi bien utilisée avec n'importe quel type de méthode (Liaw & Wiener, 2002; Domingos, 2012).

Pour les méthodes de boosting, la prédiction dépend aussi des arbres antérieurs. Dans cette approche, les points mal prédits par les arbres précédents reçoivent un poids supplémentaire par les arbres successifs. Les méthodes de boosting visent principalement à réduire le biais et éventuellement la variance tout en créant un seul apprenant fort (strong learner) parmi un ensemble d'apprenants qui sont faibles (Liaw & Wiener, 2002).

Dans Pfahringer *et al.* (2007), les auteurs soulignent que les learners de l'arbre ne sont pas très stables en raison de leur capacité d'anticipation (lookahead) limitée. Les méthodes d'ensemble tentent de surmonter les problèmes rencontrés dans les apprenants simples de l'arbre de base.

4.2 Régression symbolique et programmation génétique

La technique de régression symbolique vise à identifier une expression mathématique sous-jacente qui correspond le mieux à un ensemble de données. Elle consiste à trouver simultanément la forme des équations ainsi que les paramètres. La régression symbolique commence par former une expression initiale en combinant aléatoirement des blocs de construction mathématiques et puis continuer à former de nouvelles équations en recombinant les équations précédentes en utilisant la programmation génétique (GP) (Schmidt & Lipson, 2010).

La programmation génétique est une technique de calcul qui nous permet de trouver une solution à un problème sans connaître préalablement la forme de la solution. Elle est basée sur l'évolution d'une population de programmes informatiques où les populations sont stochastiquement transformées en de nouvelles populations génération par génération (Poli *et al.*, 2008).

La programmation génétique découvre la performance d'un programme en l'exécutant, en mesurant son résultat et en comparant ce résultat à un objectif défini. Cette comparaison est appelée «fitness». Dans le domaine de l'apprentissage automatique, ceci est équivalent à trouver 'le score', 'l'erreur' ou 'la perte'. Dans chaque génération, les programmes qui réussissent sont marqués pour la reproduction et sont ensuite utilisés pour produire de nouveaux programmes pour la génération suivante. Le croisement (crossover) et la mutation sont les principales opérations génétiques aidant à créer de nouveaux programmes à partir d'un ensemble de programmes existants. Pour l'opération de croisement, un programme enfant est généré en joignant des parties choisies au hasard à partir de deux programmes sélectionnés de la génération précédente. Cependant, pour la mutation, un programme enfant est créé à partir d'un seul parent de la génération précédente en modifiant aléatoirement un segment arbitraire (Poli *et al.*, 2008).

La programmation génétique utilise généralement les arbres dans le but de manipuler des programmes. Les appels de fonction dans l'arbre sont représentés par les noeuds et les valeurs associées aux fonctions sont représentées par les feuilles (Koza, 1992). Les programmes de la programmation génétique combinent plusieurs composantes dans des formes plus avancées. Dans ce cas, chaque composant est représenté par un arbre qui se regroupe avec d'autres arbres sous le noeud racine (Poli *et al.*, 2008).

Tout comme l'ensemble de méthodes que nous avons vu dans la section précédente, les populations initiales de la programmation génétique sont généralement générées de façon aléatoire. Ces populations initiales sont catégorisées comme pleine (full), croissance (grow) et combinée (ramped half and half) selon leur profondeur (Poli *et al.*, 2008).

Les deux méthodes full et grow limitent la profondeur maximale des individus initiaux générés. Elles diffèrent les unes des autres quant à la taille et à la forme des arbres générés. Pour la méthode

full, les arbres sont générés où toutes les feuilles sont à la même profondeur. Les arbres de la méthode grow sont générés dans différentes tailles et formes. La méthode Ramped half-and-half propose de combiner les deux méthodes full et grow. Dans cette approche, la méthode full est utilisée pour construire la moitié de la population initiale et la méthode grow est utilisée pour construire l'autre moitié (Poli *et al.*, 2008).

La programmation génétique choisit les individus de façon probabiliste en fonction de leur fitness, puis leur applique des opérations génétiques. Ce processus entraîne de meilleurs individus ayant probablement plus de programmes enfants que les individus inférieurs. Deux méthodes communes de sélection individuelle en programmation génétique sont la sélection du tournoi (tournament selection) et la sélection proportionnelle de fitness (fitness proportionate selection) (Poli *et al.*, 2008).

4.3 Apprentissage profond

L'apprentissage profond remonte aux années 1940 et a été renommé à plusieurs reprises, reflétant l'influence de différents chercheurs et de différentes perspectives. Cette appellation spécifique est très récente (Bengio *et al.*, 2015).

Un exemple typique d'un modèle d'apprentissage profond est «feedforward Deep Network» ou le perceptron multicouche (Multi-Layer perceptron, MLP) (Bengio *et al.*, 2015). MLP ne fait aucune hypothèse sur les relations entre les variables. En général, ces modèles utilisent trois couches principales: une couche d'entrée de neurones représentant le vecteur d'entrée, une ou plusieurs couches intermédiaires «cachées» et des neurones de sortie qui représentent le vecteur de sortie. Les noeuds de chaque couche sont liés à tous les noeuds des couches adjacentes. Ces liens sont utilisés pour transmettre des signaux d'un neurone à l'autre (Comrie, 1997; Mushtaq *et al.*, 2012).

Les non-linéarités sont représentées dans le réseau par les fonctions d'activation et de transfert dans chaque noeud. Chaque noeud gère un calcul de base alors que leurs liens permettent un calcul global. Le comportement global d'un réseau neuronal est influencé par le nombre de couches, le nombre de neurones dans chaque couche, la façon dont les neurones sont liés et les poids associés à chaque lien. Le poids associé à chaque lien définit comment un premier neurone influence le deuxième neurone. Les poids sont révisés durant la période d'apprentissage. Avec cette approche, les couches masquées captent les complexités dans les données tandis que les poids sont ajustés dans chaque itération afin d'obtenir la plus faible erreur dans la sortie. L'algorithme d'apprentissage utilisé est celui de la rétropropagation du gradient (gradient descent back propagation) (Bengio *et al.*, 2015; Comrie, 1997; Mushtaq *et al.*, 2012).

Dans l'approche de la rétropropagation du gradient et pendant la phase directe, le signal d'entrée est propagé par le réseau couche par couche. Dans le noeud de sortie, le signal d'erreur est calculé et est renvoyé au réseau dans ce qu'on appelle la phase de régression (backward phase). Durant cette phase, les paramètres du réseau sont modifiés afin de minimiser l'erreur de signal (Du *et al.*, 2009). Les méthodes d'apprentissage profond peuvent être utilisées dans les problèmes de régression ainsi que dans les applications de regroupement et de classification (clustering and classification).

Dans la section 7, nous examinons en détail les algorithmes cités auparavant. Nous nous intéressons à leur implémentation et verrons leurs configurations spécifiques ciblant l'utilisation de la régression. Cependant, nous devons tout d'abord examiner les ensembles de données disponibles au public ainsi que l'ensemble de données sur la qualité audiovisuelle propre à notre contexte, qui est

celui de l'INRS. Le but est de voir quel type d'information est disponible lorsque nous essayons de construire des modèles d'estimation de la qualité perçue.

5 Phase expérimentale

Nous avons d'abord généré un ensemble de données de la qualité audiovisuelle, conçu pour inclure la résolution, le débit binaire, la bande passante, le taux de perte de paquets et les facteurs influençant la gigue. Le tableau 4.1 de la page 52 montre les valeurs sélectionnées pour ces facteurs d'influence.

Les séquences audiovisuelles de référence ont été préparées avant l'évaluation en enregistrant des flux vidéo basés sur RTP, transmis sur un réseau émulé. Les vidéos ont été diffusées et enregistrées avec le serveur de vidéo sur demande VideoLan (VLC Team, 2016b) et le lecteur multimédia VLC. L'émulateur de réseau Netem (Hemminger *et al.*, 2005) est déployé afin d'introduire les conditions de test de perte de paquets et de la gigue. Dummynet (Carbone & Rizzo, 2010) est utilisé pour gérer les paramètres de la bande passante entre le serveur VOD et le client. Au total, 144 conditions de réseau ont été envisagées et 144 fichiers audio-vidéo ont été enregistrés pour différents tests subjectifs de la qualité.

En utilisant cet ensemble de données expérimental, nous avons préparé des modèles de qualité à l'aide de méthodes d'apprentissage automatique telles que les forêts d'arbres décisionnels et MLP. Nous avons mesuré les coefficients de corrélation RMSE et de Pearson (Table 1).

Table 1 – Performance des forêts d'arbres décisionnels vs MLP.

Algorithme	RMSE	Corrélation de Pearson	95% de l'intervalle de confiance
forêts d'arbres décisionnels	0.3138	0.8871	0.597
MLP	0.4207	0.8023	0.767

Les valeurs rapportées des coefficients de corrélation RMSE et de Pearson sont proches des valeurs rapportées par d'autres chercheurs, dont le gagnant de la compétition P.NAMS. Cependant, lorsque nous examinons les chiffres, nous voyons clairement un certain nombre de points aberrants où les valeurs d'estimation MOS diffèrent, d'une marge très large, des valeurs MOS réelles. Après avoir soigneusement analysé les valeurs réelles de MOS, nous avons découvert que certains d'entre elles diffèrent par plus de 2 points MOS par rapport à leur gamme réelle attendue. Des problèmes similaires ont également été signalés par Maki *et al.* (2013). La raison derrière cela est la différence entre les paramètres du canal fournis comme une information latérale et les informations de niveau de flux de bits réel. Dans une approche hybride, ce problème ne se produirait pas. En effet, les informations de l'entête du paquet et les informations de niveau de flux de bits plus précises sont utilisées.

D'autres contretemps que nous avons rencontrés sont dus au banc d'essai que nous avons utilisé. Le VLC VOD est un produit décent disponible sur le marché pour des tests simples. Cependant, il ne répond pas aux attentes quand il est utilisé dans des cas de test plus avancés. D'abord, il a un manque de soutien d'une variété de codecs vidéo et audio. Lorsque des dégradations du réseau telles que la perte de paquets sont introduites, il ne parvient pas à capturer complètement un flux vidéo qui subit une légère augmentation du taux de perte de paquets. VLC VOD ne traite certainement pas des cas d'utilisation réels où le taux de perte de paquet vidéo peut aller jusqu'à 5 %. Comme c'est une application, le fait de modifier le comportement du pipeline est presque impossible et

nécessite un changement du code source. Il ne fournit pas non plus des mesures du niveau réseau telles que le taux de perte de paquets, la gigue, le délai, le débit binaire effectif, etc.

Afin de surmonter ces problèmes, nous avions besoin d'un banc d'essai plus robuste pour générer des vidéos de référence avec des encodeurs idéaux ainsi que des paramètres de médias et de canaux. Afin de construire de meilleurs modèles, nous avons recréé notre pipeline multimédia de bout en bout en utilisant le framework GStreamer pour le streaming audio et vidéo. Un pipeline basé sur GStreamer s'est avéré être nettement plus robuste aux dégradations du réseau que le framework VLC VOD. Il nous a permis de diffuser facilement un flux vidéo à un taux de perte de paquets allant jusqu'à 5 %. GStreamer nous a également permis de collecter les statistiques RTCP pertinentes qui se sont révélées plus précises que les informations déduites du réseau. En utilisant ce banc d'essai amélioré, nous avons généré les ensembles de données de qualité audiovisuelle de l'INRS que nous présentons dans la section 6 et la section 8 et qui sont accessibles gratuitement au public. La précision des statistiques nous a finalement aidé à générer des modèles d'estimation de la qualité perçue plus performants que nous introduisons dans les sections 7 et 9.

6 Ensemble de données de la qualité audiovisuelle de l'INRS: version paramétrique

L'ensemble de données de la qualité audiovisuelle de l'INRS a été conçu pour couvrir les principaux facteurs d'influence de la compression et de la distorsion du réseau. Ces facteurs sont généralement le taux d'images vidéo, la quantification, les filtres et le taux de perte des paquets réseau. Suite à notre collaboration avec notre partenaire industriel Summit-Tech Multimedia Communications Inc., nous avons choisi la gamme de ces paramètres pour le codage vidéo H.264 comme suit: (0, 0,1, 0,5, 1 et 5 %) pour le taux de perte des paquets réseau des flux vidéo et audio, (10, 15, 20 et 25 fps) pour le taux d'images vidéo, (23, 27, 31 et 35) pour le paramètre de quantification, et (0 et 999) pour le filtre de réduction du bruit. L'ensemble de données (Demirbilek, 2016b) et les outils (Demirbilek, 2016a,c) utilisés pour créer l'ensemble de données sont accessibles au public à des fins de recherche et de développement.

6.1 Séquences vidéo et configuration des tests

La séquence audiovisuelle originale, fichier ntia_HeadShouldersFemale15_original.avi, a été obtenue à partir de la bibliothèque « the Consumer Digital Video Library » (Pinson, 2013). La vidéo consiste en un contenu tête-et-épaule (head-and-shoulder) avec un discours semblable à une conversation audiovisuelle individuelle typique (typical one-to-one audiovisual conversation). Les approches traditionnelles visent principalement les scénarios IPTV et sont donc constituées de diverses séquences vidéo qui ont des complexités de mouvement différentes. Pour l'ensemble de données sur la qualité audiovisuelle de INRS, nous ciblons toutefois les applications de vidéo-téléphonie qui consistent principalement en un contenu où la complexité des mouvements est très limitée et similaire à des cas d'utilisation. Concernant l'encodage et la fréquence des trames I (I-frames), les contenus tête et épaules (head-and-shoulder) ont tendance à se ressembler. Par conséquent, nous avons porté notre attention sur l'augmentation de la gamme des configurations d'encodage et des altérations du réseau plutôt que les variations dans le contenu head-and-shoulder.

Ce type unique de contenu peut potentiellement ennuyer rapidement les observateurs au cours de l'évaluation subjective. Dans le but d'éviter celà, nous avons divisé l'ensemble de l'expérience en plusieurs sessions et séquences. Nous avons également introduit les critères de rejet lors du post-traitement afin de pouvoir détecter les moments et les périodes d'inattention. Dans la section 6.2, nous discutons en détail de la méthodologie que nous avons suivie.

Un autre paramètre important qui influe sur la qualité globale perçue est l'utilisation fréquente de I-frames. Dans les ensembles de données mentionnés précédemment, il y a une I-frame toutes les 1-2 secondes en raison de la complexité élevée du mouvement ainsi que des altérations réparties de manière uniforme. Cependant, grâce à notre collaboration avec le partenaire industriel, nous savons que la durée entre deux I-trames successives peut aller jusqu'à des minutes pour des cas de communications en temps réel. Dans cette recherche, nous avons conservé la valeur par défaut des périodes I-frame vidéo, 10 s, pour les vidéos à faible mouvement définies par l'encodeur vidéo. Cette durée de la période vidéo I-frame est alors requise pour utiliser une séquence vidéo plus longue que les vidéos traditionnelles de 10-15 s. Le fichier ntia_HeadShouldersFemale15_original.avi a une durée de 42 s. Au cours des tests subjectifs, les observateurs ont été cependant obligés de regarder au moins les dix premières secondes des séquences et ils étaient libres de regarder le reste de la vidéo si nécessaire avant de soumettre leurs évaluations de la qualité. Cette durée a été choisie de manière à être suffisante pour que les observateurs puissent conclure leur évaluation subjective par vidéo. En fin de compte, le type du contenu et la longue durée nous ont obligé à personnaliser notre méthodologie de test (voir section 6.2).

La source audiovisuelle originale brute de 42 secondes a été codée avec le codec vidéo H.264/AVC et le codec audio AMR-WB puis multiplexée dans un conteneur de 3gp en utilisant le framework multimédia open source GStreamer (GStreamer Team, 2016a). Nous avons produit 32 fichiers audio-visuels de référence. Ces fichiers de référence avaient différentes qualités en terme du taux d'images affichées par seconde (FPS), du paramètre de quantification (QP) et des valeurs de réduction du bruit (NR). Ces valeurs sont listées dans la table 6.2 à la page 75. Les flux vidéo ont été encodés avec le profil de ligne de base à une résolution vidéo progressive de 720p. Les paramètres de codage audio ont été conservés (mono channel, 16 KHz sample rate, et 24 Kbps bit rate) pour toutes les séquences audiovisuelles. Le framework multimédia GStreamer n'utilise que le mécanisme jitter-buffer pour réguler le flux de paquets. Il n'a pas de stratégie de dissimulation des paquets et donc dans notre ensemble de données, nous supposons que les chiffres de perte de paquets rapportés correspondent aux pertes de paquets résiduelles.

Un réseau émulé a été utilisé pour transmettre et enregistrer les séquences audiovisuelles. Les flux audio et vidéo ont été capturés avec notre logiciel personnalisé basé sur GStreamer qui nous a permis de rassembler des statistiques détaillées de RTCP et de signaler de façon séparée les valeurs exactes de perte de paquets réseau pour les flux vidéo et audio. Ce logiciel est à accès libre pour le public (Demirbilek, 2016a). L'émulateur de réseau Netem a été déployé pour produire des conditions de perte de paquets réseau. La perte de paquets réseau n'a été activée qu'après la première seconde de la transmission audio et vidéo. Cela nous a permis d'obtenir des résultats plus réalistes. 160 combinaisons uniques de compression et de dégradations du réseau ont été capturées dans des fichiers A/V distincts. Un lecteur vidéo personnalisé a été développé pour recueillir les scores subjectifs (Demirbilek, 2016c).

6.2 Méthodologie de test

Onze femmes et dix-neuf hommes observateurs, âgés entre 20 et 48 ans, ont participé à l'étude. Chaque observateur a reçu des instructions écrites en anglais et a noté l'ensemble de la qualité audiovisuelle sur l'échelle de qualité catégorielle à 5 points. Les observateurs ont terminé les tests dans un environnement isolé avec un casque et un ordinateur. Les conditions de visualisation et d'écoute mentionnées dans ITU-T P.913 (1998) ont été suivies aussi près que possible. Les observateurs ont été autorisés à soumettre leurs scores subjectifs seulement après avoir regardé et écouté les 10 premières secondes de chaque séquence vidéo. Cette période a été sélectionnée en raison des intervalles vidéo I-trames décrits auparavant. L'ordre des séquences présentées a été tiré au sort au cours de la phase de planification de l'évaluation et cette configuration a été sauvegardée et suivie pour tous les observateurs. Les observateurs ont d'abord effectué une séance d'évaluation de la qualité des différents fichiers audiovisuels créés à partir de la même séquence originale. En raison du nombre élevé des conditions de test, les observateurs ont terminé les tests en deux sessions. Chacune des sessions a duré environ 45 minutes et se composait de quatre parties pour permettre aux observateurs d'avoir des pauses fréquentes si nécessaire. À chaque session, dans les deux premières parties, 80 vidéos ont été notées par les observateurs. Dans la troisième et la quatrième partie, les vidéos de la même session précédemment visionnées ont été de nouveau évaluées. Cela nous a permis de recueillir deux scores ACR pour chaque fichier de chaque observateur et de mesurer la cohérence de chaque observateur de manière indépendante. Nous avons rejeté les deux scores ACR pour le même fichier produits par un même observateur lorsque la différence entre les deux scores ACR était supérieure à 1. Nous donnons, dans la figure 6.1 de la page 76, le nombre total des scores acceptés, le temps moyen pour les évaluer ainsi que la corrélation de Pearson entre les scores individuels et les valeurs MOS pour chaque observateur. Lorsque nous examinons les scores détaillés soumis par chaque observateur (voir Tableau B.4 dans l'annexe), spécifiquement pour les observateurs qui ont 10 et plus de points rejetés, nous observons de courtes périodes de temps où des scores sont rejetés en séquence. Cette information montre que les observateurs n'étaient pas très attentifs au test pendant ces courtes périodes de temps. Enfin, au début de la première séance, on a testé l'acuité visuelle normale et la vision des couleurs pour chaque observateur et on a noté que l'observateur ayant l'ID 20 présentait une déficience de couleur rouge-vert.

6.3 Analyse

La variation de la valeur MOS pour les différents taux de perte de paquets, fréquence d'images vidéo, paramètres de quantification et valeurs de réduction du bruit sont données dans la figure 6.3 de la page 79. Nous observons que la variation du taux de perte des paquets réseau a des effets dramatiques sur la qualité perçue. Le taux de trame vidéo et le paramètre de quantification, compte tenu de la plage de valeurs, ont une influence modérée sur la valeur MOS. Si l'on considère les meilleures valeurs pour le taux de trame et le paramètre de quantification les plus faibles qui génèrent le plus petit débit binaire vidéo, il semble que le taux d'images vidéo a un avantage mineur par rapport au paramètre de quantification. Cependant, il est important de se rappeler que cet avantage est faible et est lié à des combinaisons de paramètres spécifiques. Des tentatives de modélisation détaillées révéleraient la relation complexe entre ces paramètres. La modification du paramètre de réduction du bruit a le plus petit effet sur le résultat comparé à la variation des autres paramètres.

La figure 6.4 de la page 81 montre les diagrammes de dispersion des valeurs MOS et leurs intervalles de confiance à 95% pour tous les stimuli traités. Pour des valeurs de perte de paquets

de plus en plus faibles, il existe une légère transition entre les valeurs MOS moyennes lorsque les vitesses d'image vidéo sont modifiées. Pour la gamme de valeurs de perte de paquets les plus faibles et les plus grandes, on trouve une légère transition entre les valeurs moyennes de MOS lorsque les vitesses d'image vidéo sont modifiées. Toutefois, pour les valeurs intermédiaires de perte de paquets , 0.1% et 0.5%, il existe des fluctuations dans les valeurs moyennes de MOS lorsque les vitesses d'image vidéo changent.

7 Modèles paramétriques de la qualité

Nous avons observé, à partir de nos tests expérimentaux de la section 5, la performance globale de l'ensemble des méthodes basées sur les arbres de décision. Avec le nouvel ensemble de données, nous avons étendu cette recherche et avons d'abord mené un processus de dépistage étendu pour trouver des modèles plus performants en utilisant l'atelier Weka de l'Université de Waikato. Les résultats de ce processus de sélection sont donnés dans le tableau 7.1 dans la page 85. Afin d'obtenir ces résultats, nous avons choisi la validation croisée à 10 échantillons en utilisant les paramètres par défaut pour chaque algorithme listé.

Ces résultats confirment notre constatation antérieure selon laquelle l'ensemble des méthodes basées sur les arbres de décision performent mieux que les autres méthodes. Dans le tableau 7.1 de la page 85, les forêts d'arbres décisionnels et les modèles utilisant les techniques de bootstrap performent assez bien en terme des coefficients de corrélation RMSE et de Pearson. Les modèles basés sur les réseaux de neurones ont été largement utilisés dans l'estimation de la qualité audio et vidéo. Afin de comparer les performances du modèle basé sur les arbres de décision avec les méthodes populaires, nous avons développé des modèles basés sur les forêts d'arbres décisionnels, les techniques de bootstrap, l'apprentissage profond et la programmation génétique.

Nous avons formé toutes les méthodes et avons mesuré leur précision en utilisant la validation croisée à 10 échantillons dans le cas des forêts d'arbres décisionnels, des techniques de bootstrap et de la programmation génétique. Nous nous sommes aussi interessés au cas de la validation croisée à 4 échantillons pour l'apprentissage profond.

En outre, nous avons extrait 31 paramètres tels que les débits audio et vidéo, le nombre de trames et les tailles de flux à partir des fichiers d'échantillon via l'outil Media Info Metadata Extraction (Martinez, 2010). Ces paramètres additionnels sont énumérés dans le tableau 6.3, page 77.

7.1 Modèles basés sur les arbres de décision

Nous avons utilisé l'implémentation en Python des forêts d'arbres décisionnels (RF) et des techniques de bootstrap, scikit-learn. Nous avons généré deux modèles basés sur les forêts d'arbres décisionnels et deux modèles basés sur les techniques de bootstrap. Ces modèles utilisent comme caractéristiques soit les paramètres indépendants (5 caractéristiques), soit tous les paramètres extraits (34 caractéristiques). Nous avons par la suite comparé les résultats obtenus. Pour des besoins de simplification, nous nommons le modèle de forêt d'arbres décisionnels qui utilisent tous les paramètres (paramètres indépendants et supplémentaires) par le modèle RF1. Le modèle RF2 va référer au modèle de forêt d'arbres décisionnels qui utilise uniquement les paramètres indépendants. Avec la même logique, nous appelerons le modèle basé sur les techniques de bootstrap et utilisant tous les paramètres (indépendants et supplémentaires) le modèle BG1. Le modèle basé également sur les

techniques de bootstrap et qui utilise uniquement les paramètres indépendants sera nommé par le modèle BG2.

7.2 Modèles basés sur l'apprentissage profond

Nous avons généré les modèles d'apprentissage profond à l'aide de la bibliothèque Keras qui fonctionne à l'aide de la bibliothèque Theano. Nous avons exécuté plusieurs modèles utilisant des variables indépendantes et toutes les autres variables. Nous avons expérimenté avec la grande quantité des configurations disponibles dans l'API Keras, incluant toutes les fonctions d'activation, d'optimisation, d'initialisation, d'objectif et de contrainte. Les modèles d'apprentissage profond qui utilisent uniquement des variables indépendantes ont obtenu de meilleurs résultats que les modèles utilisant toutes les variables. Au cours des expériences, nous avons généré des modèles qui avaient jusqu'à 20 couches cachées. Nous avons constaté que les modèles d'apprentissage profond qui ont une seule couche cachée surpassent les modèles qui ont plus de couches cachées. Cependant, à des fins de comparaison, nous avons inclu un modèle ayant 3 couches cachées puisque nous avons trouvé de tels modèles dans la littérature. Pous faire plus simple, nous appellerons le modèle de l'apprentissage profond n'ayant qu'une seule couche cachée DL1 et le modèle de l'apprentissage profond ayant 3 couches cachées DL2.

Pour les modèles DL1 et DL2, nous avons utilisé les configurations Keras indiquées dans le tableau 7.2 à la page 87. Il est important de noter que trouver les valeurs des variables optimales pour les modèles basés sur l'apprentissage profond est beaucoup plus difficile que d'ajuster les variables pour les modèles basés sur les arbres de décision.

7.3 Modèles basés sur la programmation génétique

Nous avons utilisé la bibliothèque Python gplearn pour implémenter des modèles basés sur la programmation génétique. La bibliothèque gplearn est compatible avec l'API fit/predict de scikit-learn et fonctionne avec le pipeline existant scikit-learn. Comme pour les modèles d'apprentissage profond, plusieurs paramètres sont à modifier. Nous avons utilisé SymbolicRegressor avec les configurations spécifiées dans le tableau 7.3, page 88. Similaires aux modèles basés sur les arbres de décision, nous avons généré deux modèles qui utilisent, comme caractéristiques, soit les paramètres indépendants (5 caractéristiques), soit tous les paramètres extraits (34 caractéristiques) et nous avons comparé les résultats. Dans un souci de simplicité, nous appellerons le nodèle basé sur la programmation génétique et qui utilise tous les paramètres (indépendants et supplémentaires) par le modèle GP1. Le modèle de programmation génétique qui utilise uniquement les paramètres indépendants sera nommé par le modèle GP2.

7.4 Résultats

Bien que tous les algorithmes testés aient bien performé sur la version paramétrique de l'ensemble de données de la qualité de l'INRS, les modèles basés sur les arbres de décision ont surpassé les modèles d'apprentissage profond et de la programmation génétique. Pour le modèle RF1, nous avons obtenu 0,340 et 0,930 en terme de corrélation RMSE et de Pearson. Ces valeurs étaient respectivement de 0,358 et 0,922 pour les modèles RF2, 0,345 et 0,928 pour les modèles BG1 et de 0,355 et 0,925 pour les modèles BG2. Les modèles RF1 et RF2 diffèrent les uns des autres dans le

Table 2 – Valeurs de RMSE et de Corrélation de Pearson pour les modèles paramétriques sans référence.

Nom du modèle	RMSE	Corrélation de Pearson
`Modèle RF1`	0.340	0.930
`Modèle RF2`	0.358	0.922
`Modèle BG1`	0.345	0.928
`Modèle BG2`	0.355	0.925
`Modèle DL1`	0.403	0.909
`Modèle DL2`	0.437	0.894
`Modèle GP1`	0.449	0.881
`Modèle GP2`	0.469	0.870

nombre de caractéristiques qu'ils utilisent. De même, les modèles BG1 et BG2 diffèrent les uns des autres dans le nombre de caractéristiques qu'ils utilisent.

Les modèles basés sur l'apprentissage profond ont également bien performé et ont atteint 0,403 en RMSE et 0,909 pour la corrélation de Pearson pour le modèle DL1. Pour le modèle DL2, le coefficient RMSE est de 0,437 et celui de Pearson atteint 0,894. Rappelons que les modèles DL1 et DL2 diffèrent les uns des autres dans le nombre de couches masquées.

Nous avons aussi obtenu les paramètres de performance pour la programmation génétique. Le coefficient RMSE est de 0.449 alors que le coefficient de corrélation de Pearson atteint 0.881 pour le modèle GP1. Pour GP2, RMSE est égale à 0,469 et la corrélation de Pearson est de 0,870. Rappellons que, selon la section 4.2, la régression symbolique mise en œuvre via la programmation génétique vise à identifier à la fois les paramètres ainsi que la forme de l'expression mathématique sous-jacente. Selon les données d'apprentissage sélectionnées, les expressions mathématiques peuvent avoir des formes différentes dans chaque exécution. L'équation 7.1 à la page 88 montre un exemple d'expression que nous avons obtenue pour l'exécution du modèle GP1. De même, un exemple d'expression pour le modèle GP2 est donné dans l'équation 7.3 à la page 89.

Dans le modèle GP2, la configuration des ensembles de données était composée uniquement de variables indépendantes et nous nous attendions à ce que l'expression mathématique sous-jacente puisse capturer les interrelations de tous les paramètres indépendants. Toutefois, comme le montre l'équation 7.3 à la page 89, le modèle n'a pas pu intégrer tous les paramètres. Nous pouvons affirmer qu'un algorithme n'a pas nécessairement besoin d'utiliser l'ensemble de l'espace des paramètres. Cependant, dans ce cas, étant donné que nous avons très peu de paramètres où chacun a un niveau différent de contribution à la qualité globale perçue, l'absence de taux de perte de paquets audio et des paramètres de réduction du bruit dans l'équation générée soulève plusieurs interrogations. Nous avons généré, à plusieurs reprises, les modèles avec les mêmes configurations et avons rencontré des problèmes similaires pour toutes les exécutions. À chaque fois, certains paramètres n'étaient pas inclus dans l'équation formée.

Le modèle GP1 a souffert de problèmes similaires. Parmi tous les paramètres disponibles, le modèle n'a pu en utiliser qu'un petit sous-ensemble. Dans les deux cas, les équations générées incorporaient non seulement les différents sous-ensembles des paramètres mais avaient aussi des formes significativement différentes, ce qui rendait impossible la formulation d'hypothèses sur les interrelations entre les paramètres. Même en supposant qu'ils auraient surpassé les autres méthodes, alors nous aurions toujours le problème de décider quelle forme de l'équation et quel ensemble de

paramètres utiliser. Dans l'ensemble, d'après nos expériences, il est difficile de conclure si les modèles basés sur la programmation génétique offrent toujours une bonne solution.

Contrairement à la programmation génétique, les modèles basés sur les arbres de décision ont tendance à utiliser tous les paramètres tant qu'ils ne sont pas limités par certaines configurations telles que la profondeur de l'arbre. Les modèles basés sur l'apprentissage profond nécessitent un traitement des paramètres (feature engineering). Traditionnellement, tous les paramètres restants sont utilisés pendant l'apprentissage.

Pour réduire la variation des paramètres, nous avons exécuté chaque modèle 10 fois consécutivement et avons pris la moyenne des métriques statistiques mesurées dans ces 10 exécutions pour chaque indicateur de performance annoncé. Notons qu'avant chaque exécution nous avons mélangé les données pour éviter toute répétition. La figure 7.1 à la page 91 montre le tracé de la courbe de RMSE et de Pearson pour ces modèles, chacun exécuté 10 fois. Les modèles des forêts d'arbres décisionnels avec les modèles basés sur les techniques de bootstrap ont obtenu la plus grande précision en terme de corrélation de Pearson. Le modèle RF1 permet non seulement d'obtenir les meilleurs résultats, mais est aussi plus précis et a moins de variation pour les métriques mesurées, ceci est valable pour toutes les exécutions effectuées.

Une autre manière efficace de visualiser la différence dans la performance est de regarder les graphiques des résidus (residual plots) des modèles. La figure 7.2 à la page 92 présente les performances respectives des modèles RF1, BG1, DL1 et GP1. Ces chiffres montrent que lorsqu'un modèle a une corrélation de Pearson plus élevée et une valeur RMSE plus faible, le graphique respectif a une forme plus compacte avec moins de valeurs aberrantes.

Dans nos expériences, nous avons pu voir que les modèles basés sur les arbres de décision performant mieux. La précision de ces modèles est meilleure et ils nécessitent moins d'effort à générer sans avoir besoin de prétraitement de sélection des caractéristiques. De plus, les forêts d'arbres décisionnels ont une grande importance qui nous aide à mieux comprendre le modèle. La figure 7.3 à la page 92 montre l'importance des modèles RF1 et RF2. Dans les deux modèles, la perte de paquets vidéo influence le résultat plus que d'autres fonctionnalités. Les deux modèles diffèrent cependant dans la façon dont les autres caractéristiques influencent le comportement du modèle. Pour le modèle RF2, nous voyons que la perte de paquets vidéo est le paramètre le plus important suivi par la perte de paquets audio, la quantification et le taux d'images vidéo. La réduction du bruit semble avoir la moindre influence sur le comportement du modèle comparée à d'autres caractéristiques. Pour le modèle RF1, le taux de perte de paquets vidéo reste la caractéristique dominante. Cependant, son influence semble être plus significative par rapport au reste des caractéristiques. De plus, certaines variables supplémentaires influencent le résultat plus que d'autres variables indépendantes. Ce comportement est dû à la corrélation des fonctionnalités. La sélection des caractéristiques des forêts d'arbres décisionnels préfère les variables avec plus de classes. Lorsqu'une des caractéristiques corrélées est utilisée, l'importance des autres caractéristiques corrélées est réduite (Strobl *et al.*, 2007).

7.5 Discussion

Nous avons formé et testé, de manière indépendante, des algorithmes d'apprentissage automatique sur chaque ensemble de données. Les modèles générés sont destinés à des cas d'utilisation spécifiques. Cependant, avec un ensemble de données couvrant plusieurs tests avec des contenus dif-

férents, les modèles basés sur l'apprentissage automatique peuvent également être construits pour des applications générales.

Selon nos expériences, nous pensons que les algorithmes basés sur les arbres de décision sont bien adaptés aux données structurées telles que l'ensemble de données de la qualité audiovisuelle de l'INRS ainsi que d'autres ensembles de données de la qualité accessibles au public. Ces algorithmes obtiennent des performances supérieures par rapport à d'autres algorithmes. Les modèles basés sur l'apprentissage profond et la programmation génétique sont inclus pour mettre les choses dans une perspective car elles sont largement utilisées dans de nombreux domaines.

Les modèles normalisés pour IPTV (ITU-T G.1071, 2015; ITU-T P.1201, 2012) et pour la vidéotéléphonie (ITU-T G.1070, 2012) visent à fournir des modèles à usage général pour autant de cas d'utilisation que possible. Ils utilisent généralement un très petit sous-ensemble de fonctionnalités qui sont disponibles parmi de nombreuses applications telles que le pourcentage de perte de paquets, le taux de trame, le taux de compression et la complexité du contenu pour les modèles bitstream. Cependant, dans cette recherche, nous avons généré des modèles d'estimation de la qualité audiovisuelle perçue en utilisant à la fois le petit sous-ensemble typique de caractéristiques utilisées dans les modèles standardisés et les données corrélées que nous avons extraites des ensembles de données. Sur la base des résultats, nous savons que l'extraction de données corrélées supplémentaires à partir de l'ensemble de données nous aide à générer des modèles plus précis lorsque des algorithmes d'apprentissage automatique appropriés sont déployés. Toutefois, ces données corrélées dépendent des codecs audio et vidéo utilisés, du format de conteneur, des outils utilisés pour extraire les caractéristiques ainsi que d'autres caractéristiques, telle que les bits par pixel, donnée dérivée par des calculs supplémentaires. Le type et la quantité de données corrélées dépendent également des caractéristiques qui sont mesurables dans le réseau. Pour des recherches similaires, nous recommandons de suivre l'approche que nous avons adoptée ici plutôt que de définir des paramètres spécifiques.

La précision des modèles basés sur les arbres de décision diminue légèrement quand il existe seulement un nombre limité de fonctionnalités disponibles, nos tests ont démontré que même dans ces conditions, ces modèles surpassent d'autres algorithmes d'apprentissage automatique. Dans la section de conclusion, nous reviendrons sur cette discussion et donnerons une comparaison plus large des algorithmes d'apprentissage automatique que nous avons utilisés dans la modélisation de la qualité perçue.

Les données corrélées peuvent aider lorsque déployées avec le bon algorithme et ceci ouvre la porte à de nombreuses autres possibilités. À titre d'exemple, les modèles peuvent facilement être étendus pour prendre en compte les positions et les durées des pertes dans les séquences, ce qui les rendraient de bons candidats pour des tâches de surveillance de la qualité. Jusqu'à cette section, nous n'avons pas investigué cet aspect vu que la version paramétrique de l'ensemble de données de qualité audiovisuelle de l'INRS n'inclut pas encore ces extensions bitstream. D'autres exemples d'applications pour lesquelles les modèles basés sur les arbres de décision peuvent être bénéfiques sont la vidéo stéréoscopique et la réalité virtuelle. Le nombre de fonctionnalités pour ces applications est en effet beaucoup plus élevé. La qualité doit être abordée avec des modèles complètement nouveaux où interviennent la perception de la profondeur, la naturalité de la scène et l'inconfort visuel (Raake *et al.*, 2011). De plus, certains algorithmes d'apprentissage automatique ne fonctionnent pas seulement mieux avec les données corrélées mais donnent également un aperçu des données elles-mêmes. Ils nous aident à comprendre et à prioriser certaines fonctionnalités qui sont très utiles pour améliorer le service lui-même.

8 Ensemble de données de la qualité audiovisuelle de l'INRS: version Bitstream

D'après les sections 5, 6 et 7, nous savons que la variation du taux de perte des paquets réseau a des effets dramatiques sur la qualité perçue. De plus, le taux d'images vidéo et le paramètre de quantification ont également une influence modérée sur la valeur MOS et le comportement du modèle. Cependant, pour la version paramétrique de l'ensemble de données de la qualité audiovisuelle de l'INRS, la valeur de ces paramètres n'est rapportée que par fichier uniquement. Nous nous attendons à ce que l'intégration de l'influence de ces paramètres par trames vidéo I et P et par trames audio améliore la précision des modèles de prédiction de la qualité perçue basés sur l'apprentissage automatique. Dans le reste de cette section, nous discutons la version bitstream de l'ensemble de données de qualité audiovisuelle de l'INRS qui inclut le taux de perte de paquets et les changements de débit sur l'individu ou le groupe de trames. Dans la section 9, nous discuterons comment ces extensions de niveau bitstream influencent le modèle généré et si les hypothèses que nous considérons ici s'avèrent vraies.

Dans les sections 5 et 7, nous démontrons que l'utilisation de paramètres dépendants améliore la précision des modèles. À la lumière de ces résultats, nous avons utilisé l'outil multimédia FFmpeg (Bellard, 2005) pour obtenir les débits binaires pour les flux audio et vidéo ainsi qu'un certain nombre de trames et la durée des flux.

En utilisant l'implémentation partielle de l'outil d'estimation de la qualité audiovisuelle UIT-T P.1201.2 (Deutsche Telekom AG : T-LABS, 2016), nous avons calculé les paramètres BitPerPixel et SceneComp pour chaque fichier vidéo de notre ensemble de données de qualité audiovisuelle.

Le travail principal que nous avons effectué est de calculer comment le taux de perte de paquets et le paramètre de quantification influencent la taille et le nombre de trames I et P. Dans un premier temps, nous avons récupéré le nombre de trames vidéo I et P, les trames audio et le pourcentage de perte pendant la transmission. De plus, nous avons rapporté la taille et le pourcentage de perte pendant la transmission des trames I individuellement, comme il n'y a qu'une seule trame I toutes les 10 secondes dans les vidéos de référence.

Nous avons cependant adopté une approche différente pour les trames P. Plutôt que de déclarer chaque trame individuelle, nous avons regroupé ces trames par seconde. Nous avons inclus dans nos calculs le pourcentage de perte et la taille moyenne des trames P. Au cours de nos premières expériences avec la modélisation basée sur l'apprentissage automatique, nous avons réalisé que les valeurs du reporting pour les trois premières périodes I-frame (soit les 30 premières secondes) sont suffisantes pour améliorer les performances du modèle et que les deux périodes I-frame vidéo rendraient simplement le modèle inutilement complexe sans amélioration significative de la précision du modèle. L'importance des 30 premières secondes est également conforme à l'analyse de la section 6 selon laquelle les observateurs n'ont pas regardé les vidéos jusqu'à la fin pendant l'évaluation subjective.

La liste complète des extensions bitstream de l'ensemble de données de qualité audiovisuelle de l'INRS mentionnées ci-dessus est donnée au tableau 8.2 dans la page 103. L'ensemble de données étendu et nouvellement créé contient des paramètres indépendants (5 fonctionnalités) à partir de l'ensemble de données paramétrique et des extensions bitstream que nous avons générées dans cette section.

9 Modèles de qualité bitstream

Nous avons construit des modèles de prévision de la qualité audiovisuelle avec référence réduite, basée sur les algorithmes que nous avons présentés dans la section 7. Avec cette approche, nous espérons trouver les réponses aux deux questions suivantes: 1) L'ajout des extensions bitstream à l'ensemble de données de qualité audiovisuelle de l'INRS permettra-t-il de développer des modèles plus précis d'estimation de la qualité perçue? 2) Comment chaque algorithme individuel va performer comparé aux modèles paramétriques?

Pour répondre à ces questions, nous avons construit des modèles bitstream avec référence réduite en utilisant l'ensemble de données étendu. Pour chaque algorithme, nous avons gardé le même framework logiciel et les mêmes paramètres que ceux utilisés par les modèles paramétriques.

9.1 Modèles basés sur les arbres de décision

Nous avons utilisé l'implémentation en Python des forêts d'arbres décisionnels (RF) et des techniques de bootstrap (BG), scikit-learn. Nous avons généré des modèles de RF et de BG qui utilisent l'ensemble de données étendu avec 125 fonctionnalités. Rappellons que la version paramétrique de l'ensemble de données de qualité audiovisuelle de l'INRS comprend un total de 34 fonctionnalités. Nous nous attendons à des changements statistiquement significatifs dans la performance du modèle avec cette augmentation radicale du nombre de fonctionnalités. Dans les deux modèles, nous avons mis la taille de l'arbre à 200 et max_features à « tous » lors de la recherche de la meilleure répartition. Nous n'avons pas limité la profondeur de l'arbre.

9.2 Modèles basés sur l'apprentissage profond

Nous avons généré les modèles d'apprentissage profond (DL) à l'aide de la bibliothèque Keras qui se trouve sur le dessus du package Theano. Fort de l'expérience décrite dans la section 7, nous savons déjà que les modèles d'apprentissage profond performent mieux seulement avec un ensemble de fonctionnalités indépendantes. Nous nous attendons donc à une diminution de la performance des modèles basés sur l'apprentissage profond présentés ici. Nous avons choisi les configurations de l'API Keras exactement similaires à celles utilisées pour les modèles paramétriques.

9.3 Modèles basés sur la programmation génétique

Nous avons utilisé la bibliothèque en Python gplearn pour implémenter des modèles basés sur la programmation génétique (GP). La bibliothèque gplearn est similaire à l'API fit/predict de scikit-learn et fonctionne avec le pipeline existant scikit-learn. Nous avons utilisé le SymbolicRegressor avec les mêmes configurations des modèles paramétriques. Dans la section 7, les modèles basés sur la programmation génétique bénéficient des données corrélées. Cependant, le processus de consolidation est lent et seuls les tests détaillés révéleraient si l'augmentation significative du nombre de fonctionnalités va améliorer la précision de ces modèles.

Table 3 – Valeurs de corrélation RMSE et de Pearson pour les modèles bitstream à référence réduite.

Nom du modèle	RMSE	Corrélation de Pearson
`Modèle RF`	0.3082	0.9439
`Modèle BG`	0.3091	0.9424
`Modèle GP`	0.4885	0.8550
`Modèle DL`	0.6356	0.8042

9.4 Résultats

Contrairement aux modèles paramétriques, les algorithmes testés n'ont pas tous bien performé sur l'ensemble de données étendu. Les modèles basés sur les arbres de décision ont surpassé largement les modèles d'apprentissage profond et ceux de la programmation génétique. Pour le modèle basé sur les forêts d'arbres décisionnels, les valeurs des coefficients de corrélation RMSE et celui de Pearson valent respectivement 0,3082 et 0,9439. Ces valeurs étaient de 0,3091 et de 0,9424 pour le modèle basé sur les techniques de bootstrap. Les modèles des forêts d'arbres décisionnels et ceux basés sur les techniques de bootstrap ont performé de manière très similaires. Toutefois, le modèle basé sur les forêts d'arbres décisionnels avait un très petit avantage par rapport au modèle basé sur les techniques de bootstrap. Les deux modèles ont également surpassé les modèles formés sur l'ensemble de données paramétriques. À partir de ces résultats, nous pouvons dire que les extensions bitstream nous ont aidés à construire des modèles plus performants.

Contrairement aux modèles basés sur les arbres de décision, les modèles basés sur l'apprentissage profond n'ont pas donné de bons résultats et ont atteint 0,6356 pour RMSE et 0,8042 pour le coefficient de corrélation de Pearson. Ceci est conforme à nos attentes que les modèles basés sur l'apprentissage profond ne fonctionnent pas bien avec un nombre élevé de fonctionnalités et nécessitent une ingénierie des fonctionnalités préalable.

Enfin, nous avons également obtenu les paramètres de performance pour la programmation génétique. Les valeurs de RMSE et de Pearson étaient respectivement de 0,4885 et de 0,8550. La régression symbolique mise en œuvre via la programmation génétique vise à identifier à la fois les paramètres et la forme de l'expression mathématique sous-jacente. Ces expressions mathématiques peuvent, en fonction des données d'apprentissage sélectionnées, avoir des formes différentes avec chaque exécution (Poli *et al.*, 2008). Ces résultats indiquent que les extensions bitstream n'ont pas aidé à obtenir des modèles plus précis compte tenu des mêmes configurations.

Comme nous avons mélangé l'ordre des lignes dans l'ensemble de données avant toute exécution des modèles, les résultats que nous avons obtenus avaient des valeurs différentes dépendamment des ensembles d'apprentissage et de test sélectionnés. Pour réduire la variation des métriques, nous avons exécuté chaque modèle consécutivement 10 fois (avec l'ensemble des données mélangées avant chaque exécution). Nous avons considéré la moyenne des métriques statistiques mesurées dans ces 10 exécutions pour chacun des indicateurs de performance. La figure 8.1 à la page 107 montre le tracé de la courbe RMSE et de Pearson pour ces modèles, chacun exécuté 10 fois. Les modèles basés sur les forêts d'arbres décisionnels et ceux des techniques de bootstrap ont obtenu la plus grande précision en terme de corrélation de Pearson. Ces modèles n'obtiennent pas seulement les meilleurs résultats, mais sont également plus précis et ont moins de variation dans les métriques mesurées pour toutes les exécutions.

Une autre manière efficace de visualiser la différence dans la performance est de regarder les graphiques des résidus. La figure 8.2 sur la page 108 montre les performances respectives des modèles des forêts d'arbres décisionnels, de ceux basés sur les techniques de bootstrap, de la programmation génétique et des modèles d'apprentissage profond. Ces chiffres montrent que lorsqu'un modèle a une corrélation de Pearson plus élevée et une valeur RMSE plus faible, le graphique respectif a une forme plus compacte avec moins de valeurs aberrantes. La différence entre les modèles basés sur les arbres de décision et les autres modèles apparaît également clairement.

Dans nos expériences, nous avons remarqué que les modèles basés sur les arbres de décision ont une meilleure précision et exigent également moins d'effort à générer sans aucune sélection préliminaire des caractéristiques. De plus, les algorithmes de forêts d'arbres décisionnels fournissent un classement important des caractéristiques qui nous aide à mieux comprendre les modèles. La Figure 8.3 à la page 109 présente les 15 caractéristiques les plus influentes pour deux modèles basés sur des forêts d'arbres décisionnels. Rappellons que l'ensemble de données étendu contient 125 fonctionnalités. Ces chiffres appartiennent aux modèles qui ont été formés sur 70% et testés sur les 30% restants des données mélangées. Chaque fois que nous avons construit un modèle basé sur les forêts d'arbres décisionnels, nous avons remarqué des changements mineurs dans l'ordre des influences des fonctionnalités. Les facteurs les plus importants dans les modèles générés sont le taux de perte de paquets vidéo et le pourcentage de perte dans le calcul de trames P. De plus, nous observons que le pourcentage de pertes dans le nombre de trames P rapporté par seconde a une influence notable sur la performance du modèle. L'ordre des secondes dans les chiffres indique une autre conclusion assez importante: presque toutes les périodes les plus influentes viennent juste après la transmission d'une trame I. Cela reflète l'effet cumulatif de la perte de paquets et l'importance de la perte de paquets au début d'une période vs au milieu ou à la fin de périodes entre deux trames I.

Une autre constatation importante est l'absence de certains paramètres dans les 15 listes de caractéristiques les plus influentes. La quantification, le taux de trames vidéo et la réduction du bruit ont moins d'influence sur le comportement du modèle comparés à d'autres caractéristiques. Ce comportement est dû à la corrélation des caractéristiques: la sélection de caractéristiques des forêts d'arbres décisionnels préfère les variables avec plus de classes et lorsque l'une des caractéristiques corrélées est utilisée, les autres caractéristiques corrélées deviennent moins importantes (Strobl *et al.*, 2007). Nous observons également l'absence des fonctionnalités BitPerPixel et Scene-Complexity que nous avons calculées dans la section 8. Ce comportement est dû à l'ensemble de données de qualité audiovisuelle de l'INRS qui ne comprend qu'un seul type de contenu.

9.5 Discussion

Les modèles normalisés pour les services IPTV (ITU-T G.1071, 2015; ITU-T P.1201, 2012) et la téléphonie vidéo (ITU-T G.1070, 2012) visent à fournir un modèle pour autant de cas d'utilisation que possible. Ils utilisent généralement un très petit sous-ensemble des fonctionnalités disponibles telles que le pourcentage de perte de paquets, le taux d'images, le taux de compression et la complexité du contenu pour les modèles bitstream. Cependant, dans cette recherche, nous avons généré des modèles d'estimation de la qualité audiovisuelle perçue en utilisant à la fois le petit sous-ensemble typique de caractéristiques utilisées dans les modèles standardisés ainsi que les données corrélées que nous avons extraites des vidéos incluses dans l'ensemble de données publiquement disponibles. Sur la base des résultats que nous avons obtenus dans cette section ainsi que des résultats rapportés dans les sections 5 et 7, nous savons que l'extraction de données corrélées supplémentaires de

l'ensemble de données nous aide à générer des modèles plus précis. Ces données corrélées dépendent toutefois de divers facteurs tels que les codecs utilisés pour l'audio et la vidéo, l'outil utilisé pour extraire les données corrélées et le nombre de caractéristiques. Pour des recherches similaires, nous recommandons de suivre l'approche que nous avons adoptée ici plutôt que de définir des paramètres spécifiques.

10 Résumé

Notre ambition initiale était de construire des modèles d'estimation de la qualité pour surveiller et maximiser la qualité globale perçue d'une communication audiovisuelle en temps réel. L'approche classique de la modélisation de la qualité audiovisuelle consiste à développer des fonctions permettant de prédire la qualité audio et vidéo de manière indépendante, puis de les combiner en utilisant une autre fonction pour prédire la qualité audiovisuelle perçue globale. L'approche alternative est de construire des modèles qui prédisent la qualité audiovisuelle directement sans passer par des fonctions intermédiaires. Les approches de modélisation par apprentissage automatique ont été appliquées avec succès à l'estimation de la qualité perçue. Dans cette recherche, nous avons adopté la deuxième approche et avons construit des modèles basés sur l'apprentissage automatique qui prédisent la qualité audiovisuelle globale dans une seule fonction. Cette dernière peut être utilisée pour la surveillance ainsi que l'amélioration de la performance du système en boucle fermée.

Afin d'atteindre nos objectifs, nous avons initialement créé un banc d'essai basé sur VLC VOD et avons généré un ensemble de données expérimentales. Nous avons élaboré des modèles basés sur deux méthodes d'apprentissage automatique; les forêts d'arbres décisionnels et les réseaux de neurones. Les connaissances que nous avons recueillies dans cette phase expérimentale nous ont permis de créer un banc d'essai plus robuste basé sur le framework multimédia GStreamer. Ce banc d'essai est capable de générer un ensemble de données reflètant les configurations contemporaines en temps réel pour la fréquence d'images vidéo, la quantification vidéo, les paramètres de réduction du bruit et le taux de perte de paquet réseau.

Nous avons ensuite utilisé l'ensemble de données de l'INRS - qui inclut les séquences audiovisuelles et leur qualité correspondante - au cours des phases d'apprentissage, de validation et de test. Nous avons construit plusieurs modèles d'estimation de la qualité perçue utilisant l'apprentissage machine. L'ensemble de données sur la qualité de audiovisuelle de l'INRS se compose de deux versions: une version paramétrique où les données d'apprentissage et de test sont obtenues à partir des couches d'application et de réseau, aucune information concernant les signaux originaux n'est utilisée. La deuxième version est bitstream où des données additionnelles provenant de la couche bitstream ainsi qu'une quantité réduite de l'information provenant du signal original sont utilisées.

À partir de ces ensembles de données, nous avons développé des modèles d'estimation de la qualité perçue paramétrique sans référence et bitstream avec référence réduite. Nous nous sommes basés sur les méthodes des forêts d'arbres décisionnels, des techniques de bootstrap, de l'apprentissage profond et de la programmation génétique.

Pour les modèles paramétriques, nous avons utilisé la version paramétrique de l'ensemble de données de l'INRS; toutes les méthodes testées ont obtenu une grande précision en terme de RMSE et du coefficient de corrélation de Pearson. Les modèles des forêts d'arbres décisionnels et des techniques de bootstrap montrent un petit avantage par rapport à l'apprentissage profond quant à la précision qu'ils fournissent sur l'ensemble de données utilisé. Les modèles basés sur la programma-

tion génétique ont moins bien performé même si leur précision est impressionnante. Nous avons également obtenu une grande précision sur les autres ensembles de données de la qualité audiovisuelle disponibles au public. Les métriques de performance sont comparables aux modèles existants formés et testés sur ces ensembles de données.

Les algorithmes des forêts d'arbres décisionnels ne performent pas seulement mieux avec les données corrélées, mais fournissent également des informations sur les données elles-mêmes et nous aident à comprendre et à prioriser certaines caractéristiques qui sont très utiles pour améliorer le service lui-même.

Sachant que les données corrélées peuvent aider lorsque déployées avec le bon algorithme, ceci ouvre la porte à diverses possibilités. Par exemple, les modèles peuvent facilement être étendus pour considérer les positions et les durées de perte dans les séquences, ce qui les rendraient de bons candidats pour la surveillance de la qualité. Ce potentiel nous a conduit à développer des modèles bitstream.

Pour les modèles bitstream, nous avons utilisé la version bitstream de l'ensemble de données de l'INRS; les modèles basés sur les forêts d'arbres décisionnels et les techniques de bootstrap ont surpassé à la fois les modèles basés sur l'apprentissage profond et la programmation génétique quant à la précision qu'ils ont obtenue. De plus, ces modèles bitstream basés sur des arbres de décision ont obtenu de meilleurs résultats comparés aux modèles paramétriques. Cependant, les modèles bitstream basés sur la programmation génétique et l'apprentissage profond ont moins bien performé comparés aux modèles paramétriques en raison d'une augmentation significative du nombre de caractéristiques dans l'ensemble de données bitstream.

Dans l'ensemble, nous concluons que le calcul de l'information bitstream mérite l'effort fourni pour la générer. Ce calcul aide à construire des modèles plus précis. Il est toutefois utile uniquement pour le déploiement des bons algorithmes.

Sur la base des résultats, nous savons que l'extraction de données corrélées supplémentaires de l'ensemble de données nous aide à générer des modèles plus précis lorsque les bons algorithmes d'apprentissage automatique sont déployés. Cependant, ces données corrélées dépendent des codecs audio et vidéo utilisés, du format du conteneur, des outils utilisés pour extraire les caractéristiques ainsi que de toute autre caractéristique, telle que bits par pixel, obtenue par des calculs supplémentaires. Le type et la quantité des données corrélées dépendent également des caractéristiques qui sont mesurables dans le réseau. Pour des recherches similaires, nous recommandons de suivre l'approche que nous avons adoptée ici plutôt que de définir des paramètres spécifiques.

Nous avons formé et testé des algorithmes d'apprentissage automatique indépendamment sur chaque ensemble de données. Les modèles générés sont destinés à être utilisés uniquement dans des cas d'utilisation spécifiques. Cependant, avec un ensemble de données couvrant plusieurs tests avec différents contenus, codecs et équipements de test, les modèles basés sur l'apprentissage automatique peuvent également être développés pour des applications générales.

Les recherches que nous avons présentées ici peuvent être étendues de plusieurs façons. En plus de s'étendre à divers contenus, codecs et équipements de test, de nombreuses dimensions de modélisation de la qualité audiovisuelle peuvent être abordées. Nous pouvons citer à titre d'exemple, la selection d'un taux de perte de paquets plus élevé pour des flux audio, la synchronisation entre les flux audio, la méthodologie suivie et l'échelle de notation.

En guise d'alternative, les données peuvent être collectées à partir du système cible soit dans un environnement contrôlé, soit directement auprès des utilisateurs finaux via le crowdsourcing. Dans

ce cas, la campagne d'évaluation subjective devra être reprise intégralement depuis le début, avec des exigences différentes pour le nombre d'observateurs, la méthodologie suivie, le nettoyage des données, etc.

Une autre extension possible pour cette recherche est de regarder au-delà de la qualité perçue, vers la qualité d'expérience. Pour des raisons pratiques, nous nous sommes limités principalement aux facteurs d'influence du système. Si nous prenons en considération le contexte et les facteurs d'influence humaines, les exigences de l'ensemble de données vont changer aussi bien que la méthodologie suivie lors de l'évaluation subjective et l'approche de modélisation adoptée. L'existence d'un ensemble de données de qualité audiovisuelle en libre accès et reflètant ces facteurs d'influence QoE serait extrêmement utile pour le milieu de la recherche.

Les applications émergentes dans les systèmes multimédias qui perçoivent la qualité n'ont pas encore reçu la même attention. Deux de ces applications sont la vidéo 3-D et la vidéo stéréoscopique. Bien que les différentes métriques 2D de la qualité objective puissent être appliquées aux couleurs et à la profondeur de l'image, aux vues de gauche et de droite d'une vidéo stéréoscopique, aucune métrique objective d'évaluation de la qualité vidéo stéréoscopique n'existe (Hewage *et al.*, 2009). En plus des facteurs d'influence du système que nous avons analysés auparavant, la profondeur perçue, le caractère naturel, la distance de base de la caméra et l'inconfort visuel jouent des rôles importants dans ces types d'applications.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ACR	Absolute Category Rating
ANN	Artificial Neural Network
$B-frame$	Bi-Predictive Picture Frame
BP	Back-Propagation
CIF	Common Intermediate Format
D	Delay
$DiffServ$	Differentiated Services
DL	Deep Learning
DT	Decision Tree
FEC	Forward Error Correction
FPS	Video Frame Rate
FR	Full-Reference
GP	Genetic Programming
HD	High Definition
HMM	Hidden Markov Model
HR	Higher Resolution
$I-frame$	Intra-Coded Picture Frame
ICT	Information and Communications Technology
$IETF$	Internet Engineering Task Force
IF	Influence Factor
$iLBC$	Internet Low Bitrate Codec
$IntServ$	Integrated Services
IP	Internet Protocol
IQX	Exponential interdependency of QoE and QoS
ITU	The International Telecommunication Union
J	Jitter

KNN	K-Nearest Neighbours
LDA	Linear Discriminant Analysis
LR	Lower Resolution
LSR	Least Squares Regression
MB	Macroblock
ML	Machine Learning
MLP	Multi-Layer Perceptron
MOA	Massive Online Analysis
MOS	Mean Opinion Score
NB	Naive Bayes
NN	Neural Network
NR	No-Reference
NR	Noise Reduction
OWD	One Way Delay
$P-frame$	Predictive Picture Frame
PLR	Packet Loss Rate
$QCIF$	Quarter Common Intermediate Format
QoE	Quality of Experience
QoS	Quality of Service
QP	Quantization Parameter
$QQVGA$	Quarter-Quarter Video Graphics Array
$Qualinet$	The European Network on Quality of Experience in Multimedia Systems and Services
$QVGA$	Quarter Video Graphics Array
R	Throughput
RF	Random Forest
$RMSE$	Root Mean Square Error
RNN	Regression Neural Networks
RR	Reduced-Reference
$RSVP$	Resource Reservation Protocol
$RTCP$	RTP Control Protocol
RTD	Round Trip Delay
RTP	Real-time Transport Protocol
SD	Standard Definition

SLA	Service Level Agreement
SVM	Support Vector Machines
$VoIP$	Voice over IP
VQA	Visual Quality Assessment
VQM	Video Quality Metric
WFL	Weber-Fechner Law

Chapter 1

Introduction

One of the standard quality assessment methods of multimedia communications is called Quality of Service (QoS), which is based on the measure of parameters such as throughput, delay, jitter and packet loss rate (Perkis, 2016). When analyzing the characteristics of a real-time multimedia communication, it is necessary to have a clear understanding of changes in these parameters occurring in the layers that enable the end-to-end communications. This is of particular importance in environments where the multimedia streams are traversing resources that have limited and fluctuating bandwidth capacity, such as wireless networks. In a wireless network, limits and fluctuations of the network capabilities are related to the changes in network topology, mobile device and sensors fading in and out of range and environmental interference. These characteristics cause the networks to have unstable capabilities (Hansen *et al.*, 2013).

In order to cope with ever-changing situations during a multi-party multimedia interaction, one of the standard approaches is to periodically measure various QoS parameters and feed them back to the transmission end-point in order to control the rate and characteristics of the streamed multimedia (Figure 1.1). Such an adaptation model generally improves the quality of audiovisual communications. The QoS measurements can also be reported for quality monitoring.

There has been significant research about the QoS feedback control in both application and network layers and various management tools and approaches have been developed (Bordetsky *et al.*, 2001). However, in recent years, the convergence of the digital media industry and the information

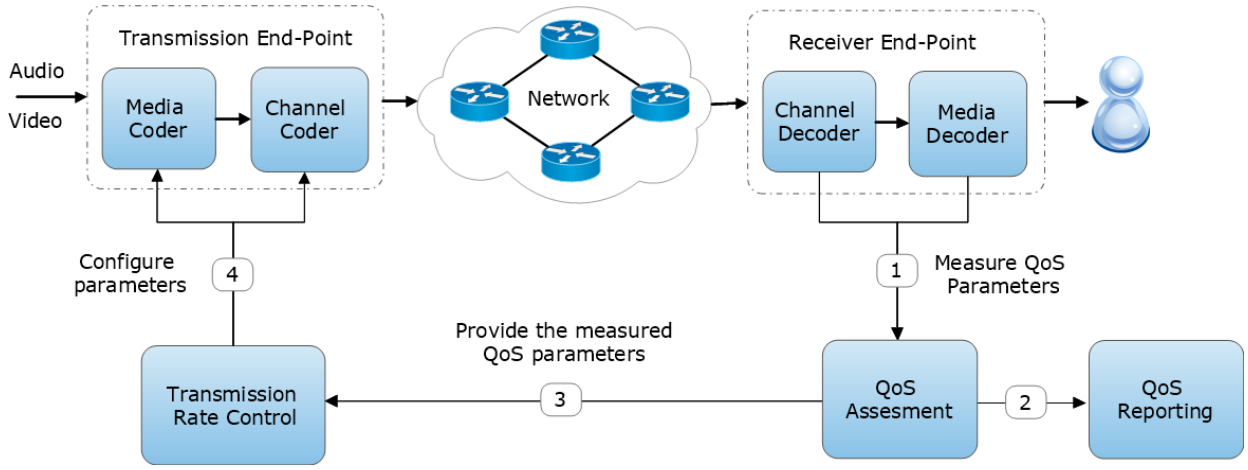

Figure 1.1 – QoS based feedback.

and communications technology (ICT) industry has led to a paradigm shift away from QoS towards the perceived multimedia presentation quality (Perkis, 2016).

While QoS primarily measures the accuracy of networked data delivery, there are additional factors affecting the user's perception of multimedia presentation quality (Perkis, 2016).

In a perceived quality based multimedia adaptation, although a better QoS usually provides a higher presentation quality, there are cases where QoS trade-offs are needed to maximize the overall perceived quality (Hansen *et al.*, 2013). These trade-offs need to be made statically and dynamically for a real-time communication. When the factors that influence the perceived quality are well understood, such as resolution over frame rate for object identification, statistical changes can be made (Hansen *et al.*, 2013). An example in (Vakili & Grégoire, 2013) shows a control mechanism for limited bandwidth situations by investigating the effect of different frame rates and compression levels on video streaming bit rate, and consequently on perceived quality.

Figure 1.2 depicts a sample architecture of a multimedia communication system where QoS parameters are measured at regular intervals and reported for monitoring as well as for parameter tuning to the transmission end-point along with the perceived quality predictions. Here, the audio and video data stream generated and encoded on the transmission end-point and then forwarded to the network. On the receiver end-point, the data stream is first decoded at the channel level and then at the media level, and finally presented to the end-user in conventional ways. While this process is taking place, in a closed loop, these measured QoS parameter values, and the quality predictions are evaluated by the transmission rate control, and media and channel encoding parameters are

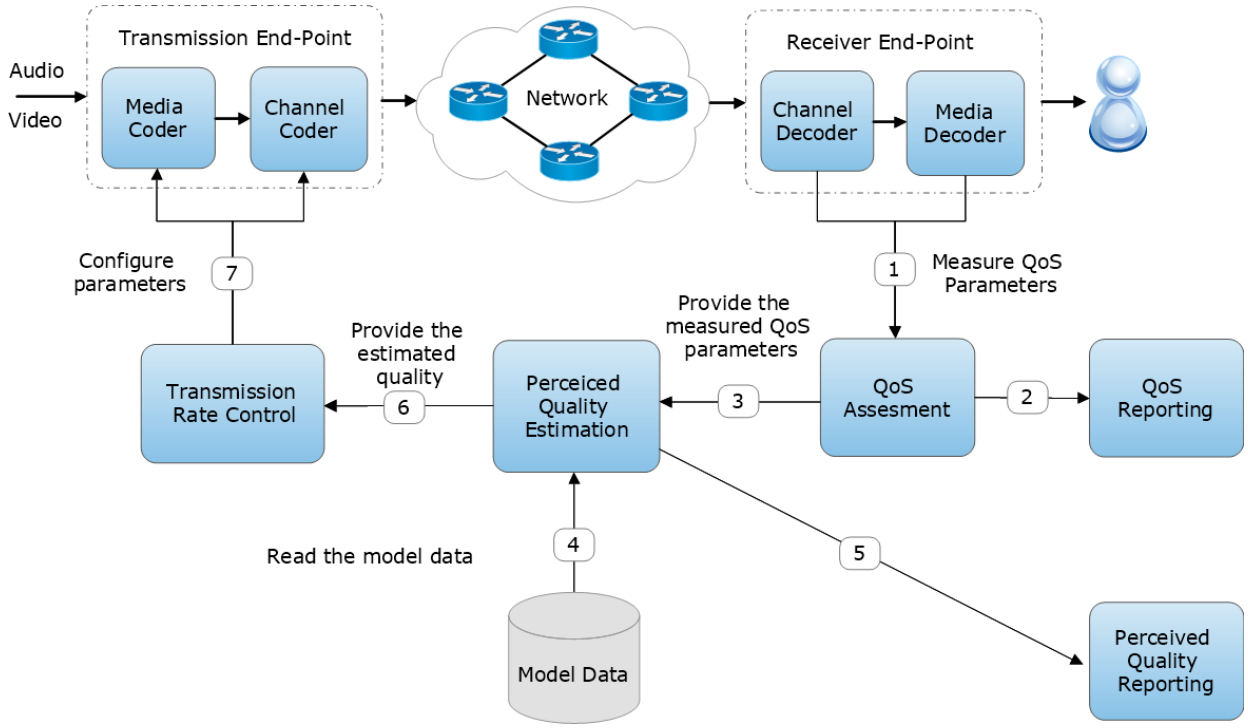

Figure 1.2 – Perceived quality based feedback.

modified to improve the user experience. Service providers can enhance their service monitoring and reporting processes dramatically by exploiting these quality predictions.

This sample architecture assumes the perceived quality modeling is already implemented properly which is the main purpose of this research.

1.1 Objectives and Scope

The classic approach to audiovisual quality modeling is to develop functions to predict audio and video quality independently and then combine them using another function to predict the overall audiovisual perceived quality. The alternative way is to build models that predict the audiovisual quality directly without any intermediate functions. Machine learning based modeling approaches have been successfully applied to estimating perceived quality (Maki *et al.*, 2013; Alreshoodi & Woods, 2013; Aroussi & Mellouk, 2014; Chen *et al.*, 2014). In this research, we have taken the second approach and have built machine learning based models that predict the overall audiovisual quality in one single function.

In order to build perceived quality estimation models, first there is a need to create a testbed that provides the full control over the complete media pipeline. Similar testbeds and some useful tools have been proposed by (Gastaldo *et al.*, 2013; Schmitt *et al.*, 2013; Kuipers *et al.*, 2010; Menkovski *et al.*, 2010b).

Having a testbed would enable us to generate reference audiovisual resources required for subjective quality assessment and to generate a dataset with quality ratings. Then we would be able to build perceived quality estimation models based on this dataset. For the perceived quality modeling, Chen *et al.* (2014) have expressed some general requirements which are in parallel to the research objectives we have:

- Reliable. Models based on independence of the QoS variables tend not to be very accurate. Proposed models should consider interrelations between these parameters and provide reliable quality predictions.
- Expressive. In ML approaches, researchers tend to give priority to the regression models due to the simplicity of the models. However, these models may be problematic since they pre-assign a certain relationship (linear, logistic, etc.) between QoS and perceived quality. Proposed models should be expressive enough to capture the complex and non-monotonic relationship between QoS and perceived quality.
- Real-time. There is the lack of real-time prediction models. While generating these models, it is important that the computational complexity and storage requirement of these models be acceptable.
- Scalable. The model should be able to readily take new variables and give a relatively accurate result as the network and user experience evolves with time.

1.2 Research Questions and Challenges of the Research

This research spans both the tools and the audiovisual quality dataset generated as well as machine learning-based models generated using this dataset. Questions regarding the creation of the tools and the dataset are more technical and discussed in the respective chapters. However, for the modeling, we have some definite questions and expect to be able to answer them as our research progresses. Some of these questions are as follows:

- How well can perceived quality estimation models be built for real-time multimedia communications on the basis of the application of machine learning-based methods?
- What is the accuracy of such models while predicting the subjective quality from objective metric values?
- How well do different ML algorithms perform over predicting the perceived quality in real-time audiovisual communication?
- Is there a universally superior ML algorithm or approach that can be applied in various configurations of the audiovisual communication?

To our knowledge and based on the literature review from (Singh & Aggarwal, 2014; Mitra *et al.*, 2014), we expect to face the following issues and limits in the research:

- The accuracy of objective audio and video quality metrics is still not good enough to replace the subjective testing.
- Perceived quality measurements may involve a large parameter space comprising several quality and context parameters. The relationships between these parameters are usually nonlinear and hard to quantify.
- Perceived quality evolves over time and with the repeated use of a service, the user perception of the quality may change. Quality measurement and prediction at a single point in time may not yield correct results and should be performed over a period of time (several days, weeks or months depending on the service or application requirements).
- For real-time applications specifically, the computational and resource complexity of metrics is very high.
- Metrics and models we build will have limited scope to use. Their performance degrades when they are used out of scope.
- The intention is to automate human perception which is not fully understood yet.
- In order to control and optimize video processing systems, frameworks are needed. Metrics we obtain are only providing the quality scores.
- Our research will be limited to available databases which for practical reasons have limited configurations.

1.3 Research Contribution and Results

In order to fulfill our goals, we have initially created a VLC VOD based testbed and have generated an experimental dataset, and have built Random Forests and Neural Networks based machine learning models. The knowledge that we have gathered in this experimental phase has enabled us to create a more robust testbed based on the GStreamer multimedia framework that is capable of generating a dataset that reflects contemporary real-time configurations for video frame rate, video quantization, noise reduction parameters and network packet loss rate.

Taking advantage of the GStreamer based testbed, we have generated the INRS audiovisual quality dataset that includes both the audiovisual sequences and their corresponding quality rating. The INRS audiovisual quality dataset consists of two instances; a parametric version where the training and test data is obtained from the application and network layers and no information regarding the original signals is used, and a bitstream version where additionally data from bitstream layer as well as reduced amount of information from the original signal is used. Both the dataset and the tools used to create the dataset are publicly available for research and development purposes and are listed at the end of this chapter.

Then we have used the INRS dataset during the training, validation and test phases and have built several machine learning based perceived quality estimation models. We have built No-Reference parametric and Reduced-Reference bitstream perceived quality estimation models based on the Random Forests, Bagging, Deep Learning and Genetic Programming methods. We have primarily used the INRS audiovisual quality dataset which includes various media compression and network distortion degradations typically seen in real-time communications.

For parametric models, we have used the parametric version of the INRS dataset and found that all of the mentioned methods have achieved high accuracy in terms of RMSE and Pearson correlation. Random Forests and Bagging based models show a small edge over Deep Learning with respect to the accuracy they provide on the INRS dataset we have used. Genetic Programming based models fell behind even though their accuracy is impressive as well. We have also obtained high accuracy on the other publicly available audiovisual quality datasets and the performance metrics are comparable to the existing models trained and tested on these datasets.

For bitstream models, we have used the bitstream version of the INRS dataset and found that Random Forests and Bagging based models have outperformed both the Deep Learning and Genetic Programming based models with respect to the accuracy they provide on the extended dataset that we have used. Further, these Decision Trees based models have performed better compared to the parametric models. However, both the Genetic Programming and Deep Learning based bitstream models fell behind the parametric models due to a significant increase in the number of features in the bitstream dataset. Overall we conclude that computing the bitstream information is worth the effort it takes to generate and helps to build more accurate models. However, it is useful only for the deployment of the right algorithms.

Considering both the parametric models and bitstream models, we conclude that the Decision Trees based algorithms are well suited to the No-Reference parametric models as well as to the Reduced-Reference bitstream models.

In light of these results, our contribution to the research can be summarized as follows:

- The INRS audiovisual quality dataset that consists of a parametric and a bitstream version. This dataset includes reference videos, transmitted videos, detailed and consolidated quality scores and all parameters.
- Machine Learning based No-Reference parametric and Reduced-Reference bitstream perceived quality estimation models.
- Open source resources: VLC and GStreamer based testbeds, subjective assessment player, configuration scripts and machine learning codes for experimental, parametric and bitstream models.

1.4 Structure of this Thesis

In Chapter 2 and Chapter 3, we present some background information on various aspects of the audiovisual quality modeling and widely used machine learning algorithms and concepts. We explain various perceived quality modeling approaches, statistical metrics used for evaluating model performance and best practices used in modeling when measuring test accuracy over a relatively small set of data in Chapter 2. In this chapter, we also explain the standardized audiovisual quality prediction models and describe how these models differ from the approach we have taken in this

research. This chapter is also very helpful to understand the audiovisual quality dataset configurations that we introduce in Chapter 6. In Chapter 3, we go through some of machine learning algorithms and concepts widely used in perceived quality modeling and describe the algorithms we use in this research in more detail.

Chapter 4 explains our preliminary research towards building perceived quality estimation models. Both the dataset and the models generated in this chapter provide some inside information on how to generate a dataset with more realistic configurations and how to build more accurate models. Both the dataset and the models in this chapter should be taken as an experimental phase rather than the eventual outcome of the research.

The testbeds that we have used for generating the main and experimental datasets are introduced in Chapter 5. This chapter introduces a VLC VOD based testbed that we have used during the experimental phase and a GStreamer multimedia framework based testbed that we have used to generate our main dataset. In this chapter, we also compare our experience using both of the technologies and share some lessons that we have learned along the way. Both testbeds are open source.

We list existing publicly available audiovisual quality datasets in Chapter 6 and explain the INRS audiovisual quality dataset we have generated for this research in details. In this chapter, we introduce only the parametric version of this dataset.

In Chapter 7, we introduce No-Reference parametric models that we have built and share the results obtained for Random Forests, Bagging, Genetic Programming and Deep Learning methods using the INRS audiovisual quality dataset. In this chapter, we also share the results for the Random Forest based models trained and tested on the publicly available datasets.

In Chapter 8, we introduce Reduced-Reference bitstream audiovisual quality prediction models that are generated using the bitstream extensions of the INRS audiovisual quality dataset. As these bitstream extensions are not available in the parametric version introduced in Chapter 6, we present the extensions and the motivation and method for generating these extensions in this chapter.

Chapter 9 reemphasizes the motivation of the research, discusses various aspects of the models we have introduced and sums up the research.

1.5 Publications and Open Source Resources

Some parts of this thesis are based on the content presented in the following journals, conferences, and publications:

- Edip Demirbilek and Jean-Charles Grégoire. Towards reduced reference parametric models for estimating audiovisual quality in multimedia services. *IEEE International Conference on Communications (ICC), 2016,* IEEE.
- Edip Demirbilek and Jean-Charles Grégoire. Multimedia communication quality assessment testbeds. *arXiv preprint arXiv:1609.06612, (2016).*
- Jean-Charles Grégoire. Multimedia communication quality assessment testbed. *GStreamer Conference, 2016.*
- Edip Demirbilek and Jean-Charles Grégoire. INRS audiovisual quality dataset. *Proceedings of the 2016 ACM Multimedia Conference, 2016.*
- Edip Demirbilek and Jean-Charles Grégoire. Machine learning based parametric audiovisual quality prediction models for realtime communications. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)(Accepted).*
- Edip Demirbilek and Jean-Charles Grégoire. Machine learning based bitstream audiovisual quality prediction models for realtime communications. *IEEE International Conference on Multimedia and Expo, 2016 (Submitted).*
- Edip Demirbilek and Jean-Charles Grégoire. Perceived quality prediction models: Taking advantage of correlated data. *Springer Quality and User Experience Journal: Topical Collection on Managing QoE of Future Networks and Applications (Submitted).*

The INRS audiovisual quality dataset, the testbeds and all other tools and machine learning codes are publicly available at GitHub repository:

- *GStreamer Multimedia Quality Testbed.*
  https://github.com/edipdemirbilek/GStreamerMultimediaQualityTestbed
- *VLC Multimedia Quality Testbed.*
  https://github.com/edipdemirbilek/VLCVODMultimediaTestbed
- *Subjective Assessment Video Player.*
  https://github.com/edipdemirbilek/SubjectiveAssesmentVideoPlayer

- *The INRS Audiovisual Quality Dataset*
  https://github.com/edipdemirbilek/TheINRSAudiovisualQualityDataset
- *Machine Learning Models.*
  https://github.com/edipdemirbilek/MachineLearning

Chapter 2

Perceived Quality

In this chapter, we first look at various aspects of multimedia communications quality such as Quality of Experience (QoE), subjective quality assessment and perceived quality modeling.

In the second part of the chapter, we present recommendations relevant for conducting research similar to ours. We explain various perceived quality modeling approaches, statistical metrics used for evaluating model performance and best practices used in quality modeling when measuring test accuracy over a relatively small set of data. In this part, we also explain the standardized audiovisual quality prediction models and describe how these models differ from the approach we have taken in this research. This part is also very helpful to understand the audiovisual quality dataset configurations that we introduce in Chapter 6. It is important to note that, we will only discuss the relevant recommendations, best practices, and standards and by no means it should be regarded as a comprehensive survey.

Last part of the chapter is dedicated to the audiovisual quality elements and features such as Delay/Latency, Jitter/Deviation in delay, Packet loss rate, Throughput and Bandwidth.

This chapter is partly based on the content presented in the following conference and journals:

– Edip Demirbilek and Jean-Charles Grégoire. Machine learning based parametric audiovisual quality prediction models for realtime communications. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)(Accepted).*

– Edip Demirbilek and Jean-Charles Grégoire. Machine learning based bitstream audiovisual quality prediction models for realtime communications. *IEEE International Conference on Multimedia and Expo, 2016 (Submitted).*
– Edip Demirbilek and Jean-Charles Grégoire. Perceived quality prediction models: Taking advantage of correlated data. *Springer Quality and User Experience Journal: Topical Collection on Managing QoE of Future Networks and Applications (Submitted).*

2.1 Perceived Quality Modelling

The European Network on Quality of Experience in Multimedia Systems and Services (Qualinet) whitepaper (Le Callet *et al.*, 2012) defines the Quality of Experience (QoE) as follows: *Quality of Experience (QoE) is the degree of delight or annoyance of the user of an application or service. It results from the fulfillment of his or her expectations with respect to the utility and/or enjoyment of the application or service in the light of the user's personality and current state.*

Additionally, it defines characteristics of a user, service, application, or context whose actual state or setting may have an influence on the QoE. These influence factors may be interrelated but still can be broadly classified into three categories, namely Human Influence Factors (IF), System Influence Factors, and Context Influence Factors.

Human IFs are complex and strongly interrelated and related to demographic and socio-economic background, physical and mental constitution, or emotional state (Le Callet *et al.*, 2012).

System IFs describe the properties and characteristics that determine the technically produced quality of an application or service. These factors are related to media capture, coding, transmission, storage, rendering, and reproduction/display, as well as to the communication of information itself—from content production to user—and can be further sub-grouped into 4 categories: (1) Content-related System IFs, (2) Media-related System IFs, (3) Network-related System IFs and (4) Device-related System IFs (Le Callet *et al.*, 2012).

Context IFs are related to situational properties that describe the user's environment in terms of physical, temporal, social, economic, task, and technical characteristics (Le Callet *et al.*, 2012).

The definition for QoE and its characteristics described by the Qualinet whitepaper is one of the most recent standardized works. In the literature, there have been many attempts to model, predict or find a correlation between QoE and IFs. Most authors concentrate on network-oriented QoS parameters such as Packet Loss Rate (PLR), Jitter (J), Delay (D), and Throughput (R) that are a subset of the System IFs (Paudyal *et al.*, 2014; Maia *et al.*, 2014). In this research we will also focus on System IFs and therefore use the term "perceived quality" rather then "Quality of Experience" as the IFs we analyze are only a subset of all potential IFs.

Collecting the opinion of a set of users via subjective experiments is one of the standard ways of measuring the perceived quality of telecommunication systems. However, conducting such experiments is costly and time-consuming, and impossible to perform for real-time communications (Li *et al.*, 2014), which is the focus of our research. An alternative approach is to conduct these experiments once to collect quality scores and then develop instrumental methods to predict the mean of the users' perception of service quality.

Predicting the perceived quality from objective measurements provides only partial solutions. However, these partial solutions do provide insight into the principals of how perceived quality is affected by network QoS parameters. It is possible to correlate the QoS parameters with the measured perceived quality metrics and build an effective perceived quality-aware QoS model (Alreshoodi & Woods, 2013; Rifai *et al.*, 2011).

There are many standardization, industrial, and research bodies developing subjective and objective video quality assessment metrics as well as objective models for estimating perceived quality (Paudyal *et al.*, 2014). The main objectives of the existing QoS and perceived quality correlation models are: (1) predicting the perceived quality with only knowledge of the QoS parameters and (2) finding suitable QoS parameters for a desired perceived quality (Alreshoodi & Woods, 2013).

A literature review of perceived quality and QoS correlation models for multimedia services (Alreshoodi & Woods, 2013; Song & Tjondronegoro, 2014; Battisti *et al.*, 2014) provides the following general correlation approaches:

- Video Quality Metric (VQM) based Mapping Model
- Statistical Analysis method
- Machine Learning (ML) methods
- Crowdsourcing for subjective tests

– Resource Arbitration System
– Considering equipment and environment factors
– Quantitative and Qualitative Assessment

In the following sections, we will revisit this topic in more detail when discussing the standardized perceived quality estimation models. As a general note, from these published approaches, quality assessment systems that exploit ML paradigms very promisingly attain a high degree of prediction accuracy (Alreshoodi & Woods, 2013; Aroussi & Mellouk, 2014; Chen *et al.*, 2014). ML based approaches provide a theoretical and methodological framework to quantify the relationship between perceived quality and network QoS.

Both the standardized approaches and the ML based models are generated by utilizing some training and test dataset that has similarity to the target multimedia system. The model is built using the training dataset, its accuracy is calculated over a test dataset and the model configuration is then stored in a persistent storage in order to be used later. The form and content of this configuration might greatly vary depending on the modeling approach taken. Once the model is generated, the training and test dataset are not needed anymore and the model can be reconstructed in the target system from this configuration which we call "model data". Figure 2.1 depicts the relationship between the training dataset, the modeling process, and the model data.

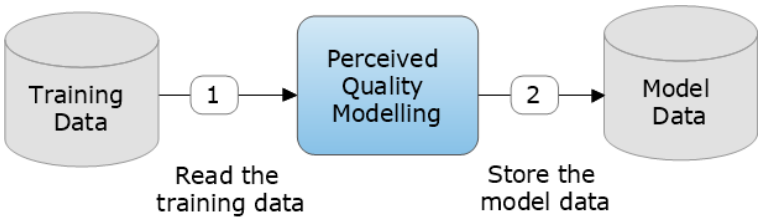

Figure 2.1 – Perceived quality modeling.

The training dataset typically includes subjective quality ratings for a number of audiovisual files with the respective measured QoS parameters. There are different methodologies to follow in order to generate a training dataset. No matter what methodology is followed, it is important to have a dataset that reflects the use case of the target system which depends on the content, encoders, media and channels configurations, protocols, software frameworks used, etc. In the telecommunications domain, a best practice is to collect subjective scores for a specific media resource from many observers and then to take the mean of all subjective scores. The subjective scores can be collected from a well-planned test environment as well as from the live target system (Figure 2.2). Depending

on where these subjective scores are collected, the requirements for the test methodology vary greatly. During the modeling, these mean opinion scores with the accompanying QoS parameters are used to generate the model rather than the individual scores. In the following sections, we will discuss various recommendations and best practices followed while conducting such experiments.

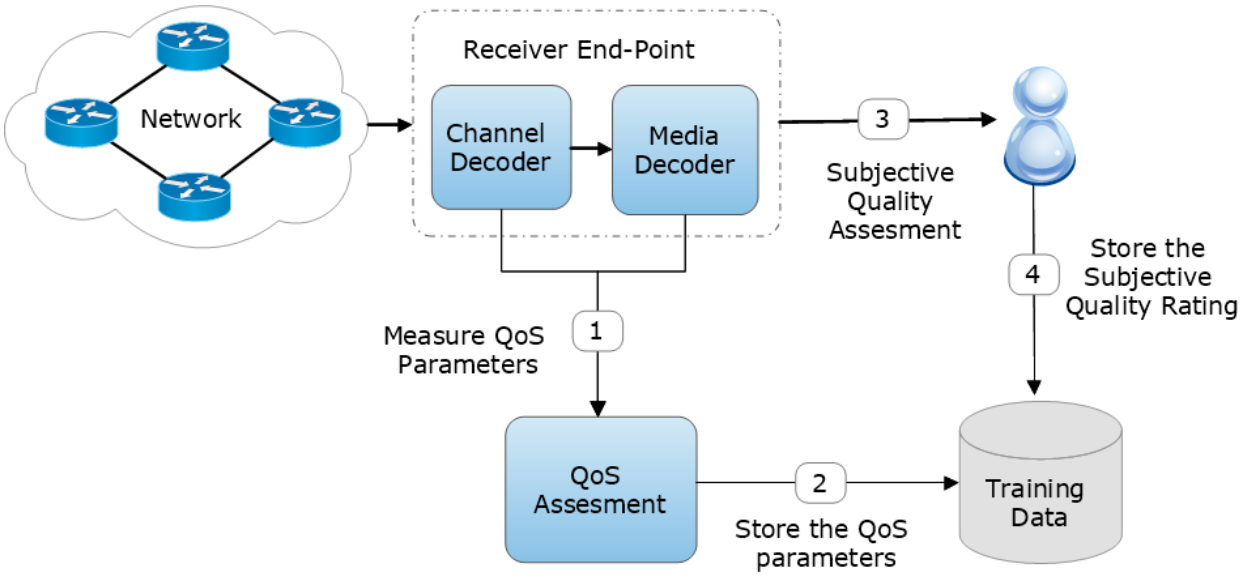

Figure 2.2 – Gathering the training data.

Another important aspect of the modeling is finding out how close is the model's assessment compared to the actual perceived quality of the system. There are numerous metrics for measuring the relationship between the outcome of the modeling and the data used. We will discuss these metrics in Section 2.4 in detail. It is obvious that the data used to measure the "accuracy" of the model has to be similar to the training data used. But the question is how similar should two datasets be.

The desired test dataset should reflect the target system and have similar configuration to the training dataset. A common method is to generate two datasets that have similar but not exactly identical configurations and then use the training dataset to train the model while using the test dataset to measure the test error in order to define the accuracy of the model (Figure 2.3). However, this approach is not always possible when the amount of data is small. We will discuss a potential solution to this issue in more detail in section 2.5.

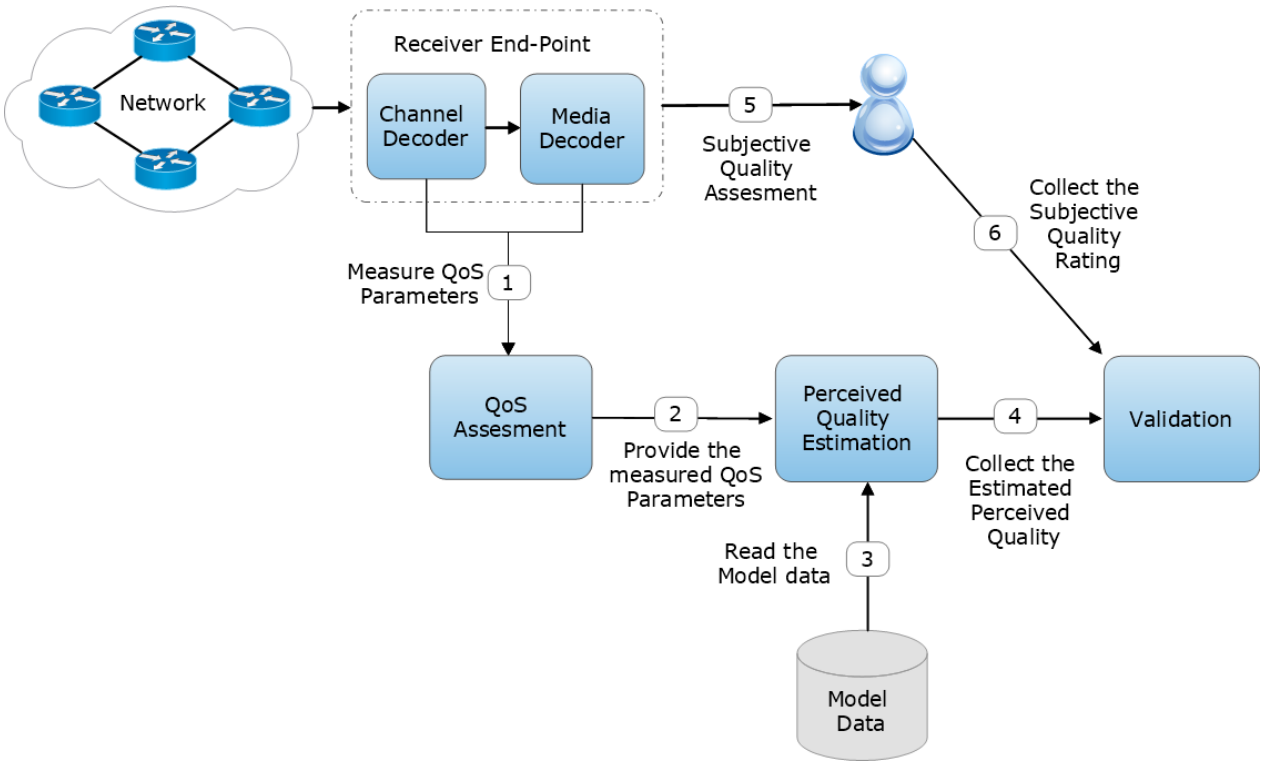

Figure 2.3 – Quality model validation.

2.2 Subjective Quality Assessment

The International Telecommunication Union (ITU) has various recommendations on how multimedia quality tests should be conducted. Commonly used recommendations for audiovisual quality tests are ITU-T P.913 (1998), ITU-T P.920 (1996) and ITU-T P.1401 (2012). These recommendations provide guidance on test methods and the test material to be used, and also describe the test environment and specify the number of subjects and possible subject screening. A list of valid ITU-T recommendations that are relevant to this research is given below.

The guidelines for various aspects of the subjective quality assessment:

– "Recommendation ITU-T P.911 (1998), Subjective audiovisual quality assessment methods for multimedia applications."
– "Recommendation ITU-T P.913 (2016), Methods for the subjective assessment of video quality, audio quality and audiovisual quality of Internet video and distribution quality television in any environment."

- "Recommendation ITU-T P.920 (2000), Interactive test methods for audiovisual communications."
- "Recommendation ITU-T P.800.1 (2006), Mean Opinion Score (MOS) terminology."
- "Recommendation ITU-T P.910 (2008), Subjective video quality assessment methods for multimedia applications."
- "Recommendation ITU-T P.1401 (2012), Methods, metrics and procedures for statistical evaluation, qualification and comparison of objective quality prediction models."

Some standardized quality models:

- "Recommendation ITU-T G.1070 (2012), Opinion model for video-telephony applications."
- "Recommendation ITU-T G.1071 (2015), Opinion model for network planning of video and audio streaming applications."
- "Recommendation ITU-T P.1201 (2012), Parametric non-intrusive assessment of audiovisual media streaming quality."
- "Recommendation ITU-T P.1202 (2012), Parametric non-intrusive bitstream assessment of video media streaming quality."
- "Recommendation ITU-T G.107 (2015), The E-model: a computational model for use in transmission planning."

Subjective assessment methodologies and the choice of the rating scale are discussed extensively in (ITU-T P.910, 1999; Huynh-Thu *et al.*, 2011). Huynh-Thu *et al.* (2011) have showed that there are no overall statistical differences between the different scales used. They have also showed that the single-stimulus presentation provides highly repeatable results even if different scales are used. Single-stimulus methods including the "Absolute Category Rating" (ACR) are frequently utilised for conducting subjective quality experiments as they are easy and fast to implement and the presentation of the stimuli is similar to that of the common use of the systems. The ACR method enables efficient rating of several files in a session and it is well suited for qualification tests. Two widely used ACR scales are the 5-point and 11-point scales. Categorical 5-point scale is commonly used in telecommunications where the labels "excellent", "good", "fair", "poor", and "bad" translate to the values 5, 4, 3, 2, and 1 when calculating the Mean Opinion Scores (MOS) (ITU-T P.910, 1999).

2.3 Quality Models

Audiovisual quality relies on audio and video quality and their interactions. The overall audiovisual quality can be estimated via a function directly regardless of how degradations affect the audio and video quality individually. An alternative approach is to predict the intermediate audio and video quality individually and then integrate these into an overall audiovisual quality. Raake *et al.* (2011) suggest that a complex model that uses the intermediate audio and video functions would generate more accurate predictions.

There are different approaches in building audiovisual quality models. Raake *et al.* (2011) have categorized the models based on the type of data they use.

- Parametric quality models predict the impact of encoding configurations and network impairments on multimedia quality. They typically use information extracted from packet headers and have no access to the packet payload data. These methods are well suited to the cases where the payload data is encrypted (Dubin *et al.*, 2016).
- Planning models are similar to the parametric models but differ on where the input information is acquired from. Planning models are based on service information available during the planning phase while parametric models take the input information acquired from an existing service.
- Media or signal-based models include aspects of human perception and assess the physical characteristics of the dispatched signal. They utilize the decoded signal as input to calculate a quality score.
- Bit-stream based models exploit information from the elementary stream. These models typically process both the headers and the payload of the video bit stream. They process the bit stream header to extract transport-related information such as Transport Stream (TS) and/or Realtime Transport Protocol (RTP) time stamps and sequence numbers for packet loss detection. They process the payload of the video bitstream to extract a number of features such as the picture type, the number of slices, the Quantization Parameter (QP), the motion vector, the type of each macroblock (MB) and its partitions, and the transform coefficients of the prediction residual.
- Hybrid quality assessment models exploit information from the packet headers, the elementary stream and the reconstructed pictures. The information of the reconstructed pictures is

obtained from the processed video sequence generated by an external decoder rather than from an internal decoder within the model.

Quality models can also be grouped relying on the type of additional information they process. Full-Reference (FR) models typically process the original source sequence while Reduced-Reference (RR) models use only a limited amount of information derived from the source sequence. No-Reference (NR) models use transmitted sequences without using any information from the original signal.

One important aspect of the perceived quality modeling is that an objective model is not expected to predict an average subjective opinion more accurately than an average test subject. The uncertainty of the subjective votes is calculated via standard deviation and its corresponding Confidence Interval (CI). These statistical parameters are aimed at determining the uncertainty of the subjects per file, or per test condition (ITU-T P.1401, 2012).

2.4 Statistical Metrics

Traditionally, the performance of a model is evaluated using three statistical metrics which are used to report the model performance's accuracy, consistency, and linearity/monotonicity (ITU-T P.1401, 2012).

The accuracy of a model is usually determined by a statistical interpretation of the difference between the MOS values of the subjective test and its prediction on a generalized scale. An accurate model aims to make predictions with the lowest error in terms of "Root Mean Square Error" (RMSE) used during the subjective tests (ITU-T P.1401, 2012). ITU-T Rec. P.863 (Beerends *et al.*, 2013) recommends to convert this value to the "epsilon-modified RMSE" to compare the results across different scales (Garcia, 2014).

The perceived quality predictions have to have consistently low error margins over the range of test subjects. The model's consistency is reported by calculating either the residual error distribution or outlier ratio. Computing the outlier ratio requires finding the outliers that are determined as the points for which prediction error surpasses the 95% confidence interval (ITU-T P.1401, 2012; Garcia, 2014).

In the literature, there are two commonly used metrics for computing the linearity of a model: the Spearman rank coefficient and the Pearson correlation coefficient. The Pearson correlation coefficient is used whenever the sampled data has a near-normal distribution. In other cases, the Spearman rank coefficient is utilized to qualify the linearity between the predicted and the actual subjective quality scores (ITU-T P.1401, 2012; Garcia, 2014).

2.5 Cross Validation

Quality model predictions are compared to the actual quality scores to evaluate the performance of the models. However, in the case of the limited amount of training and test data set, K-fold or "leave-one-out cross-validation" is used to report the performance of the quality model. In the K-Folds approach, the available data is split into K folds. In each step, K-1 folds are used to train the data and the single remaining fold is used to measure the accuracy of the model. This procedure is repeated K times using a different portion of the available data as test data. Data splitting can be done randomly as well as stratifying the folds. Depending on the selected K folds, the outcome of the model might vary. In order to make the predictions more robust and independent of the selected K folds, it is recommended to repeat the procedure several times and then take the average of these runs for each metric. The common practice is to use stratified 10-fold cross-validation (Garcia, 2014; Maki *et al.*, 2013).

2.6 Standardized Audiovisual Quality Prediction Models

Some of the standardized methods have been created through conducting competitions and selecting the models that have achieved the highest prediction accuracy. In this section, three audiovisual quality prediction models for streaming services and video telephony applications are briefly explained.

The ITU-T P.1201 (2012) model is intended for estimating the audiovisual quality of streaming services. It is a non-intrusive packet-header information based model for the service monitoring and benchmarking of User Datagram Protocol (UDP) based streaming. The model supports both lower resolution applications such as mobile TV and higher resolution applications such as Internet

Protocol television (IPTV). The model uses the information retrieved from the packet header as well as information provided out of the band. It provides separate predictions of audio, video, and audiovisual quality as output in terms of the five-point MOS scale. The model has been validated for compression, packet loss and re-buffering impairments of audio and video with different bitrates. Video content of different spatiotemporal complexity with different keyframes, frame rates, and video resolutions is selected. The model was tested over 1166 samples at lower resolutions and tested over 3190 samples at higher resolutions. RMSE and Pearson correlation (Garcia, 2014) values for audiovisual modeling were evaluated as 0.470 and 0.852, respectively for lower resolution applications and 0.435 and 0.911, respectively for higher resolution applications. Detailed performance figures are included in (ITU-T P.1201, 2012).

The ITU-T G.1071 (2015) model is recommended for network planning of audio and video streaming services. This recommendation addresses higher resolution application areas like IPTV and lower resolution application areas like mobile TV. The application of the models is limited to QoE/QoS planning, and quality benchmarking and monitoring is outside the scope of this recommendation. The model takes network planning assumptions such as video resolution, audio, and video codec types and profiles, audio and video bitrates, packet-loss rate and distribution as input. As for output, it provides the separate predictions of audio, video and audiovisual quality defined on the five-point MOS scale. Use cases such as re-buffering degradation of audio, and video, transcoding situations, the effects of audio level, noise and delay, audiovisual streaming with significant rate adaptation are not contained by the model. The model has been tested for low-resolution areas with ITU-T P.1201.1 training and validation databases and has achieved 0.50 RMSE and 0.83 Pearson correlation for audiovisual quality estimation. For high-resolution areas, it was tested on the ITU-T P.1201.2 training and validation databases and has achieved 0.51 RMSE and 0.87 Pearson correlation for audiovisual quality estimation.

The ITU-T G.1070 (2012) is a planning model recommended for video telephony. In this model, overall multimedia quality is computed from network and application parameters as well as terminal equipment parameters. It proposes an algorithm that estimates videophone quality for QoE and QoS planners. It provides estimates of multimedia quality that take interactivity into account to allow planners to avoid under-engineering. The model contains three main functions for assessing speech quality, video quality, and overall multimedia quality. The speech quality estimation function is similar to the E-model (ITU-T G.107, 2003) and takes speech codec type, packet loss rate, bit

rate, and talker echo loudness rating as input parameters. The video function is generated for "head-and-shoulders" content and takes video format, display size and codec type, packet loss rate, bit rate, key frame interval, frame rate as input parameters. The multimedia function integrates video alone and speech alone quality by including the audiovisual asynchrony and the end-to-end delay. The accuracy of the multimedia communication quality assessment model in terms of Pearson correlation was 0.83 for QVGA and 0.91 for QQVGA resolution on the given datasets. Application of the model is limited to QoE and QoS planning and other applications such as quality benchmarking and monitoring is not covered by the recommendation.

The models mentioned above and recent proposals to improve them are based on the integration of audio and video quality estimates to predict the overall audiovisual quality. They rest on the fusion theory that the auditory and visual signals are assessed separately to obtain auditory and visual qualities and then fused together to assess the overall perceived quality (You *et al.*, 2010). This integration of audiovisual quality leads us to the following alternative integration models (Belmudez, 2015):

$$Q_{av} = \alpha + \beta.Q_a + \gamma.Q_v + \zeta.Q_a.Q_v \tag{2.1}$$

$$Q_{av} = \alpha + \beta.Q_a.Q_v \tag{2.2}$$

where Q_{av} denotes audiovisual quality, Q_a audio quality and Q_v video quality. The symbols α, β, γ and ζ are constants.

In Equation 2.1, which is also known as late fusion theory, a few factors affect the overall audiovisual quality: individual Q_a and Q_v and audiovisual asynchrony. The four fusion parameters do not have commonly agreed upon values even though many researchers have adopted the late fusion equation. The second fusion equation also known as early fusion theory indicates that auditory and visual information are integrated into an early phase of the human cognition. You *et al.* (2010) could not conclude whether late fusion or early fusion between auditory and visual information influence human cognition more in audiovisual quality assessment. In Chapter 3 we will revisit the mathematical models from a broader perspective.

ITU-T Rec. P.1201, G.1070, and G.1071 have achieved high prediction accuracy against the test datasets provided. However, these methods by definition also have limited application areas and cover limited coding technologies. Therefore, researchers have been attempting to improve these models ever since (Garcia, 2014; Belmudez, 2015).

In standardized models, audio, video, and audiovisual predictions are performed by their respective functions whose output is then forwarded to the audiovisual function to predict audiovisual quality. Another alternative approach is to implement the audiovisual function in a way that would not require intermediate predictions for audio and video quality and still be able to capture all those complex interrelations between influence factors. Machine learning based techniques have been successfully applied in implementing these functions (Gastaldo *et al.*, 2013; Maki *et al.*, 2013). With machine learning techniques, we can with less effort build prediction models that fit specific use cases and achieve high accuracy. Historically, Neural Networks (NN) based approaches have been used extensively. In this research, in addition to the Deep Learning (DL) models, we evaluate Decision Tree (DT) based ensemble methods and Genetic Programming (GP) to implement the audiovisual quality function to predict the perceived quality directly from the parameters extracted from application and network layers. The machine learning based models tend to capture the complex relationships between influence factors no matter if the dataset is generated for IPTV services or video-telephony in mind.

2.7 Quality of Service Elements

QoS is defined as network performance measured in terms of some parameters, such as throughput, packet loss, latency/delay, and jitter. QoS analysis consists of three main tasks; QoS measurement, QoS visualization, and QoS prediction. QoS measurement tracks network packets from senders to receivers in order to detect QoS problems. QoS visualization provides visual analytics to network administrators in order to help identify QoS problems, and QoS prediction aims to predict QoS using network traffic data to help in planning and in taking action proactively to prevent any network problem (Senturk, 2014).

QoS parameter measurement techniques can be grouped into two general approaches: active measurement and passive measurement. In active measurement approaches, probing traffic is inser-

ted into the network under investigation in order to measure the QoS of the network. On the other hand, the passive measurement approaches utilize only the observational traffic data collected from the network (Chevul, 2006).

The Internet Engineering Task Force (IETF) has proposed two models to provide QoS in the network: Integrated Services (IntServ) and Differentiated Services (DiffServ). IntServ is a flow based QoS model which creates a kind of virtual circuit connecting the source and the destination. As a requirement, all routers in the path are informed using the RSVP signaling protocol about the requirements to be met. IntServ defines two classes of services: guaranteed service and controlled load service. The guaranteed service is designed for supporting real-time media traffic with guaranteed minimum delay whereas the controlled load service is designed for applications having a fair tolerance and which are not sensitive to network delay (Shah & Parvez, 2014).

IETF later introduced DiffServ to overcome the limitations of IntServ. In the new proposal, there were two fundamental changes. The first change was about the location of processing tasks. In the new model, main processing was removed from network core to the edge of the network to solve the scalability problems. The second change was to switch from per-flow service to per class service. In per class service, the routers route the packet based on a class of service definition in the packet thus solving the service type limitation problem (Shah & Parvez, 2014). However, in general in the Internet, neither model is implemented for the general public who is left with a «best effort» service.

From published research, there is a consensus among the researchers that the following four common parameters influence QoS the most in a network (Horvat *et al.*, 2013):

- Delay/latency
- Jitter/deviation in delay
- Throughput and bandwidth
- Packet loss/error rate and burst

In the rest of this chapter, we will concentrate on these four QoS parameters and expand our discussion about how to measure each specific parameter as well as the cause and implication of these parameters on the quality perceived by the user.

2.7.1 Delay/Latency

Depending on the type of the multimedia application, delay constraints play a big role. While streaming applications tend to be more flexible w.r.t. delay, real-time communications require strict delay constraints to be met since the knowledge of the end-to-end delay can be used for Service Level Agreement (SLA) validation (Wang *et al.*, 2004).

The IETF IP Performance Metrics Working Group (IPPM-WG) defines two kinds of metrics for the end-to-end delay; Round Trip Delay (RTD) (Zekauskas *et al.*, 1999b) and One Way Delay (OWD) (Zekauskas *et al.*, 1999a). Those metrics are also the foundations for many other QoS metrics measurements, such as bandwidth, jitter, and packet loss measurements (Wang *et al.*, 2004).

In OWD measurements, both passive and active measurement techniques are utilized. In passive measurement case, a unique packet identity and a time stamp are recorded at the sending node and another timestamp for the same packet is recorded upon arrival in the receiving node. Then the receiver transfers the packet id with the timestamp back to the sender where the OWD is calculated using the difference between the sender timestamp and receiver timestamp for the same packet. In this method, the clocks of the sender and receiver have to be synchronized (Wang *et al.*, 2004).

OWD can be also measured without clock synchronization by utilizing the RTD measurement results. In that case, the symmetry is assumed between the forward and reverse path in the route that the packet traversed and RTD value is halved to obtain the OWD value estimation (Wang *et al.*, 2004).

In VOIP systems, the end-to-end delay has the contribution from multiple processing levels and nodes. These delays are propagation, compression, packetization and packet switching delays. Propagation delay is the time taken by a packet to travel from one end point to another end point through the network. Compression delay is the time spent while compressing the data in order to save bandwidth. This is typically done with the codecs where further mechanism like look-ahead is applied. Packetization delay is the time spent for preparing the data to be used by the relevant service. Usually, the codec used governs this delay as well. Packet switching delay is the duration when a router or a switch buffers a packet and decides which interface to direct the packet. The ITU-T G.114 (2003) recommends between 150 and 400 ms as acceptable OWD values for an end-to-end delay in VOIP systems. Latency greater than 400 ms is unacceptable (Singh *et al.*, 2014).

2.7.2 Jitter/Deviation in Delay

Because of the queuing effect at routers, transmission of data packets through Internet Protocol (IP) networks varies. It is also possible that routing may change with time. Eventually, these factors cause variations in the arrival time of the packets at the receiver ends which is called jitter or deviation in delay (Singh *et al.*, 2014).

Jitter affects the perceived quality of conversation worse than delay. Since the data packets do not arrive at the destination at the regular interval, the decoder on the receiver side might not be able to construct a continuous conversation when the deviation exceeds acceptable thresholds (Singh *et al.*, 2014; Angrisani *et al.*, 2013).

Jitter is a relevant issue in both audio and video streaming and real-time services. To assess the QoS to the end-user, jitter measures are commonly compared to thresholds that vary depending on application requirements. When the jitter value exceeds these threshold values only for short time intervals with a relatively low repetition rate, the quality perceived by the user might not significantly decrease. However, when the jitter value gets closer to the threshold without exceeding for long time intervals with high repetition rates, it might indeed influence the quality perceived by the user significantly. As a result, finding a correlation between jitter metric and the quality perceived by the end user is both a very useful and challenging issue (Angrisani *et al.*, 2013; Shah & Parvez, 2014).

In general, the jitter threshold requirement per application is set considering the maximum tolerable values. Even though when jitter value exceeds the predefined threshold value causes unreliability of the service, the service degradations might not be perceived by the end user every time as in specific cases we mentioned above (Angrisani *et al.*, 2013).

Calyam *et al.* (2004) made a comparative study to obtain performance bounds for network metrics such as delay, jitter, and loss based on objective and subjective quality assessment of various audio and video streams. Using the video conferencing tasks, they have concluded that end-user perception of audio-visual quality is more sensitive to the variations in end-to-end jitter than to variations in delay or loss. In (Angrisani *et al.*, 2013) authors proved that the videos characterized by high motion are more affected by this issue.

2.7.3 Throughput and Bandwidth

Throughput and bandwidth are often associated with each other and used interchangeably but there is actually a difference. Bandwidth is the peak amount of data that can be sent per unit time. Throughput is however the actual achievable bandwidth and holds the most informative value from a user perspective (Shah & Parvez, 2014; Zhao *et al.*, 2014; Holik *et al.*, 2014).

From the wireless channel perspective, the available bandwidth of a link is related to the effective channel capacity on the available channel idle time. Therefore, the available bandwidth estimation is mainly based on the effective channel capacity and available channel idle time. Both the data rate set by the physical interface card and the random factors in the network define the channel capacity (Zhao *et al.*, 2014).

There are two basic approaches to differentiate the methods of measuring available bandwidth: active and passive probing vs. iterative and direct methods.

Active probing introduces an additional overhead to the network, degrading network performance and affecting the accuracy of the bandwidth estimation. In general, this approach does not work well in wireless networks and performs poorly in terms of accuracy in comparison with the passive based approaches (Zhao *et al.*, 2014; Holik *et al.*, 2014).

In a passive probing approach, the bandwidth is estimated via the utilization of the information collected in a passive manner. Many existing passive probing approaches monitor the channel usage based on the Carrier sense multiple access (CSMA) schemes of IEEE 802.11 via sensing radio medium (Zhao *et al.*, 2014; Holik *et al.*, 2014).

In iterative and direct methods, the capacity is unknown a priori in iterative or is known a priori in direct and probing is then done with gradually increasing speeds (Holik *et al.*, 2014).

Random backoff in wireless networks, unfortunately, prevents using most tools. Holik *et al.* (2014) claim that the Pathload (Jain & Dovrolis, 2002) and DietTOPP (Johnsson *et al.*, 2004) are the only tools that provide accurate results when set to fixed packet size of 1500 bytes. Both of these tools use the iterative approach of measuring.

2.7.4 Packet Loss, Error Rate and Burst

In the previous sections, we have seen that DiffServ can help the service providers meet SLA requirements. DiffServ can help control delay and jitter. However, it falls short when it comes to packet loss due to lower-layer errors or network connectivity issues (Greengrass *et al.*, 2009).

In some cases, delay and jitter result in increased load in router queues. When the load exceeds the threshold values, network nodes start discarding the packets which cause irritating discontinuities in multimedia conversations. One solution to prevent these issues is to use the Transmission Control Protocol (TCP) retransmission mechanism. However, retransmission causes some packets to arrive very late which interrupts the audio-visual communication in real-time applications. In the case of consecutive (burst) packet loss, the communication quality drops significantly (Singh *et al.*, 2014). The effect of burst losses is particularly pronounced for low bit rate video (Liang *et al.*, 2008).

The severity impact of packet loss on perceived quality also depends on which packets are being discarded. Aroussi & Mellouk (2014) have shown that "not all packets are equal". The loss of Intra-coded picture (I-frame) frames has the greatest impact compared to other frames since the I-frames are used by the decoders as reference frames during the decoding. Therefore, in congestion when packet discard is necessary, forwarding I-frames and discarding Predictive (P-frame) and Bi-predictive picture (B-frame) frames has a less severe impact on the quality perceived by the end user. However, in practice network nodes usually drop tail packets which does not discriminate between the packets depending on the frames they carry (Joskowicz & Sotelo, 2013; Santos *et al.*, 2014; Greengrass *et al.*, 2009).

In the literature, authors tend to publish their results with respect to the percentage of IP packet losses (Joskowicz & Sotelo, 2013). That approach highly depends on the codec technology used and knowing the percentage of the dropped packets does not make it easier to compare different works published. Knowing the details about the dropped frame types, however, provides more insight and a better ground for comparing them.

There are a variety of techniques to mitigate the effects of packet loss and inter-frame error propagation, and thereby to increase the robustness of video communication over lossy networks. Examples of recent work in this area includes intra/inter-mode switching, dynamic control of pre-

diction dependency using multi-frame memory, forward error correction (FEC), channel-adaptive packet scheduling, and the use of multiple description coding and packet path diversity (Liang *et al.*, 2008).

Cermak (2005) maps three metrics of network performance into judged quality of video conferencing: bandwidth, latency, and packet loss. They have observed via statistical analysis that packet loss was the most important network performance parameter in predicting the subjective quality of video conferencing. They have made the following observations.

- Users can compensate for delay by changing their behavior. However, such a compensation is less likely possible with packet loss.
- Both bandwidth and latency have much less effect on judgments of quality than packet loss, whether random or bursty.
- The main benefit of high throughput appears to be good quality when the content includes motion or the concatenation of motion and detail. However, lower bandwidth introduces aesthetics issues.
- Latency does not affect the aesthetics of speech sounds or video images. Bandwidth, on the other hand, does clearly affect video images.
- Packet loss affects the intelligibility of the communication. Behavioral adjustments do not work very well.
- Packet loss also affects the aesthetics of both video quality and speech quality. This combination of effects is what makes packet loss such a powerful variable in quality assessment experiments.
- Also, unlike a fixed latency, packet loss arrives randomly and thus is unpredictable. Finding behavioral adjustments for random events is more difficult than for predictable events.

2.8 Summary

Measuring QoS parameters and using this information in a feedback control in both application and network layers is a well-understood phenomenon. However, reporting and tuning multimedia transmission control parameters based on the observed QoS parameters is not necessarily an ideal way of obtaining a superior perceived quality. As the Qualinet whitepaper stated, the QoE or as in our case perceived quality is based on multiple influence factors, some of which are interrelated and

hard to qualify. However, it is possible to establish a relationship between these influence factors and the perceived quality to a certain degree.

There are various established practices on modeling approaches and recommendations on methodologies to follow when collecting the data required for this modeling attempts. Additionally, models such as ITU-T P.1201 (2012), ITU-T G.1071 (2015) and ITU-T G.1070 (2012) have been standardized for estimating the audiovisual quality of streaming services and video telephony. However, these models are intended for specific use cases and their performance decreases dramatically when used out of context.

The complete list of QoS parameters utilized frequently in perceived quality modeling varies from research to research. However, there is a consensus among the researchers that delay/latency, jitter, packet loss rate, throughput and bandwidth are common parameters influencing QoS the most in a network. In general, measuring these QoS parameters is a well-studied problem.

Chapter 3

Machine Learning Algorithms and Mathematical Models

This chapter is dedicated to machine learning (ML) based perceived quality modeling, machine learning algorithms and mathematical models. In this chapter, we also explain the algorithms that we have specifically used in this research–Decision Trees based ensemble methods, Genetic Programming and Deep Learning Algorithms–in detail.

This chapter is partly based on the content presented in the following conference and journals:

- Edip Demirbilek and Jean-Charles Grégoire. Machine learning based bitstream audiovisual quality prediction models for realtime communications. *IEEE International Conference on Multimedia and Expo, 2017 (Submitted).*
- Edip Demirbilek and Jean-Charles Grégoire. Machine learning based parametric audiovisual quality prediction models for realtime communications. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)(Accepted).*
- Edip Demirbilek and Jean-Charles Grégoire. Perceived quality prediction models: Taking advantage of correlated data. *Springer Quality and User Experience Journal: Topical Collection on Managing QoE of Future Networks and Applications (Submitted).*

In machine learning, the inference rules are carried out in an inductive manner in which general rules are drawn from specific observations via finding patterns which allow to make predictions

for upcoming events. There are three types of learning in ML, namely: supervised, unsupervised and semi-supervised. Perceived quality estimation models fall into supervised and semi-supervised learning (Aroussi & Mellouk, 2014).

When the predictions for upcoming events are expressed on a discrete scale the problem is called a classification. However, when it is expressed on a continuous scale, it is a regression problem (Aroussi & Mellouk, 2014; Alpaydin, 2014). In perceived quality modeling, classification models are used to predict the perceived quality as a class and regression models are used to approximate the perceived quality as a continuous function of QoS parameters (Aroussi & Mellouk, 2014).

Most of the proposed supervised ML systems can be classified into two categories: offline and online models. Building the model from a set of collected data and then using that model to predict the upcoming event from newly arrived data is an offline learning process. On the other hand, in online learning both training the model and the deployment of the model for predictions take place simultaneously. In that case, as new data arrives, the model at hand might evolve and predictions made might be different than in previous steps (Aroussi & Mellouk, 2014). That adaptive behavior is indeed very useful in dynamic environments like real-time multimedia communications.

In short, ML-based perceived quality prediction models can be classified into two categories: offline and online learning models. Furthermore, each of these categories consists of regression and classification methods.

3.1 Offline Models

There are several proposed offline perceived quality prediction models based on regression analysis. We can differentiate these models further by subgrouping them into Least Squares Regression (LSR) models and those using Regression Neural Networks (RNN) (Aroussi & Mellouk, 2014). Least Squares Regression (LSR) models attempt to predict upcoming events by applying various mathematical formulas in order to find a deterministic relationship between the QoS parameters (inputs) and the perceived quality (result). Khorsandroo *et al.* (2013) have grouped these LSR models from a mathematical form perspective into three groups as follows:

Table 3.1 – Mathematical models.

Name	Form	Relation
(1) Weber-Fechner Law	Logarithmic	$Perceived\ Quality = k.ln(QoS)$
(2) IQX Hypothesis	Exponential	$Perceived\ Quality = \alpha.e^{-\beta.QoS} + \gamma$
(3) Stevens' Power Law	Power	$Perceived\ Quality = k.QoS^{\beta}$

Each of these mathematical models proposes a QoS-perceived quality relationship from a different perspective, not necessarily using same QoS parameters.

1. Based on the psychological Weber-Fechner Law (WFL), in the proposed models (Reichl *et al.*, 2010a,b), speech quality, which is measured by the Mean Opinion Score (MOS), is considered as a logarithmic function of bit rate or loss rate in Voice over IP (VoIP) services (Aroussi & Mellouk, 2014). This briefly says that perceived quality may change as QoS parameters change (Khorsandroo *et al.*, 2013).

2. Hoßfeld *et al.* (2007, 2008); Fiedler *et al.* (2010) and Fiedler & Hoßfeld (2010) claim that perceived quality fluctuates according to its current level and proposed an exponential relationship between perceived quality and QoS where perceived quality is expressed in terms of MOS as functions of loss or reordering ratio (Aroussi & Mellouk, 2014). In other words, during a multimedia session where the overall perceived quality is described as very good, even slight degradations may have a severe and sensible impact. However, at low-quality levels, a further decrease in bitrate or increase in loss rate does not affect the perceived quality as significantly as before, since the user tends to be already dissatisfied with the perceived quality (Reichl *et al.*, 2010a). Hoßfeld *et al.* (2007) and Hoßfeld *et al.* (2008) have conducted experiments to verify this hypothesis using the voice codecs iLBC and G.711 in VoIP scenarios. They tested the IQX hypothesis for different QoS parameters that are packet loss, delay, and jitter. Fiedler *et al.* (2010) and Fiedler & Hoßfeld (2010) tested the same hypothesis for streaming services and claimed to have better accuracy compared to logarithmic based models. The IQX hypothesis presumes that perceived quality change related to QoS is a function of the currently perceived quality level (Khorsandroo *et al.*, 2013).

3. In video streaming services, the relationship between perceived quality and QoS (packet loss) is claimed to be in the form of a power function (Aroussi & Mellouk, 2014; Khorsandroo & Noor, 2012; Khorsandroo *et al.*, 2012, 2013). Khorsandroo *et al.* (2013) conducted the experiments and observed how the video streaming quality was affected by packet loss, VoIP quality was affected by delay and web browsing quality was affected by available bandwidth. They have

concluded that for different QoS parameters, the growth of perceived quality magnitude may vary differently and formulated this as the power relation between perceived quality and QoS.

Regression Neural Networks (RNN) are good candidates for modeling the nonlinear relationships between perceived quality and multiple QoS parameters where LSR methods often fall short. RNN models use multilayer feedforward neural network consisting of three main layers. The input neurons layer represents the number of selected QoS parameters, one or more hidden neurons layer, and single output neuron corresponding to the perceived quality prediction. The learning algorithm used is gradient descent backpropagation (Aroussi & Mellouk, 2014). Some of the work based on RNN is as follows:

1. Du *et al.* (2009) applied RNN models to get perceived quality estimates in terms of degradation mean opinion score (DMOS) using delay, jitter, loss ratio, bandwidth, burst, congestion period and disorder packets as input QoS parameters. They have used the Back-Propagation (BP) Neural Network (NN) toolbox in Matlab. Based on the experiments, they have adjusted the input network parameters to get the ideal output to satisfy the users' need.
2. Machado *et al.* (2011) took the delay, jitter, and loss ratio parameters and approximated QoE measurement for video streaming. In order to conduct the experiments, they have used NS-2 (Issariyakul & Hossain, 2011) for simulating the computer networks, the Evalvid (Klaue *et al.*, 2003) framework to perform traffic generation as well as the offline video evaluation, the MSU Video Quality Measurement tool for objective video quality assessment and Weka (Hall *et al.*, 2009) to perform transformations on the data, sorting tasks, regression, clustering, association and visualization.
3. Similarly Calyam *et al.* (2012) generated RNN based models using jitter, loss ratio and bit rate as input QoS parameters. They claimed to have a novel methodology to build real-time and No-Reference perceived quality models for IPTV applications. The models they built are using neural network principles for multiple resolutions (QCIF, QVGA, SD, and HD) video sequences, popular codec combinations (MPEG-2 video with MPEG-2 audio, MPEG- 4 video with AAC audio, and H.264 video with AAC audio) and bit rates for different network health conditions. Their models can be used for online perceived quality estimation given measurable network factors such as jitter and loss.

In the offline learning models for the classification we see a lot of different ML approaches being utilized. In the previous section, we have seen that machine learning algorithms are useful for building regression models from nonlinear multiple parameters. In the next paragraphs, we will concentrate on the classification models.

Agboma & Liotta (2008) and Agboma & Liotta (2012) have used the Linear Discriminant Analysis (LDA) to create models having video bit rate and frame rate QoS parameters as input. Their model attempt to predict the perceived quality as acceptable or unacceptable and depending on the terminal and video content type claimed to have generated several prediction models with a precision from 75% to 85% for the mobile handset, PDA and laptop terminals. However, they have pointed that the linear dependencies can accurately model user predictions only within limited boundaries.

Menkovski *et al.* (2009); Menkvoski *et al.* (2009) have utilized other ML methods such as Sequential Minimal Optimization implementation of Support Vector Machines (SVM) and the C4.5 Decision Trees (DT) algorithm to enhance accuracy rate and reflect systems dynamics. In their work they have used four video QoS parameters: bit rate, frame rate, spatial and temporal information and predicted the perceived quality as acceptable or unacceptable. They claim to have accuracy between 77-95% estimation for each terminal type such as mobile, PDA and laptop. The models they have proposed can be used to predict perceived quality for real-time communications as well. Here it is important to realize that the published results are for binary classification rather than multi-class classification.

Mushtaq *et al.* (2012) have compared various ML classification models using nine parameters, which are gender, frequency of viewing, interest, delay, jitter, loss, conditional loss, motion complexity and resolution in the context of video streaming delivery over cloud networks. They have used the WEKA tool and compared six classifying models: SVM, DT, Naive Bayes (NB), k-Nearest Neighbours (k-NN), Random Forest (RF) and Artificial Neural Network (ANN) models and concluded that the RF and DT methods are most appropriate for estimating the perceived quality as a class (five classes according to the MOS score) on the data sets they have. Additionally, they have shown that when a statistical analysis of classification is done, RF performs slightly better than DT. They have reported the following exact number; RF with 74.8% of correctly classified instances, DT model with 74% of correctly classified data.

3.2 Online Models

The number of available online learning methods for correlating QoS and perceived quality is much smaller than for available offline methods. Menkovski *et al.* (2010a,b) propose an online perceived quality prediction model based on given QoS metrics from continuous real-time user feedback. In contrast with the offline learning methods presented before, their model does not require a priori execution of subjective studies. They built their models based on Hoeffding Trees, Hoeffding Trees with functional nodes, Adaptive Hoeffding Trees with functional nodes, Hoeffding Option Trees, Hoeffding Option Trees with functional nodes and Adaptive Hoeffding Option Trees with functional nodes. They have combined these algorithms with ensemble methods such as bagging and boosting that are capable of online learning. For an Online Learning System, they have used the Massive Online Analysis (MOA) (Bifet *et al.*, 2010) ML platform for data stream mining which has implementations of Hoeffing Trees, Hoeffding Option Trees,Oza Bagging and Oza Boosting algorithms. MOA is based on the Weka ML data mining platform and it is optimized for fast stream mining, which implies a lot of online data passing through the classifier. At the end of their experiment, they concluded that the Oza Bagging ensemble with Hoeffding Option Trees NB Adaptive as a base classifier achieved the best results both in overall accuracy and variation of accuracy.

3.3 Open Issues

While ML tools can play an important role in the area of visual quality assessment (VQA) there are some still open issues (Gastaldo *et al.*, 2013). In existing ML-based approaches to VQA, the "more features are the better" hypothesis does not hold true in every case. In fact, such an approach may eventually affect the performance of the prediction system because of the "curse of dimensionality" which requires exponential growth in data for supporting the results when the number of features increased. Therefore there is the need for powerful solutions that combine ML with feature selections (Gastaldo *et al.*, 2013).

There is the tendency to use support vector machine (SVM)-based frameworks since SVM can deal with the "curse of dimensionality" and because of publicly available off-the-shelf software implementations such as libSVM (Chang & Lin, 2011). While SVM can be considered one of the

most effective classification model available in the ML area, several other powerful options exist to tackle regression problems (Gastaldo *et al.*, 2013).

Another issue with these ML-based prediction systems is experimental reproducibility. In order to improve the reliability of these approaches, essential details about the model selection must be published so that same experiments can be conducted on other publicly available datasets with robust validation procedures (Gastaldo *et al.*, 2013).

The contemporary machine learning landscape consists of countless algorithms and their implementations in various libraries. Some of these methods are intended for classification only problems. However, a lot of algorithms are suitable for classification as well as regression problems. Below we briefly describe the machine learning methods that we have investigated in details and fine tuned to obtain the best performance in this research. The reasoning behind choosing these methods is given in Chapter 4, Chapter 7 and Chapter 8. Even though we utilize these methods for regression, in the summary below we did not limit ourselves to their regression usage only. The specific configurations we have used for these methods during the implementation are given in respective chapters.

3.4 Machine Learning Algorithms

3.4.1 Decision Tree Based Ensemble Methods

Decision Trees (DT) are hierarchical data structures that can be used for classification and regression problems effectively using the divide-and-conquer strategy. A Decision Tree is composed of internal decision nodes where a test is applied to a given input and branches to a classification or regression value by the leaf nodes. The estimation process originates at the root node, traverses the decision nodes until a leaf node is hit (Alpaydin, 2014; Mushtaq *et al.*, 2012).

The tree structure allows a fast discovery of nodes that cover an input. In a binary tree, traversing each decision nodes exclude half of the cases. Due to fast convergence and ease of interpretation, they are sometimes preferred over more accurate methods (Alpaydin, 2014).

The estimation can be computed in a parametric model as well as a nonparametric model. In parametric estimation, the model is built over the whole input space from the training data and

a static tree structure is formed. Then the same model is used to make estimations as test data arrives. In the nonparametric approach, the tree structure is not static and, during the learning process, it grows as branches and leaves are added (Alpaydin, 2014).

Decision Trees have low bias and very high variance which bring over-fitting issues when they grow very deep. To reduce variance, Decision Tree-based ensemble methods have been developed. Random Forests (RF) are such an ensemble learning method for classification and regression that utilize several Decision Trees models to obtain a better prediction performance. During the training, an array of Decision Trees are formed and a randomly chosen subset of training data is used to train each tree. In a classification problem, the inputs are submitted to every tree in the RF in order to get a vote for a class. An RF model collects all votes and then picks the class with the highest number of votes. That behavior reduces the high variance issues we have mentioned above. However, since there is a tradeoff between bias and variance, RF classification introduces a small increase in the bias while reducing the variance. Overall, it still provides significant improvements in terms of classification accuracy (Breiman, 2001; Mushtaq *et al.*, 2012).

Rather than searching for a single superior model, researchers have noticed that combining many variations produce better results with a little extra effort. As we see for the Random Forests, ensemble learning models generate many classifiers and combine their results. This approach has been gaining a lot of interest recently. Two well-known methods of ensemble learning are boosting and bagging. In both methods, the learning algorithm combine the predictions of multiple base models (Liaw & Wiener, 2002; Oza, 2005; Domingos, 2012).

When we consider the Decision Trees based models with Bagging methods, each tree is constructed with a random variation of training data set. The prediction is fulfilled via the simple majority vote to improve the stability and accuracy. This approach greatly reduces the variance as well as helping to avoid over-fitting issues, but it slightly increases the bias. Although it is usually applied to Decision Trees, it can be used with any type of method as well (Liaw & Wiener, 2002; Domingos, 2012).

In boosting methods, prediction depends on the earlier trees as well. In this approach, the points incorrectly predicted by previous trees are given extra weight by the successive trees. Boosting methods primarily target the reduction of the bias and possibly of the variance while creating a single strong learner out of a set of weak learners (Liaw & Wiener, 2002).

Pfahringer *et al.* (2007) emphasize that the tree learners are not very stable due to limited lookahead ability. Ensemble methods attempt to overcome such problems found in simple base tree learners.

3.4.2 Symbolic Regression and Genetic Programming

Symbolic regression technique aims to identify an underlying mathematical expression that best fits a dataset. It consists of finding both the form of equations and the parameters simultaneously. Symbolic Regression starts by forming an initial expression by randomly combining mathematical building blocks and then continue forming new equations by recombining previous equations using Genetic Programming (GP) (Schmidt & Lipson, 2010).

GP is a computation technique that enables us to find a solution to a problem without knowing the form of the solution in advance. It is based on the evolution of a population of computer programs where populations are transformed stochastically into new populations generation by generation (Poli *et al.*, 2008).

GP discovers the performance of a program by running it, measuring its outcome, and then comparing the result to some objective. This comparison is called fitness. In machine learning domain this would be equal to finding the 'score', 'error' or 'loss'. In each generation, the programs that do well are marked to breed and then are used to produce new programs for the following generation. The crossover and mutation are the main genetic operations for creating new programs from existing ones. In the crossover, a child program is generated by joining randomly chosen parts from two selected programs from the previous generation. In mutation, however, a child program is created from a single parent from the previous generation by randomly modifying a randomly selected segment (Poli *et al.*, 2008).

GP usually utilizes trees in order to manipulate the programs. In the tree, function calls are represented by nodes and values associated with the functions are represented by leaves (Koza, 1992). GP programs combine multiple components in more advanced forms. In that case, each component is represented by a tree that grouped together with other trees under the root node (Poli *et al.*, 2008).

Similar to the ensemble methods we have seen in the previous section, initial GP populations are typically randomly generated as well. These initial populations are categorized as full, grow and ramped half-and-half depending on their depth (Poli *et al.*, 2008).

Both full and grow methods limit the maximum depth of the initial individuals generated. They differ from each other with respect to the size and the shape of the trees generated. In the full method, trees are generated where all the leaves are at the same depth. In the grow method, trees are generated in various sizes and shapes. Ramped half-and-half method proposes a combination of both full and grow methods. In this approach, the full method is used to construct the half of the initial population and the grow method is used to construct the other half of the initial population (Poli *et al.*, 2008).

GP selects the individuals probabilistically based on their fitness and then applies the genetic operations to them. This process causes better individuals to have likely more child programs than inferior individuals. Two common individual selection methods in GP are tournament selection and fitness proportionate selection (Poli *et al.*, 2008).

3.4.3 Deep Learning

Deep Learning dates back to the 1940s and has been re-branded many times, reflecting the influence of different researchers and different perspectives. It has only recently become called "Deep Learning" (Bengio *et al.*, 2015).

A typical example of a Deep learning model is the feedforward Deep Network or Multi-Layer perceptron (MLP) (Bengio *et al.*, 2015). A Multi-Layer Perceptron makes no assumptions about relationships among variables. In general, these models use three main layers: one input neurons layer that represents the input vector, one or more intermediator "hidden" layers and output neurons that represent the output vector. Nodes in each layer are linked to all nodes in adjoining layers. These links are used to forward signals from one neuron to the other (Comrie, 1997; Mushtaq *et al.*, 2012).

Nonlinearities are represented in the network by the activation and transfer functions in each node. Each node handles a basic computation while their links enable an overall computation. The overall behavior of a Neural network is influenced by the number of layers, the number of

neurons in each layer, how the neurons are linked and the weights associated with each link. The weight associated with each link defines how a first neuron influences the second neuron. During the training period, the weights are revised. With that approach, hidden layers capture the complexities in the data while the weights are adjusted in each iteration in order to obtain the lowest error in the output. The learning algorithm used is gradient descent backpropagation (Bengio *et al.*, 2015; Comrie, 1997; Mushtaq *et al.*, 2012).

In the back propagation approach, during the forward phase, the input signal is propagated through the network layer by layer. In the output node, the error signal is computed and then this error signal is sent to the network in backward direction which is called the backward phase. During this backward phase, network parameters are modified in order to minimize the signal error (Du *et al.*, 2009). Deep Learning methods can be used in regression problems as well as clustering and classification applications.

In Section 7.2 we will look into the details of these algorithms once again but this time from an implementation point of view and mention their specific configurations targeting regression usage. However, before that we need to look at the publicly available datasets as well as the INRS audiovisual quality dataset to see what kind of information is available to us when attempting to build perceived quality estimation models.

3.4.4 Other Commonly Used ML Algorithms

Least Squares Regression (LSR)

Least Squares Regression (LSR) is called as “regression”, “linear regression” or “least squares” by many people in most cases and is one of the most widely used modeling methods. The goal is to find the unknown parameters of a function that minimize the sum of the squared deviations between the actual data and the generated model. The function found would be in linear form even though the line it represents might not be a straight line. In this case, the word “linear” means “linear in the parameters” or “statistically linear” (Natrella, 2010).

In practice, LSR makes very efficient use of the data and aims to obtain results even from small data sets. The theory behind the linear regression is well-understood as well (Natrella, 2010).

However, LSR models have limitations in the shape to represent the data as the range of the data increases. Furthermore, it is sensitive to outliers and a few unusual data points can sometimes seriously skew the results of the least squares analysis. These characteristics enforce various validation schemes which then require additional processing as well as reduces the training data size (Natrella, 2010).

Support Vector Machines (SVM)

In 1995, Cortes and Vapnik developed Support Vector Machines (SVMs) for binary classification. The idea is to find an optimal separating hyperplane between two classes by maximizing the margin between the classes' closest points. The points lying on the margin boundaries are called support vectors. When new data arrives, it is only tested against this hyperplane and classification is done based on which side of the hyperplane they are located. If the data points fall in the margin area, then the support vectors are updated and a new optimal hyperplane is calculated by finding the maximum margin again. This behavior does not only provide good classification performance, but also leaves much room for the correct classification on future data (Meyer & Wien, 2014; Wu *et al.*, 2008; Mushtaq *et al.*, 2012).

In the case of overlapping classes, data points that are located on the "wrong" side of the hyper plane are weighted down to reduce their influence. Sometimes finding a linear separating hyper plane between two classes is not possible. In that case, data points are projected to a usually higher-dimensional space where data points become linearly separable. This projection is realized via various kernel techniques (Meyer & Wien, 2014; Mushtaq *et al.*, 2012).

SVM methodologies have many attractive features where geometric intuition, elegant mathematics, theoretical guarantees, and practical algorithms meet. They do not have any local minima issues that we face with neural networks and decision trees. That characteristic provides stable, reproducible and largely independent experiment. For example, if two users apply the same SVM model with the same parameters to the same data, they will get the same solution. In Neural Networks, the results dependent on the particular algorithm and the starting point used (Bennett & Campbell, 2000). Training usually does not require a large dataset and is insensitive to the number of dimensions. As a result, SVMs are considered a must try algorithm for diverse machine learning applications (Wu *et al.*, 2008).

However, SVM techniques involve quadratic optimization and scale badly as data size grows. Additionally, the extensive search must be conducted in the parameter space for the correct choice of the kernel before results can be trusted (Meyer & Wien, 2014).

SVM can be extended to perform numerical calculations to perform regression analysis as well as ranking elements (Wu *et al.*, 2008).

Support Vector Machines (SVMs) and related kernel methods have become increasingly popular tools (Bennett & Campbell, 2000). LibSVM is one of the most widespread used SVM software package; it includes several extensions for binary and multi-class classification, outlier/novelty detection and regression (Meyer & Wien, 2014).

K-Nearest Neighbours (KNN)

K-Nearest Neighbour (KNN) method is an instance based ML method for classification that is easy to understand and easy to implement. It searches for a group of k objects in the training set and bases the assignment of a label on the predominance of a particular class in this neighborhood. To classify an unlabelled object, the distance of this object to the labeled objects is computed, its k-nearest neighbors are identified, and the class labels of these nearest neighbors are then used to determine the class label of the object based on the majority class (Wu *et al.*, 2008; Mushtaq *et al.*, 2012).

KNN classifiers are considered to be lazy learners. Constructing the model from the training data is a relatively cheap process. However labeling newly arrived data requires a lot of labor since it requires computing the distance of the unlabelled object to all the objects in the labeled set, which can be expensive particularly for large training sets (Wu *et al.*, 2008).

KNN classifiers can perform well in many situations. They are particularly well suited for multi-modal classes as well as applications in which an object can have many class labels. Some researchers found that KNN outperformed SVM, which is a much more sophisticated classification scheme. However, it is worthy to note that there is a trade-off between model training and classifying the newly arrived data in both cases (Wu *et al.*, 2008).

There are some downsides to using the KNN classifiers. On issue is the choice of the "k". If k is chosen to be too small for the given data set, then the classification done can be sensitive to outlier points. On the other hand, if k is larger than the optimum classification, then the neighborhood may include too many points from other classes (Wu *et al.*, 2008).

Naive Bayes (NB)

The Naive Bayes (NB) classifier is a probabilistic model that uses the joint probabilities of features to make an estimation of a class. The learning is based on the Bayes rule that assumes the attributes are all conditionally independent of one another given class. This assumption dramatically reduces the number of parameters that must be estimated to learn the classifier. The naive part of the classifier also comes from the assumption that all inputs are conditionally independent of each other which enables the parameters to be learned separately. As a result this simplifies and speeds up the computation operations (Mushtaq *et al.*, 2012; Rish, 2001; Mitchell, 2015).

Naive Bayes method is very easy to construct and does not need any complicated iterative parameter estimation schemes. As a result, it may be readily applied to huge data sets. It is easy to interpret as users unskilled in classifier technology can understand why it is making the classification it makes. Furthermore, even though it is based on poor assumptions, it often does surprisingly well (Wu *et al.*, 2008).

Naive Bayes is widely used for both discrete and continuous inputs. However, Naive Bayes is a learning algorithm with greater bias but lower variance, compared to other methods such as Logistic Regression. If this bias is acceptable given the actual data, Naive Bayes is preferable (Mitchell, 2015).

Oza Bagging and Boosting

Oza Bagging modifies the general bagging approach for online learning in the following manner. Each example of data is presented to a base classifier K times, where K is a random variable with Poisson distribution. Authors claimed that the online bagging classifier converges to the batch bagging classifier given certain conditions and a training set where the examples tend to infinity (Meyer & Wien, 2014).

In boosting methods, the prediction depends on the earlier trees and successive trees give extra weight to points incorrectly predicted by earlier predictors. In the end, a weighted vote is taken for prediction. Boosting methods primarily targeting to reduce the bias and possibly the variance while creating a single strong learner out of a set of weak learners (Liaw & Wiener, 2002).

In general, ensemble methods seek to overcome problems inherent in all greedy tree learners. Tree learners have limited lookahead ability and therefore are not very stable. In practice, bagging methods appear to be working successfully while boosting not so much (Pfahringer *et al.*, 2007).

Hoeffding Option Trees

Hoeffding Trees model is designed to handle extremely large training sets that are too large to remain in memory and must be processed from a stream in a single pass. Since the data is processed sequentially only in a single pass with a constant processing speed, it makes this model a good candidate for online learning problems as well as high-speed data streams (Meyer & Wien, 2014; Pfahringer *et al.*, 2007).

Hoeffding Trees model assumes that the distribution generating examples does not change over time. The problem of deciding exactly how many examples are necessary for training is solved by using a statistical result known as the Hoeffding bound. But since the learner has only a partial view of the data, it imposes to have less confident split criteria's for selected attributes. At the beginning of the learning process, this model has the tendency to be less-accurate (Meyer & Wien, 2014; Yusuf & Reddy, 2012).

Hoeffding Option Trees are Hoeffding Trees with option nodes alongside the standard decision nodes and leaf nodes. Option nodes enable several tests instead of a single test per node and split the decision path several ways. The main motivation is that introducing option nodes removes the need for selecting the best splitting attribute where several attributes appear to be equally discriminative (Yusuf & Reddy, 2012; Pfahringer *et al.*, 2007).

Option nodes are introduced when splitting decisions are ambiguous, which then helps to avoid excessive and unnecessary tree growth and reduce memory consumption. In a sense, option trees represent a middle ground between single trees and ensembles in a way to represent multiple trees

in a single structure. They are capable of producing useful, and interpretable, additional model structure without consuming too many resources (Yusuf & Reddy, 2012; Pfahringer *et al.*, 2007).

One of the main benefits of the Hoeffding trees is the amount of memory they require. However, option trees have a tendency to grow very rapidly if not controlled. There are various strategies to handle this problem, including limiting the number of options allowed locally per node (Pfahringer *et al.*, 2007).

3.5 Mathematical Models

Weber-Fechner Law (WFL) and Stevens' Power Law

In 1834, German physiologist Ernst Heinrich Weber stated that human sensory system is able to notice differences as soon as the basic physical stimulus changes for more than a constant proportion of its actual magnitude (Reichl *et al.*, 2010a). From thousands of psychophysical experiments, Ernst Weber discovered that over a large dynamic range, and for many parameters, the threshold of discrimination between two stimuli increases linearly with stimulus intensity (Dehaene, 2003). For example, experiments showed that when we held a weight in our hand, we can notice the increase of the weight of approximately 3%, independently of the actual size of the original weight (Reichl *et al.*, 2010a).

Later, in addition to what Weber discovered, Gustav Fechner showed that the external stimulus can be scaled into a logarithmic internal representation of sensation which is known as Weber-Fechner Law (WFL) today. The WFL is applicable to broad range of scenarios such as human vision where magnitude measured on a logarithmic scale and hearing where the intensity of the sound is measured on a decibel scale (Reichl *et al.*, 2010a).

However, recently Stevens discussed that internal representation of sensation is a power function rather than a logarithm (Dehaene, 2003). Additionally, he pointed that the power relationship between each external stimulus and its internal representation is different than another.

IQX Exponential

The IQX hypothesis states that the subjective sensibility of the perceived quality is higher when the experienced quality is very high and lower when the experienced quality is low. As an example, a single spot on the white table cloth in a starred restaurant disturbs significantly more than if it occured in a beer pub (Hoßfeld *et al.*, 2008).

When we extend this to QoS and perceived quality correlation, the IQX hypothesis can be formulated as follows: the change of perceived quality depends on the current level of perceived quality given the same amount of change of the QoS value. That means, when we have a high perceived quality, changes in QoS have a significant impact on the perceived quality. However, when the perceived quality is already very low, the perceived change in the perceived quality is much lower given the same degree of changes in the QoS parameters (Hoßfeld *et al.*, 2008).

3.6 Summary

Most of the proposed supervised ML based or mathematical perceived quality estimation models can be classified into two categories: offline and online models. In offline learning, building the model from a set of collected data happens before using the model to predict the upcoming event from newly arrived data. On the other hand, in online learning both training the model and the deployment of the model for predictions take place simultaneously.

The models can also be further qualified as regression and classification models where in the regression case, perceived quality estimation is in the form of a continuous function, and in the classification case, perceived quality estimation is expected to fall into pre-defined quality classes.

In this research, we use Decision Trees based ensemble methods, Genetic Programming and Deep Learning algorithms for audiovisual perceived quality prediction.

Chapter 4

Experimental Dataset and Two Preliminary Parametric Models

This chapter explains our preliminary research towards building perceived quality estimation models. Both the dataset and the models generated in this chapter provide some inside information on how to generate a dataset with realistic configurations and how to build more accurate models. Both the dataset and the models in this chapter should be taken as an experimental phase rather than the eventual outcome of the research.

We have taken a parametric model approach by using the additional side information available that makes it possible to build a model meeting real-time requirements. We attempt to estimate the audiovisual quality directly from the system influence factors by creating the model with machine learning algorithms that have been successfully applied to estimating the perceived quality. We have mainly used the Random Forests ensemble methods during model training. However, we have provided the comparative results with Multi-layer Perceptron (MLP) methods that have been widely used in image assessment, video assessment, and video and voice quality estimation.

This chapter is partly based on the content presented at the following conference:

– Edip Demirbilek and Jean-Charles Grégoire. Towards reduced reference parametric models for estimating audiovisual quality in multimedia services. *IEEE International Conference on Communications (ICC), 2016,* IEEE.

4.1 Related Work

The Qualinet Multimedia Databases set v5.5 compiled by (Fliegel, 2014) provides a list of some publicly available audiovisual datasets and models built based on them. Some of these were created by (Goudarzi *et al.*, 2010; Pinson *et al.*, 2012, 2013; Robitza *et al.*, 2012; Maki *et al.*, 2013). In Chapter 6 we will explain these datasets in more detail. As the work we present in this chapter is more experimental, we will postpone comparison with other datasets until we present the eventual models in Chapter 7. We have presented the standardized models in Chapter 2.

Some of these datasets contain quality ratings for audiovisual subjective tests while others contain quality ratings for audio, video, and audiovisual test separately. Researchers have built audiovisual quality models using the data provided in these audiovisual datasets that have a variety of configurations. Some models aim to estimate the perceived quality directly while others try to deduct a model accurate enough for estimating the audiovisual quality by using the separate audio and video quality estimates.

As recent developments show, parametric models achieve high accuracy to a degree in estimating the perceived audiovisual quality with limited resources. In this chapter, we try to build similar models by estimating the audiovisual quality directly by using Random Forest and Neural network machine learning methods for specific target network configurations.

4.2 An Audiovisual Quality Dataset

Creating an audiovisual quality dataset requires finding optimum dataset configurations, creating required test tools, building test setups, producing content, generating the files under specific network conditions, preparing the subjective test methodology and conducting the tests while following standard compliant procedures. Publicly available datasets help us to avoid this significant amount of work and enable us to compare the performance of various models on the same dataset. However, a model's performance has to be assessed in the target environment with the required application and network parameters. Additionally, to the best of our knowledge, none of the available datasets include the whole parameter space we needed to experiment.

We have created audiovisual content specifically for the early phase of this research. Figure 4.1 depicts a scene in the generated reference video being played by the custom video player developed to collect the subjective scores without disclosing the resource's quality. The video consists of slow-moving scenes where a person reads a passage from a book. This content is chosen to be similar to a typical one-to-one audiovisual conversation.

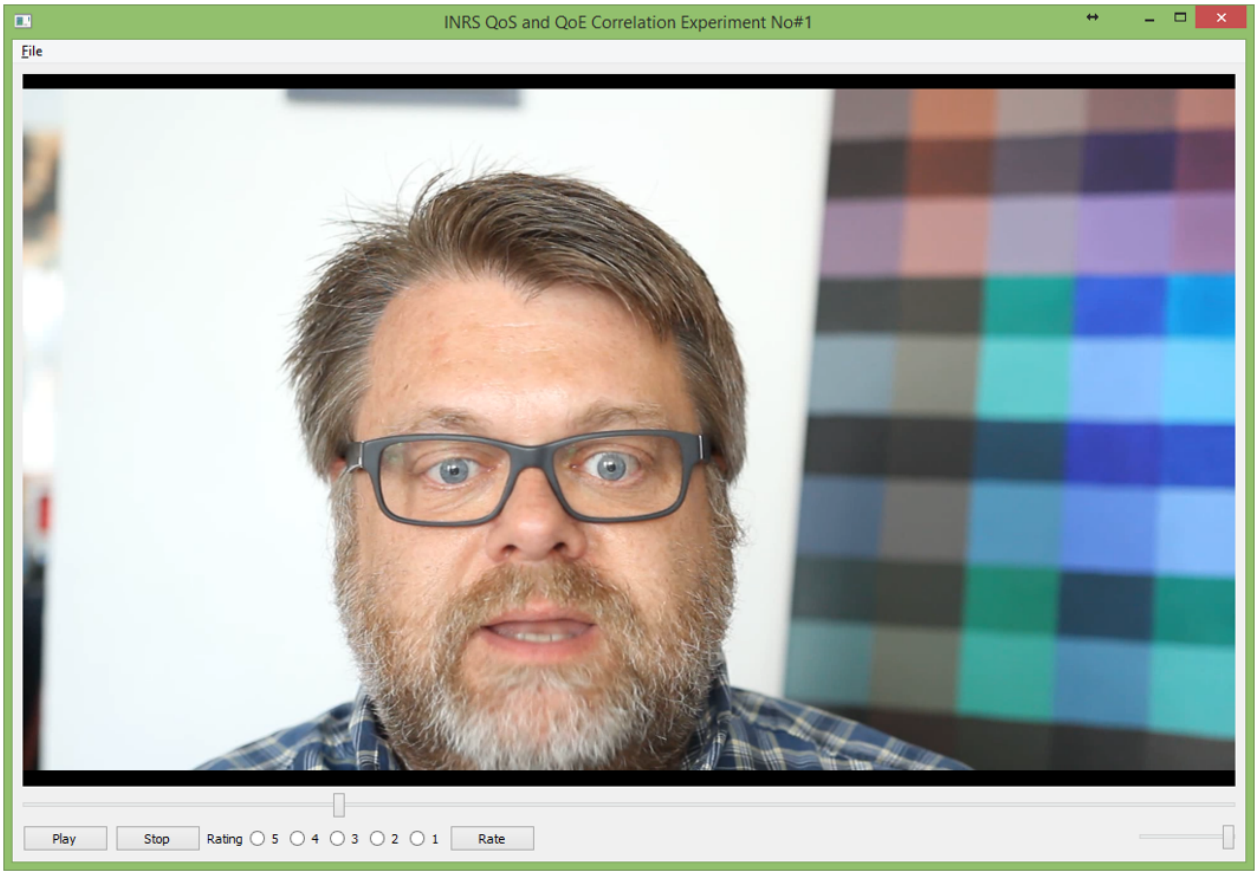

Figure 4.1 – A scene from the generated reference video file.

The dataset includes the resolution, bit rate, bandwidth, packet loss rate, and jitter influence factors. Table 4.1 shows the selected values for these influence factors. Each of these influence factor values is assigned as follows.

From the reference video file, we have created 6 MPEG audio-video files with a leading commercial video editing software. These 6 source files consisted of 3 target bit rates (High Quality, Medium Quality, and Low Quality) for two resolution levels. To match the capabilities of contemporary mobile platforms HD 1080 and HD 720 resolution levels were selected. Table 4.2 shows the source files generated where the bitrates computed as an average over one second of transmission.

Table 4.1 – Audivisual quality dataset influence factors.

Resolution	HD1080 (1920x1080 pixels) HD720 (1280x720 pixels)
Bit Rate	High Quality(HQ) Medium Quality(MQ) Low Quality (LQ)
Bandwidth	High Bandwidth (2x Max Bit Rate) Low Bandwidth (Max Bit rate)
Packet Loss rate (%)	0, 0.1, 0.5
Jitter (ms)	0, 10, 50, 100

Each source files was encoded with MPEG video version 2 video codec with Main@High 1440 profile and MPEG Audio version 1 layer 2 audio codecs contained in a MPEG-TS container. The video frame rate was set to 25 fps and the audio sampling rate set to 48.0 KHz.

Table 4.2 – Source files generated.

File	Overall Bit Rate (Kbps)	Video Max Bit Rate (Kbps)	Audio Bit Rate (Kbps)
MPEG2_HD_720_LQ.ts	1389	1477	128
MPEG2_HD_720_MQ.ts	3461	3664	128
MPEG2_HD_720_HQ.ts	8040	8313	128
MPEG2_HD_1080_LQ.ts	2871	3227	128
MPEG2_HD_1080_MQ.ts	7457	8069	128
MPEG2_HD_1080_HQ.ts	13.1 Mbps	18083	128

In our preliminary experiments, we have selected packet loss rate (PLR) between 0-5% and observed that a PLR greater than 0.5% reduces the perceived quality significantly. This is mostly due to the testbed and the limitations of the codec technologies supported. We have obtained similar limitations with the bandwidth configurations. Initially, we have tested 4 different bandwidth levels for a given bit rate configuration and observed that only two out of these 4 levels are relevant to our real-life conditions and eventually only kept these two for our study.

The selected bandwidth-pair configurations are intended to provide one configuration for no-limitation on bandwidth (High Bandwidth) while the other one is intended to provide slightly less bandwidth than max bitrate (Low Bandwidth), which causes only small degradation in perceived quality.

We have conducted various iperf tests to measure the bandwidth in Low Bandwidth setting for each file and have found that available bandwidth was 2.8% less than max bitrate for each file. We have taken into account that iperf adds a small bias to the measurement since iperf measures

the available bandwidth using a TCP stream, while the bandwidth limitation sets the bandwidth available for IP packets (Nussbaum & Richard, 2009).

The test sequences were prepared prior to the assessment by recording RTP-based video streams transmitted over an emulated network. The videos were streamed and recorded with the VideoLan Video-on-Demand (VOD) Server (VLC Team, 2016b) and VLC media player. The Netem network emulator (Hemminger *et al.*, 2005) was deployed in order to introduce the packet loss and jitter test conditions. Dummynet (Carbone & Rizzo, 2010) was used to manage the bandwidth settings between the VOD server and the client. A total of 144 network conditions were considered and respectively 144 audio-video files are recorded for subjective quality tests. The first part of Chapter 5 explains how we have technically realized this testbed in details.

The participants consisted of 24 graduate level INRS students. They had various cultural backgrounds and all of them were fluent in English which allowed us to deliver the test guidelines and answer any questions raised during the training session in English for all. The observers' age ranged between 20 and 37 years old.

The viewing and listening conditions specified in ITU-T P.911 (2016) were followed as much as feasible. Observers were asked to rate each audiovisual quality on the 5-point ACR categorical quality scale and were allowed to submit their subjective scores after watching/listening to the first 10 seconds. The order of the rendered sequences was randomly drawn before assessment but was the same for all observers. Observers initially performed a training session and completed the tests between 30-45 min in a single assessment session. Observers were allowed to have a pause halfway through the test.

4.3 Two Preliminary Parametric Models

The models we mention in this section were trained on the 5-point ACR MOS scale where the scores for a given audiovisual configuration are averaged over all observers. We have constructed various parametric models and measured their performance in terms of accuracy, consistency, and linearity. We have seen earlier that these terms are represented by the following statistical metrics; Root-Mean-Square-Error (RMSE), the outlier ratio which is typically defined as the points for which the prediction error exceeds the 95% confidence interval and the Pearson correlation coefficient.

Table 4.3 – Preliminary dataset parameters.

File Size	Audio Sampling Count
Duration	Audio Frame Count
Overall Bit Rate	Audio Video Delay
Frame Count	Audio Stream Size
Video Duration	Source File Max Bit Rate
Video Bit Rate	Network Bandwidth
Video Width	Measured Network Bandwidth For Max Bit Rate
Video Height	Network Jitter
Video Frame Count	Network Packet Loss Rate
Video Bits/(Pixel*Frame)	Missing Bandwidth
Video Stream Size	Bit Rate Range
Audio Duration	

We have extracted the features from the file headers such as bits per pixel in each video frame, audio-video delay, duration, frame count, video and audio stream sizes, etc. and additional side information such as network packet loss, network jitter, and bandwidth configurations. We have kept the feature space the same across all machine learning models we have tried. The list of all parameters is given in Table 4.3.

In our quick Weka (Witten & Frank, 2005) experiments, we have witnessed an overall superior performance of the Decision Trees based ensemble methods. Out of all available ensemble methods, Random Forests showed better accuracy in terms of RMSE values calculated. In order to put the Random Forest model's performance into relation to neural networks based models, we have decided to build two models based on Random Forest and Neural Networks. Maki *et al.* (2013) showed that MLP models perform better compared to RNN models. Therefore as neural network implementation, we have used Multi-layer Perceptron.

First, we have trained an MLP regression model using a single hidden layer where the number of input neurons equaled the number of input features. The tangent hyperbolic function was chosen as the activation function of the hidden nodes, while the linear function was chosen for the output neuron as in (Maki *et al.*, 2013). The learning rate was set to 0.02 and the number of iterations was set to 100 for a gradient descent to perform on the neural network's weights.

Second, we have trained a Random Forests regression model that fits a number of classifying Decision Trees on various subsamples of the dataset and used averaging to improve the predictive

accuracy and control over-fitting. The number of trees in the forest was set to 100 with no restriction on the depth of the tree and all features are used.

Initially, the dataset was shuffled and then both methods were trained and their accuracy measured on the test MOS data using 10-Fold cross-validation. To reduce the variation, as a common practice, we have run this process 10 times and have taken the average of the measured statistical metrics. These figures are shown in Table 4.4. It is clear that Random Forests methods outperform Multi-layer perceptron methods in terms of all metrics computed.

Table 4.4 – Random Forests vs Multi-Layer Perceptron Performance.

Algorithm	RMSE	Pearson Correlation	95% Confidence Interval
Random Forests	0.3138	0.8871	0.597
Multi-layer Perceptron	0.4207	0.8023	0.767

The difference in performance is much easier to realize in a graphical interpretation. Figure 4.2 shows actual MOS vs predicted MOS for both Random Forests and Multilayer Perceptron methods. The figures also show that the Random Forest method makes a more accurate estimation.

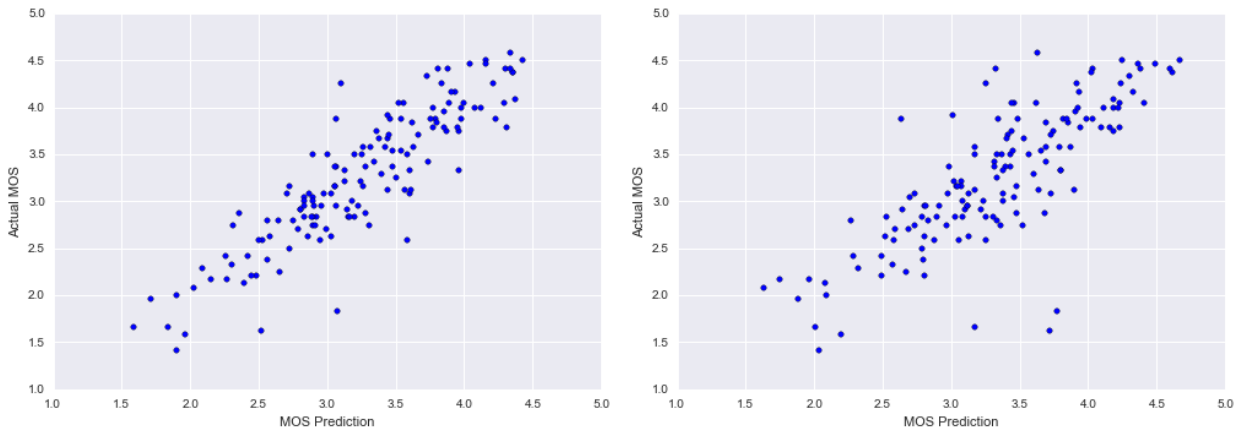

Figure 4.2 – Actual MOS vs predicted MOS: Random Forests (Left) and Multi-Layer Perceptron (Right).

We have intentionally kept all features when training both models. With some feature selection pre-processing the MLP performance might be improved. The beauty of the Random Forests method is that it handles feature selection automatically as well as telling us feature importance which would be extremely useful while adapting the service quality based on the quality predictions made. In Figure 4.3 it is shown that packet loss rate, network jitter, and bandwidth information, provided as side information, play the most important role when estimating perceived quality. It is important to note that changing the range of value of parameters would influence the perceived quality differently.

The values here are the side information but not the actual data collected from the bit stream level. This automatically makes it clear that a hybrid approach that incorporates the packet level information with more accurate bitstream level information would produce much better predictions.

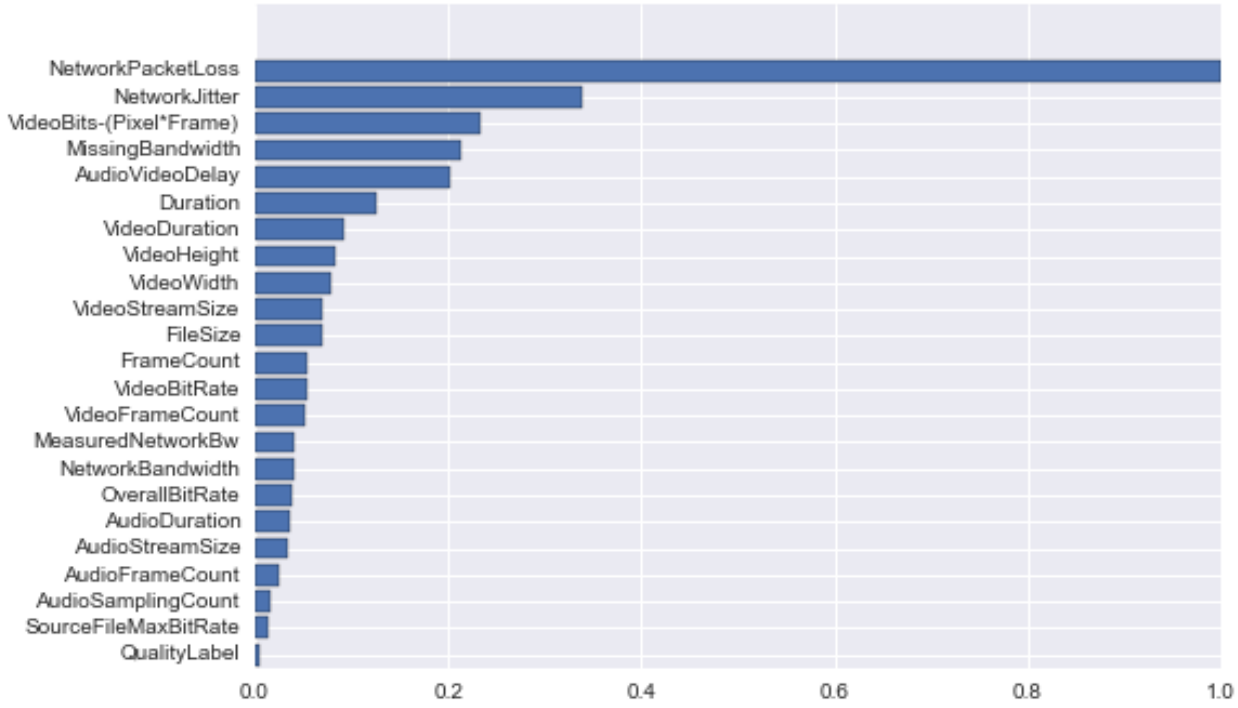

Figure 4.3 – Random Forests feature importance: Network PLR, jitter and bandwidth information have the most influence on estimating the perceived quality.

4.4 Discussion

We have obtained high RMSE and Pearson correlation coefficient values. However, when we look at the figures, we clearly see the outlier points where the MOS estimation values differ from actual MOS value by a large margin. After carefully analyzing the actual MOS values, we have discovered that some of them differ by more than 2 MOS points compared to their actual expected range. Similar issues have been reported by Maki *et al.* (2013) as well. The reason behind this is the difference between the channel parameters provided as side information and the actual bit stream level information. In a hybrid approach, where the packet header information and more accurate bit-stream level information is used, this problem would not occur.

Other setbacks we have faced are due to the test bed we have used. The VLC VOD is a decent off-the-shelf product for simple tests. However, it falls short of expectations when it is used in more advanced test cases. Foremost, it has a lack of support for a variety of video and audio codecs. When network impairments such as packet loss are introduced, it fails to capture entire video stream with only a minor increase in packet loss rate. It certainly did not support real-world use cases where up to 5% video packet loss rate is expected. As it is off-the-shelf, changing the pipeline behavior is next to impossible and requires source code change. It also does not provide stream level network measurements such as packet loss rate, jitter, delay, effective bit rate etc.

To overcome these issues, we need a more robust testbed where we can generate reference videos with ideal encoders, and media and channel parameters. In the following chapter, we will present both the VLC VOD based testbed as well as the new improved GStreamer based testbed. With the dataset generated via this new testbed, we will train and test new models and compare their performances with existing work.

4.5 Summary

We have generated an audiovisual quality dataset in order to gather data for building two Reduced-Reference parametric models for estimating audiovisual quality in multimedia services. We have trained the models using Random Forests and Multi-Layer Perceptron machine learning methods. In terms of RMSE, Pearson Correlation coefficient value and 95% confidence interval boundaries, Random Forests based methods outperformed Multi-Layer Perceptron methods.

Building a parametric model requires less effort compared to bitstream level or signal level models. Using the side information proved to be invaluable in terms of improving the performance metrics. However, the model might suffer from the imperfections contained in the side information.

The parametric model based on the Random Forests achieved high accuracy. Random Forests methods also provide built-in feature importance properties that give insight about which parameters are more influential on the user perceived service quality. This information would be very useful when adapting the service quality based on the quality predictions made.

Instead of side information which has potential imperfections, bit stream level measurements can be used to obtain more accurate quality estimations. However, this approach would require peeking into the bit stream and require more effort to build such a model.

In the next chapters, we will concentrate on hybrid modeling approaches where both packet header and bitstream level information is used to build perceived audiovisual quality models.

Chapter 5

Multimedia Communication Testbeds

In our experimental tests presented in Chapter 4, we have used the VLC VOD software to generate reference audiovisual files with various degree of coding and network degradations. We have successfully built machine learning based models on the subjective quality dataset we have generated using these files. However, imperfections in the dataset introduced by the multimedia framework we have used prevented us from achieving the full potential of these models.

In order to build better models, we have re-created our end-to-end multimedia pipeline using the GStreamer framework for audio and video streaming. A GStreamer based pipeline proved to be significantly more robust to network degradations than the VLC VOD framework and allowed us to stream a video flow at a loss rate up to 5% packet very easily. GStreamer has also enabled us to collect the relevant RTP Control Protocol (RTCP) statistics that proved to be more accurate than network-deduced information. The accuracy of the statistics eventually helped us to generate better performing perceived quality estimation models.

In this chapter, we present the implementation of these VLC and GStreamer based multimedia communication quality assessment testbeds with the references to their publicly available code bases. We also compare our experience using both technologies and share some lessons that we have learned along the way.

This chapter is partly based on the content presented in the following conference and publication:

– Edip Demirbilek and Jean-Charles Grégoire. Multimedia communication quality assessment testbeds. *arXiv preprint arXiv:1609.06612, (2016).*
– Jean-Charles Grégoire. Multimedia communication quality assessment testbed. *GStreamer Conference, 2016.*

5.1 Introduction

Using a live network would be the preferred way to conduct our research. However, the possibility of many unknown features makes it difficult to evaluate live experiments. In order to collect subjective assessment ratings for various file configurations and network settings, we need to be able to manage the network behavior to achieve reliable and reproducible results.

In our quest to achieve reproducible experiments, we have created two multimedia communication quality assessment testbeds. The first testbed makes use of VLC Video-on-Demand (VOD) technology. The second testbed is based on the GStreamer multimedia framework and took considerable effort to design, develop and test. The data set generated using this latter testbed is presented in Chapter 6.

We suggest to refer to Chapter 4 and Chapter 7 for the reasoning behind specific configurations, and refer to this chapter for how these specific configurations are technically implemented.

5.2 Tools

Multimedia quality testbeds enable us to experiment with various encoding configurations through the multimedia frameworks used for streaming and to introduce network impairments through network emulators. In our research we have used the VLC VOD (VLC Team, 2016b) and GStreamer (GStreamer Team, 2016a) multimedia frameworks in order to establish end-to-end pipelines for multimedia streaming. We have also considered libjitsi (Ivov, 2013), an advanced Java media library for secure real-time audio/video communication, during the implementation. However, the absence of media capture features made it unsuitable for our work. For network impairments, we had to consider various criteria while picking the right software among the numerous available solutions. Historically, network emulators have captured the behavior of the links in terms of

queue size, limited bandwidth, the probability of loss and propagation delay. Two most popular and flexible representatives of this class are DummyNet (Rizzo, 1997) and NISTNet (Carson & Santay, 2003) which we have used to introduce network impairments for the testbeds. Other emulator and simulator examples are Netpath (Agarwal *et al.*, 2005), LANforge-ICE (Candela Tech., 2015), AnueSystem Ethernet emulator (Emulators, 2007), NS-2 (McCanne *et al.*, 1997), NS-3 (Riley & Henderson, 2010), ENDE (Yeom & Reddy, 2001) and Satellite Lab (Dischinger *et al.*, 2008). In the rest of this section, we will discuss various properties of the tools that we have chosen to use for building the testbeds.

5.2.1 VideoLAN software and Video-on-Demand (VOD)

VideoLAN (VLC Team, 2016b) is a software solution for video streaming and distributed under the GNU General Public License (GPL). It is designed to stream videos on high bandwidth networks. It includes VLC (initially VideoLAN Client) which can be used as a server to stream MPEG-1, MPEG-2 and MPEG-4 files, DVDs and live videos on the network in unicast or multicast mode or used as a client to receive, decode and display MPEG streams. It is also used to play files from the local disk on multiple operating systems including Linux, Windows, Mac OS X (De Lattre *et al.*, 2002). Although the list here seems to be long, the lack of support for streaming some popular video formats such as H264 and VP8, although it is possible to playback them from the file, is noteworthy. In the following sections, we will see that this feature was one of the main reason for building multiple testbeds.

There is one particular thing about MPEG. MPEG is an audio and video codec standard with several versions called MPEG-1, MPEG-2, MPEG-4. MPEG is also a container format, sometimes referred to as MPEG System. There are several such systems: ES, PS, and TS. As an example; a MPEG video from a DVD is actually composed of several streams (called Elementary Streams, ES) where there is one stream for video, one or more streams for audio, another for subtitles. These different streams are mixed together into a single Program Stream (PS). For example, the .VOB files found in a DVD are actually MPEG-PS files which are not adapted for streaming video through a network or by satellite, for which another format called Transport Stream (TS) was designed. In our tests, we have used the TS format (De Lattre *et al.*, 2002).

VideoLAN solution provides both GUI wizards and command line tools for unicast, multicast and Video on Demand (VoD) streaming needs. Due to its flexibility and ease of use, we have used the VoD feature during our tests.

5.2.2 VLC Python Bindings

Subjective video assessment requires a custom software to be used in order to run specific scenarios while allowing the observer to interact with the system with ease and to self-operate the rating process. Developing a video player with the features required for subjective assessment would take significant effort.

The VLC Python bindings (VLC Team, 2016c) provide complete access to the libVLC API (VLC Team, 2016a) without requiring any compilation and work with multiple VLC versions. In our research, we have created an application on top of the runnable example player included in the VLC python bindings.

5.2.3 GStreamer Multimedia Framework

The GStreamer multimedia framework (GStreamer Team, 2016a) is based on the GLib 2.0 (Gnome Developer, 2016) object model for object-oriented design and inheritance and consists of a comprehensive core library to allow construction of graph-based arbitrary pipeline structures. Through a plug-in architecture, it supports numerous container formats, streaming protocols, codecs, metadata, video and audio configurations. We have constructed the second test bed on top of the GStreamer multimedia framework to address some shortcomings of the VLC framework. In Section 1.1.6 we will discuss these shortcomings in detail.

The features of GStreamer are accessible either through simple API or command line tools. In our research, we have initially created our complex pipelines using command line tools and then created the same pipeline through the Python bindings (GStreamer Team, 2016b). We should note that, with our testbed, each side is implemented independently and could use either approach, without concern for interoperability. As a matter of fact, we have benefited from this feature greatly in our work.

5.2.4 DummyNet

DummyNet is a network emulator developed over a decade ago which has become very popular over time. It has been a standard component of the FreeBSD from the beginning and of Mac OS since 2006. It is designed to have an easy learning curve and one can set up the emulator with as few as two commands and then master additional features as needed (Carbone & Rizzo, 2010). One of the advantages of using DummyNet is that it works on both incoming and outgoing packets. However, it does not allow to emulate degraded network conditions such as packet duplication or corruption (Nussbaum & Richard, 2009).

5.2.5 Netem/TC

Netem is a network emulation facility built into Linux's Traffic Control (TC) subsystem. TC/-Netem use the same principle as DummyNet to capture the ongoing only packets and use a set of rules and queues to store the packets and forward them to the operating system or to the network (Nussbaum & Richard, 2009).

TC allows shaping, scheduling, policing and dropping network traffic. When traffic is shaped, the rate of transmission is controlled on egress. Traffic shaping is also possible on ingress via policing. With scheduling, it improves the interactivity of the traffic that needs it while still guaranteeing bandwidth to bulk transfers on egress. Both on ingress and egress, traffic exceeding a specific bandwidth level can also be dropped (Almesberger, 1999).

TC allows traffic processing via three kinds of object: qdiscs, classes, and filters. Qdisc is short for "queueing discipline" and is used to enqueue the traffic destined to a specific interface. Some qdiscs can contain classes which contain further qdiscs to enqueue the traffic. With this mechanism, a qdisc may prioritize certain kinds of traffic by trying to dequeue from certain classes before others. A filter is used by a qdisc to determine which class a packet will be enqueued (Almesberger, 1999).

5.2.6 DummyNet vs Netem/TC

Both DummyNet and TC/Netem are no longer prototypes and have reached production quality. Furthermore, they both are freely available and used by large communities of researchers (Nussbaum & Richard, 2009).

In the case of bandwidth limitation, DummyNet simply computes the delay to add to a specific packet based on the configured bandwidth and the current state of the queue. TC uses a Token-Bucket algorithm to shape traffic (Nussbaum & Richard, 2009).

Nussbaum & Richard (2009) have conducted detailed bandwidth experiments with both solutions and concluded that DummyNet is slightly more accurate than TC when limiting bandwidth. During our tests, we have experienced similar results in terms of streaming quality and decided to use DummyNet for bandwidth limiting configurations. However, it is important to note that by design DummyNet does not allow one to achieve very high emulated bandwidth since the timer frequency might lead to burstiness which leads to unrealistic traffic (Nussbaum & Richard, 2009).

While performing slightly better at bandwidth limitation, DummyNet does not contain any built-in feature to emulate jitter. Therefore, rather than creating some workaround with DummyNet that has not been tested extensively, we decided to introduce jitter and delay using the TC/Netem. During the tests, we have also observed that TC handles packet loss configurations better than DummyNet in terms of quality achieved in streaming and therefore the decision was made to use TC for that purpose as well.

5.3 VLC VOD Based Multimedia Communication Quality Assessment Testbed

We have built a dedicated setup to conduct the streaming tests. To do so, we have used two dedicated workstations running the Ubuntu OS with the VideoLAN software solution. To manage network traffic effectively we have used both DummyNet and TC/Netem network emulation solutions. DummeyNet is used to manage the available bandwidth while TC/Netem is used to control jitter and delay. Predefined source video files are streamed from the VLC VoD server towards VLC Client and saved on the client local disc. Each file is saved with a file name that includes the network

configuration set for that specific streaming case. Figure 5.1 depicts the high-level design of this test setup. Detailed implementation of this testbed is given in Appendix 1.1.

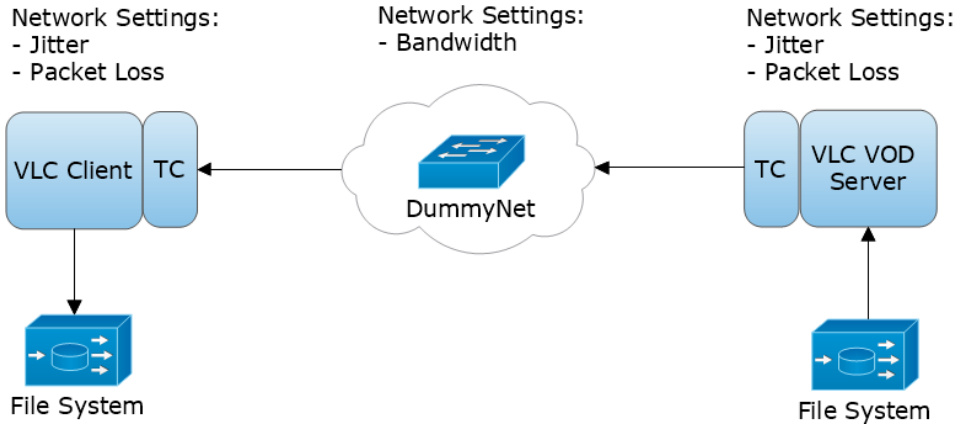

Figure 5.1 – VLC VOD based multimedia communication quality assessment testbed.

5.4 GStreamer Based Multimedia Communication Quality Assessment Testbed

Due to the limitations of the VLC VoD based test bed, we have implemented a second testbed using the GStreamer multimedia framework for media streaming. When introducing network impairments, we have utilized only the Netem/TC tool to apply a packet loss rate to multimedia traffic. Jitter and additional bandwidth limitations were not introduced. As a result, the DummyNet tool was not needed anymore. Figure 5.2 depicts the high-level design of this test setup. Detailed implementation of this testbed also is given in Appendix 1.2.

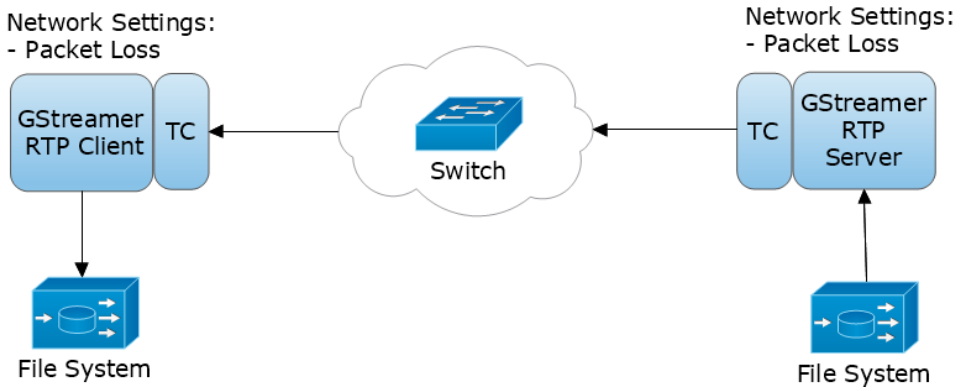

Figure 5.2 – GStreamer based multimedia communication quality assessment testbed.

5.5 Subjective Assessment Video Player

A custom video player (Demirbilek, 2016c) was developed to collect the subjective scores. This video player allowed to user to rate the quality of the video file on the 5-point ACR categorical quality scale. The menu that allows the user to make a choice and the button to submit that choice appears only after watching and listening to the first 10 seconds of video sequences. This period was selected due to the nature of the test case. The menu structure allows the user to carry a training session for rating the quality of the various audiovisual files created from the same original sequence. The video assessment consisted of two sessions of four parts each. The videos file name is read from a specifically sorted list stored in a file depending on the selection of the session and part. The video player also allows a user to pause the media playing if needed. During the assessment, users can follow the number of sequences rated from the top bar. At the end of each part, the user is informed about progress and further instructions are given for the next step. A screen shot of this software playing an example video is given in Figure 5.3.

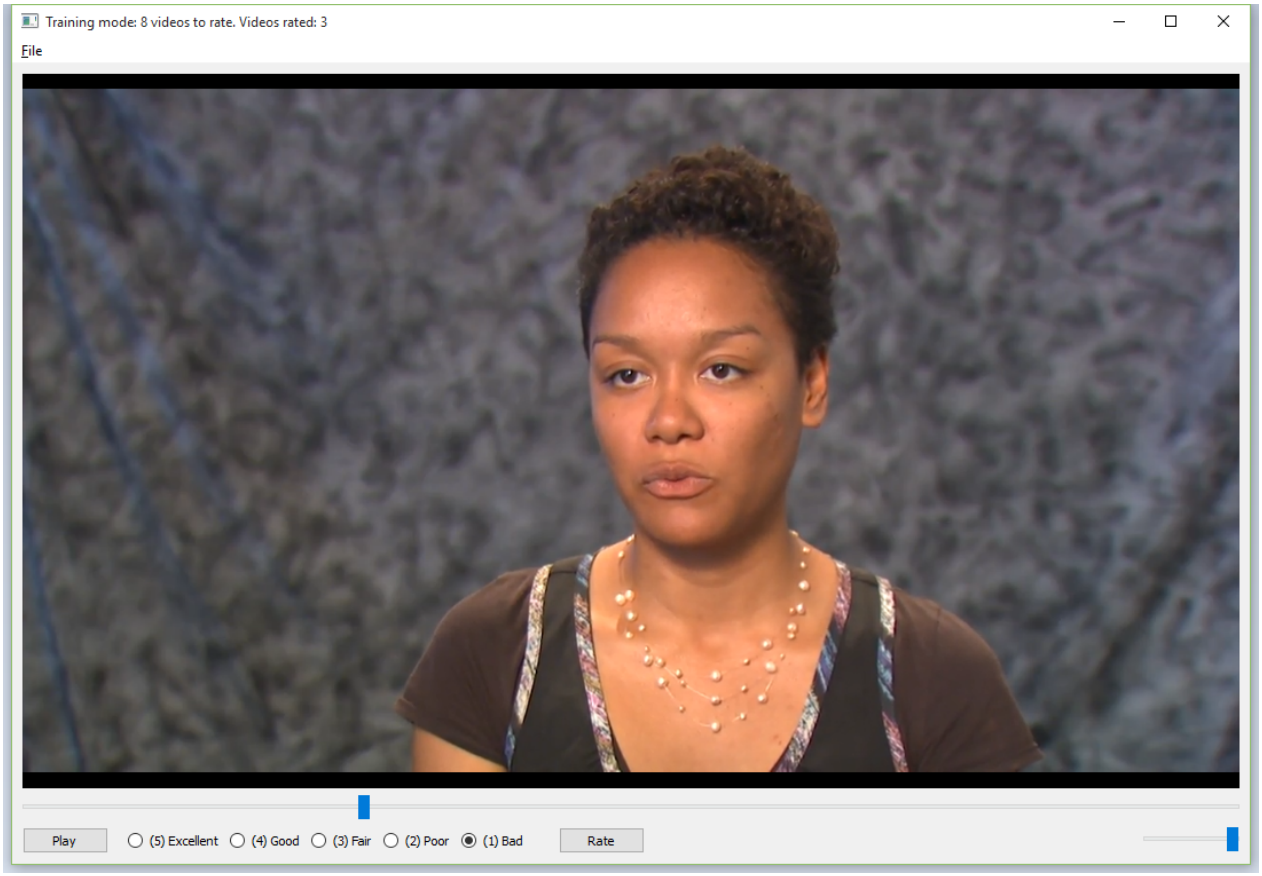

Figure 5.3 – A scene from the generated reference video file.

	Off-The-Shelf	Learning Curve	Applications	Streaming Codecs	Network Impairments	RTCP Statistics
VLC Video-on-Demand	Yes	Easy	Basic Scenarios	MPEG	Not Robust	No
GStreamer	No	Moderate	Real-World Scenarios	Most Modern Codecs	Robust	Yes

Table 5.1 – VLC VOD vs GStreamer Framework for multimedia communication quality assessment testbed.

5.6 Summary

We have created testbeds based on the VLC Video-on-Demand product and Gstreamer multimedia framework. DummyNet and Netem/TC were used to introduce network impairments to the media streams.

The VLC VoD was capable of handling simple scenarios that did not involve heavy network impairments. Specifically, it failed to capture entire video stream with rates greater than 0.5 percent. This is significantly lower than real-world use cases where up to 5% video packet loss rate is expected. As the VLC VoD is off-the-shelf, changing the pipeline behavior is next to impossible and it also does not provide stream level network measurements.

In order to build better models, we have re-created our end-to-end multimedia pipeline using the GStreamer framework for audio and video streaming. A GStreamer based pipeline proved to be significantly more robust to network degradations than the VLC VOD framework and allowed us to stream a video flow at a loss rate up to 5% packet very easily. GStreamer has also enabled us to collect the relevant RTCP statistics that proved to be more accurate than network-deduced information. The accuracy of the statistics eventually helped us to generate better performing perceived quality estimation models. A brief comparison of the VLC VOD and GStreamer is also given in Table 5.1.

Although during the implementation we have faced some minor setbacks, overall, creating our testbed on top of GStreamer framework turned out to be a wise decision and we strongly recommend it for similar work.

Chapter 6

The INRS Audiovisual Quality Dataset: Parametric Version

In this chapter, we first list existing publicly available audiovisual quality datasets and then explain the parametric version of the INRS audiovisual quality dataset which we have generated for this research in details.

This chapter is partly based on the content presented in the following conference and journal:

- Edip Demirbilek and Jean-Charles Grégoire. INRS audiovisual quality dataset. *Proceedings of the 2016 ACM Multimedia Conference, 2016.*
- Edip Demirbilek and Jean-Charles Grégoire. Machine learning based parametric audiovisual quality prediction models for realtime communications. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)(Accepted).*

6.1 Existing Publicly Available Audiovisual Datasets

The Qualinet Multimedia Databases set v5.5 (Fliegel, 2014) provides a list of some publicly available audiovisual databases and models.

6.1.1 University of Plymouth Dataset (PLYM)

Goudarzi *et al.* (2010) have created an audiovisual quality database to discover methods for audiovisual quality predictions for video calls for wireless applications. They have conducted subjective tests for 60 audio, 60 video and 60 audiovisual test conditions. They have used 6 low-motion and different spatial complexity video sequences where each video had a 176x144 resolution and lasted between 7 and 14 seconds. They have used 2 video frame rates and 5 packet loss rates. The videos were encoded with the H.263 video codec and the G.711 μlaw audio codec. Absolute Category Rating (ACR) (ITU-T P.913, 1998) with a discrete 9-level quality scale for low bit-rate evaluations was used during the experiments. They have conducted subjective tests to assess audio, video and audiovisual quality with 16 observers in each part.

6.1.2 TUM 1080p50 Dataset (TUM)

Keimel *et al.* (2012) have created subjective quality datasets for high definition videos in 1080p25 and 1080p50 formats. From these datasets, only the 1080p50 dataset consisted of audiovisual sequences. Five 10 seconds long source video sequences were encoded with the H.264/AVC video codec, each with one of the four chosen bitrates depending on the motion complexity of the source sequences in the range of 2 Mbit/s to 40 Mbit/s. The subjective test was carried with a projector and a 7.1 surround sound system following the single-stimulus SSMM methodology. Twenty-one observers participated in the study and rated the perceived quality for 20 different data points using a discrete 11-level quality scale from 0 to 10. The encoding impairments introduced into the videos were blurring, visible blocking or ringing artifacts, and flicker and similar effects.

6.1.3 VQEG Dataset (VQEG)

Pinson *et al.* (2012) and Pinson *et al.* (2013) have conducted subjective tests on the same audiovisual material at six different international laboratories to find out the most appropriate way to perform audiovisual quality testing. Overall ten different datasets were produced. Each dataset contained subjective scores for 10 audiovisual source sequences lasting 10 seconds. Eight of audiovisual source sequences contained music or singing in English, and 2 contained speech with background noise. The video sequences had a 640x480 video resolution and a 30 fps frame rate.

The H.264/AVC video codec with 6 bitrate levels was used for encoding. Advanced Audio Coding (AAC) was used to encode audio streams at 8, 32, and 64 kbps. For each audio and video sequence, three coding qualities are selected: high, medium, and low. The experiment included five randomly picked impaired variant of each source and the original source sequences. Observers have rated the overall quality of 60 video clips on the ACR using a 5-point discrete quality rating scale. In each lab, between 9 and 35 observers took part in the experiment. The authors inferred that the number of observers was the most substantial control variable for a reproducible subjective experiment. They have recommended to have 24 or more observers for ACR tests and to have 35 observers in the public environment to get the same Student's t-test sensitivity.

6.1.4 Made for Mobile Dataset (Vienna)

Robitza *et al.* (2012) have created a video dataset especially designed to evaluate content production rules and video quality between mobile and television. The Made for Mobile Database consisted of 19 pairs of extracted video sequences from 22 professionally produced clips. Researchers designed the experiment based on pair comparisons, with 18 observers. Each pair consisted of a made for mobile and a made for television video, with a duration between 8 and 15 seconds. The H.264 video codec's baseline profile was used to encode the videos at an average bit rate of 500 Kbit/s and at the picture size of 854x480. The video sequence was multiplexed with uncompressed PCM audio.

6.1.5 VTT Dataset (VTT)

Maki *et al.* (2013) have created an audiovisual quality dataset to predict the audiovisual quality in streaming services. They have conducted audiovisual quality assessment tests using H.264 video and AAC audio streams in streaming services. They have analyzed the influence of several quality factors. The video sequences were encoded with the H.264 baseline profile into different bitrates depending on their resolution. The audio was encoded with the AAC codec at two different bitrates. Subjective audio, video, and audiovisual scores were collected in one assessment session from 24 observers for 125 streamed video sequences. Video samples varied by the impact of resolution, movement quantity (MQ), packet loss rate, and mean loss burst size. The authors have altered the Degradation Category Rating (DCR) method to collect audio, video and audiovisual ratings in one session. The content is selected with the low, medium and high movement quantity. Maki et al.

have built Reduced-Reference parametric models for audio, video and audiovisual quality estimation following the Pseudo-Subjective Quality Assessment (PSQA) methodology and have tested Random Neural Networks (RNN) and Multilayer Perceptrons (MLP). They have trained the models and have reported that the MLP model is performing better. They have also demonstrated that incorporating the Reduced-Reference metrics improves the performance of these models in certain uses.

Additionally, the TVM and P.NAMS training and validation datasets (Garcia, 2014) have led to ITU-T standards ITU-T P.1201 (2012), ITU-T P.1201.1 (2012), ITU-T P.1201.2 (2012) and ITU-T G.1071 (2015). These datasets have only recently been made publicly available (Garcia *et al.*, 2016) during the writing of this thesis and therefore are not included in the results we have published.

Existing audiovisual datasets are invaluable for building perceived quality estimation models for one-way streaming based services. However, some of these datasets listed in Table 6.1 also have some limitations in terms of real-time communication use cases, mainly because they have been generated with a different scope in mind. The University of Plymouth dataset (Goudarzi *et al.*, 2010) is interesting with respect to real-time communications, however, video frame rates and the video resolution selected no longer reflect contemporary applications. This research specifically targets one-to-one real-time communications while exploring the encoding configurations and network impairments often seen in contemporary video-telephony applications.

Table 6.1 – Publicly available audiovisual quality datasets.

	PLYM	TUM	VQEG	Vienna	VTT	INRS
ContentType	6x	5x	5x	19x	4x	1x
Duration(s)	7 - 14	10	10	8 - 15	10	42
Motion	Low	Low-High	Low-High	Low-High	Low-High	Low
Bitrates	1x /Seq	4x /Seq	3x /Seq	1x /Seq	3x /Seq	4x /Seq
Resolutions	144p	1080p	480p	480p	480,720,1080p	720p
FR	8, 15	50	30	25	25-30	4x (10-25)
PLR %	5x (0.01-0.20)	No	No	No	5x (0.3-4.8)	5x (0-5)
MLB	No	No	No	No	1.0-3.0	No
Noise Red.	No	No	No	No	No	2x
Video Codec	H.263	H.264	H.264	H.264	H.264	H.264
Audio Codec	G.711	PCM	AAC	PCM	AAC	AMR-WB
Audio MOS	60	No	No	No	125	No
Video MOS	60	No	No	No	125	No
AVMOS	60	20	60	No	125	160
Observers	16	21	9-35	18	24	30
Scale	9	11	5	11	5	5
Year	2010	2012	2012-2013	2012	2013	2016

It is tempting to address multiple motion complexity and video resolutions levels and multiple contents in the same subjective dataset. However, that approach significantly lessens the effective number of data points per motion complexity, resolution level, and content type because of practical limits. In Section 6.2 we will discuss the reasoning behind the specific content selected for the INRS audiovisual quality dataset. The audio and video codecs used influence the results obtained as well. The AAC audio codec or uncompressed audio are often used because they extend to music. However, in the telecommunication domain, a speech codec like Adaptive Multi-Rate Wide-Band (AMR-WB) is more often used. In this research, we have kept frame rate and quantization range as large as possible alongside noise filter and network packet loss rate. We have also used AMR-WB to encode speech-only audio streams.

All of these audiovisual databases have a variety of configurations. Researchers have built audiovisual quality models using the data provided in these databases. Some models aim to estimate the perceived quality directly by conducting the audio-video subjective tests while others conduct audio, video, and audio-video test separately and try to deduct a model accurate enough for estimating the audiovisual quality by using the separate audio and video quality estimates as parameters.

As recent developments show, No-Reference and Reduced-Reference parametric models achieve high accuracy in estimating perceived quality with limited resources. In Chapter 7 and Chapter 8, we try to build and analyze similar models by estimating the audiovisual quality directly using Random Forest, Bagging, Genetic Programming and Deep Learning machine learning methods based on the dataset presented here.

6.2 The INRS Audiovisual Quality Dataset: Parametric Version

The INRS audiovisual quality dataset has been designed to span the most important compression and network distortion influence factors. These factors are typically video frame rate, quantization and filters, and network packet loss rate. From our collaboration with the industrial partner, Summit-Tech Multimedia Communications Inc., we have chosen the range of these parameters for the H.264 video encoding as follows: (0, 0.1, 0.5, 1 and 5%) for network packet loss rate in both video and audio streams, (10, 15, 20 and 25 fps) for video frame rate, (23, 27, 31 and 35) for quantization parameter, and (0 and 999) for the video noise reduction filter.

6.2.1 Video Sequences and Test Setup

The original audiovisual sequence, ntia_HeadShouldersFemale15_original.avi file, has been obtained from "the Consumer Digital Video Library" (Pinson, 2013). The video consists of head-and-shoulder content with a speech similar to a typical one-to-one audiovisual conversation. Traditional approaches mostly target the IPTV scenarios and therefore consist various video sequences that have different motion complexities. For the INRS audiovisual quality dataset, however, we target video-telephony applications that mostly consist of head-and-shoulder content where the range of motion complexity is much limited and similar along the use cases. In terms of encoding and the video I-frame frequency, head-and-shoulder contents tend to resemble each other to a great extent and therefore in our research we have given our attention to increasing the range of encoding configurations and network impairments rather than variations in head-and-shoulder content.

This single type of content can potentially bore the observers quickly during the subjective assessment. In order to prevent that, we have divided the whole experiment into multiple sessions and parts, and we have also introduced the rejection criteria during post processing in order to be able to detect moments and periods of inattention. In Section 6.2.2 we discuss the methodology that we have followed in more details.

Another important setting that influences the overall perceived quality is the frequent use of I-frames. In the previously mentioned datasets, there is an I-frame every 1-2 seconds because of high motion complexity as well as for uniformly spreading impairments under network packet loss. However, from our collaboration with the industrial partner, we know that the duration between two following I-frames can be up to the order of minutes in one-to-one real-time communications. In this research, we have kept the default value for the video I-frame periods, 10 s, for the low-motion videos set by the video encoder. This duration of the video I-frame period then required to use a longer video sequence than traditional 10-15 s long videos. The ntia_HeadShouldersFemale15_original.avi file has 42 s long video and audio sequences. During the subjective tests, however, observers were required to watch at least the first 10 s of the sequences whereas they were free to watch the rest of the video before submitting their quality ratings. This duration was selected to be large enough to see how many I-frame cycles and overall duration it required for observers to conclude their subjective assessment per videos. In the end, the type of the content and the longer duration then obliged us to customize our testing methodology (see Section 6.2.2).

Table 6.2 – Media compression parameters and network impairments.

	Video	Audio
`Frame Rate`	10, 15, 20, 25	Mono, 16kHz, 24 kbps
`Quantization Parameter`	23, 27, 31, 35	Mono, 16kHz, 24 kbps
`Noise Reduction`	0, 999	Mono, 16kHz, 24 kbps
`Packet Loss Rate (%)`	0, 0.1, 0.5, 1, 5	0, 0.1, 0.5, 1, 5

The original 42 seconds long audiovisual raw source was encoded with the H.264/AVC video codec and the AMR-WB audio codec and then multiplexed into a 3gp container with the GStreamer open source multimedia framework (GStreamer Team, 2016a) to produce 32 reference audiovisual files. These reference files had various qualities in terms of selected video frame rate (FPS), the quantization parameter (QP) and noise reduction (NR) values. These values are listed in Table 6.2. Video streams were encoded with the constrained-baseline profile at a 720p progressive video resolution. Audio encoding settings were kept the same -mono channel, 16 kHz sample rate, and 24kbps bit rate- for all audiovisual sequences. The multimedia framework, GStreamer, uses only the jitter buffer mechanism to smooth out the packet flow and has no forward error correction strategy and hence in our dataset we assume that the packet loss figures reported are the residual packet losses.

An emulated network was used to transmit and record the audiovisual sequences. The audio and videos streams were captured with our GStreamer based custom developed software which enabled us to gather detailed RTCP statistics and report the exact network packet loss values for video and audio streams separately. The detailed information on the technical implementation of this testbed and how each of these configuration settings are implemented is presented in Appendix A. The Netem network emulator was deployed to produce network packet loss conditions. Network packet loss was only activated after the first second of the audio and video transmission. This allowed us to get more realistic results. Thirty two reference files, under 5 conditions of loss has resulted in 160 unique combinations of compression and network impairments in distinct A/V files. A custom video player was developed to collect the subjective scores (Demirbilek, 2016c).

6.2.2 Testing Methodology

Eleven female and nineteen male observers, ranging between 20 and 48 years old, took part in the study. Each observer was given written instructions in English and allowed to rate the overall

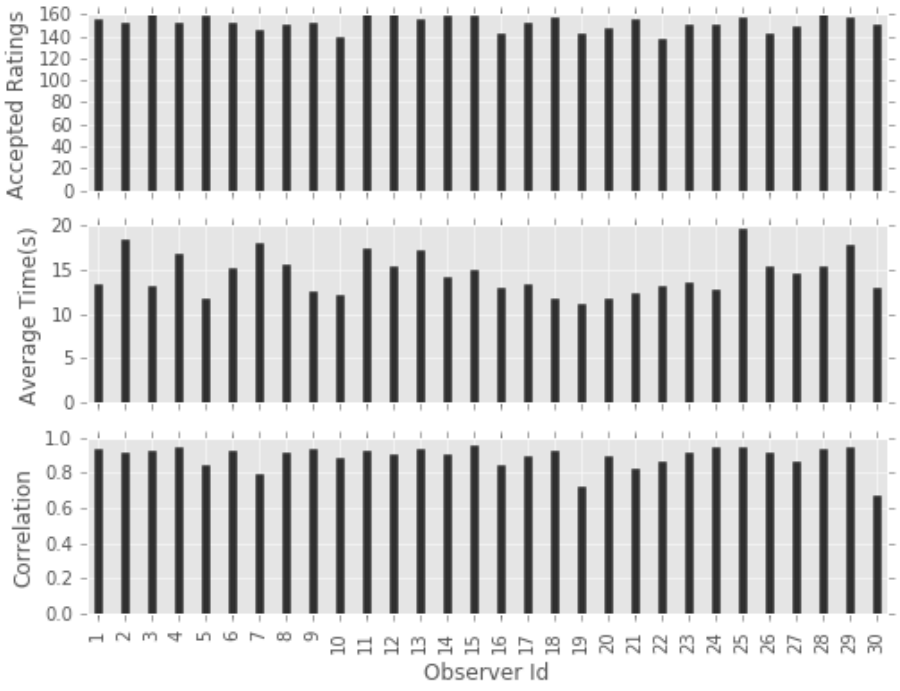

Figure 6.1 – Accepted subjective scores, average time to rate and Pearson correlation coefficient per observer.

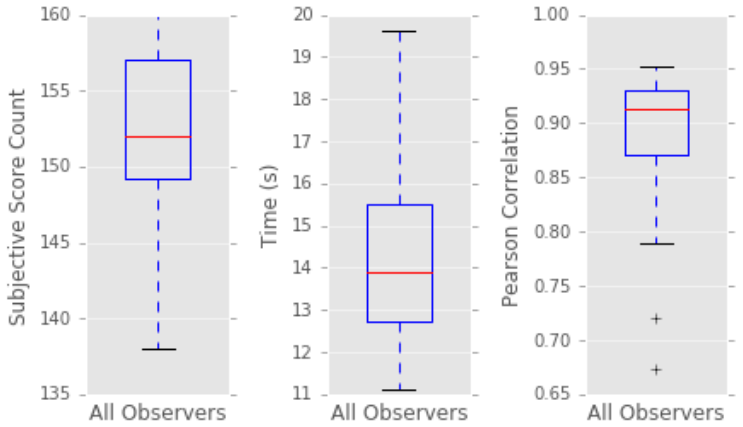

Figure 6.2 – Variation for accepted subjective scores count, average time to rate and Pearson correlation coefficient.

audiovisual quality on the 5-point ACR categorical quality scale. Observers completed the tests in a sound isolated environment using headphones and a computer. Viewing and listening conditions mentioned in ITU-T P.913 (1998) were followed as close as possible. Observers were allowed to submit their subjective scores only after watching and listening to the first 10 seconds of video sequences. This period was selected because of the video I-frame intervals. The order of the presented sequences was drawn randomly during the planning phase of the assessment and this configuration

Table 6.3 – Independent parameters.

General File Size	Video Stream Size Proportion
General Duration	Key Int
General Overall Bit Rate	Key Int Min
General Frame Rate	Audio Duration
General Frame Count	Audio Bit Rate
General Stream Size	Audio Frame Count
General Stream Size Proportion	Audio Stream Size
General Data Size	Audio Stream Size Proportion
General Footer Size	Video Octets Received
Video Duration	Video Packets Received
Video Source Duration	Video Packets Lost
Video Bit Rate	Audio Octets Received
Video Frame Count	Audio Packets Received
Video Bits (Pixel*Frame)	Audio Packets Lost
Video Stream Size	

was saved and followed for all observers. Observers at first performed a training session rating the quality of the various audiovisual files created from the same original sequence. Because of the high number of test conditions, observers completed the tests in two sessions. Each session took about 45 minutes and consisted of 4 parts to allow observers to have frequent pauses if needed. In each session, in the first two parts, 80 videos were rated by the observers. In the third and fourth parts, previously seen videos from the same session were rated again. This allowed us to collect two ACR scores for each file from each observer and measure the consistency of each observer independently. We have rejected both ACR scores for the same file from the same observers when the difference between both ACR scores was more than 1. In Figure 6.1, we give the total number of accepted scores, the average time to rate as well as the Pearson correlation between the individual scores and the MOS values for each observer. A different interpretation of the same data using the box plots is given in Figure 6.2. When we look into the detailed scores submitted by each observer (see Table B.4 in Appendix), specifically the observers that have 10 and more scores rejected, we observe short time periods where there are bursts of rejected scores. This information shows that observers were not very attentive to the test during these short periods of time and applying rejection criteria has allowed us to discard those scores. At the beginning of the first session, each observer was tested for normal visual acuity and color vision and it was noted that the observer with ID 20 had the red-green color deficiency.

6.2.3 Analysis

The variation in MOS value for various packet loss rates, video frame rates, quantization parameters, and noise reduction values are given in Figure 6.3. We observe that the change in the network packet loss rate has dramatic effects on the perceived quality. Both the video frame rate and the quantization parameter, given the value range, have a moderate influence on the MOS value. Considering the best values for the lowest frame rate and quantization parameter that generate the smallest video bitrate, it seems that the video frame rate has a minor advantage over the quantization parameter. However, it is important to remember that this advantage is small and bound to specific parameter combinations. Detailed modeling attempts would reveal the complex relationship between those parameters. Changing the video noise reduction parameter has the smallest effect on the perceived quality compared to the change in other parameters in our test environment. However, the video noise reduction filter does still play an important role. Minimizing the potential noise in the original videos can help us to gain the maximum amount of compression which in the end allows us to select higher frame rates and the quantization parameter for available bandwidth in a contended environment. In our tests, reducing the noise in the original videos has helped us to reduce the file sizes by up to 24%.

Figure 6.4 shows the scatter plots of MOS values and their 95% confidence intervals for all the processed stimuli. For lower and higher packet loss values, there is a soft transition between average MOS values as the video frame rates changed. However, for intermediate packet loss values, 0.1% and 0.5%, there are fluctuations in average MOS values as the video frame rates change. However, for intermediate packet loss values, 0.1% and 0.5%, there are fluctuations in average MOS values as the video frame rates change. To find out the root cause of these fluctuations, we have looked at the actual packet loss values per stream in each file and have also computed how these packet loss values affected the I-frames and P-frames. One of these fluctuations is happening at PLR=0.1%, NR=0 and FPS-15 for QP=23. Our analysis has shown that for QP=23 case, the actual packet loss value for the audio stream is 0.14, while the actual audio packet loss value for other QP cases are between 0.5 and 0.9. Another visible fluctuation is happening at PLR=0.1, NR=999 and FPS=10 for the QP=27. Our analysis revealed that at the 10th second, P-frames for QP=27 in the video stream are dramatically affected compared to other cases. During the modeling, we will be using the actual packet loss values reported and therefore we expect the parametric models to capture them. However, the fluctuations happening due to the unexpected patterns in the I- and P-frames

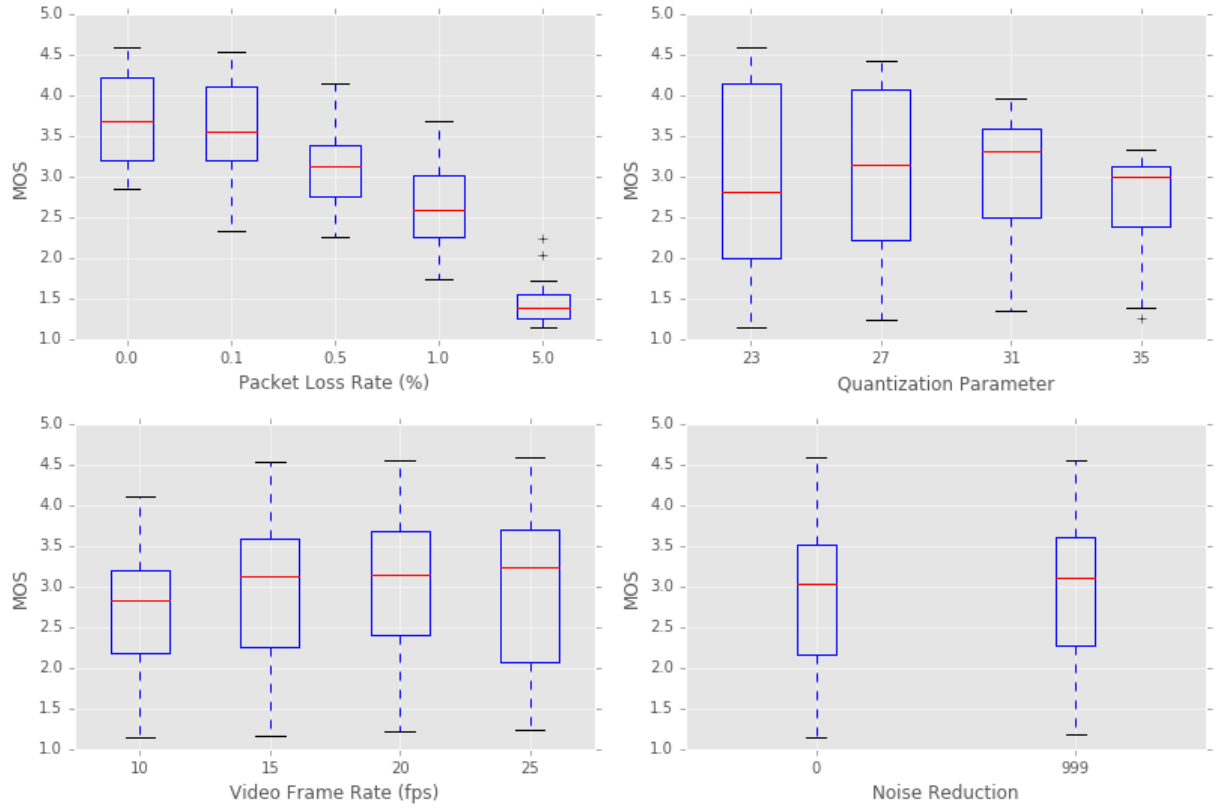

Figure 6.3 – Variation in MOS value for packet loss rate, video frame rate, quantization parameter and noise reduction.

can be captured only by bitstream models. In our research, to see if parametric models and future bitstream models do really capture those differences, we have kept the dataset as is and have not removed any cases.

6.3 Summary

The INRS audiovisual quality dataset is composed of 160 unique configurations for audiovisual content including various media compression and network distortion parameters such as video frame rate, quantization and noise reduction parameters, and packet loss rate. The compression and network distortion parameter range values have been selected to match real-time communications use cases. The H.264 video codec at 720p resolution and the AMR-WB audio codec have been used for encoding video and audio streams. Thirty observers have rated the overall audiovisual quality on the Absolute Category Rating (ACR) 5-level quality scale in a controlled environment. The dataset includes MOS values, packet loss rates measured at bit stream level for both video and

audio streams, compression parameters and various packet header information (Table 6.3). We have used open source software for producing source audiovisual sequences, end-to-end streaming, and a custom video player.

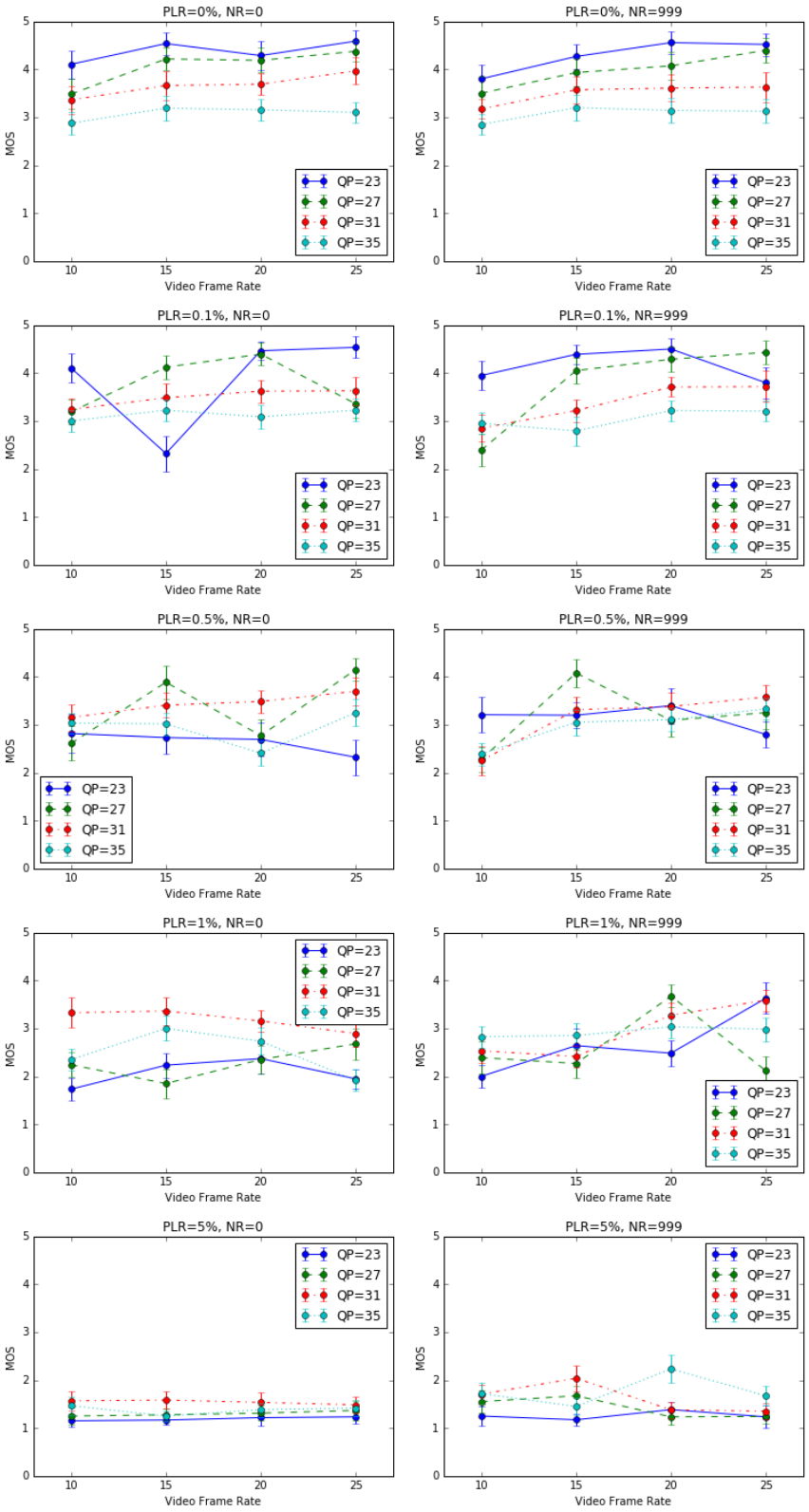

Figure 6.4 – Perceived quality at video frame rates from 10 to 25 for different packet loss rates. The whiskers represent the 95% confidence intervals of the subjective test results for the perceived audiovisual quality.

Chapter 7

Parametric Quality Models

In this chapter, we introduce No-Reference parametric models that we have built and share the results obtained for Random Forests, Bagging, Genetic Programming and Deep Learning methods using the parametric version of the INRS audiovisual quality dataset that we have presented in Chapter 6. We also share the results for the Random Forest based models trained and tested on some of the publicly available datasets.

This chapter is partly based on the content presented in the following journals:

- Edip Demirbilek and Jean-Charles Grégoire. Machine learning based parametric audiovisual quality prediction models for realtime communications. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)(Accepted).*
- Edip Demirbilek and Jean-Charles Grégoire. Perceived quality prediction models: Taking advantage of correlated data. *Springer Quality and User Experience Journal: Topical Collection on Managing QoE of Future Networks and Applications (Submitted).*

7.1 Introduction

The classic approach to audiovisual quality modeling is to develop functions to predict audio and video quality independently and then combine them using another function to predict the overall audiovisual perceived quality. The alternative way is to build models that predict the audiovisual

quality directly without any intermediate functions. Machine learning based modeling approaches have been successfully applied to estimating perceived quality (Maki *et al.*, 2013). In this research, we have taken the second approach and have built machine learning based models that predict overall audiovisual quality in one single function. In order to realize that, we have used the INRS audiovisual quality dataset–that includes both the audiovisual sequences and their corresponding quality rating–during the training, validation and test phases.

We have obtained the training and test data from the application and network layers, and have not used any information regarding the original audiovisual sequences. We have built several models using the Decision Trees based ensemble methods, Genetic Programming and Deep Learning methods. Out of all available ensemble methods, Random Forests and Bagging methods performed the best in the metrics we have used. Random Forests based models have also provided feature importance information that is very valuable for further research. For the Deep Learning methods, we have tested models containing up to 20 hidden layers and identified the optimum number of hidden layers. Once we have found the best performing model, we have run it over other publicly available datasets and compared the results with other existing models already tested.

7.2 Machine Learning Based Audiovisual Quality Models

7.2.1 Preliminary Weka Tests

From our experimental tests in Chapter 4, we have observed the overall superior performance of the Decision Trees based ensemble methods trained and tested on an audiovisual quality datasets. Here with this new dataset, we have extended that research and initially conducted an extensive screening process to find out better performing models using the Weka workbench. The results of this screening process are given in Table 7.1. In order to obtain these results, we have chosen the 10-Fold cross-validation with the default parameter settings for each listed algorithm.

These results confirm our previous finding that Decision Trees based ensemble methods outperform other methods. In Table 7.1, Random Forests, and Bagging models perform quite well in terms of RMSE and Pearson correlation values calculated. Neural Networks based models have been widely used in audio and video quality estimation as well. In order to compare the Decision Trees

Table 7.1 – Weka test results.

Category	Method	Correlation	RMSE
Meta	AdditiveRegression	0.815	0.571
	Bagging	0.911	0.397
	Dagging	0.815	0.581
	EnsembleSelection	0.897	0.427
	RandomCommittee	0.868	0.483
	RandomSubSpace	0.898	0.424
	RegressionByDiscretization	0.862	0.493
Trees	AlternatingModelTree	0.875	0.485
	DecisionStump	0.727	0.665
	ExtraTree	0.871	0.490
	M5P	0.899	0.423
	RandomForest	0.906	0.407
	RandomTree	0.822	0.566
	REPTree	0.866	0.485
Functions	GaussianProcesses	0.836	0.530
	IsotonicRegression	0.857	0.498
	LeastMedSq	0.827	0.593
	LinearRegression	0.835	0.535
	MultilayerPerceptron	0.723	0.791
	PaceRegression	0.815	0.560
	PLSClassifier	0.814	0.567
	RBFRegressor	0.884	0.452
	SimpleLinearRegression	0.815	0.560
	SMOreg	0.823	0.567
Lazy	CAAR	0.766	0.657
	LWL	0.775	0.614
Rules	ConjunctiveRule	0.723	0.669
	DecisionTable	0.798	0.585
	M5Rules	0.890	0.441

based model's performance with popular methods, we have built models based on Random Forests, Bagging, Deep Learning and Genetic Programming.

All methods were trained and their accuracy measured on the test MOS data using 10-Fold cross-validation in Random Forests, Bagging and Genetic Programming and 4-Fold cross-validation in Deep Learning. The independent parameters were video frame rate, quantization, video noise reduction, video packet loss rate and audio packet loss rate.

Additionally, we have extracted 31 other parameters such as audio and video bitrates, frame counts, stream sizes from sample files via Media Info Metadata Extraction Tool (Martinez, 2010). These additional parameters are listed in Table 6.3.

7.2.2 Decision Trees Based Models

We have used the Python scikit-learn's implementation of Random Forests (RF) and Bagging (BG). We have generated two Random Forest based models and two Bagging models that used either only the independent parameters as features (5 features) or all extracted parameters as features (34 features) and have compared the results. For the sake of simplicity, we will call the Random Forest model that used all parameters (independent and additional parameters) the RF1 model, and the Random Forest model that used only the independent parameters the RF2 model. We similarly call the Bagging model that used all parameters (independent and additional parameters) the BG1 model, and the Bagging model that used only the independent parameters the BG2 model. For all models we have set the tree size to 200, max_features to all when looking for the best split and no limits on the tree depth.

7.2.3 Deep Learning Based Models

We have generated the Deep Learning (DL) models using the Keras Deep Learning library that runs on top of the Theano library. We have run several models using both independent variables and all variables and have experimented with the vast amount of the configurations available in the Keras API including all available activation, optimization, initialization, objective and constraint functions and dropout layers. Deep Learning models that use only independent variables performed better compared to models that used all variables. During the experiments, we have generated models that had up to 20 hidden layers. We have found that Deep Learning models that have only one hidden layer outperform models that have more hidden layers. However, for the purpose of comparison, we have included a model that had 3 hidden layers since in literature we came across such models. For the sake of simplicity, we will call the Deep Learning model that had only one hidden layer DL1 and the Deep Learning model that had 3 hidden layers DL2.

Table 7.2 – Keras Deep Learning configurations.

Variable	Value
`Model Type`	Sequential
`Neural network Layers`	Dense
`Activation Function`	Tanh and Softplus
`Initialization Function`	Uniform
`Optimizer`	Adadelta
`Loss Function`	MSE
`Batch Size`	4
`Epoch Size`	440

For both the DL1 and the DL2 models, we have used the Keras configurations shown in Table 7.2. It is important to note that finding the optimum variable values for Deep Learning based models is significantly more difficult than tuning the variables for Decision Trees based models.

7.2.4 Genetic Programming Based Models

We have used the Python gplearn library to implement Genetic Programming based models. The gplearn library is familiar to scikit-learn fit/predict API and works with the existing scikit-learn pipeline. Similar to the Deep Learning models, there are many parameters to tweak. We have used the SymbolicRegressor with the configurations specified in Table 7.3. Similar to the Decision Trees based models, we have generated two models that used either only the independent parameters as features (5 features) or all extracted parameters as features (34 features) and have compared the results. For the sake of simplicity, we will call the Genetic Programming based model that used all parameters (independent and additional parameters) the GP1 model, and the Genetic Programming model that used only the independent parameters the GP2 model.

7.2.5 Results

Event though all tested algorithms performed well on the parametric version of the INRS quality dataset, Decision Trees based models outperformed both the Deep Learning and Genetic Programming models. For the RF1 model, we have obtained 0.340 and 0.930 in terms of RMSE and Pearson correlation values. These values were 0.358 and 0.922 for the RF2, 0.345 and 0.928 for BG1 and 0.355 and 0.925 for BG2 models, respectively. The RF1 and the RF2 models differ from each other

Table 7.3 – gplearn Genetic Programming configurations.

Variable	Value
`Model Type`	Symbolic Regression
`Population Size`	5000
`Generations`	200
`Tournament Size`	20
`Stopping Criteria`	0.0
`Init Method`	full
`Transformer`	True
`Comparison`	True
`Trigonometric`	False
`Metric`	RMSE
`Parsimony Coefficient`	0.001
`P Crossover`	0.9
`P Subtree Mutation`	0.01
`P Hoist Mutation`	0.01
`P Point Mutation`	0.01
`P Point Replace`	0.05
`Max Samples`	0.8
`Jobs`	1
`Random State`	None

in the number of features they use. Similarly, The BG1 and the BG2 models differ from each other in the number of features they use.

Deep Learning based models also performed well and have achieved 0.403 RMSE and 0.909 Pearson correlation for the DL1 model and 0.437 RMSE and 0.894 Pearson correlation for the DL2 model. Recall that the DL1 and the DL2 models differ from each other in the number of hidden layers.

Last but not least, we have obtained the performance metrics for Genetic Programming as well. These values were 0.449 RMSE and 0.881 Pearson correlation for the GP1 model and 0.469 RMSE and 0.870 Pearson correlation for the GP2 model. Recall from Section 3.4.2 that Symbolic Regression implemented via Genetic Programming aims to identify both the parameters and the form of underlying mathematical expression simultaneously. These mathematical expressions depending on the training data selected might have different forms in each run. Equation 7.1 shows an example expression we have obtained for the GP1 model.

$$A \times \log(B + C) \tag{7.1}$$

where

$$A = \left| \frac{1}{-max(PLR_V, StrSizeProp_A) - \log(\sqrt{\frac{Duration_A}{FPS}})} \right| \tag{7.2a}$$

$$B = \frac{\log(\frac{1}{Duration_A}) + BitRate}{-max(PLR_V, StrSizeProp_A) - \log(\sqrt{\frac{Duration_A}{FPS}})} \tag{7.2b}$$

$$C = \frac{max(PLR_V, StrSizeProp_A) + BitRate}{PLR_V + StrSizeProp_A} + StrSize_A + OctRec_A \tag{7.2c}$$

where $BitRate$ is Overall Bit Rate, FPS is Overall Frame Rate, $Duration_A$ is Audio Duration, $StrSize_A$ is Audio Stream Size, $StrSizeProp_A$ is Audio Stream Size Proportion, PLR_V is Video Packet Loss Rate and $OctRec_A$ is Audio Octets Received.

Similarly, an example expression for the GP2 model is given in Equation 7.3.

$$\log(FPS + QP - PLR_V - \sqrt{PLR_V}) - \sqrt{PLR_V} \tag{7.3}$$

where FPS is Video Frame Rate, QP is Quantization Parameter and PLR_V is Video Packet Loss Rate.

In the GP2 model, the dataset configuration consisted of independent variables only and we have expected the underlying mathematical expression to capture the interrelations of all of the independent parameters. However, as shown in Equation 7.3, the model was not able to incorporate all of the parameters. It can be argued that an algorithm does not necessarily need to utilize all of the parameter space. However, in this case, considering we have very few parameters where each has a different level of contribution to the overall perceived quality, the lack of the audio packet loss rate and noise reduction parameters in the equation generated raises red flags. We have generated the models with the same configurations several times and we have experienced similar issues in all runs where each time some significant parameters were not included in the equation formed.

Table 7.4 – RMSE and Pearson correlation values for No-Reference parametric models.

Model Name	RMSE	Pearson Correlation
`RF1 Model`	0.340	0.930
`RF2 Model`	0.358	0.922
`BG1 Model`	0.345	0.928
`BG2 Model`	0.355	0.925
`DL1 Model`	0.403	0.909
`DL2 Model`	0.437	0.894
`GP1 Model`	0.449	0.881
`GP2 Model`	0.469	0.870

The GP1 model has suffered from similar issues. Out of all available parameters, the model was able to use only a small subset. In both cases, the equations generated were not only incorporating a different subset of parameters but also had significantly different forms and made it impossible for us to make any assumptions about the interrelations between the parameters. Even assuming they would have out-performed the other methods, then we would still have the problem of deciding which form of the equation and which parameter set to use. Overall, based on our experiments, it is difficult to conclude if the Genetic Programming based models always provide a good solution.

Contrary to Genetic Programming, Decision Trees based models tend to use all parameters as long as they are not limited by certain configurations such as tree depth. Deep Learning based models require feature engineering and then traditionally all of the remaining parameters are used during the training.

To reduce the variation in metrics we have trained and tested each model consecutively 10 times and have taken the average of the measured statistical metrics in these 10 runs for every reported performance indicators. Please note that before each run we have shuffled the data to avoid repetition. Figure 7.1 shows the box plot of RMSE and Pearson correlation for these models each run 10 times. Random Forest based models with Bagging based models achieved the highest accuracy in terms of Pearson correlation. The RF1 model not only achieves best results but also is more precise and has less variation in measured metrics in all runs.

Another effective way of visualizing the difference in performance is to look at the scatter plots of the models. Figure 7.2 shows the respective performance of the RF1, BG1, DL1 and the GP1 models. These figures show that when a model has higher Pearson correlation and lower RMSE value, the respective graph has a more compact shape with fewer outliers.

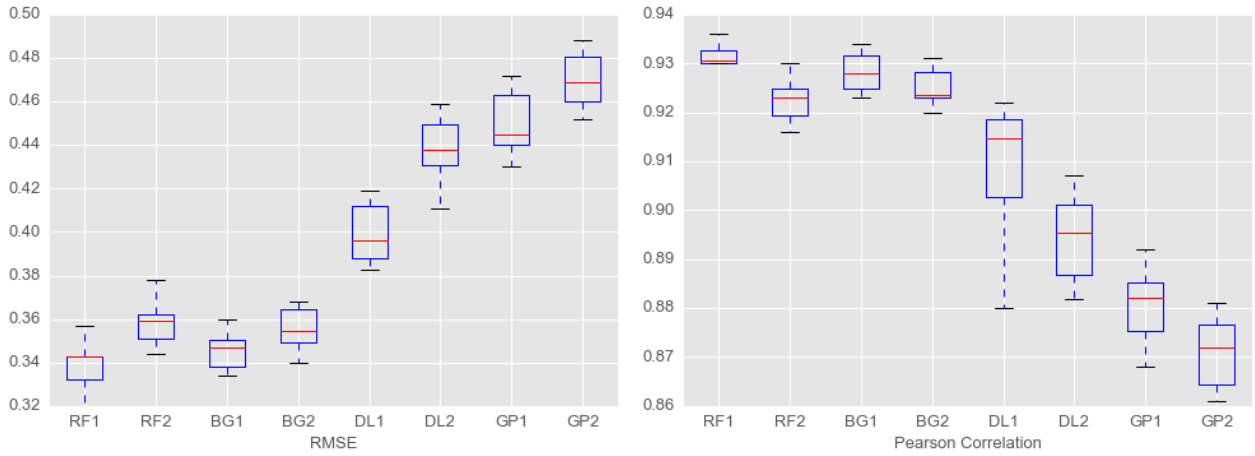

Figure 7.1 – BoxPlot of RMSE and Pearson correlation values for Random Forests, Bagging, Deep Learning and Genetic Programming based models.

In our experiments, we have witnessed the Decision Trees based models performing better in accuracy and precision while also requiring less effort to generate without any feature selection pre-process. Additionally, Random Forests have feature importance which helps us understand the model much better. Figure 7.3 shows the feature importance of the RF1 and the RF2 models. In both models, video packet loss influences the outcome more than other features. However, they differ from each other in the way the other features influence the model's behavior. In the RF2 model, we see video packet loss as the most important parameter followed by audio packet loss, quantization and video frame rate. Noise reduction seems to have the least influence on model behavior compared to other features. In the RF1 model, video packet loss rate is still the dominant feature. However, its influence seems to be more significant compared to rest of the features. Moreover, some additional variables influence the outcome more than other independent variables. This behavior is due to the correlation of features. Random Forests feature selection prefers variables with more classes. When one of the correlated features is used, the importance of the other correlated features is reduced (Strobl *et al.*, 2007).

7.3 Training and Testing Parametric Quality Models on Publicly Available Audiovisual Quality Datasets

The models we have built in this chapter are comparable to the ITU-T G.1071 (2015) and ITU-T G.1070 (2012) models in terms of the type of data they use. Running these two models on

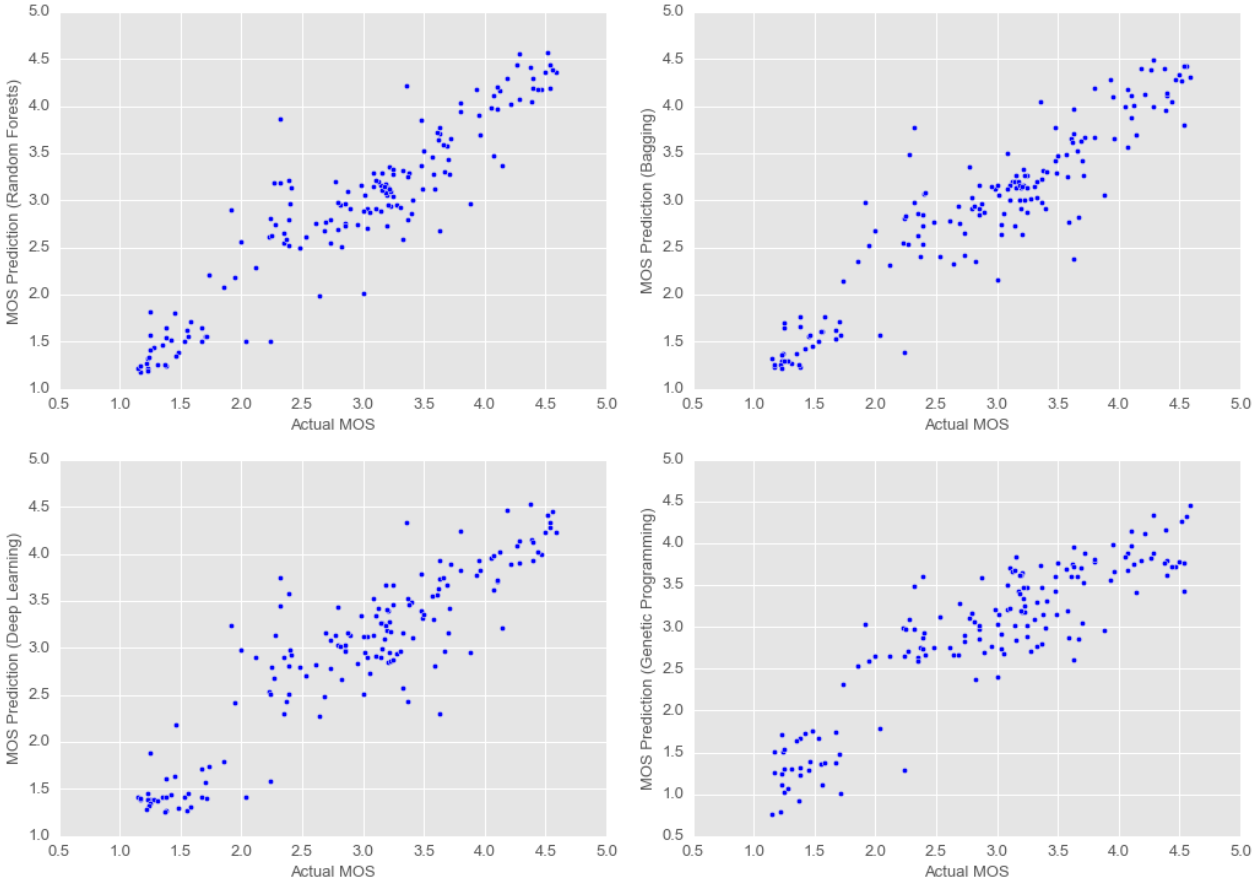

Figure 7.2 – MOS estimation vs actual values.

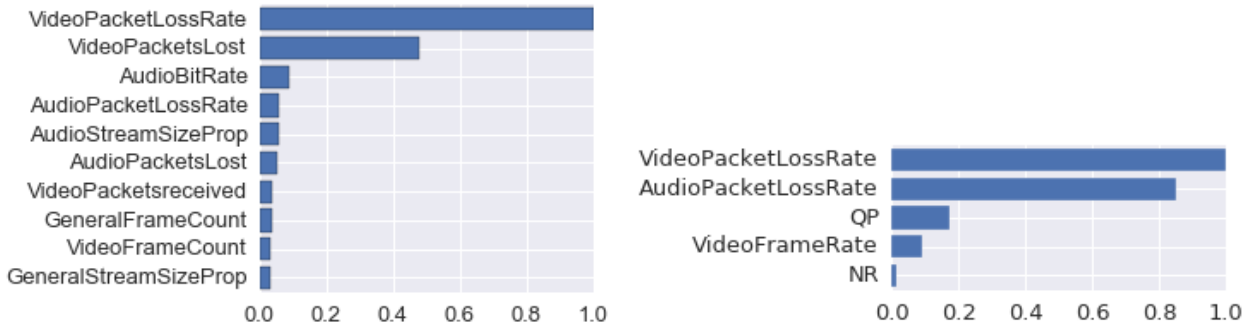

Figure 7.3 – Feature importance for the RF1 and RF2 models.

the parametric version of the INRS quality dataset was not possible as these models have limited coverage and do not support the AMR-WB audio codec combined with H.264 720p video resolution. The ITU-T G.1071 (2015) supports AMR codec only in low resolutions up to HVGA (480 x 320) resolution. The ITU-T G.1070 (2012) model provides model constants, m, only for 2.1 and 4.2-inch video display sizes and also does not provide speech coding-distortion and packet-loss-robustness constants for AMR codec. Comparison with the ITU-T P.1201 (2012) model is not given as it utilizes

bitstream level information and the models we have introduced in this chapter are neither designed to utilize such information nor does the INRS quality dataset, in its parametric version contain bitstream level information. Even if we provide that information, it still wouldn't be possible to run the ITU-T P.1201 (2012) model on our dataset as it does not support the AMR-WB codec.

We have witnessed the superior performance of the Decision Trees based ensemble models compared to the Deep Learning and Genetic Programming based models on the parametric version of the INRS audiovisual dataset. To put these results in perspective we have trained and tested the Random Forest based model on some publicly available audiovisual quality datasets (Table 6.1) and compared the results with the existing models, if any, that have been generated using these datasets. Note that some datasets listed in this section were not designed with real-time communications in mind.

Made for Mobile (Vienna) (Robitza *et al.*, 2012) has no subjective scores associated with it. The VTT dataset (Maki *et al.*, 2013) looks similar to the INRS dataset but only the scores with a subset of the independent variables are provided. The video files are not provided either. As a result, we did not run the models on this dataset since it would not be possible to make a reasonable comparison. It is also important to remember that we have trained and tested the Random Forests based models separately for each of the publicly available datasets. We did not attempt to generate one universal model that fits all cases.

In the rest of this section, we describe only the models built based on the publicly available audiovisual quality datasets. Previously, in Section 6.1, we have already described these datasets in detail.

7.3.1 University Of Plymouth Dataset (PLYM)

Goudarzi *et al.* (2010) have built No-Reference regression models to predict audiovisual quality from frame rate, packet loss rate using this dataset. They have also built Full-Reference models to estimate audiovisual quality by integrating the Full-Reference audio and video quality metrics such as PESQ and PSNR. They have obtained accuracy between 84% and 93% in terms of R-squared depending on weather application and network parameters are used or additional Full-Reference voice and video quality metrics are used as well. Konuk *et al.* (2015) have also used this

dataset to generate content aware audiovisual quality Reduced-Reference models using spatial and temporal characteristics of the videos and have achieved 0.78 and 0.81 accuracy in terms of Pearson correlations. Models created in both instances are based on the various form of the fusion equations we have presented above.

We have trained and tested the Random Forests based model on this dataset using the independent and other additional variables we have extracted from the files provided. A full list of the parameters we have used is given in Table B.1 in Appendix. We did not use Full-Reference metrics such as PESQ and PSNR or spatiotemporal characteristics and still have achieved high accuracy of 0.512 RMSE and 0.848 Pearson correlation.

7.3.2 TUM 1080p50 Dataset (TUM)

We have trained and tested the Random Forests based model on the TUM 1080p50 dataset and have obtained 0.439 RMSE and 0.956 Pearson correlation using independent variables already provided in the dataset and dependent variables that we have extracted from the files. This dataset also includes the PSNR Full-Reference metric. When we integrate PSNR values, the model's accuracy increases to 0.424 RMSE and 0.9714 Pearson correlation. The list of parameters we have used in this case are given in Table B.2 in Appendix.

7.3.3 VQEG Dataset (VQEG)

Pinson *et al.* (2013) have conducted subjective tests through the same audiovisual material at six different international laboratories. Overall ten different datasets each having 60 data points were produced. In order to measure the Random Forest based model performance, we have trained and tested the model on the global VQEG MOS dataset. The performance metrics we have obtained were 0.344 RMSE and 0.9159 Pearson correlation without using any Full-Reference or Reduced-Reference metrics. The full list of the parameters we have extracted is given in Table B.3 in Appendix.

7.4 Discussion

We have trained and tested machine learning algorithms on each dataset independently. The models generated are meant to be used in that specific use case only. However, with a dataset spanning several tests with different contents, machine learning based models can also be built for general purpose applications.

Based on our experiments, we believe that the Decision Trees based algorithms are well suited for structural data like the INRS audiovisual quality dataset and other publicly available quality datasets, and achieve superior performance compared to other algorithms. The deep learning and genetic programming based models are included as a point of comparison as they are widely used in many domains.

The standardized models for IPTV (ITU-T G.1071, 2015; ITU-T P.1201, 2012) and video telephony (ITU-T G.1070, 2012) aim to provide general purpose models for as many use cases as possible. They typically use a very small subset of the features that are available among many applications such as packet loss percentage, frame rate and compression rate, and content complexity for bitstream models. However, in this chapter, we have built audiovisual perceived quality estimation models using both the typical small subset of features used in standardized models and the correlated data that we have extracted from the datasets. Based on the results, we know that extracting additional correlated data from the dataset helps us to generate more accurate models when suitable machine learning algorithms are deployed. However, this correlated data depends on the audio and video codec used, the container format, the tools used to extract the features as well as any other features derived by additional computations. The type and the amount of correlated data also depend on what features are measurable within the network. For similar research, we recommend following the approach we have taken here rather than pinpointing specific parameters a priori.

The accuracy of the Decision Trees based models decrease slightly when only a limited number of features are available, but our tests have demonstrated that even in those conditions, Decision Trees based models out-perform other machine learning algorithms. In the Conclusion section, we will revisit this discussion and give a broader comparison of the machine learning algorithms that we have used in perceived quality modeling.

As we have seen in this chapter, parametric quality models predict the impact of encoding configurations and network impairments on multimedia quality. They typically use information extracted from packet headers and have no access to the packet payload data. These methods are well suited to the cases where the payload data is encrypted (Dubin *et al.*, 2016).

Knowing that the correlated data can help when deployed with the right algorithm, opens the door to many other possibilities. For instance, the models can easily be extended to consider the loss positions and durations in the sequences which would make them good candidates for quality monitoring tasks. In this chapter, we have not investigated that as the parametric version of the INRS audiovisual quality dataset does not include that bitstream information. Certain Machine Learning algorithms do not only work better with correlated data but also provide insights into the data itself and help us to understand and prioritize certain features which are quite valuable for improving the service itself.

7.5 Summary

We have generated the INRS audiovisual quality dataset made of subjective scores for the combination of video frame rate, video quantization, noise reduction parameters and network packet loss rate. The value range of these parameters was set with real-time communications in mind.

We have built No-Reference parametric perceived quality estimation models based on the Random Forests, Bagging, Deep Learning and Genetic Programming methods. We have primarily used the parametric version of the INRS audiovisual quality dataset which includes various media compression and network distortion degradations typically seen in real-time communications. All of the mentioned methods have achieved high accuracy in terms of RMSE and Pearson correlation.

Random Forests and Bagging based models show a small edge over Deep Learning with respect to the accuracy they provide on the parametric version of the INRS dataset we have used. Genetic Programming based models fell behind even though their accuracy is impressive as well. We have also obtained high accuracy on the other publicly available audiovisual quality datasets and the performance metrics are comparable to the existing models trained and tested on these datasets.

Table 7.5 – Comparing Random Forests, Bagging, Deep Learning and Genetic Programming based models.

Method	Learning Curve	Tuning	Execution Time	Model Readability	Feature Selection Required	Accuracy	Overall Perf.
Random Forests	Easy	Easy	Linear	Difficult	No	Very Good	#1
Bagging	Easy	Easy	Linear	Difficult	No	Very Good	#2
Deep Learning	Steep	Difficult	Exponential	Difficult	Yes	Very Good	#3
Genetic Prog.	Steep	Difficult	Exponential	Easy	No	Good	#4

Table 7.5 depicts the overall comparison of the Random Forests, Bagging, Deep Learning and Genetic Programming methods in the context of this research. Note though that this table should not be perceived as a universal guide as these methods would behave differently in different context.

Decision Trees based models have a relatively easy learning curve compared to other methods. These methods are easy to tune as the number of input parameters are very limited and the execution times increase only linearly with increasing tree size. Both Deep Learning and Genetic Programming require significant know-how to start with, have a large number of parameters that take significantly longer time to tune and their execution times tend to increase exponentially when the value of certain parameters increases. As a matter of fact, we have run out of memory during the Genetic Programming model training and had to limit the extent of some parameters such as *Population Size* and *Generations*.

In terms of feature engineering, based on this research, only Deep Learning methods require feature selection. Decision Trees and Genetic Programming based methods tend to benefit from the correlated data for the size of given parameter space and dataset.

All of the methods that we have assessed rated good to very good (ref. Table 7.4). Overall, in a context similar to this research, we believe it makes sense to try Decision Trees based ensemble methods first and then compare with Deep Learning with very few layers and optionally with Genetic Programming.

No-Reference parametric models are well suited for network planning tasks. However, in order to monitor transmission quality, we need models that use the information retrieved from the bit stream

level such as loss positions and durations in the sequence. The question is, would Decision Trees based ensemble models still demonstrate superior performance with the bitstream level models as well or would Deep Learning and Genetic Programming based models capture those complex interrelations better? We further research this issue in the next chapter.

Chapter 8

Bitstream Quality Models

In this chapter we utilize the bitstream level information such as loss positions and durations in the sequence and hope to create more accurate models. In order to realize that, we have used the INRS audiovisual quality dataset. As that parametric dataset did not contain bitstream information but provided both reference and transmitted videos, we have computed bitstream extensions to build Reduced-Reference bitstream models. We have then used this extended dataset for the Machine Learning-based model's training, validation, and test phases. Eventually, we have compared the results that we have achieved on the extended dataset using the bitstream models with the results reported by the parametric models that we have presented in Chapter 7.

The models we introduce here are bitstream since we have obtained the training and test data from the media frame level as well as the application and network layers, and Reduced-Reference as we compare the frames on the sending-end with the frames received on the receiving-end. We have built several bitstream models using the Decision Trees based ensemble methods, Genetic Programming, and Deep Learning methods in order to compare their respective performance with the parametric models proposed in Chapter 7. Out of all available ensemble methods, Random Forests and Bagging methods performed better compared to the parametric models. Random Forests based models have also provided feature importance information that is very valuable for further research. The performance of the Deep Learning methods and Genetic Programming based models decreased due to the significant increase in the number of parameters we have added in order to monitor the difference in the frames transmitted.

This chapter is partly based on the following content:

- Edip Demirbilek and Jean-Charles Grégoire. Machine learning based bitstream audiovisual quality prediction models for realtime communications. *IEEE International Conference on Multimedia and Expo, 2016 (Submitted).*
- Edip Demirbilek and Jean-Charles Grégoire. Perceived quality prediction models: Taking advantage of correlated data. *Springer Quality and User Experience Journal: Topical Collection on Managing QoE of Future Networks and Applications (Submitted).*

In the next section, we first mention the ITU-T P.1201 (2012) again since the approach we have taken here to compute some of the bitstream information is based on its model. However, we took a few additional steps when generating the Reduced-Reference content which will be introduced in section 8.2.

8.1 ITU-T Recommendation P.1201 Model

The ITU-T P.1201 (2012) model is intended for estimating the audiovisual quality of streaming services. It is a non-intrusive packet-header information based model for the service monitoring and benchmarking of UDP-based streaming. The model supports both lower resolution applications such as mobile TV and higher resolution applications such as IPTV. The model uses the information retrieved from the packet header as well as information provided out of the band. It provides separate predictions of audio, video, and audiovisual quality as output in terms of the five-point mean opinion score (MOS) scale. The model has been validated for compression, packet loss, re-buffering impairments of audio and video with different bitrates. Video content of different spatiotemporal complexity with different keyframes, frame rates, and video resolutions was selected. The ITU-T Rec. P.1201 model was tested over 1166 samples at lower resolutions and tested over 3190 samples at higher resolutions. RMSE and Pearson Correlation (Garcia, 2014) values for audiovisual modeling were evaluated as 0.470 and 0.852, respectively, for lower resolution applications and 0.435 and 0.911, respectively for higher resolution applications. Detailed performance figures are included in ITU-T P.1201 (2012).

The model recommended for higher resolution applications (ITU-T P.1201.2, 2012) is a combination of the quality impairment based models where the quality model is based on the audio and

video quality terms, and the impairment model is based on the audio and video impairment factor terms linked to the degree of compression and the transmission errors. The overall model takes the following mathematical form:

$$Q_{AV} = \omega_1.Q_{AV} + \omega_2.IF_{AV} \tag{8.1}$$

where ω_1 and ω_2 are constants, Q_{AV} denotes quality model and IF_{AV} denotes the impairement model.

Both quality and impairment models consist of complex mathematical forms that can be found in ITU-T P.1201.2 (2012). The impairment model requires the calculation of averaged number of bits per pixel (BitPerPixel) and scene complexity (SceneComp) parameters. The scene complexity parameter is calculated using video resolution, video frame rate, the number of scenes in the video sequence and the number of GOPs in the scene (ITU-T P.1201.2, 2012). The implementation of the calculation of these parameters is given in the partial implementation of the ITU-T P.1201.2 audiovisual quality estimation tool at (Deutsche Telekom AG : T-LABS, 2016). We have integrated the BitPerPixel and SceneComp parameter evaluations from this tool into the overall computation of the bitstream extensions to the INRS audiovisual quality dataset. These extensions are explained in detail in Section 8.2.

8.2 The INRS Audiovisual Quality Dataset: Bitstream Version

We have designed the INRS audiovisual quality dataset to span the most important compression and network distortion influence factors. These factors are typically video frame rate, video quantization, filters and network packet loss rate. The range of these parameters for the H.264 video encoding are (0, 0.1, 0.5, 1 and 5%) for network packet loss (PLR) rate in both video and audio streams, (10, 15, 20 and 25 fps) for video frame rate (FPS), (23, 27, 31 and 35) for quantization (Quant), and (0 and 999) for the noise reduction (NR) filter. These parameters are summarized in Table 8.1 as well.

From Chapter 4, Chapter 6 and Chapter 7 we know that the change in the network packet loss rate has dramatic effects on the perceived quality. Moreover, the video frame rate and the quantization parameter have also a moderate influence on the MOS value and the model behavior. However, in the current version of the INRS audiovisual quality dataset, the value of these parameters are

Table 8.1 – Media compression parameters and network impairments.

	Video	Audio
FPS	10, 15, 20, 25	Mono, 16kHz, 24 kbps
Quant	23, 27, 31, 35	Mono, 16kHz, 24 kbps
NR	0, 999	Mono, 16kHz, 24 kbps
PLR (%)	0, 0.1, 0.5, 1, 5	0, 0.1, 0.5, 1, 5

reported per file only. We expect that incorporating the influence of these parameters per video I and P frames, and per audio frames would improve the accuracy of the machine learning-based perceived quality prediction models. In the rest of this section, we discuss the extensions we have brought to the INRS audiovisual quality dataset in order to gather packet loss rate and bit rate changes on the individual or group of frames. In Section 8.3 we will discuss how these bitstream level extensions influence the model generated and if the assumptions we hold here turn out to be true.

In Chapter 4 and Chapter 7, have demonstrated that using dependent parameters improves the accuracy of the models. In light of these findings, we have used the FFmpeg multimedia system (Bellard, 2005) to obtain the bitrates for audio and video streams, a number of frames, and the duration of the streams.

Using the partial implementation of the ITU-T P.1201.2 Audiovisual quality estimation tool at Deutsche Telekom AG : T-LABS (2016), we have computed the BitPerPixel and SceneComp parameters per video files in the INRS audiovisual quality dataset.

The main work we have done here is, however, computing how packet loss rate and quantization parameter influence the size and the count of I and P frames. Initially, we have retrieved the count of the video I and P frames, audio frames and the loss percentage during the transmission for each of these. Additionally, we have reported the size and the loss percentage during the transmission of the video I frames individually as there is only one I frame in every 10 sec in the reference videos.

We have however taken a different approach for the video P frames and audio frames. Instead of reporting each individual frame, we have grouped these frames on per second basis and reported the loss percentage in the count and the average size of video P frames and the loss percentage in the count of audio frames. During our initial experiments with machine learning based modeling, we have realized that reporting values for the first three video I frame periods (i.e. the first 30

Table 8.2 – Bitstream dataset parameters.

Video Frame Rate
Video Noise Reduction
Video Quantization Parameter
Video Packet Loss Rate
Audio Packet Loss Rate
Video Start Time
Video Duration
Video Bit Rate
Video NB Frames
Audio Duration
Audio Duration
Audio Bit Rate
Audio NB Frames
Video Bits / (Pixel * Frame)
iFrames Per Scene
Content Complexity
iFrames Count
iFrame (n) Size, n = {0,1, ... ,5}
pFrames Count
aFrames Count
iFrames Count Difference (%)
iFrame (n) Size Difference (%), n = {0,1, ... ,5}
pFrames Count Difference (%)
pFrames at Second (n) Count Difference (%), n = {0,1, ... ,30}
pFrames at Second (n) Mean Size Difference (%), n = {0,1, ... ,30}
aFrames Count Diff (%)
aFrames at Second (n) Count Difference (%), n = {0,1, ... ,30}

seconds) is sufficient for improving model performance and the two remaining video I frame periods would just make the model unnecessarily complex without any significant improvement in model accuracy. The significance of the first 30 seconds is also in line with the analysis in Chapter 6 that the observers did not watch the videos to the end during the subjective assessment.

The complete list of the bitstream extensions to the INRS audiovisual quality dataset mentioned above are given in Table 8.2. The newly generated extended dataset contains independent parameters (5 features) from the parametric dataset and the bitstream extensions that we have developed in this chapter.

In Chapter 7, we have built Random Forests (RF), Bagging (BG), Genetic Programming and Deep Learning based perceived quality estimation models using the parametric version of the INRS

Table 8.3 – RMSE and Pearson correlation values for No-Reference parametric models.

Model Name	RMSE	Pearson Correlation
`RF1 Model`	0.340	0.930
`RF2 Model`	0.358	0.922
`BG1 Model`	0.345	0.928
`BG2 Model`	0.355	0.925
`DL1 Model`	0.403	0.909
`DL2 Model`	0.437	0.894
`GP1 Model`	0.449	0.881
`GP2 Model`	0.469	0.870

audiovisual quality dataset, which includes base independent variables (5 features) as depicted in Table 8.1. Packet loss percentage for audio and video streams are computed from the actual network measurements and are reported separately. Additionally, it includes dependent variables extracted from the transmitted videos by a third party software (29 more features). For each of the algorithms mentioned above we have built two models as follows:

- Two Random Forest based models: The RF1 model that used only the independent parameters as features (5 features) and the RF2 model that used all extracted parameters as features (34 features).
- Two Bagging based models (BG1 and BG2) following the same approach as Random Forests.
- Two Genetic Programming based models (GP1 and GP2) following the same approach as Random Forests.
- Two Deep Learning based models: The DL1 model that had only one hidden layer and the DL2 model that had three hidden layers.

For these models, we have obtained the RMSE and Pearson Correlation values, presented in Table 8.3.

All models performed overall well while Decision Trees based models outperformed both the Deep Learning and Genetic Programming models. In the following section, we will present the models we have built based on the extended dataset that contains independent parameters (5 features) from the parametric dataset and the bitstream extensions described in Section 8.2.

8.3 Reduced-Reference Bitstream Audiovisual Quality Prediction Models

We have built Reduced-Reference bitstream audiovisual quality prediction models based on the algorithms we have introduced in Chapter 7. With this approach we are hoping to find the answers to the following two questions: 1) Will adding the bitstream extensions to the INRS audiovisual quality dataset help to build more accurate perceived quality estimation models? 2) How will each individual algorithm perform compared to parametric models?

To answer these questions, we have built Reduced-Reference bitstream models using the extended dataset as follows. We have kept the software framework and the settings for each algorithm the same as in the parametric models.

8.3.1 Decision Trees Based Models

We have used the Python scikit-learn's implementation of Random Forests (RF) and Bagging (BG). We have generated Random Forest and Bagging models that used the extended dataset with 125 features. Please recall that the parametric version of the INRS audiovisual quality dataset included a total of 34 features. We expect some statistically significant changes in the model's performance with this drastic increase in the number of features. In both models, we have set the tree size to 200 and max_features to all when looking for the best split and no limits on the tree depth.

8.3.2 Deep Learning Based Models

We have generated the Deep Learning (DL) models using the Keras Deep Learning library that run on top of the Theano package. From Chapter 7, we already know that the Deep learning models perform best with the independent feature set only. Therefore we expect a decrease in the performance of the Deep Learning based models introduced here. We have kept the Keras API configurations exactly the same as in parametric models.

Table 8.4 – RMSE and Pearson Correlation values for Reduced-Reference bitstream models.

Model Name	RMSE	Pearson Correlation
`RF Model`	0.3082	0.9439
`BG Model`	0.3091	0.9424
`GP Model`	0.4885	0.8550
`DL Model`	0.6356	0.8042

8.3.3 Genetic Programming Based Models

We have used the Python gplearn library to implement genetic Programming (GP) based models. The gplearn library is compatible to scikit-learn fit/predict API and works with the existing scikit-learn pipeline. We have used the SymbolicRegressor with the configurations exactly the same was as in parametric models. In Chapter 7, Genetic Programming based models are reported to benefit from the correlated data. However, only the detailed tests would reveal if the significant increase in the number of features still improves the accuracy of these models.

8.3.4 Results

In contrast to the parametric models, not all tested algorithms performed well on the extended dataset. Decision Trees based models outperformed both the Deep Learning and Genetic Programming models by a great margin. For the Random Forests based model, we have obtained 0.3082 and 0.9439 in terms of RMSE and Pearson correlation values. These values were 0.3091 and 0.9424 for the Bagging based model. The Random Forests and Bagging based models performed very similarly while the Random Forests based model had a very small edge over the Bagging based model. Both models have also outperformed the parametric models (Table 8.3). From these results, we can say that the bitstream extensions have helped us to build better performing models.

Contrary to the Decision Trees based models, Deep Learning based models did not perform well and have achieved 0.6356 RMSE and 0.8042 Pearson correlation. This is in line with our expectations that Deep Learning based models do not perform well with an increased number of features and require prior feature engineering.

Last but not least, we have obtained the performance metrics for Genetic Programming as well. These values were 0.4885 RMSE and 0.8550 Pearson correlation. Symbolic Regression implemented

via Genetic Programming aims to identify both the parameters and the form of underlying mathematical expression simultaneously. These mathematical expressions depend on the training data selected and might have different forms in each run (Poli *et al.*, 2008). These results indicate that bitstream extensions did not help to achieve more accurate models given the same configurations.

Since we have shuffled the order of the rows in the dataset each time before running the models, the results that we have obtained had different values depending on the training and tests sets selected. To reduce the variation in metrics, we have run each model consecutively 10 times (with the entire dataset shuffled before each run) and have taken the average of the measured statistical metrics in these 10 runs for every reported performance indicators. Figure 8.1 shows the box plot of RMSE and Pearson correlation for these models each run 10 times. Random Forest based models with Bagging based models have achieved the highest accuracy in terms of Pearson correlation. These models do not only achieve best results but also are more precise and have less variation in measured metrics in all runs.

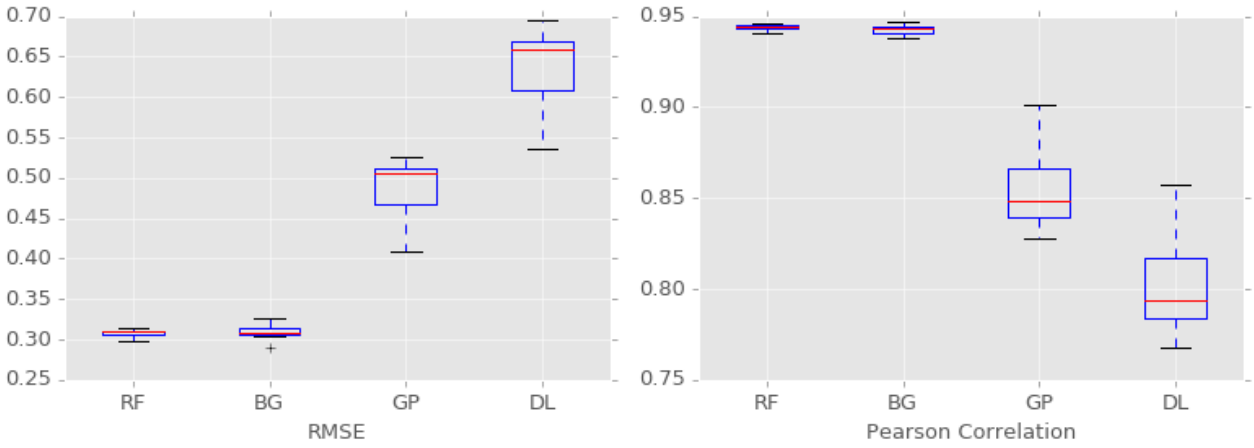

Figure 8.1 – BoxPlot of RMSE and Pearson correlation values for Random Forests, Bagging, Deep Learning and Genetic Programming based models.

Figure 8.2 shows the scatter plots of the Random Forests, Bagging, Genetic Programming and the Deep Learning based models. These figures show that when a model has higher Pearson correlation and lower RMSE value, the respective graph has a more compact shape with fewer outliers. The difference between the Decision Trees based models and other models is visually visible as well.

In our experiments, we have witnessed Decision Trees based models perform better in accuracy and precision and also requiring less effort to generate without any preliminary feature selection. Additionally, Random Forests algorithms provide feature importance ranking which helps us un-

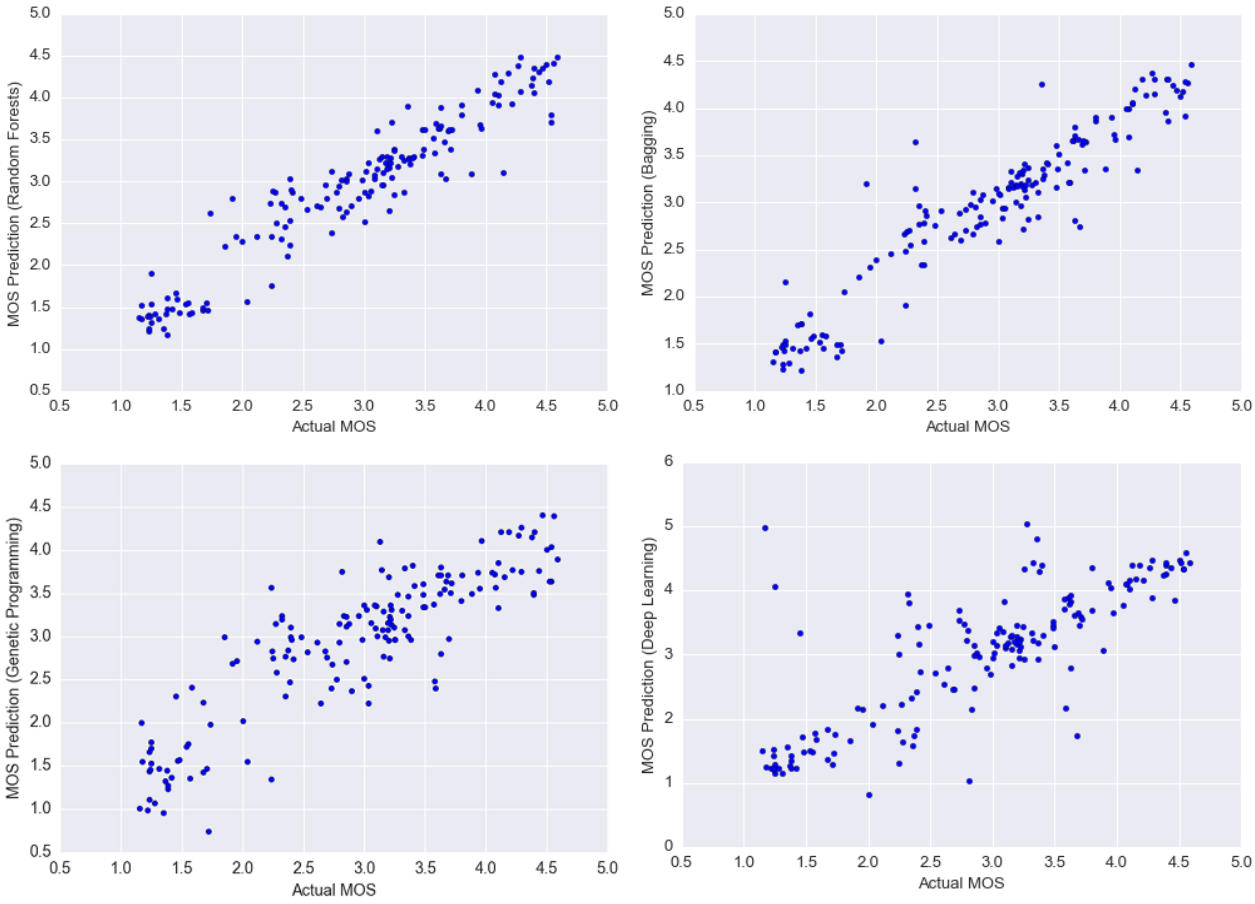

Figure 8.2 – MOS estimation vs actual values for Random Forests, Bagging, Genetic Programming and Deep Learning based models.

derstand the models much better. Figure 8.3 shows the 15 most influential features for two Random Forests based models. Please recall that the extended dataset contains 125 features. These figures belong to the models that trained on 70% and tested on remaining 30% of shuffled data. Every time we have built a Random Forests based model, we have witnessed some minor changes in the order of feature influences. Video Packet Loss rate and the loss percentage in the video P frame count are the most significant factors on the models generated. Audio and Video bit rate and loss percentage in the audio frame count also appear on both figures. Additionally, we observe that the loss percentage in the video P frame count reported per second has a noticeable influence on the model's performance. The order of seconds in the figures point another quite important finding: Almost all most influential one second periods come right after the video I frame transmission. This reflects the cumulative effect of packet loss and the importance of packet loss at the beginning of a period vs at the middle or at the end of periods between two video I frames.

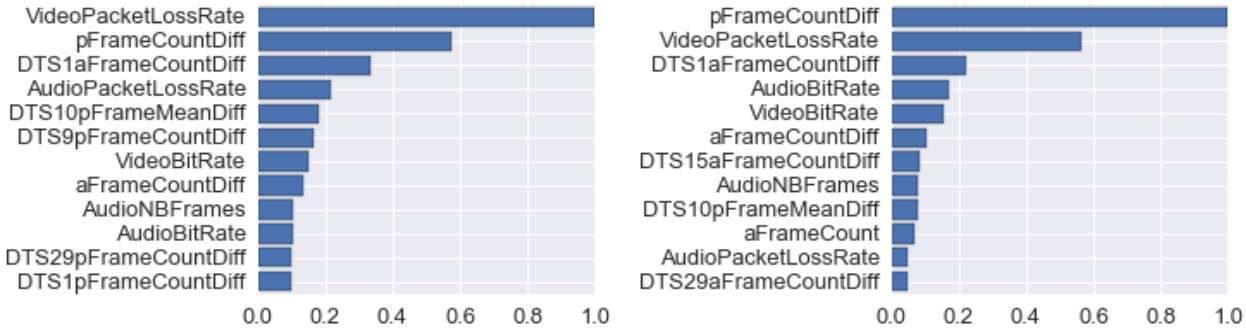

Figure 8.3 – Feature importance for the Random Forests based models.

Another important finding is the lack of certain parameters in the 15 most influential features lists. Quantization, video frame rate, and noise reduction have less influence on model behavior compared to other features. This behavior is due to the correlation of features: Random Forests feature selection prefers variables with more classes and when one of the correlated features is used, the importance of the other correlated features is reduced (Strobl *et al.*, 2007). We also observe the lack of the BitPerPixel and Scene-Complexity features that we have computed in Section 8.2. This behavior is easily explained by the fact that the INRS audiovisual quality dataset includes only one type of content.

8.4 Discussion

The standardized models for IPTV (ITU-T G.1071, 2015; ITU-T P.1201, 2012) and video telephony (ITU-T G.1070, 2012) aim to provide a model for as many use cases as possible. They typically use a very small subset of the features available such as packet loss percentage, frame rate and compression rate, and content complexity for bitstream models. However, in this research, we have generated audiovisual perceived quality estimation models using both the typical small subset of features used in standardized models and the correlated data that we have extracted from the videos included in the publicly available dataset. Based on the results that we have obtained in this chapter as well as results reported in Chapter 4 and Chapter 7, we know that extracting additional correlated data from the dataset helps us to generate more accurate models. However, this correlated data depends on various factors such as the audio and video codec used, the tool used to extract the correlated data and number of features. For audiovisual quality modeling that

takes advantage of the correlated data, we recommend following the approach we have taken here rather than pinpointing specific parameters.

Bit-stream based models exploit information from the elementary stream. These models typically process both the headers and the payload of the video bit stream. They process the bit stream header to extract transport-related information such as Transport Stream (TS) and/or Realtime Transport Protocol (RTP) time stamps and sequence numbers for packet loss detection. The increased complexity of the model, as in our case, has resulted in an improved accuracy of the model. However, these models are not suited to the cases where the payload data is encrypted.

8.5 Summary

Perceived quality prediction models for multimedia services vary greatly depending on the type of data and on the amount of information related to the original signal used. In this research, we have built machine learning-based Reduced-Reference bitstream audiovisual quality prediction models using the INRS audiovisual quality dataset. As that parametric version of the INRS audiovisual quality dataset did not contain bitstream information but provided both reference and transmitted videos, we have computed its bitstream extensions to build the Reduced-Reference bitstream models. We have compared the performance of the Decision Trees based ensemble methods, Genetic Programming and Deep Learning models on this extended dataset and have also compared these results with the results of the No-Reference models on the parametric dataset. Decision Trees based ensemble methods outperformed Deep Learning and Genetic Programming based models when Reduced-Reference bitstream data was used and outperformed all existing No-Reference parametric models that were trained and tested on the parametric dataset. Our studies show that Decision Trees based approaches are well suited for No-Reference parametric models as well as for Reduced-Reference bitstream models.

We have built Reduced-Reference bitstream perceived quality estimation models based on the Random Forests, Bagging, Deep Learning and Genetic Programming methods. We have added bitstream extensions to the INRS audiovisual quality dataset which includes various media compression and network distortion degradations typically seen in real-time communications.

Random Forests and Bagging based models have outperformed both the Deep Learning and Genetic Programming based models with respect to the accuracy they provide on the extended dataset that we have used. Further, these Decision Trees based models have performed better compared to the existing No-Reference parametric models that have been built using the parametric version of the INRS audiovisual quality dataset.

However, both the Genetic Programming and Deep Learning based models fell behind the parametric models due to a significant increase in the number of features in the extended dataset.

Overall we conclude that computing the bitstream information is worth the effort it takes to generate and helps to build more accurate models. However, it is useful only for the deployment of the right algorithms. Our studies have proved that Decision Trees based algorithms are well suited to the No-Reference parametric models as well as to the Reduced-Reference bitstream models.

Chapter 9

Conclusion and Future Research Directions

9.1 Conclusion

Our original ambition was to build quality estimation models to monitor and to maximize the overall perceived quality of an audiovisual real-time communication. The classic approach to audiovisual quality modeling is to develop functions to predict audio and video quality independently and then combine them using another function to predict the overall audiovisual perceived quality. The alternative way is to build models that predict the audiovisual quality directly without any intermediate functions. Machine learning based modeling approaches have been successfully applied to estimating perceived quality. In this research, we have taken the second approach and have built machine learning based models that predict the overall audiovisual quality in one single function that can be utilized for monitoring as well as improving system performance in a closed loop.

In order to fulfill our goals, we have initially created a VLC VOD based testbed and have generated an experimental dataset, and have built Random Forests and Neural Networks based machine learning models. The knowledge that we have gathered in this experimental phase has enabled us to create a more robust testbed based on the GStreamer multimedia framework that is capable of generating a dataset that reflects contemporary real-time configurations for video frame rate, video quantization, noise reduction parameters and network packet loss rate. Both the

dataset and the tools used to create the dataset have been made publicly available for research and development purposes.

Then we have used the INRS dataset–that includes both the audiovisual sequences and their corresponding quality rating–during the training, validation and test phases and have built several machine learning based perceived quality estimation models. The INRS audiovisual quality dataset has two variants; a parametric version where the training and test data is obtained from the application and network layers, and no information regarding the original signals is used, and a bitstream version where additional data from the bitstream layer as well as reduced amount of information from the original signal is used.

Having these datasets, we have built No-Reference parametric and Reduced-Reference bitstream perceived quality estimation models based on the Random Forests, Bagging, Deep Learning and Genetic Programming methods.

In parametric models, we have used the parametric version of the INRS dataset and all of the tested methods have achieved high accuracy in terms of RMSE and Pearson correlation. Random Forests and Bagging based models show a small edge over Deep Learning with respect to the accuracy they provide on the INRS dataset we have used. Genetic Programming based models fell behind even though their accuracy is impressive as well. We have also obtained high accuracy on the other publicly available audiovisual quality datasets and the performance metrics are comparable to the existing models trained and tested on these datasets.

Random Forests algorithms do not only work better with the correlated data but also provide insights into the data itself and help us to understand and prioritize certain features which are quite valuable for improving the service itself.

Knowing that the correlated data can help when deployed with the right algorithm, opens door to many other possibilities. For instance, the models can easily be extended to consider the loss positions and durations in the sequences which would make them good candidates for quality monitoring tasks. This potential has lead us to build bitstream models.

In bitstream models, we have used the bitstream version of the INRS dataset, and Random Forests and Bagging based models have outperformed both the Deep Learning and Genetic Programming based models with respect to the accuracy they have achieved. Further, these Decision

Trees based bitstream models have performed better compared to the parametric models. However, both the Genetic Programming and Deep Learning based bitstream models fell behind the parametric models due to a significant increase in the number of features in the bitstream dataset.

Overall we conclude that computing the bitstream information is worth the effort it takes to generate and helps to build more accurate models. However, it is useful only for the deployment of the right algorithms.

Based on the results, we know that extracting additional correlated data from the dataset helps us to generate more accurate models when suitable machine learning algorithms are deployed. However, this correlated data depends on the audio and video codec used, the container format, the tools used to extract the features as well as if any other features, such as bits-per-pixel, derived by additional computations. The type and amount of the correlated data also depend on what features are measurable within the network. For similar research, we recommend following the approach we have taken here rather than pinpointing specific parameters.

9.2 Limits of the Work

There are a number of points in our work which would need to be addressed to extend its scope.

We have compared the performance of the various machine learning algorithms using the RMSE and Pearson correlation coefficient. Testing the significance of two algorithms in terms of correlation (via the z-scores) and RMSE (via the epsilon-insensitive RMSE) provides additional information when comparing two models which have not been explored in this research.

Another improvement can be achieved by increasing the size of the dataset. Measuring the test error of a base learner on a limited size dataset via cross validation potentially suffers from overfitting. One solution to this problem is to use completely two different datasets during the training and test phases of model development.

Feature selection process can also be improved to obtain better models. Random Forests based models handle the feature selection automatically. However, increasing the number of features might not always be the most optimum solution even though they appear to benefit from that increase. From earlier chapters, we know that certain features dominate the overall model performance. It

is possible to create more compact models using a limited number of features and still achieve statistically similar results. Same can also be said for the size of the trees in the ensemble methods. With clever pruning techniques, trees having less depth might also achieve statistically similar results.

Additionally, even though we have explored the encoder configurations targeting the real time communication, the interactivity dimension of the real-time communication has not been explored in the research.

Finally, we have trained and tested machine learning algorithms on each dataset independently. The models generated are meant to be used in that specific use cases only. However, with a dataset spanning several tests with different content, codec, and test equipment, machine learning based models can also be built for general purpose applications.

9.3 Future Research Directions

Beyond addressing some of the limitations we have exposed, this work can be extended in many other ways. In addition to extending to various content, codec and test equipment, there are many other dimensions of audiovisual quality modeling that can be addressed which include, but are not limited to, selecting a higher packet loss rate for audio streams, synchronization between audio and video streams, rating scale and methodology followed.

Alternatively, the data can be collected from the target system either in a controlled environment or directly from end users via crowdsourcing. In that case, subjective assessment campaign would have to be redesigned from the beginning with different requirements for the number of observers, the methodology followed, data cleanup, etc.

Another possible extension to this research is looking beyond perceived quality and into Quality of Experience. For practical reasons in this research, we have limited ourself mostly to the system influence factors. Considering the context and human influence factors would change the requirements for the dataset, the methodology followed during the subjective assessment as well as the modeling approach taken. Having an open source audiovisual quality dataset that reflects those QoE influence factors would be extremely useful for the research community.

There are also emerging applications in multimedia systems that perceived quality modeling has not been explored much. Stereoscopic video and and virtual reality are great examples to these applications. For these applications, the number of features is much higher and the quality must be addressed with completely new types of models where the quality aspects include depth perception, the naturalness of a scene and visual discomfort (Raake *et al.*, 2011). Although different 2D objective quality metrics can be applied to the color and depth images, left and right views of a stereoscopic video, there is no existing objective metric merely for stereoscopic video quality evaluations (Hewage *et al.*, 2009).

References

Agarwal S, Sommers J & Barford P (2005). Scalable network path emulation. *Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2005. 13th IEEE International Symposium on*, IEEE, pages 219–228.

Agboma F & Liotta A (2008). QoE-aware QoS management. *Proceedings of the 6th international conference on advances in mobile computing and multimedia*, ACM, pages 111–116.

Agboma F & Liotta A (2012). Quality of experience management in mobile content delivery systems. *Telecommunication Systems*, 49(1):85–98.

Almesberger W (1999). Linux network traffic control–implementation overview. *5th Annual Linux Expo*, numéro LCA-CONF-1999-012, pages 153–164.

Alpaydin E (2014). *Introduction to machine learning*. MIT press.

Alreshoodi M & Woods J (2013). Survey on QoE-QoS correlation models for multimedia services. *arXiv preprint arXiv:1306.0221*.

Angrisani L, Capriglione D, Ferrigno L & Miele G (2013). Internet protocol packet delay variation measurements in communication networks: How to evaluate measurement uncertainty? *Measurement*, 46(7):2099–2109.

Aroussi S & Mellouk A (2014). Survey on machine learning-based qoe-qos correlation models. *Computing, Management and Telecommunications (ComManTel), 2014 International Conference on*, IEEE, pages 200–204.

Battisti F, Carli M & Paudyal P (2014). QoS to QoE mapping model for wired/wireless video communication. *Euro Med Telco Conference (EMTC), 2014*, IEEE, pages 1–6.

Beerends JG, Schmidmer C, Berger J, Obermann M, Ullmann R, Pomy J & Keyhl M (2013). Perceptual objective listening quality assessment (POLQA), the third generation ITU-T standard for end-to-end speech quality measurement part i—temporal alignment. *Journal of the Audio Engineering Society*, 61(6):366–384.

Bellard F (2005). *FFmpeg multimedia system*. https://www.ffmpeg.org/about.html. 2016-11-22. Online.

Belmudez B (2015). *Audiovisual Quality Assessment and Prediction for Videotelephony*. Springer.

Bengio Y, Goodfellow IJ & Courville A (2015). Deep learning. *An MIT Press book in preparation*.

Bennett KP & Campbell C (2000). Support vector machines: hype or hallelujah? *ACM SIGKDD Explorations Newsletter*, 2(2):1–13.

Bifet A, Holmes G, Kirkby R & Pfahringer B (2010). Moa: Massive online analysis. *Journal of Machine Learning Research*, 11(May):1601–1604.

Bordetsky A, Brown K & Christianson L (2001). A feedback control model for managing quality of service in multimedia communications. *Telecommunication Systems*, 17(3):349–371.

Breiman L (2001). Random forests. *Machine learning*, 45(1):5–32.

Calyam P, Chandrasekaran P, Trueb G, Howes N, Ramnath R, Yu D, Liu Y, Xiong L & Yang D (2012). Multi-resolution multimedia qoe models for IPTV applications. *International Journal of Digital Multimedia Broadcasting*, 2012.

Calyam P, Sridharan M, Mandrawa W & Schopis P (2004). Performance measurement and analysis of h. 323 traffic. *Passive and Active Network Measurement*, Springer, pages 137–146.

Candela Tech. (2015). *LANforge ICE Network Emulator*. http://www.candelatech.com/datasheet_ice.php. 2016-11-22. Online.

Carbone M & Rizzo L (2010). Dummynet revisited. *ACM SIGCOMM Computer Communication Review*, 40(2):12–20.

Carson M & Santay D (2003). NIST net: a linux-based network emulation tool. *ACM SIGCOMM Computer Communication Review*, 33(3):111–126.

Cermak G (2005). Packet loss, bandwidth and latency affect judged quality of videoconferencing. *Proceedings of the First International Workshop on Video Processing and Quality Metrics for Consumer Electronics*, pages 1–6.

Chang CC & Lin CJ (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27.

Chen Y, Wu K & Zhang Q (2014). From QoS to QoE: A survey and tutorial on state of art, evolution and future directions of video quality analysis.

Chevul S (2006). *On application-perceived Quality of Service in wireless networks*. Department of Telecommunication Systems, School of Engineering, Blekinge Institute of Technology.

Comrie AC (1997). Comparing neural networks and regression models for ozone forecasting. *Journal of the Air & Waste Management Association*, 47(6):653–663.

De Lattre A, Bilien J, Daoud A, Stenac C, Cellerier A & Saman JP (2002). *VideoLAN Streaming Howto*. http://www.videolan.org/doc/streaming-howto/en. 2016-11-22. Online.

Dehaene S (2003). The neural basis of the weber–fechner law: a logarithmic mental number line. *Trends in cognitive sciences*, 7(4):145–147.

Demirbilek E (2016a). *GStreamer Multimedia Quality Testbed*. https://github.com/edipdemirbilek/GStreamerMultimediaQualityTestbed. 2016-11-22. Online.

Demirbilek E (2016b). *The INRS Audiovisual Quality Dataset*. https://github.com/edipdemirbilek/TheINRSAudiovisualQualityDataset. 2016-11-22. Online.

Demirbilek E (2016c). *Subjective Assesment Video Player*. https://github.com/edipdemirbilek/SubjectiveAssesmentVideoPlayer. 2016-11-22. Online.

Demirbilek E (2016d). *VLC Multimedia Quality Testbed*. https://github.com/edipdemirbilek/VLCVODMultimediaTestbed. 2016-11-22. Online.

Deutsche Telekom AG: T-LABS (2016). *ITU-T P.1201.2 Audiovisual Quality Estimation Tool*. http://vqegstl.ugent.be/?q=P.1201.2. 2016-11-22. Online.

Dischinger M, Haeberlen A, Beschastnikh I, Gummadi KP & Saroiu S (2008). Satellitelab: adding heterogeneity to planetary-scale network testbeds. *ACM SIGCOMM Computer Communication Review*, 38(4):315–326.

Domingos P (2012). A few useful things to know about machine learning. *Communications of the ACM*, 55(10).

Du H, Guo C, Liu Y & Liu Y (2009). Research on relationship between qoe and qos based on bp neural network. *Network Infrastructure and Digital Content, 2009. IC-NIDC 2009. IEEE International Conference on*, IEEE, pages 312–315.

Dubin R, Dvir A, Pele O & Hadar O (2016). Real time video quality representation classification of encrypted http adaptive video streaming-the case of safari. *arXiv preprint arXiv:1602.00489*.

Emulators HPP (2007). Anue systems. *Inc.(Nov. 2007)*.

Fiedler M & Hoßfeld T (2010). Quality of experience-related differential equations and provisioning-delivery hysteresis. *21st ITC specialist seminar on multimedia applications-Traffic, performance and QoE, Miyazaki, Japan*.

Fiedler M, Hossfeld T & Tran-Gia P (2010). A generic quantitative relationship between quality of experience and quality of service. *Network, IEEE*, 24(2):36–41.

Fliegel K (2014). Qualinet multimedia databases v5. 5.

Garcia MN (2014). *Parametric Packet-based Audiovisual Quality Model for IPTV Services*. Springer.

Garcia MN, List P, Feiten B, Wüstenhagen U & Raake A (2016). Audio-video databases for h. 264-bitstream-based quality assessment of iptv services. *8th International Conference on Quality of Multimedia Experience (QoMEX 2016)*.

Gastaldo P, Zunino R & Redi J (2013). Supporting visual quality assessment with machine learning. *EURASIP Journal on Image and Video Processing*, 2013(1):1–15.

Gnome Developer (2016). *GLib 2.0 object model*. https://developer.gnome.org/gobject/stable/. 2016-11-22. Online.

Goudarzi M, Sun L & Ifeachor E (2010). Audiovisual quality estimation for video calls in wireless applications. *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, IEEE, pages 1–5.

Greengrass J, Evans J & Begen AC (2009). Not all packets are equal, part 2: The impact of network packet loss on video quality. *Internet Computing, IEEE*, 13(2):74–82.

GStreamer Team (2016a). *GStreamer: open source multimedia framework*. https://gstreamer.freedesktop.org/. 2016-11-22. Online.

GStreamer Team (2016b). *GStreamer Python Bindings Supplement*. https://gstreamer.freedesktop.org/modules/gst-python.html. 2016-11-22. Online.

Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P & Witten IH (2009). The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18.

Hansen JP, Hissam S *et al.* (2013). Assessing QoS trade-offs for real-time video. *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2013 IEEE 14th International Symposium and Workshops on a*, IEEE, pages 1–6.

Hemminger S *et al.* (2005). Network emulation with netem. *Linux conf au*, Citeseer, pages 18–23.

Hewage CT, Worrall ST, Dogan S, Villette S & Kondoz AM (2009). Quality evaluation of color plus depth map-based stereoscopic video. *IEEE Journal of Selected Topics in Signal Processing*, 3(2):304–318.

Holik F, Horalek J, Marik O, Neradova S & Zitta S (2014). The methodology of measuring throughput of a wireless network. *Computational Intelligence and Informatics (CINTI), 2014 IEEE 15th International Symposium on*, IEEE, pages 279–284.

Horvat G, Zagar D & Matic T (2013). Analysis of QoS parameters for multimedia streaming in wireless sensor networks. *ELMAR, 2013 55th International Symposium*, IEEE, pages 279–282.

Hoßfeld T, Hock D, Tran-Gia P, Tutschku K & Fiedler M (2008). Testing the IQX hypothesis for exponential interdependency between qos and qoe of voice codecs ilbc and g. 711. *Proceedings of the 18th ITC Specialist Seminar on Quality of Experience*, pages 105–114.

Hoßfeld T, Tran-Gia P & Fiedler M (2007). Quantification of quality of experience for edge-based applications. *Managing Traffic Performance in Converged Networks*, Springer, pages 361–373.

Huynh-Thu Q, Garcia MN, Speranza F, Corriveau P & Raake A (2011). Study of rating scales for subjective quality assessment of high-definition video. *IEEE Transactions on Broadcasting*, 57(1):1–14.

Issariyakul T & Hossain E (2011). *Introduction to network simulator NS2*. Springer Science & Business Media.

ITU-T G.107 (2003). ITU-T RECOMMENDATION G.107: The e model, a computational model for use in transmission planning.

ITU-T G.1070 (2012). ITU-T RECOMMENDATION G.1070: Opinion model for video-telephony applications.

ITU-T G.1071 (2015). ITU-T RECOMMENDATION G.1071: Opinion model for network planning of video and audio streaming applications.

ITU-T G.114 (2003). ITU-T RECOMMENDATION G.114: One-way transmission time.

ITU-T P.1201 (2012). ITU-T RECOMMENDATION P.1201: Parametric non-intrusive assessment of audiovisual media streaming quality.

ITU-T P.1201.1 (2012). ITU-T RECOMMENDATION P.1201.1: Parametric non-intrusive assessment of audiovisual media streaming quality - lower resolution application area.

ITU-T P.1201.2 (2012). ITU-T RECOMMENDATION P.1201.2: Parametric non-intrusive assessment of audiovisual media streaming quality - higher resolution application area.

ITU-T P.1401 (2012). ITU-T RECOMMENDATION P.1401: Methods, metrics and procedures for statistical evaluation, qualification and comparison of objective quality prediction models.

ITU-T P.910 (1999). ITU-T RECOMMENDATION P.910: Subjective video quality assessment methods for multimedia applications. *International Telecommunications Union, Geneva*.

ITU-T P.911 (2016). ITU-T RECOMMENDATION P.911: Methods for the subjective assessment of video quality, audio quality and audiovisual quality of internet video and distribution quality television in any environment. *International Telecommunications Union, Geneva*.

ITU-T P.913 (1998). ITU-T RECOMMENDATION P.913: Subjective audiovisual quality assessment methods for multimedia applications. *International Telecommunications Union, Geneva*.

ITU-T P.920 (1996). ITU-T RECOMMENDATION P.920: Interactive test methods for audiovisual communications. *International Telecommunications Union Radiocommunication Assembly*.

Ivov E (2013). Hangout-like video conferences with jitsi videobridge and xmpp.

Jain M & Dovrolis C (2002). Pathload: A measurement tool for end-to-end available bandwidth. *In Proceedings of Passive and Active Measurements (PAM) Workshop*, Citeseer.

Johnsson A, Melander B & Björkman M (2004). Diettopp: A first implementation and evaluation of a simplified bandwidth measurement method. *Second Swedish National Computer Networking Workshop*, Citeseer, volume 5.

Joskowicz J & Sotelo R (2013). Considerations on packet loss incidence on the perceived video quality in digital television. *Workshop on Communications-IEEE LATINCOM*, 1 pages.

Keimel C, Redl A & Diepold K (2012). The tum high definition video datasets. *Quality of Multimedia Experience (QoMEX), 2012 Fourth International Workshop on*, IEEE, pages 97–102.

Khorsandroo S & Noor RM (2012). A generic quantitative relationship between quality of experience and packet loss in video streaming services. *Ubiquitous and Future Networks (ICUFN), 2012 Fourth International Conference on*, IEEE, pages 352–356.

Khorsandroo S, Noor RM & Khorsandroo S (2012). The role of psychophysics laws in quality of experience assessment: a video streaming case study. *Proceedings of the International Conference on Advances in Computing, Communications and Informatics*, ACM, pages 446–452.

Khorsandroo S, Noor RM & Khorsandroo S (2013). A generic quantitative relationship to assess interdependency of QoE and QoS. *KSII Transactions on Internet and Information Systems (TIIS)*, 7(2):327–346.

Klaue J, Rathke B & Wolisz A (2003). Evalvid–a framework for video transmission and quality evaluation. *International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, Springer, pages 255–272.

Konuk B, Zerman E, Akar GB & Nur G (2015). Content aware audiovisual quality assessment. *Signal Processing and Communications Applications Conference (SIU), 2015 23th*, IEEE, pages 966–969.

Koza JR (1992). *Genetic programming: on the programming of computers by means of natural selection*. volume 1. MIT press.

Kuipers F, Kooij R, De Vleeschauwer D & Brunnström K (2010). Techniques for measuring quality of experience. *Wired/wireless internet communications*, Springer, pages 216–227.

Le Callet P, Möller S, Perkis A *et al.* (2012). Qualinet white paper on definitions of quality of experience. *European Network on Quality of Experience in Multimedia Systems and Services (COST Action IC 1003)*.

Li W, Wang J, Xing C, Fei Z & Kuang J (2014). A real-time QoE methodology for AMR codec voice in mobile network. *Science China Information Sciences*, 57(4):1–13.

Liang YJ, Apostolopoulos JG & Girod B (2008). Analysis of packet loss for compressed video: Effect of burst losses and correlation between error frames. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(7):861–874.

Liaw A & Wiener M (2002). Classification and regression by randomforest. *R news*, 2(3).

Machado VA, Carlos N, Silva RSO, Melo AM, Silva M, Francês CR, Costa JC, Vijaykumar NL & Hirata CM (2011). *A new proposal to provide estimation of QoS and QoE over WiMAX networks*.

Maia OB, Yehia HC & de Errico L (2014). A concise review of the quality of experience assessment for video streaming. *Computer Communications*.

Maki T, Kukolj D, Dordevic D & Varela M (2013). A reduced-reference parametric model for audiovisual quality of iptv services. *Quality of Multimedia Experience (QoMEX), 2013 Fifth International Workshop on*, IEEE, pages 6–11.

Martinez J (2010). *MediaInfo*.

McCanne S, Floyd S, Fall K, Varadhan K *et al.* (1997). *Network simulator ns-2*.

Menkovski V, Exarchakos G & Liotta A (2010a). Online learning for quality of experience management. *Annual Machine Learning Conference of Belgium and The Netherlands*, 6 pages.

Menkovski V, Exarchakos G & Liotta A (2010b). Online qoe prediction. *Quality of Multimedia Experience (QoMEX), 2010 Second International Workshop on*, IEEE, pages 118–123.

Menkovski V, Oredope A, Liotta A & Sánchez AC (2009). Predicting quality of experience in multimedia streaming. *Proceedings of the 7th International Conference on Advances in Mobile Computing and Multimedia*, ACM, pages 52–59.

Menkvoski V, Oredope A, Liotta A & Sánchez AC (2009). Optimized online learning for qoe prediction. *Proc. of the 21st Benelux conference on artificial intelligence*.

Meyer D & Wien FT (2014). Support vector machines. *The Interface to libsvm in package e1071*.

Mitchell TM (2015). Machine learning. 2014.

Mitra K, Zaslavsky A & Åhlund C (2014). QoE modelling, measurement and prediction: A review. *arXiv preprint arXiv:1410.6952*.

Mushtaq MS, Augustin B & Mellouk A (2012). Empirical study based on machine learning approach to assess the qos/qoe correlation. *Networks and Optical Communications (NOC), 2012 17th European Conference on*, IEEE, pages 1–7.

Natrella M (2010). NIST/SEMATECH e-handbook of statistical methods.

Nussbaum L & Richard O (2009). A comparative study of network link emulators. *Proceedings of the 2009 Spring Simulation Multiconference*, Society for Computer Simulation International, 85 pages.

Oza NC (2005). Online bagging and boosting. *Systems, man and cybernetics, 2005 IEEE international conference on*, IEEE, volume 3, pages 2340–2345.

Paudyal P, Battisti F & Carli M (2014). A study on the effects of quality of service parameters on perceived video quality. *Fifth European workshop on visual information processing (EUVIP)*.

Perkis A (2016). Electronic imaging & signal processing quality of experience (QoE) in multimedia applications.

Pfahringer B, Holmes G & Kirkby R (2007). New options for hoeffding trees. *AI 2007: Advances in Artificial Intelligence*, Springer, pages 90–99.

Pinson M (2013). The consumer digital video library [best of the web]. *Signal Processing Magazine, IEEE*, 30(4):172–174.

Pinson M, Janowski L, Pépion R, Huynh-Thu Q, Schmidmer C, Corriveau P, Younkin A, Callet PL, Barkowsky M & Ingram W (2012). The influence of subjects and environment on audiovisual subjective tests: An international study. *Selected Topics in Signal Processing, IEEE Journal of*, 6(6):640–651.

Pinson M, Schmidmer C, Janowski L, Pépion R, Huynh-Thu Q, Corriveau P, Younkin A, Le Callet P, Barkowsky M & Ingram W (2013). Subjective and objective evaluation of an audiovisual subjective dataset for research and development. *Quality of Multimedia Experience (QoMEX), 2013 Fifth International Workshop on*.

Poli R, Langdon WB, McPhee NF & Koza JR (2008). *A field guide to genetic programming*. Lulu.com.

Raake A, Gustafsson J, Argyropoulos S, Garcia M, Lindegren D, Heikkila G, Pettersson M, List P, Feiten B *et al.* (2011). Ip-based mobile and fixed network audiovisual media services. *Signal Processing Magazine, IEEE*, 28(6):68–79.

Reichl P, Egger S, Schatz R & D'Alconzo A (2010a). The logarithmic nature of QoE and the role of the weber-fechner law in qoe assessment. *Communications (ICC), 2010 IEEE International Conference on*, IEEE, pages 1–5.

Reichl P, Tuffin B & Schatz R (2010b). Economics of logarithmic quality-of-experience in communication networks. *Telecommunications Internet and Media Techno Economics (CTTE), 2010 9th Conference on*, IEEE, pages 1–8.

Rifai H, Mohammed S & Mellouk A (2011). A brief synthesis of QoS-QoE methodologies. *Programming and Systems (ISPS), 2011 10th International Symposium on*, IEEE, pages 32–38.

Riley GF & Henderson TR (2010). The ns-3 network simulator. *Modeling and Tools for Network Simulation*, Springer, pages 15–34.

Rish I (2001). An empirical study of the naive bayes classifier. *IJCAI 2001 workshop on empirical methods in artificial intelligence*, IBM New York, volume 3, pages 41–46.

Rizzo L (1997). Dummynet: a simple approach to the evaluation of network protocols. *ACM SIGCOMM Computer Communication Review*, 27(1):31–41.

Robitza W, Pitrey Y, Nezveda M, Buchinger S & Hlavacs H (2012). Made for mobile: a video database designed for mobile television. *Sixth International Workshop on Video Processing and Quality Metrics for Consumer Electronics (VPQM)*.

Santos CEM, Ribeiro EP & Pedroso CM (2014). The application of neural networks to improve the quality of experience of video transmission over ip networks. *Engineering Applications of Artificial Intelligence*, 27:137–147.

Schmidt M & Lipson H (2010). Symbolic regression of implicit equations. *Genetic Programming Theory and Practice VII*, Springer, pages 73–85.

Schmitt M, Gunkel S & Cesar P (2013). A quality of experience testbed for video-mediated group communication. *Multimedia (ISM), 2013 IEEE International Symposium on*, IEEE, pages 514–515.

Senturk MB (2014). *A Computational Framework for Quality of Service Measurement, Visualization and Prediction in Mission Critical Communication Networks*. Thèse de doctorat, Arizona State University.

Shah JL & Parvez J (2014). Evaluation of queuing algorithms on QoS sensitive applications in ipv6 network. *Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on*, IEEE, pages 106–111.

Singh HP, Singh S, Singh J & Khan S (2014). VoIP: State of art for global connectivity—a critical review. *Journal of Network and Computer Applications*, 37:365–379.

Singh R & Aggarwal N (2014). State of the art and research issues in video quality assessment. *Engineering and Computational Sciences (RAECS), 2014 Recent Advances in*, IEEE, pages 1–6.

Song W & Tjondronegoro DW (2014). Acceptability-based qoe models for mobile video. *Multimedia, IEEE Transactions on*, 16(3):738–750.

Strobl C, Boulesteix AL, Zeileis A & Hothorn T (2007). Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC bioinformatics*, 8(1):1.

Vakili A & Grégoire JC (2013). QoE management for video conferencing applications. *Computer Networks*, 57(7):1726–1738.

VLC Team (2016a). *LibVLC API*. https://www.videolan.org/developers/vlc/doc/doxygen/html/group___libvlc.html. 2016-11-22. Online.

VLC Team (2016b). *VideoLan*. https://www.videolan.org. 2016-11-22. Online.

VLC Team (2016c). *VLC Python Bindings*. https://wiki.videolan.org/python_bindings. 2016-11-22. Online.

Wang J, Zhou M & Li Y (2004). Survey on the end-to-end internet delay measurements. *High Speed Networks and Multimedia Communications*, Springer, pages 155–166.

Witten IH & Frank E (2005). *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

Wu X, Kumar V, Quinlan JR, Ghosh J, Yang Q, Motoda H, McLachlan GJ, Ng A, Liu B, Philip SY *et al.* (2008). Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1):1–37.

Yeom I & Reddy AN (2001). ENDE: An end-to-end network delay emulator tool for multimedia protocol development. *Multimedia Tools and Applications*, 14(3):269–296.

You J, Reiter U, Hannuksela MM, Gabbouj M & Perkis A (2010). Perceptual-based quality assessment for audio–visual services: A survey. *Signal Processing: Image Communication*, 25(7):482–501.

Yusuf BR & Reddy PC (2012). Mining data streams using option trees. *International Journal of Computer Network and Information Security (IJCNIS)*, 4(8):49.

Zekauskas MJ, Kalidindi S & Almes G (1999a). A one-way delay metric for IPPM.

Zekauskas MJ, Kalidindi S & Almes G (1999b). A round-trip delay metric for IPPM.

Zhao P, Yang X, Yu W, Dong C, Yang S & Bhattarai S (2014). Toward efficient estimation of available bandwidth for IEEE 802.11-based wireless networks. *Journal of Network and Computer Applications*, 40:116–125.

Appendix A

Multimedia Communication Testbeds

1.1 VLC VOD Based Multimedia Communication Quality Assessment Testbed

We have built a dedicated test setup to conduct the streaming tests. To do so, we have used two dedicated workstations running the Ubuntu OS with the VideoLAN software solution. To manage network traffic effectively we have used both DummyNet and TC/Netem network emulation solutions. DummeyNet is used to manage the available bandwidth while TC/Netem is used to control jitter and delay. Predefined source video files are streamed from the VLC VoD server towards VLC Client and saved on the client local disc. Each file is saved with a file name that includes network configurations set for that specific streaming case. Figure A.1 depicts the high-level design of this test setup.

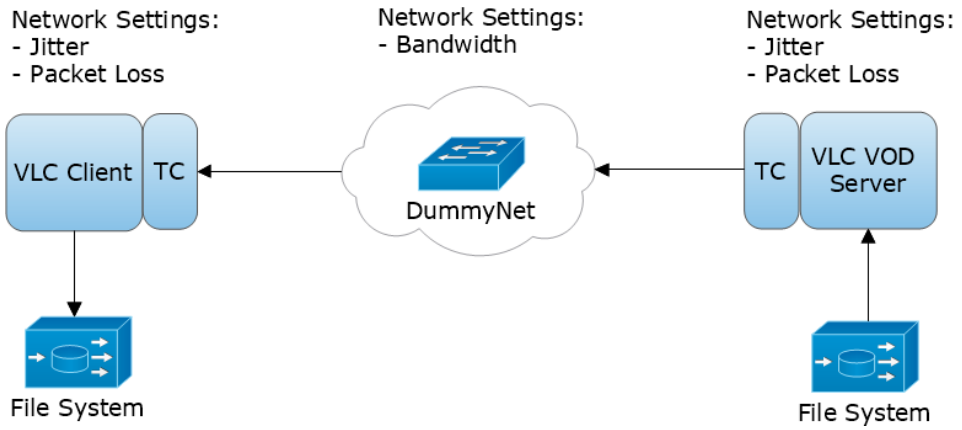

Figure A.1 – VLC VOD based multimedia communication quality assessment testbed.

1.1.1 Configuring the Workstations

Each workstation runs Ubuntu kernel v3.16.0-38 and has two network interfaces (eth0 and eth1). The eth0 interface is configured to provide access through ssh while the eth1 interface is configured for streaming purposes only.

1.1.2 Configuring DummyNet

A custom hardware running FreeBSD 9.1-RELEASE is configured to run DummyNet with bridge0 interface enabling traffic between workstations that are physically connected to the two available ports (vr1, vr2). Additionally, vr0 interface is used for remote terminal connections over ssh. The vr1 and vr2 interfaces operate in full-duplex mode and enable the traffic between two workstations.

Every time a system reboot occurs, the bridge interface has to be enabled with the following command;

```
% sysctl net.link.bridge.ipfw=1
```

and then the bridge interface need to be created as follows;

```
% ifconfig bridge create
% ifconfig bridge0 addm vr1 addm vr2 up
% ifconfig vr1 up
% ifconfig vr2 up
% ifconfig bridge0 up
```

where
- the first line creates the bridge interface.
- in the second line the vr1 and vr2 interfaces are added to the bridge.
- in line 3,4 and 5 the vr1, vr2 and bridge0 interfaces are set to operate.

In order to set the bandwidth limits, the following commands are executed;

```
% ipfw -f flush
% ipfw add 3000 pipe 1 ip from any to any
% ipfw pipe 1 config bw $BWKbit/s
```

where $BW denotes the desired bandwidth setting in Kbits.

1.1.3 Configuring TC

TC is used to configure the delay, jitter and packet loss behavior of the network. During our initial tests to find out the target network test settings, we observed that the delay parameter can be avoided due to the nature of our streaming tests. However, this parameter will become crucial in real-time communications.

The following commands configure the delay, jitter and packet loss rate on the eth1 interface:

```
% /sbin/tc qdisc del dev eth1 root
% /sbin/tc qdisc add dev eth1 root handle 1:1 netem delay $DELAY $JITTER
% /sbin/tc qdisc add dev eth1 parent 1:1 handle 10:1 netem loss $PLR
```

where the first line makes sure we start from a clean slate; in the second line delay and jitter parameters are set in terms of milliseconds, and in the last line packet loss rate is set in terms of percentage. The full range of parameters with the proper syntax to be run on both workstations is given in the Demirbilek (2016d).

1.1.4 Configuring The VLC VoD Server

Configuring the VLC VoD server consists of two steps. In the first step VLC system is configured to run with a telnet interface:

```
% cvlc -I telnet
    --telnet-password videolan
    --rtsp-host 0.0.0.0
    --rtsp-port 5554
```

where "cvlc" executable is used to control VLC command-line instances.

In the second step, VoD objects need to be created. In order to avoid any transcoding related issues on the server side, we have pre-encoded the source files. To do so, we have defined two resolution levels: 720p and 1080p, and 3 bitrate quality levels (High Quality, Middle Quality, and Low Quality) for each resolution. In the next section, where we discuss target network settings, we discuss the details of these pre-encoded video files. Since we had 6 source files to be streamed, we needed to create 6 different objects with the following syntax.

```
% telnet localhost 4212 videolan
```

and then

```
% new ts_id vod
% setup ts_id input ``filename.ts''
% setup ts_id enabled
```

where for each source video file, first a VoD object is created and then a video file is assigned to that object. In the last line, the video object is primed for streaming.

These steps could be automated by creating a vlm.conf file that includes the set of commands for creating 6 VoD objects; then by simply running the following command, manual telnet interaction would be avoided:

```
% vlc -I telnet
    --telnet-password videolan
    --vlm-conf vlm.conf
    --rtsp-host 0.0.0.0
    --rtsp-port 5554
```

1.1.5 Configuring the VLC Client and Streaming

In order to start streaming, no configurations are needed on the VLC Client side. The command for streaming and saving the file with the given name is as follows:

```
% cvlc -v rtsp://$IP:$PORT/$VoD
    --sout-mux-caching=$MC
    --file-caching=$FC
    --rtsp-frame-buffer-size=$FBS
    --sout=``\#std{access=file,dst=$FILENAME.ts}''
% vlc://quit
```

where
- $IP and $PORT define the IP and port number of the VLC VoD Server.
- $VoD defines the VoD object identifier created on the server. This has to be the same identifier as the one set on the server side.
- $MC, $FC and $FBS denotes Mux Caching, File Caching, and Frame Buffer Size Caching respectively.
- $FILENAME denotes the file name. During the test, we actually used more complicated post fixes to be able to identify source file and network parameters. Please refer to the source code in Demirbilek (2016d).

1.1.6 Shortcomings of VLC Multimedia Framework

The VLC VoD is a decent off-the-shelf product for simple tests. However, it falls short of expectations when it is used in more advanced test cases. Foremost, it has a lack of support for a variety of video and audio codecs. When network impairments such as packet loss are introduced, it fails to capture entire video stream with only a minor increase in packet loss rate. It certainly did not support real-world use cases where up to 5% video packet loss rate is expected. As it is off-the-shelf, changing the pipeline behavior is next to impossible and requires source code change. It also does not provide stream level network measurements such as packet loss rate, jitter, delay, effective bit rate etc.

1.2 GStreamer Based Multimedia Communication Quality Assessment Testbed

Due to the limitations of the VLC VoD based testbed that we have mentioned in Section 1.1.6, we have implemented a second testbed using the GStreamer multimedia framework for media streaming. To introduce network impairments, we have utilized only the Netem/TC tool to apply packet loss rate to multimedia traffic. Jitter and additional bandwidth limitations were not introduced. As a result, the DummyNet tool was not needed anymore. Figure A.2 depicts the high-level design of this test setup.

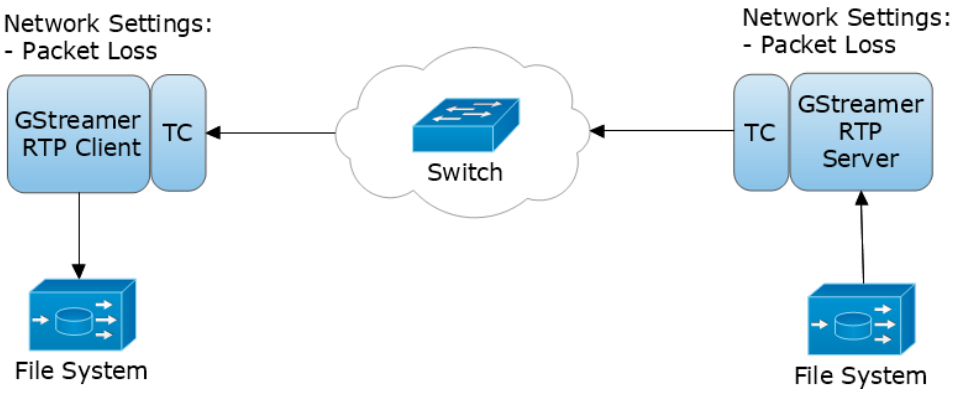

Figure A.2 – GStreamer based multimedia communication quality assessment testbed.

1.2.1 Configuring the Workstations

Workstation configurations are kept same as the VLC based test configurations. Each workstation runs Ubuntu kernel v3.16.0-38 with two network interfaces (eth0 and eth1). The eth0 interface is configured to provide remote access through ssh interface while the eth1 interface is configured for streaming purposes only.

1.2.2 Configuring the TC

TC is used to introduce packet loss impairments onto the media streams.

The Following set of commands are used to configure packet loss rate on the eth1 interface:

```
% /sbin/tc qdisc del dev eth1 root
% /sbin/tc qdisc add dev eth1 root handle 1:1 netem delay 0ms 0ms
% /sbin/tc qdisc add dev eth1 parent 1:1 handle 10:1 netem loss $PLR
```

where the first line makes sure we start from a clean slate. The delay and jitter parameters are set to 0 ms as they were not required anymore. Packet loss rate is set in terms of percentage. The full range of parameters with the proper syntax to be run on both workstations is given in reference Demirbilek (2016a).

1.2.3 GStreamer RTP Client and Server Pipelines

In order to address the limitations of the VLC, we have developed custom RTP server and client pipelines using the GStreamer multimedia framework.

RTP Server pipeline and elements are as follows:

```
% gst-launch-1.0 -ve rtpbin name=rtpbin
    filesrc location=$AV_FILE ! queue ! qtdemux name=dem
    dem. ! queue ! rtph264pay ! rtpbin.send_rtp_sink_0
        rtpbin.send_rtp_src_0 ! udpsink host=$DEST port=5000
```

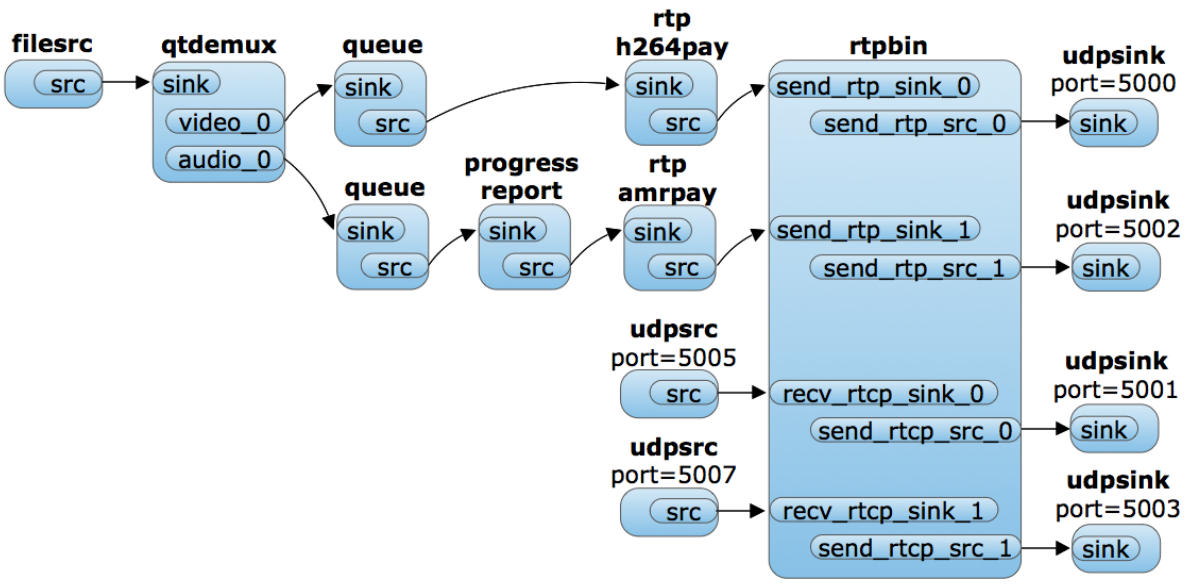

Figure A.3 – GStreamer RTP server pipeline.

```
        rtpbin.send_rtcp_src_0 ! udpsink host=$DEST port=5001
            sync=false async=false
        udpsrc address=$SRC  port=5005 ! rtpbin.recv_rtcp_sink_0

    dem. ! queue ! rtpamrpay ! rtpbin.send_rtp_sink_1
        rtpbin.send_rtp_src_1 ! udpsink host=$DEST port=5002
        rtpbin.send_rtcp_src_1 ! udpsink host=$DEST port=5003
            sync=false async=false
        udpsrc address=$SRC port=5007 ! rtpbin.recv_rtcp_sink_1
```

where the RTP server creates two sessions and streams audio on one, video on the other, with RTCP on both sessions. The video is sent on port 5000, with its RTCP stream sent on port 5001 and received on port 5005. Audio is sent on port 5002, with its RTCP stream sent on port 5003 and received on port 5007.

RTP client pipeline and elements are as follows:

```
% VIDEO_CAPS=``application/x-rtp, media=(string)video,
    clock-rate=(int)90000, encoding-name=(string)H264''
% AUDIO_CAPS=``application/x-rtp, media=(string)audio,
    clock-rate=(int)16000, encoding-name=(string)AMR-WB,
    encoding-params=(string)1, octet-align=(string)1''

% gst-launch-1.0 -ve rtpbin name=rtpbin latency=$LATENCY
    udpsrc caps=$VIDEO_CAPS address=$SRC port=5000 !
    rtpbin.recv_rtp_sink_0 rtpbin. ! rtph264depay ! queue !
    h264parse ! queue ! qtmux name=mux !
    filesink location=$AV_FILE udpsrc address=$SRC port=5001 !
    rtpbin.recv_rtcp_sink_0 rtpbin.send_rtcp_src_0 !
    udpsink host=$DEST port=5005 sync=false async=false
```

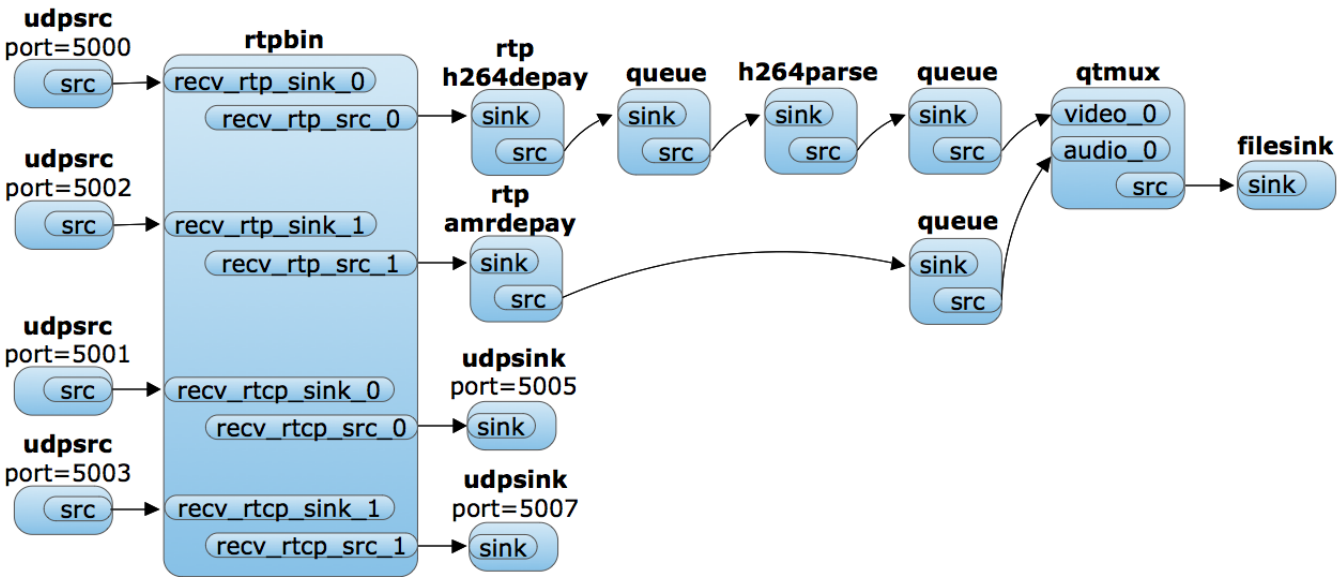

Figure A.4 – GStreamer RTP client pipeline.

```
udpsrc caps=$AUDIO_CAPS address=$SRC port=5002 !
rtpbin.recv_rtp_sink_1 rtpbin. ! rtpamrdepay ! queue ! mux.
udpsrc address=$SRC port=5003 !
rtpbin.recv_rtcp_sink_1 rtpbin.send_rtcp_src_1 !
udpsink host=$DEST port=5007 sync=false async=false
```

where the RTP client creates two RTP sessions, one for video and one for audio. The video is received on port 5000, with its RTCP stream received on port 5001 and sent on port 5005. Audio is received on port 5002, with its RTCP stream received on port 5003 and sent on port 5007.

These two pipelines are also visualized in Figure A.3 for the server and in Figure A.4 for the client.

In the listing above, the audio and video caps are given for a specific configuration only. For each file, depending on the media configuration, separate caps were generated on the server side and then used on the client side to create the pipeline.

Since one of the objectives was to be able to make accurate stream level measurements of RTCP statistics, we have implemented both of these pipelines also in Python using the GStreamer Python API. With the Python implementation, we were able to trace the RTCP messages and extract only the required fields and write their values to file system for post-processing. These statistics proved to be invaluable and have allowed us to develop more accurate perceived quality estimation models.

One of a trivial but very important element of the pipeline is capturing the End-of-Stream messages. the 'e' portion of the "-ve" parameters given to the "gst-launch-1.0" serves that purpose. In the Python implementation, this is achieved by following the signaling messages forwarded through the pipeline and taking necessary actions based on the type of the messages. The source code for both RTP Server and Client implementation is given in Demirbilek (2016a).

Appendix B

Audiovisual Quality Datasets

Table B.1 – University Of Plymouth Dataset Parameters.

ContentType	VideoBitRate	AudioSamplingCount
GeneralFileSize	VideoFrameRate	AudioStreamSize
GeneralDuration	VideoFrameCount	AudioStreamSizeProportion
GeneralOverallBitRate	VideoBitsPixelFrame	AudioInterleaveVideoFrames
GeneralStreamSize	VideoStreamSize	AudioInterleaveDuration
GeneralStreamSizeProportion	VideoStreamSizeProportion	FPS
VideoDuration	AudioDuration	PER

Table B.2 – TUM 1080p50 Dataset Parameters.

ContentType	TotalBitRate	AudioDuration
RatePoint	BitsToAvoid	AudioBitRate
TotalBits	GeneralFileSize	AudioSamplingCount
TotalBitsI	AudioFileSize	AudioStreamSize
TotalBitsP	AudioOverallBitRate	AudioStreamSizeProportion
TotalBitsB	AudioStreamSizeProportion	MeanLumaPSNR

Table B.3 – VQEG Dataset Parameters.

ContentType	GeneralOverallBitRate	AudioStreamSize
VParam	GeneralOverallBitRateMax.	AudioStreamSizeProportion
Aparam	GeneralDataSize	
GeneralFileSize	AudioBitRate	

Table B.4 – Rejected Scores Per Observer in the INRS Audiovisual Quality Dataset.