IDIAP COMMUNICATION

UNE APPLICATION DE RECONNAISSANCE DU LOCUTEUR : LE USER-CUSTOMIZED PASSWORD SPEAKER VERIFICATION

Jérôme Kowalczyk

IDIAP–COM 04-02

JUNE 2004

Dalle Molle Institute for Perceptual Artificial Intelligence • P.O.Box 592 • Martigny • Valais • Switzerland

phone +41 − 27 − 721 77 11
fax +41 − 27 − 721 77 12
e-mail secretariat@idiap.ch
internet http://www.idiap.ch

Table des matières

1 Présentation de l'Idiap 4
1.1 Institut Dalle Molle d'Intelligence Artificielle Perceptive 5
1.2 L'activité de Recherche 5
1.3 Organisation de l'IDIAP 5
1.4 Les différents projets académiques 6
1.4.1 Le projet IM2 : Interactive Multimodal Information Management 6
1.4.2 Le projet AMI :Augmented Multiparty Interaction 6
1.4.3 Les autres projets 6

2 Etat de l'art dans le domaine de la reconnaissance du locuteur 7
2.1 Introduction 8
2.2 Règle de décision en vérification du locuteur 8
2.3 Les paramètres acoustiques : l'extraction de coefficients 9
2.4 Le seuil de décision 9
2.5 Vérification du locuteur indépendante du texte 9
2.6 Vérification du locuteur dépendante du texte 10
2.7 Vérification du locuteur basée sur mot de passe personnalisé 10
2.7.1 Applications 11

3 Le développement de l'application 12
3.1 Cadre du projet 13
3.1.1 Intérêt du point de vue de la recherche 13
3.1.2 Intérêt du point de vue industriel 13
3.2 Objectifs du projet 13
3.3 Réalisation du projet 14
3.3.1 Rappels des notions mathématiques et statistiques 14
3.3.2 Le "User Customized Password Speaker Verification" 14
3.3.3 Entrainement des modèles 17
3.3.4 Les technologies employées 17
3.3.5 Présentation de l'application de démonstration 20
3.3.6 Reflexion sur les modifications à apporter à l'application 22
3.3.7 Réalisation d'un programme de test et validation du code 22
3.3.8 L'intégration de la classe au sein de l'application 24
3.3.9 Phase de test, choix et ajustement des hyperparamètres 24
3.4 Difficultés rencontrées et apports techniques 24
3.4.1 Difficultés 24
3.4.2 Apports techniques 25

4 Analyse **27**
4.1 Le réseau informatique et le Système d'Information 28
4.1.1 Le LAN 28
4.1.2 Une information centralisée 28
4.2 Les différences majeures entre une entreprise et un institut de recherche 28
4.2.1 Mesures de performances 28
4.2.2 Gestion de projet 29
4.3 Le management au sein de l'institut 29
4.3.1 La notion d'interculturalité 29
4.3.2 TAM : Tuesday Afternoon Meetings: 29

Bibliographie **31**

Introduction

C'est au sein de l'Institut Dalle Molle d'Intelligence Artificielle Perceptive situé dans la région du Valais en Suisse que j'ai effectué mon stage de fin d'étude.

Ayant choisi l'option Ingénierie Informatique pour le Multimédia (IIM) comme spécialisation de cinquième année, j'ai recherché un stage pouvant me permettre d'appliquer les enseignements que j'ai reçus pendant ces derniers mois de cours académiques. Cette option nous a en effet permis d'aborder un domaine que je connaissais peu, le traitement de la parole, et je fut particulièrement sensibilisé et interessé par celui-ci.

Le projet qui m'a été confié au cours de ces mois de stage a consisté à incorporer un travail de recherche effectué à l'institut au sein d'une application de démonstration. Intégré au groupe "speech processing", j'ai réalisé ce travail en collaboration avec le doctorant en charge du projet et avec le responsable informatique qui avait développé l'application que je devais faire évoluer.

Réaliser ce produit multimédia m'a offert l'opportunité d'utiliser des outils de développement informatique nouveaux, de mettre en pratique toutes les notions liées au développement orienté Objet et à la création d'Interfaces Homme-Machine au sein d'une structure de très haut niveau technique, très stimulante intellectuellement parlant.

Ce mémoire a pour but de présenter le déroulement de la réalisation du projet. Dans la première partie, je présenterai l'institut et son mode de fonctionnement. Dans un second temps, je rappelerai les points théoriques essentiels dans le domaine de la reconnaissance du locuteur. La troisième partie décrira les aspects techniques et méthodologiques du travail que j'ai réalisé. Enfin, la dernière partie sera consacrée à une reflexion sur les particularités et les besoins d'une structure de recherche mais aussi sur les méthodes employées pour faire collaborer au sein d'une même entité des chercheurs venant des quatres coins du monde.

Chapitre 1

Présentation de l'Idiap

1.1 Institut Dalle Molle d'Intelligence Artificielle Perceptive

Créé en 1991 par la Ville de Martigny, le canton du Valais, l'Université de Genève, l'Ecole Polytechnique Fédérale de Lausanne (EPFL) et Swisscom, l'Institut Dalle Molle d'Intelligence Artificielle Perceptive (IDIAP) est un institut de recherche semi-privé localisé à Martigny. Spécialisé dans le domaine de la reconnaissance vocale, du traitement informatique de l'image, et du "machine learning" (prédiction statistique, intelligence artificielle et optimisation de systèmes), l'IDIAP cherche à s'assurer une place de leader national, européen, et même mondial dans ces domaines de compétences.

Pour cela, elle forme de jeunes chercheurs au travers des doctorats, mais cherche aussi à attirer des universitaires avant la fin de leurs études pour leur donner l'occasion de découvrir le monde de la recherche.

La valorisation des résultats constitue donc un soucis permanent et c'est pourquoi les travaux réalisés sont constamment diffusés, aussi bien dans les milieux scientifiques que dans les milieux industriels.

1.2 L'activité de Recherche

L'ensemble des activités de l'IDIAP a pour objectif l'avancement de la recherche et le développement de prototypes, dans le domaine de l'interaction entre l'homme et la machine.

Les chercheurs de l'IDIAP concentrent essentiellement leurs efforts dans les technologies qui coordonnent des modes d'entrée naturels (telles que parole, image, crayon, toucher, gestes de la main, mouvements de la tête et/ou du corps et même des capteurs physiologiques) avec des sorties de systèmes multimédia comme parole, sons, image, graphique 3D et animation.

1.3 Organisation de l'IDIAP

L'année 2003 fût une année de forte croissance pour l'IDIAP, il a donc été nécessaire de repenser la structure de l'institut. Ainsi les trois groupes de recherche initiaux sont maintenant decomposés en six directions thématiques :

- **Machine Learning:** algorithmes pour la classification, modélisation statistique, techniques connectivistes ...
- **Speech and Audio Processing:** traitement du signal audio, localisation de sources acoustiques, reconnaissance de la parole multilangue ...
- **Computer Vision:** détection d'objets, reconnaissance de mouvements, de textes, d'expressions faciales, extraction d'informations exploitables à partir de séquences vidéos, tracking ...
- **Information Retrieval:** extraction d'informations à partir de données multimédia, reconnaissance et catégorisation de données écrites ...
- **Biometric Authentification:** verification du locuteur, reconnaissance du visage, authentifications multimodales ...
- **Multimodal Interaction:** IHM, développement des systèmes utiles aux réunions, développement de browser multimédia, interaction avec les Smartphone, interface cerveau-machine ...

Le recouvrement entre ces thèmes n'en demeure que plus important, garantissant ainsi une meilleure circulation de l'information parmi les chercheurs.

1.4 Les différents projets académiques

1.4.1 Le projet IM2 : Interactive Multimodal Information Management

L'Idiap est à la tête du Pôle de Recherche National (NCCR) sur la Gestion Interactive et Multimodale de Systèmes d'Information (IM2).

Ce Pôle a pour buts tant l'avancement de la recherche que le développement de prototypes dans le domaine des interactions homme-machine.

Le domaine des interactions multimodales couvre une gamme très large d'activités et d'applications comprenant la reconnaissance et l'interprétation de langages parlés, écrits et gesticulés, la vision par ordinateur, l'indexation automatique de documents multimedia et leur gestion. Les autres thèmes importants sous-jacents sont la protection du contenu informationnel, le contrôle d'accès et la structuration, la récupération et la présentation de l'information multimédia.

Ce Pôle regroupe un grand nombre de partenaires issus de nombreuses institutions universitaires (EPFL, Université de Genêve, Université de Fribourg, Université de Berne, EHTZ) ainsi qu'une HES (Sion), le CSEM (Neuchâtel) et une quinzaine d'industries. Le pôle entretient également de nombreux contacts à l'étranger, notamment sous forme d'un accord d'échange de jeunes chercheurs avec l'ICSI à Berkeley, Californie.

La plupart des travaux de recherche réalisés à l'Idiap le sont dans le cadre du projet IM2.

1.4.2 Le projet AMI :Augmented Multiparty Interaction

Depuis 2003, l'Idiap coordonne un projet dont il a eu l'initiative : le projet AMI. Il se focalise sur les technologies multimodales capables de supporter les interactions humaines (*Smart Meeting Room, remote meeting assistants*). Le projet a ainsi pour but d'améliorer la qualité des données extraites lors des enregistrements de réunions et de rendre les interactions humaines temps-réel plus efficaces.

Il regroupe 14 partenaires européens et 1 partenaire américain (4 instituts de recherche, 5 partenaires académiques, 5 partenaires industriels, et la W3C).

1.4.3 Les autres projets

L'Idiap fait partie de nombreux autres projets :

- **Le projet PASCAL (Pattern Analysis, Statistical modelling and Computational Learning) :** il s'agit d'un réseau d'excellence dans le domaine des interfaces multimodales, réunissant de nombreuses universités européennes. L'EC (European Commity) souhaite que l'Idiap joue un rôle de relai entre les projets AMI et PASCAL.
- **EURON (European Robotics Network):** C'est un autre réseau d'excellence auquel l'Idiap participe de par son activité de recherche dans le domaine de l'interface Cerveau-Machine.
- ...

Chapitre 2

Etat de l'art dans le domaine de la reconnaissance du locuteur

Avant de commencer à implémenter le système de vérification du locuteur basé sur un mot de passe personnalisé au sein de l'application de démonstration, une première phase de rappels des notions et des méthodes employées dans le domaine de la reconnaissance du locuteur fût nécessaire. Je vais donc décrire ici l'état de l'art de ce domaine et plus particulierèment l'état de l'art de la vérification du locuteur.

2.1 Introduction

L'expression vocale est une caractéristique propre du locuteur, les variations individuelles ont deux origines essentielles. D'une part les caractéristiques de l'appareil de phonation (conduit vocal, cordes vocales, ...) qui influencent les formants, la valeur du pitch etc ..., indépendamment de la phrase prononcée, d'autre part une même phrase n'est pas prononcée de la même manière par deux locuteurs : on observe des différences de débits d'élocution dans l'étendue des variations de pitch etc... souvent liées à des paramètres d'ordre social (culture, niveau social) ou d'ordre géographique (accents).

La reconnaissance du locuteur [1] est un terme générique regroupant les problèmes relatifs à la reconnaissance vocale ou à la vérification du locuteur sur la base de l'information contenue dans le signal acoustique.

Ainsi, l'**identification du locuteur** consiste à identifier un locuteur parmi un ensemble fini de locuteurs en utilisant un enregistrement vocal.

La **vérification du locuteur** consiste à vérifier l'identité proclamée par un locuteur. Dans ce cas, les utilisateurs acceptés par le système seront définis comme des **clients** alors que ceux qui sont rejetés seront définis comme des **imposteurs**.

On peut diviser la vérification du locuteur en deux catégories :

- **la vérification du locuteur indépendante du texte** : il s'agit de systèmes qui n'ont aucune connaissance a priori du mot ou de la phrase qui va être prononcé. L'utilisateur peut donc dire ce qu'il souhaite lors de l'utilisation.
- **la vérification dépendante du texte** : ces systèmes ont une connaissance préalable de ce qui va être prononcé; ainsi lors de l'utilisation, le texte répété par l'utilisateur fait parti d'un vocabulaire restreint et prédéfini.

2.2 Règle de décision en vérification du locuteur

Pour vérifier l'identité proclamée par l'utilisateur d'un système de vérification du locuteur, on fait intervenir un test d'hypothèse binaire : soit l'utilisateur est un client, soit c'est un imposteur. Pour modéliser cette hypothèse, on cherche à estimer :

- $P(S_c|X)$ qui est la probabilité que X ait été prononcé par le locuteur proclamé S_c.
- $P(\overline{S_c}|X)$ qui est la probabilité que X ait été prononcé par un autre locuteur, autre que S_c ($\overline{S_c}$ représente donc dans ce cas l'ensemble de tous les locuteurs rivaux possibles).

et nous accepterons le locuteur comme un client si :

$$P(S_c|X) > P(\overline{S_c}|X) \tag{2.1}$$

En appliquant le théorème de Bayes, notre test d'hypothèse 2.1 peut être redéfini comme suit :

$$p(X|S_c)P(S_c) > p(X|\overline{S_c})P(\overline{S_c}) \quad \textit{soit} \quad \frac{p(X|S_c)}{p(X|\overline{S}_c)} > \frac{P(\overline{S}_c)}{P(S_c)} = \delta \tag{2.2}$$

On va donc chercher à estimer d'une part le rapport de vraisemblance $\frac{p(X|S_c)}{p(X|\overline{S}_c)}$ en utilisant des modèles statistiques et on va comparer ce rapport au *prior* $\frac{P(\overline{S}_c)}{P(S_c)}$ que l'on remplace généralement par un seuil δ.

La manière d'estimer le rapport de vraisemblance sera différente selon que le système soit dépendant ou indépendant du texte.

2.3 Les paramètres acoustiques : l'extraction de coefficients

Pour estimer le rapport de vraisemblance, on emploie des modèles statistiques qui ne peuvent pas traiter directement le signal temporel. On s'est donc inspiré des connaissances en matière de reconnaissance de la parole pour extraire des paramètres caractéristiques de ce signal. Les systèmes actuels de vérification du locuteur utilisent donc des paramètres basés sur le spectre à court-terme et moyen-terme, ainsi que sur l'énergie [2]. Par ailleurs, afin d'atténuer l'effet des variations du canal de transmission, on utilise différentes techniques telles que la soustraction cepstrale (on soustrait à chaque vecteur acoustique la moyenne de ces vecteurs).

2.4 Le seuil de décision

Chaque type de système va determiner des scores (ou des rapports de vraisemblance) estimés à partir du modèle statistique associé à l'identité proclamée et des modèles *du monde* (ou modèles *anti-clients*), sur la base d'une séquence de vecteurs acoustiques X prononcée par le locuteur.

Ces scores sont alors comparés à un seuil. On comprend alors aisément que la détermination de ce seuil est souvent déterminante dans le bon fonctionnement de tout système de vérification. S'il est trop élevé, le *taux de faux rejet*, soit les clients rejetés comme étant imposteurs, sera trop important, alors que s'il est trop bas, *le taux de fausse acceptation*, soit des imposteurs reconnus comme des clients, sera innaceptable.

Le choix de cette valeur de seuil dépend du type d'application que l'on cherche à développer. Ainsi dans le cas de système à haute sécurité, on estimera le seuil de telle sorte que les accès imposteurs soient le moins nombreux possibles. Dans le cadre du travail de recherche aucune contrainte n'est réellement imposée et l'on choisit d'utiliser, de manière arbitraire, la méthode de l'**EER** (Equal Error Rate) pour estimer ce paramètre.

L'EER correspond au point de fonctionnement pour lequel le taux de faux rejet est égal au taux de fausse acceptation.

Si ce seuil EER est estimé sur l'ensemble d'entraînement, il sera alors appelé *seuil a priori*, alors qu'il sera appelé *seuil a posteriori* s'il est estimé sur l'ensemble du test. Il est clair que pour des systèmes réels, cette valeur a posteriori n'est pas accessible; on supposera alors souvent que la base d'entraînement est suffisamment représentative des conditions réelles ou on cherchera à remplacer les performances EER par la moyenne des valeurs de taux de rejet et de fausse acceptation (HTER[1]).

2.5 Vérification du locuteur indépendante du texte

Ce type de système n'impose aucune contrainte au locuteur, il peut prononcer ce qu'il veut lors des phases d'enrollement et des phases de test. De ce fait, on peut dire que ce système est particulièrement convivial pour l'utilisateur. Toutefois, les modèles entraînés ne sont pas vraiment robustes et cette caractéristique constitue le point négatif majeur de ce type de système.

Depuis maintenant quelques années, l'approche utilisant les modèles GMM[2] permet de modéliser des distributions statistiques relativement complexes, ce qui en a fait l'approche dominante utilisée en vérification du locuteur indépendante du texte [3].

1. Half Total Error Rate
2. Gaussians Mixture Models : ensemble de distributions statistiques, appelées distributions gaussiennes, caractérisées chacune par un vecteur moyenne et une matrice de covariance.

Pour modéliser les imposteurs, on utilise un modèle GMM *du monde* dont les paramètres acoustiques sont estimés sur un ensemble d'entraînement contenant du texte prononcé par un grand nombre d'utilisateurs, (hommes et femmes). Ce modèle, entraîné par l'algorithme EM (Expectation-Maximization) [2], est utilisé pour estimer la vraisemblance $p(X|\overline{S_c})$.

Pour chaque locuteur, on utilise un modèle GMM qui dérive du modèle *du monde* et qui est adapté au locuteur avec ses propres données grâce à une adaptation MAP (Maximum A Posteriori) [4]. Ce modèle va permettre d'estimer $p(X|S_c)$.

On peut alors comparer le rapport de vraisemblance obtenu par le ratio des scores de ces deux GMM à la valeur de seuil pour accepter ou refuser le locuteur.

2.6 Vérification du locuteur dépendante du texte

Contrairement au système décrit dans la section précédente, une contrainte est imposée à l'utilisateur : le texte qu'il doit prononcer est en effet imposé par le système, soit de manière statique, soit de manière aléatoire. Le système connait par avance le mot ou la phrase qui va être prononcé. On associe donc chaque locuteur à un modèle qui encode à la fois le contenu lexical et les caractéristiques du locuteur. Une première approche, l'approche DTW [3] a été à la base de nombreux modèles commerciaux. Toutefois, à l'heure actuelle, l'approche la plus courante est l'approche HMM [4].

Le mot de passe associé à chacun des locuteurs est représenté par un modèle HMM gauche-droite [5] représentatif du contenu lexical du mot de passe et dont les paramètres sont ceux du modèle *du monde* (lui aussi un modèle HMM gauche-droite) adaptés avec les données propres à l'utilisateur.

Le score $p(X|S,M)$ est alors calculé grâce à l'algorithme Viterbi [6] [5] appliqué au modèle M du locuteur S_c.

Le score $p(X|\overline{S_c},M)$ est calculé par l'algorithme Viterbi appliqué au modèle M de $\overline{S_c}$ qui représente tous les autres locuteurs possibles.

Comme précédemment on compare le ratio des valeurs ainsi estimées avec le seuil de décision.

2.7 Vérification du locuteur basée sur mot de passe personnalisé

La méthode SV-UCP (Speaker Verification based on User-Customized Password) développée au sein de l'Idiap, en est encore au stade de la recherche. Elle peut être assimilable à une vérification dépendante du texte, la seule différence tient du fait que le mot n'est pas connu du système avant que l'utilisateur ne le répète et qu'il va donc falloir trouver sa transcription phonétique de manière automatique [7] [8]. On demande au locuteur de prononcer son mot de passe à L reprises, en on crée un modèle HMM gauche-droite pour chacune de ces répétitions. Il s'agit de la méthode que j'ai du implémenter au sein de l'application de démonstration.

Elle est caractérisée par la règle de décision suivante (explicitée ici pour un seul des L modèles clients) :

on cherche à estimer $P(M_c,S_c|X)$ qui représente la probabilité *a posteriori* que le bon locuteur S_c ait prononcé le bon mot M_c considérant la séquence acoustique observée X. Cette probabilité sera comparée à $P(M_c,\overline{S_c}|X)$ qui represente la probabilié *a posteriori* qu'un autre locuteur, en l'occurence un imposteur, ait prononcé le bon mot de passe et à $P(\overline{M_c},S|X)$ qui représente la probabilité *a posteriori* que n'importe quel locuteur (y compris le client) ait prononcé n'importe quel mot. De cette

3. Dynamic Time Warping : méthode basée sur une déformation temporelle des modèles *de référence*

4. Hidden Markov Models : on distingue les modèles HMM ergodiques, où tout état du modèle est relié à tout autre et à lui-même, des modèles HMM gauche-droite, où la topologie permet de modéliser la séquentialité des états.

5. Algorithme qui ne considère que le meilleur chemin, c'est-à-dire la séquence d'états la plus probable dans le modèle HMM. Il s'agit d'un cas simplifié de l'algorithme Forward-Backward où toutes les séquences d'états possibles sont prises en compte

manière, on peut définir mathématiquement la règle de décision de la sorte :

$$(S,M) = (S_c,M_c) \quad si \quad P(M_c,S_c|X) \geq P(M_c,\overline{S}_c|X) \tag{2.3}$$

$$et \quad P(M_c,S_c|X) \geq P(\overline{M_c},S|X) \tag{2.4}$$

En utilisant la règle de Bayes, les deux équations précédentes peuvent s'écrire de la sorte :

$$\frac{p(X|M_c,S_c)}{p(X|M_c,\overline{S}_c)} \geq \left[\frac{P(M_c,\overline{S}_c)}{P(M_c,S_c)} = \Delta_1\right] \tag{2.5}$$

$$\frac{p(X|M_c,S_c)}{p(X|\overline{M}_c,S)} \geq \left[\frac{P(\overline{M}_c,S)}{P(M_c,S_c)} = \Delta_2\right] \tag{2.6}$$

où $\Delta 1$ et $\Delta 2$ sont les seuils de décision.

- $p(X|M_c,S_c)$ est la *vraisemblance* de la prononciation, étant donné le modèle HMM du client (M_c,S_c).
- $p(X|M_c,\overline{S}_c)$ est la *vraisemblance* de la prononciation, étant donné le modèle HMM des autres locuteurs $(M_c,\overline{S}_c)$.
- $p(X|\overline{M}_c,S)$ est la *vraisemblance* de la prononciation étant donné le modèle qui représente les autres mots (sauf le mot de passe).

En passant dans le domaine logarithmique et en normalisant chacune des probabilités par le nombre de trames de la phrase de test, on obtient les deux *rapports de vraisemblance* (*log likelihood ratio*, LLR) suivants :

$$LLR_s = \frac{1}{N}[\log p(X|M_c,S_c) - \log p(X|M_c,\overline{S_c})] \geq \omega_1 \tag{2.7}$$

$$LLR_u = \frac{1}{N}[\log p(X|M_c,S_c) - \log p(X|\overline{M_c},S)] \geq \omega_2 \tag{2.8}$$

N est la longueur du mot (le nombre de trames acoustiques) aprés la suppression des trames de silence. La première de ces règles de décision est utilisée pour vérifier que le locuteur qui prononce le bon mot de passe est le bon client. Cette vérification est basée sur les caractéristiques du locuteur et répresente la partie de vérification du locuteur (speaker verification). La seconde régle de décision est utilisée pour vérifier que ce qui est prononcé est bien le bon mot de passe. Il s'agit de la vérification de la prononciation (utterance verification).

On calcule ensuite des *rapports de vraisemblance* globaux (moyenne des rapports de vraisemblance partiels estimés pour chaque modèle), avant de combiner ces scores de manière pondérée pour obtenir un unique score final qui sera comparé à la valeur seuil.

2.7.1 Applications

Les **applications potentielles des systèmes de vérification** sont nombreuses : on pense notamment à la sécurisation des cartes d'accès (cartes de crédit ou de téléphone), le contrôle d'accès à des bases de données (téléphoniques ou bancaires) et batiments protégés, le commerce électronique, les services d'information et de réservations, l'accès distant à un réseau LAN, en somme tout ce qui nécessite une identification de la part de l'utilisateur.

Je viens de présenter les différents systèmes de vérification du locuteur, dont celui que j'ai du intégrer au sein de l'application de démonstration, le SV-UCP. Cette application telle qu'elle a été développée réalise une vérification du locuteur sur le mode du texte indépendant et avec une vérification du visage (localisation et adaptation de modèles GMM), les deux modes étant couplés afin d'accepter ou rejeter un utilisateur.

Après avoir ajouté le nouveau mode de vérification du locuteur, le module de vérification du visage pourra donc être utilisé indifféremment avec le module de vérification du locuteur texte indépendant ou avec le SV-UCP.

Chapitre 3

Le développement de l'application

3.1 Cadre du projet

Dans le cadre du projet MULTI et grâce aux fonds de la FNS (Fonds National Suisse), il a été proposé à un des doctorants de l'Idiap de travailler sur un nouveau système de vérification du locuteur : le Speaker Verification based on User-Customized Password (SV-UCP). Le but de ce projet est d'améliorer "l'état de l'art" dans les systèmes de vérification du locuteur, où l'Idiap possède une position reconnue de leader.

De manière plus spécifique, le but de ce projet réside dans la recherche de nouvelles alternatives dans les systèmes de vérification du locuteur, où celui-ci est autorisé à choisir son mot de passe, en le prononçant plusieurs fois de suite.

En effet, il a été demontré par différentes études que les nouveaux systèmes de vérification du locuteur devaient être plus conviviaux et ergonomiques. De ce fait, le SV-UCP est un bon moyen de faciliter l'interaction entre l'homme et la machine.

3.1.1 Intérêt du point de vue de la recherche

Il y a de nombreuses motivations qui peuvent amener à développer les activités de recherche liant simultanément la reconnaissance du texte et du locuteur:

- Dans la réalité, quand une personne essaie de reconnaître un locuteur, les informations de sens du mot et d'identité du locuteur sont indissociables.
- Dans la plupart des systèmes dépendants du texte, une phase de reconnaissance de la parole est nécessaire afin de segmenter la phrase ou le mot avant de le comparer aux différents modèles déjà entraînés.
- Les différents résultats obtenus en incluant des réseaux de neurones artificiels au sein des systèmes de vérification du locuteur ont été très prometteurs et de ce fait, ce domaine constitue un axe de recherche à explorer et développer.

3.1.2 Intérêt du point de vue industriel

L'interêt pour ces nouvelles technologies n'a cessé de croître ces dernières années, et beaucoup de services basés sur la reconnaissance de la voix du locuteur ont été développés, essentiellement dans le domaine des télécommunications et des services financiers. Il est évident que les moyens traditionnels de sécurité, dans les environnements de téléphonie mobile par exemple, seront à terme remplacés par des protections gérées par la voix de l'utilisateur. Pour les fournisseurs de "nouvelles technologies", ce marché incluant à la fois des produits et des services semble être très attractif. Les études les plus optimistes prévoient une progression annuelle de ces systèmes de l'ordre de 35 à 40%.

3.2 Objectifs du projet

Les différents projets de recherche doivent être exploitables et montrables à l'ensemble des partenaires académiques et économiques de l'Idiap, qui doivent avoir une certaine visibilité de l'activité de l'institut.

Ainsi, une application de vérification biométrique était déjà développée avant mon arrivée. Cette application réalise la vérification de l'identité de l'utilisateur sur la base de son empreinte vocale mais aussi sur détection et vérification du visage.

Le but de mon projet était donc d'intégrer au sein de cette application le nouveau système de vérification de la parole, le SV-UCP, tout en laissant la possibilité d'utiliser le système de vérification préalablement implémenté.

3.3 Réalisation du projet

Pour mener à bien ce projet, il était nécessaire de le décomposer en plusieurs étapes et de les planifier :

1. Rappels des notions mathématiques et statistiques impliquées.
2. Compréhension des spécificités du SV-UCP.
3. Entrainement des modèles : l'utilisation des scripts pour la variation des paramètres et validation des résultats.
4. Prise en main des différentes technologies informatiques impliquées : QT, Torch et debuggers.
5. Compréhension et prise en main de l'application existante.
6. Réflexion sur les modifications à apporter à l'application.
7. Réalisation d'un programme de test et validation du code.
8. Intégration à l'application.
9. Phase de test, choix et ajustement des hyperparamètres.

Je vais à présent décrire chacune de ces étapes, en décrivant les tâches réalisées et les difficultés rencontrées.

3.3.1 Rappels des notions mathématiques et statistiques

Je devais appréhender, en premier lieu, les différentes notions mathématiques que j'allais devoir utiliser pour comprendre en détail le système que je devais intégrer. La plupart des ces notions (explicitées préalablement dans 2) avaient été abordées lors des enseignements académiques de l'ENIC, mais il était tout de même nécessaire de me "remettre à niveau". Pour ce faire, j'ai lu plusieurs ouvrages et rapports scientifiques et j'ai réalisé divers Travaux Pratiques sous Matlab, préparés par mon tuteur de stage pour les enseignements qu'il dispense dans plusieurs universités (EPFL, Berkeley), sur le thème des distributions gaussiennes, des algorithmes de maximisation et des chaînes de Markov cachées.

Ce travail préalable était indispensable et dura environ 3 semaines.

3.3.2 Le "User Customized Password Speaker Verification"

Le travail de recherche a été effectué par le doctorant en charge du projet. Aussi je n'expliquerai pas les différents systèmes intermédiaires qui ont été testés mais me contenterai de décrire en détail le système définitif qui a été intégré à l'application de démonstration.

Dans la plupart des systèmes de vérification du locuteur dépendants du texte, le système possède une connaissance *a priori* du mot du passe du client (en l'occurence la transcription phonétique). Dans le SV-UCP, le système ne possède pas cette information au moment de la phase d'enregistrement d'un nouveau client: celui-ci peut choisir son mot de passe sans aucune contrainte sur le vocabulaire. Cette caractéristique rend ce système plus convivial pour l'utilisateur ; par ailleurs, la vérification simultanée du mot et du locuteur ajoute un degré de vérification supplémentaire. Néanmoins, ces systèmes posent quelques problèmes pratiques:

- **L'inférence du modèle du mot de passe:** il s'agit ici de déterminer de manière automatique la transcription phonétique du mot de passe choisi par le client à partir de quelques répétitions. La transcription phonétique inférée doit être représentative du contenu lexical du mot de passe, cela nécessite un bon système de reconnaissance de la parole.
- **L'adaptation du locuteur:** il s'agit de créer un modèle acoustique (HMM) pour chaque client. Comme les données d'entrainement fournies par chaque client sont limitées, cette étape consiste à adapter les modèles HMM inférés pour obtenir un modèle dépendant du client.
- **La decision :** il s'agit de définir la manière avec laquelle on accepte ou on refuse un utilisateur.

Le choix des modèles

J'avais besoin, afin d'implémenter ce système, de 3 modèles acoustiques différents, entraînés et donnant des résultats satisfaisants.

- Le premier d'entre eux était un Multi-Layer-Perceptron Θ (MLP) indépendant du locuteur utilisé lors de la phase d'inférence. Ce MLP est composé de 234 entrées (soit 9 vecteurs consécutifs de 26 coefficients chacun, afin de prendre en compte le contexte), de 600 unités cachées et de 36 sorties, chacune d'entre elles correspondant à un phonème spécifique de la langue française. La valeur du nombre d'unités cachées a été determinée après l'entrainement de plusieurs MLP (entre 450 et 700 avec un pas de 50) : celui étant composé de 600 unités cachées donnant le meilleur taux de reconnaissance de phonème.
- Le second modèle est un modèle HMM (λ), indépendant du locuteur comprenant les 36 modèles de phonèmes indépendants du contexte. Chaque phonème est représenté par un modèle HMM gauche-droite à 3 états, chacun de ces états étant représenté par 3 gaussiennes. Ce modèle est utilisé d'une part comme une distribution *a priori* pour l'adaptation MAP des modèles d'un nouveau client (il s'agit d'une adaptation où seule les moyennes des gaussiennes sont modifiées avec les paramètres de l'utilisateur) et pour construire le *modèle du monde* utilisé pour estimer le score de normalisation lors de verification de l'identité proclamée par le locuteur.
- Le dernier modèle est un modèle de mélange de gaussiennes (GMM) (Λ) composé de 200 gaussiennes. Ce modèle est utilisé pour estimer le score de normalisation afin de vérifier le mot prononcé.

Tout ces modèles ont été entraînés en utilisant la base de donnée *PolyPhone* [1] [9] avec les coefficients MFCCs.

L'extraction de coefficients

La première étape du système consiste à extraire les coefficients MFCC (Mel-Frequency Cepstral Coefficients). Les paramètres MFCC sont des coefficients cepstraux issus du calcul des énergies du signal obtenues à partir d'une banque de filtres en échelle de fréquence Mel. On extrait généralement 12 coefficients cepstraux pour la reconnaissance du locuteur. A ces coefficients, on ajoute le logarithme de l'énergie globale calculée dans le domaine temporel.

L'inférence

Cette étape requiert un bon système de reconnaissance de la parole, indépendant du locuteur, dont l'unité est le phonème. Pour cette étape, l'utilisation d'un système hybride HMM/MLP [2][10] paraissait tout à fait appropriée puisqu'il donne un bon taux de reconnaissance des phonèmes. Pour chaque séquence acoustique $X = \{x_1,x_2,...,x_n\}$ associée à une répétition du mot de passe, le MLP fournit, pour chaque trame acoustique x_n, les probabilités *a posteriori* $p(q_k|x_n)$ de chaque phonème q_k. En utilisant ces probabilités ainsi que le modèle HMM ergodique, un décodage Viterbi est appliqué afin de trouver la séquence optimale des phonèmes. On obtient L transcriptions phonétiques, chacune d'entre elles étant associée à une répétition de mot du passe.

On souhaite maintenant choisir "la meilleure" transcription phonétique pour créer le modèle HMM du mot de passe du client.

Une première approche consistait à choisir parmi ces L transcriptions phonétiques celle qui représente le mieux le contenu lexical du mot de passe. Mais il s'est avéré que garder toutes les transcriptions phonétiques et construire un modèle HMM pour chacune d'elles permettait d'obtenir des résultats significativement meilleurs. Ainsi, pour chaque transcription phonétique inferée, on construit un modèle HMM strictement gauche-droite en concaténant les modèles de phonèmes qui constituent la transcription phonétique. Les modèles de phonèmes utilisés sont ceux du modèle λ. On a alors L modèles HMM

1. Une base de données francophone téléphonique
2. Dans ces systèmes, le MLP (dont chacune des sorties correspond à un phonème) est utilisé pour estimer les probabilités *à posteriori* locales des différents états (chaque état correspond à un phonème) d'un modèle HMM.

(M_c^l,λ) qui sont indépendants du locuteur. La prochaine étape va donc consister à adapter ces modèles de phonème et à les rendre dépendants du locuteur.

Adaptation du locuteur

Une fois les modèles du mot de passe construits, une procédure d'adaptation MAP (Maximum A Posteriori) est appliquée. Cette procédure consiste à adapter (en utilisant toute les données d'entrainement) la moyenne des gaussiennes de chaque modèle de phonème qui compose le modèle inféré. A la fin de cette étape, on se retrouve avec L modèles HMM (M_c^ℓ,λ_c) gauche-droite dépendants du locuteur. La figure 3.1 représente le schéma des trois premiers blocs du système (extraction, inférence, adaptation).

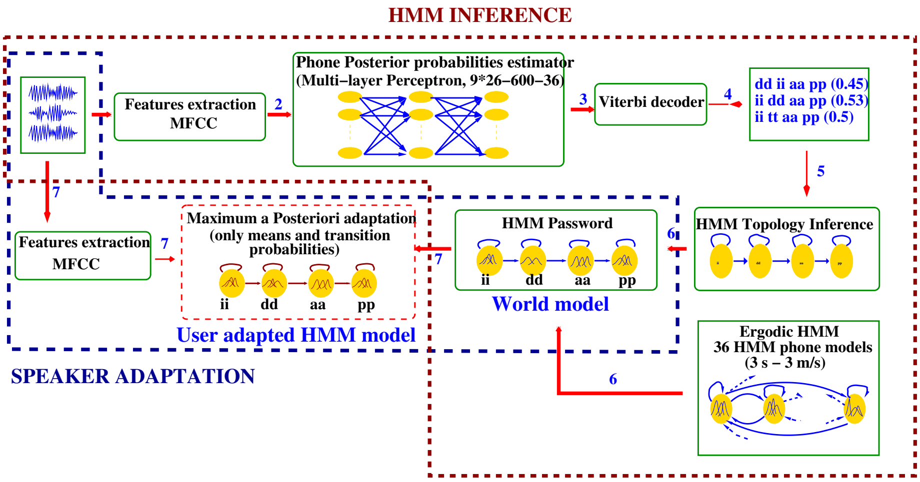


FIG. 3.1 – *Inférence et adaptation appliquée à un des cinq modèles*

La règle de décision

Maintenant que l'on a défini les différents modèles impliqués dans l'estimation des deux rapports de vraisemblance (cf. *Le choix des Modèles*), pour la vérification du locuteur et la vérification du mot prononcé, les deux équations 2.7 et 2.8 peuvent s'écrire de la manière suivante :

$$LLR_s = \frac{1}{N}[\log p(X|M_c,\lambda_c) - \log p(X|M_c,\lambda)] \geq \omega_1 \quad (3.1)$$

et

$$LLR_u = \frac{1}{N}[\log p(X|M_c,\lambda_c) - \log p(X|\Lambda)] \geq \omega_2 \quad (3.2)$$

Le score final de la partie de vérification du locuteur est une moyenne des LLR_s calculés pour chacun des L modèles clients (2.7) et sera estimé comme suit :

$$LLR1 = \frac{1}{L}\left[\sum_{\ell=1}^{L} LLR_s^\ell\right] \quad (3.3)$$

Le score final de la partie de vérification de la prononciation est une moyenne des LLR_u calculés pour chacun des L modèles clients (2.8) et sera estimé comme suit:

$$LLR2 = \frac{1}{L}\left[\sum_{\ell=1}^{L} LLR_u^\ell\right] \quad (3.4)$$

La décision finale peut alors être prise de deux manières :

- Cascadage des 2 étapes : D'abord on réalise l'étape de vérification de la prononciation. Si le score estimé excède un seuil prédefini, on réalise l'étape de vérification du locuteur. Dans ce cas, le locuteur est accepté si les deux scores dépassent leurs valeurs de seuil respectives.
- Combinaison pondérée des deux scores : Dans ce cas, le score final sera estimé selon la règle suivante:

$$SF = \alpha LLR1 + (1-\alpha)LLR2 \tag{3.5}$$

  où $0 \leq \alpha \leq 1$. Ce score sera comparé à un seuil Δ indépendant du client.

Les valeurs du paramètre α et le seuil Δ doivent alors être optimisés.

Les différents tests effectués ont montré que la pondération des deux scores améliorait la performance du système.

Le système que je viens de présenter utilise donc d'une part un système hybride HMM/MLP pour trouver la transcription phonétique de chacune des répétitions du mot de passe du client et d'autre part un modèle HMM pour représenter à la fois le mot de passe et les caractéristiques du client.

Il s'est avéré que ce système donne des résultats proches d'un système ayant une connaissance *a priori* de la transcription phonétique du mot de passe.

On comprend également aisément qu'un grand soin doit être apporté au choix et à l'entraînement des modèles *du monde* utilisés pour la normalisation du score mais aussi à l'ajustement des valeurs de seuil lors de l'étape de décision.

Une plate-forme de recherche concernant ce projet avait été développée par le doctorant. Ainsi il existait plusieurs programmes informatiques qui réalisaient de manière indépendante et à partir de données statiques chacune des tâches décrites précédemment (six programmes en tout). La tâche principale du projet allait donc être l'incorporation de tous ces blocs au sein d'un même application et le traitement des données de manière dynamique, *à la volée.*

3.3.3 Entrainement des modèles

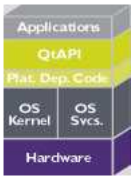
L'entraînement des modèles s'est fait à l'aide de **Torch** et de scripts écrits en Perl. L'utilisation des scripts permet de réaliser facilement l'entraînement des modèles en lançant automatiquement une série de programmes et en faisant varier les hyperparamètres. Pour l'entraînement des réseaux de neurones, il était nécessaire de faire varier le nombre d'unité cachées; pour les modèles HMM, plusieurs modèles ont été entraînés où l'on cherchait à faire varier le nombre d'états par modèles et le nombre de gaussiennes ; enfin pour les modèles GMM, seul le nombre de gaussiennes était modifié.

Cette phase d'optimisation des modèles de référence a été l'une des plus longues du projet car la qualité de l'application dépendait en partie de la qualité de ces modèles.

3.3.4 Les technologies employées

L'application de démonstration nécessite d'une part l'utilisation de librairies pour la réalisation de l'interface (couches hautes) et d'autre part de librairies pour la gestion de "l'intelligence artificielle" (couches basses). C'est ainsi que les outils qui ont été choisis sont **QT**[3] et **Torch**[4].

Pour la réalisation de ce projet, je ne suis pas intervenu dans le choix des différentes technologies de développement, dans la mesure où l'objectif était de faire évoluer l'application et non pas de la créer.



Toolkit QT

Edité par la société Trolltech, **QT** est un *toolkit* graphique complet écrit en C++ et dont le mode de licence est double (libre ou propriétaire). Il est relativement simple à utiliser (ressemble quelque peu à Swing) et possède de nombreux outils et extensions (parseurs XML, bases de données SQL, générateur d'interfaces Qt Designer, utilisation de l'unicode avec la classe QString, compilateur g++, debugger GDB). Il est installé par défaut sur Kde, environnement graphique de bureau Open Source destiné aux stations de travail Unix/Linux.

L'une des principales caractéristiques de QT est la portabilité. Il s'agit en effet d'un produit multiplate-forme, disponible sous Linux/Unix, X11 et Windows, et qui permet par simple recompilation de la même base de code de réaliser une application native sur le système désiré (QT est implémenté sur les couches basses des systèmes graphiques).

La gestion des interactions entre objets est réalisée grâce à un mécanisme de Signaux et de Slots (Les objets émettent des signaux quand ils reçoivent des événements et sont liés à des slots qui appelent automatiquement une ou plusieurs méthodes). Il s'agit de l'aspect central de QT ce qui le rend sûr (typage) et modulaire mais nécessite une phase de précompilation.

Le générateur d'interfaces interactives, QT Designer, est l'environnement utilisé pour concevoir et implémenter les interfaces utilisateurs construites avec QT. Grâce à Designer, il est simple de moduler les composants visuels (*Widgets* assimilables aux *Controls* en terminologie Windows). Pour chacun des formulaires créés, Designer génère un fichier XML (extension .ui).

Sa modularité, sa portabilité et sa gratuité (sous Linux) font de QT l'outil adéquat pour la réalisation de cette application.

Torch

Torch est une librairie de *Machine Learning*, sous licence BSD, écrite en C avec le concept de classes trouvées en C++ et développée au sein même de l'Idiap. Ce concept de classes apporte un aspect modulaire alors que le C apporte la robustesse et l'efficience, nécessaire pour la mise en oeuvre d'algorithmes souvent très lourds.

L'objectif de **Torch** est de contenir tout "l'état de l'art" en matière de *Machine Learning*, que ce soit pour la résolution de problèmes statiques ou dynamiques. Ainsi ***Torch*** contient un grand nombre de packages permettant d'implémenter et d'utiliser des Réseaux de Neurones Artificiels, des modèles HMM, des modèles GMM .... Tout ceci peut être utilisé pour le traitement du signal (extraction et prédiction), pour le traitement de l'image et de la vidéo (reconnaissance du visage, des mouvements ou de l'écriture), ou encore pour le traitement de la parole (reconnaissance de la voix ou du locuteur).

Concepts de base de Torch : **Torch** a été développé selon un paradigme orienté Objet et implémenté en C++. Pour simplifier la modification d'algorithmes existants ou pour implémenter de nouveaux algorithmes ou nouvelles méthodes, une stratégie modulaire a été adoptée en déterminant les classes de base suivantes :

- `DataSet` : cette classe permet de manipuler les données. De nombreuses sous-classes offrent la possibilité de manipuler les données de manière statique ou dynamique, et d'y accéder à travers

3. www.trolltech.com/products/qt/
4. www.torch.ch

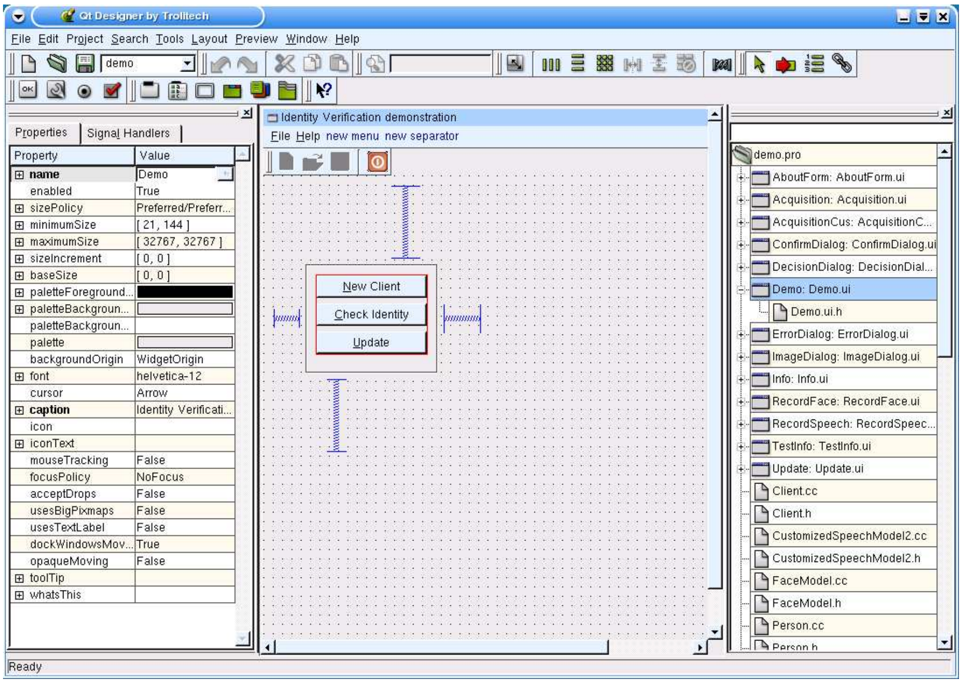


Fig. 3.2 – *L'environnement de développement QT Designer*

la mémoire ou directement sur le disque (pour un gros ensemble de données par exemple).

- `Machine` : cette classe représente toutes les boites noires, qui étant donné des données d'entrées et quelques paramètres, ressortent des valeurs en sorties. Il s'agit par exemple de Réseaux de Neurones Artificiels, de HMM, *etc.*
- `Trainer` : cette classe est utilisée pour régler et ajuster les paramètres d'une `Machine`, étant donné un critère particulier ou des données d'entrée. Elle sert aussi à tester les `Machines` suivant certaines données d'entrée.
- `Measurer` : cette classe permet d'écrire dans des fichiers, différentes valeurs caractéristiques, comme l'erreur de classification, le *log likelyhood*, *etc.*

De manière générale, l'idée de **Torch** peut se résumer de la manière suivante : tout d'abord les `DataSet` permettent de charger les données d'entraînement ou de test. Le `Trainer` passe ces données à la `Machine` qui les traite et génère des valeurs de sorties. Ces valeurs sont ensuite utilisées pour régler les paramètres de la `Machine`. Au cours de ce processus, un ou plusieurs `Measurer` peuvent être utilisés pour visualiser les performances du système.

Comparaison avec les bibliothèques de *Machine Learning* existantes : La plupart des algorithmes implémentés dans **Torch** était déjà disponible au sein d'autres bibliothèques standards telle que HTK (pour l'entraînement des HMM pour les tâches de reconnaissance de la parole), SVMLight et SVMTorch (pour l'entraînement des Machines à Vecteur Support) ou encore différents packages indépendants utilisés pour les réseaux de neurones. Toutefois, il n'existe aucun environnement de programmation qui réunisse tous ces algorithmes, permettant à un utilisateur de tester plusieurs solutions pour une même tâche, et ainsi d'utiliser les mêmes techniques pour évaluer les performances.

Torch constitue donc l'outil idéal pour la gestion de l'intelligence artificielle de l'application, dans la mesure où il est le seul à regrouper toutes les bibliothèques nécessaires au traitement du signal, à l'utilisation des `Machines` (leur entraînement et leur test) et aussi au traitement et au passage des données entre les différents blocs du système.

Outils de debuggage

Valgrind : C'est un logiciel Open Source de détection de problèmes de gestion de la mémoire pour x86-GNU/Linux. Il vérifie les opérations de lecture - écriture de la mémoire. Il peut détecter des problèmes tels que la mémoire non-initialisée, la mémoire libérée trop tôt, la lecture/écriture au-delà des blocks alloués, la lecture/écriture à des endroits inappropriés de la pile ou encore des erreurs d'utilisation des threads.

GDB : C'est l'acronyme de Gnu DeBugger. C'est un debugger puissant dont l'interface est totalement en ligne de commande. On peut aussi le trouver encapsulé dans des interfaces graphiques, comme XXGDB ou DDD. GDB est publié sous la licence GNU et gratuit.

3.3.5 Présentation de l'application de démonstration

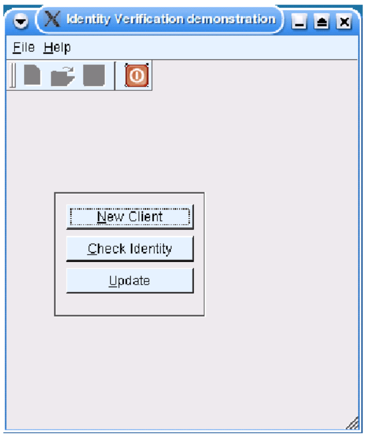


FIG. 3.3 – *Interface de démarrage de l'application*

L'application de démonstration réalise une vérification biométrique où la reconnaissance du locuteur et du visage sont combinées pour accepter ou refuser l'utilisateur.

La reconnaissance du locuteur est basée sur un système indépendant du texte, où des modèles GMM *du monde* sont adaptés à partir des données propres à l'utilisateur.

La reconnaissance du visage se fait en deux étapes : dans un premier temps, une phase de localisation est effectuée puis une phase d'adaptation où l'on adapte également des modèles GMM *du monde* avec les données spécifiques à l'utilisateur du système.

Structure globale

L'application est structurée selon 3 couches :

– QT qui gère l'interface et les interactions utilisateurs.
  Un utilisateur est invité dans un premier temps à s'enregistrer; durant cette phase, il sera amené à entrer diverses données le concernant (nom, prénom, âge, sexe), puis à enregistrer plusieurs fois son visage et sa voix afin d'entraîner les différents modèles statistiques.

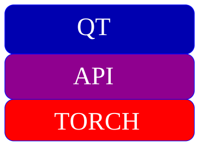


Fig. 3.4 – *Représentation des 3 couches logicielles de l'application*

Pour la phase de test, l'utilisateur choisit d'abord l'identité qu'il veut tester (que ce soit la sienne pour un accès valide ou une autre pour un accès imposteur) avant de capturer ses données vocales et visuelles. Il sera alors accepté ou rejeté.
Enfin, il est possible de visualiser à travers l'application l'ensemble des données et sessions relatives à chaque utilisateur.
L'interface de l'application suit le schéma suivant :

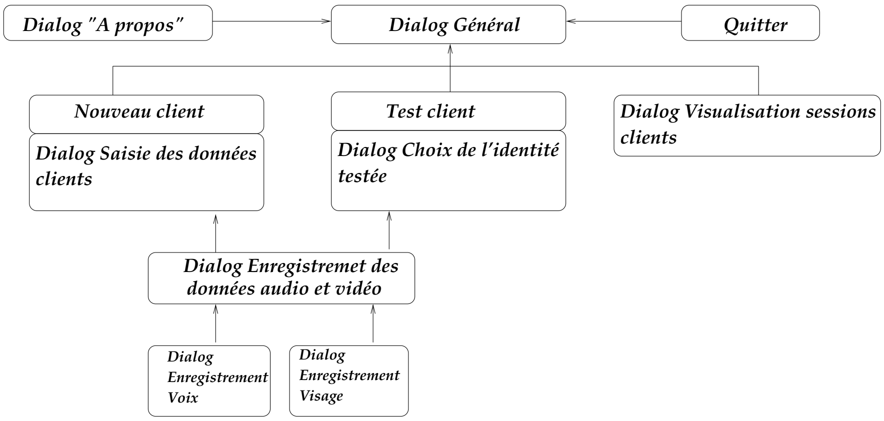


Fig. 3.5 – *Schéma représentant l'interface de l'application*

– Une API dédiée à l'application qui centralise à travers une classe principale, la classe *Client*, toutes les données nécessaires et relatives à un utilisateur. A la fin de la session d'enregistrement d'un nouveau client, un dossier client est créé, dans lequel se trouve d'une part un fichier (*info.tet*) contenant les informations du client et des informations sur les données ayant servi à l'entraînement des modèles (*path* des fichiers audio et vidéo) et d'autre part les deux modèles (*face et speech*) entraînés du client.
  Lors de chaque session de test ces deux modèles seront utilisés pour scorer et les résultats (accès client ou imposteur) seront enregistrés dans le fichier d'informations du client.
  Les classes de l'API sont les suivantes :
  – Classe *Person* qui regroupe les données informatives, dont un identifiant unique pour chaque client.
  – Classe *Session* qui gère les sessions d'entraînement, d'enregistrement, qui permet de les ajouter mais aussi de les supprimer pour un utilisateur.
  – Classe *Score* qui centralise les résultats au moment du test.
  – Classe *SpeechModel* qui gère la partie traitant de la parole pour la vérification.
  – Classe *FaceModel* qui gère la partie traitant de la vérification du visage.
  – Classe *Client* qui instancie chacune de ces classes et constitue donc la classe centrale de

cette API. Cette classe est instanciée au niveau interface.

– Torch qui est utilisé par les 2 classes *SpeechModel* et *FaceModel* afin d'extraire des coefficients à partir des données brutes (.wav et .ppm), afin d'entraîner les modèles clients et afin de tester ces modèles et les modèles anti-clients pour scorer. C'est la classe *SpeechModel* que j'ai été amenée à redéfinir afin de pouvoir intégrer la nouvelle classe réalisant la vérification du locuteur à partir d'un mot choisi par celui-ci.

3.3.6 Reflexion sur les modifications à apporter à l'application

Pour intégrer le nouveau système SV-UCP au sein de l'API, il était nécessaire de modifier la structure de celle-ci en ajoutant une classe générique virtuelle pour la gestion du *speech*.

Le système initial indépendant du locuteur était défini par la classe *SpeechModel* qui regroupait 5 méthodes :

- *createDataSet()* qui retourne les vecteurs de coefficients issus du traitement du fichier audio.
- *enroll()* qui permet de réaliser l'étape d'entraînement des modèles clients.
- *test()* qui permet de scorer, en fonction des données de test et des modèles clients et anti-clients.
- *load ()* qui permet de charger le modèle client
- *save ()* qui permet de sauver le modèle client

Les différents modèles sont chargés au niveau interface (QT) et sont passés en paramètre du constructeur de la classe.

Pour implémenter la classe réalisant la vérification du locuteur à partir d'un mot de passe personnalisé, il était nécessaire de redéfinir les 5 méthodes citées précédemment. J'ai donc défini une classe virtuelle appelée *SpeechModel* qui déclare ces méthodes comme étant virtuelles. Les deux classes qui définissent les deux systèmes, *TextIndeSpeechModel* (anciennement SpeechModel) et *UserCustomizedSpeechModel*, héritent donc de cette classe générique.

3.3.7 Réalisation d'un programme de test et validation du code

Avant d'intégrer la nouvelle classe *UserCustomizedSpeechModel* au sein de l'interface graphique, il était nécessaire de l'implémenter et de la tester grâce à un programme où les différents arguments nécessaires (modèles, fichier audio) étaient passés de test en ligne de commande. De cette manière, il était bien facile de détecter les problèmes d'implémentation, plutôt que de passer à chaque fois par l'interface graphique.

J'ai donc été amené à redéfinir le constructeur et implémenter chacune des méthodes virtuelles définies dans *SpeechModel*.

Constructeur : CustomizedSpeechModel(HMM^{**} **world-model-hmm ,** *DiagonalGMM** **world-model-gmm ,** *const char** **mlp-model-str ,** ***LexiconInfo****** **lex ,** *Grammar** **grammar,** *const char** **phone-list-str) : SpeechModel()**

A chaque instanciation de la classe, différents modèles sont chargés :

- Le modèle HMM ($world - model - hmm$) *du monde* préalablement entraîné, nécessaire pour l'adaptation du modèle client et pour la normalisation au moment de la phase de test (test du bon locuteur).
- Le modèle GMM ($world - model - gmm$) *du monde* préalablement entraîné, nécessaire pour la normalisation au moment de la phase de test (test du bon mot de passe).
- Le modèle MLP ($mlp - model - str$) préalablement entraîné, utilisé pour réaliser l'inférence au moment de la phase d'enrollement. Dans ce cas, je ne passe en paramètre que le *path* du modèle et je serai amener à construire mon MLP durant la phase d'inférece.

- Une liste de phonème($phone - list - str$) qui liste les 36 phonèmes de la langue française et nécessaire à Torch.
- Une grammaire (*grammar*), nécessaire aussi à Torch lors des phases d'adaptation et de test. Ici il s'agit d'un grammaire libre puisqu'aucune contrainte n'est imposée sur le mot de passe.

createDataSet($Session^{**}$ **raw-session , ***int* **n-raw-session)**

Cette classe réalise l'extraction MFCC et retourne les vecteurs de coefficients, en l'occurence 26 coefficients pour chaque trame du fichier audio (sachant que l'on prend une trame toutes les 10 ms). Les paramètres qui sont passés lors de l'appel de la méthode sont les sessions (d'enrollement ou de test) qui ont servi à la capture des données audio (et qui contiennent donc le *path* de ces données), ainsi que le nombre de ces sessions. Ainsi cette méthode sera appelée au moment de l'enrollement et au moment du test.

enroll($Session^{**}$ **train-data , ***int* **n-train-data) :**

C'est l'implémentation de cette méthode qui fut la plus compliquée, car elle réalise à la fois l'inférence des cinq mots de passe et l'adaptation des cinq modèles à partir des données issues des cinq répétitions. Les paramètres passés à cette méthode sont ceux passés à *createDataSet()*. Elle regroupe les étapes suivantes :

- Appel de la méthode *CreateDataSet()* pour pouvoir obtenir les coefficients issus de l'extraction.
- Création du modèle hybride HMM/MLP, en créant indépendamment chacune de ces machines, puis en associant les sorties du MLP aux distributions *à priori* des différents états du modèle HMM ergodique.
- Génération et stockage de la transcription phonétique pour chaque répétition.
- Création du modèle HMM qui va être adapté et dont les distributions d'émission avant adaptation sont celles du modèle HMM *du monde* chargé au constructeur.
- Réalisation de l'adaptation des 5 modèles clients.

test($Session^{**}$ **test-data , ***real** **results)**

Cette méthode réalise le scorage pour les 5 modèles clients et pour les 5 modèles HMM anti-clients en utilisant la grammaire issue de chacune des transcriptions phonétiques et pour le modèle GMM. Le score final est alors retourné et traité par la classe *Client* pour y être combiné avec le score obtenu par la reconnaissance du visage puis par la classe *Score* pour être sauvegarder.

saveXFile($XFile^{*}$ **xfile)**

Cette méthode permet de sauver les 5 modèles au sein d'un même fichier (le fichier SpeechModel, passé en paramètre ici sous le nom *xfile*). Elle permet aussi d'enregistrer, sous forme de *tags*, différentes informations utiles telles que le nombre de phonèmes, le nombre d'états par modèle de phonème, le nombre de gaussiennes, le nombre de répétitions du mot de passe, la transcription phonétique trouvée par le système hybride HMM/MLP pour chacune de ces répétitions.

loadXFile($XFile^{*}$ **xfile)**

Cette méthode permet de charger les modèles et les différentes valeurs qui viennent d'être citées.

Cette implémentation a été validée lorsque les résultats obtenus correspondaient aux résultats obtenus à partir de la plate-forme de recherche. Le code de cette classe (voir annexes B et C) n'a été validé que lorsque Valgrind ne détectait plus aucun problème de mémoire et que le debugger GDB ne détectait aucun problème particulier lors de l'éxecution.

3.3.8 L'intégration de la classe au sein de l'application

Le but du projet était d'intégrer ce nouveau type de vérification en modifiant le moins possible le code existant. Ainsi en créant cette classe virtuelle *SpeechModel*, le modèle de *speech* client pouvait être instancié par cette classe virtuelle, puis défini au sein d'une méthode spécifique (*createSpeechModel()*) où les différents modèles seraient chargés et le bon constructeur appelé selon le choix de l'utilisateur concernant le type de vérification qu'il souhaite effectuer. Les différences majeures à effectuer au sein de l'application étaient donc la création et le chargement des modèles nécessaires au constructeur de la classe *UserCustomizedSpeechModel* (créés au niveau interface, puisque QT permet de parcourir les arborescences du système et donc d'accéder à ces données) et le chargement des valeurs de seuil, caractéristiques du système et utilisés lors de l'acceptation ou du rejet d'un client. Cette étape n'a pas posé de problèmes particuliers dans la mesure ou le code avait été validé auparavant.

3.3.9 Phase de test, choix et ajustement des hyperparamètres

Une fois le système intégré, il faut réaliser une phase de test afin d'une part de détecter les bugs éventuels et d'autre part d'ajuster les hyperparamètres. En effet, nous avons vu précédemment (2) que les systèmes étaient dépendants des valeurs de seuil choisies pour la validation.

Ceux-ci ont été déterminés grâce à une expérience effectuée sur une base données francophone, PolyVar [9]. Dès lors, il est possible de simuler des accès clients et imposteurs sur le système et d'exploiter les résultats afin de déterminer les valeurs caractéristiques de seuil (voir annexeA, qui est un script permettant de simuler ces différents accès).

Toutefois, ces simulations ne sont pas réalisées dans les conditions réelles d'utilisation et doivent donc être ajustées par une phase de test.

Il faut donc mettre en place une procédure de test qui permette de récolter suffisamment d'informations pour déterminer des valeurs de seuil plus en rapport avec les conditions réelles d'utilisation.

Je n'ai malheureusement pas eu le temps de réaliser ces expériences et les valeurs de seuil sont toujours celles estimées grâce à la base de données. Toutefois cette tâche peut toujours être effectuée après mon départ, dans la mesure où il s'agit uniquement de tester l'application.

3.4 Difficultés rencontrées et apports techniques

3.4.1 Difficultés

Un tel développement ne va pas sans son lot de situations problématiques. Elles n'ont jamais entravé la réalisation de l'application mais en ont parfois ralenti la progression.

Les problèmes d'entraînements de modèles

Cette phase fut certainement la plus longue et la plus laborieuse du projet dans la mesure où un modèle peut mettre plusieurs jours à s'entraîner suivant le nombre de données d'entraînement et la puissance de la machine utilisée. Pour réaliser l'implémentation de l'application, il était nécessaire de n'utiliser que **Torch** pour la gestion de l'intelligence artificielle. Or pour valider son système, le doctorant en charge du projet avait lui aussi entraîné des modèles et développé une plate-forme informatique de recherche composée de programmes réalisant de manière indépendante chacune des phases du système : ses modèles, en particulier le MLP, avait été entraînés avec des coefficients générés par HTK.

De ce fait, lors de l'entraînement de mes modèles, les résultats étaient nettement moins bons que les siens et il était clair que ces modèles ne seraient pas utilisables.

Il a donc fallu apporter un soin particulier à reproduire avec Torch une extraction de coefficients exactement similaire à celle effectuée par HTK.

Le traitement des données "à la volée" entre les différents blocs

Comme je viens de l'expliquer, des programmes informatiques indépendants réalisaient chacune des phases du système.

Chacun de ses programmes récupérait les données de manière statique et écrivait ses résultats dans un fichier.

Or au moment de l'intégration au sein de l'application de démonstration, les données devaient être gardées en mémoire, et traitées à la volée, ce qui a engendré de nombreux problèmes de mémoire et d'intégrité des données.

Par ailleurs, l'entraînement des cinq modèles ralentissait considérablement l'utilisation du système lors de la phase d'enrollement (entre 1 et 2 minutes). Il a donc été nécessaire de baisser le nombre d'itérations lors de cette phase afin de rendre le "délai d'attente utilisateur" plus acceptable.

Les conditions d'enregistrement

Comme je l'ai dit précédemment, le système est sensible aux différents appareils servant à la capture des données ainsi qu'aux conditions d'enregistrement. C'est pourquoi des différences notables ont pu être constatées lors des phases de test. En effet, l'extraction de coefficient effectuée n'est pas robuste face au bruit, et son amélioration pourrait permettre d'améliorer les performances du système. Il fallait donc veiller à toujours conserver les mêmes conditions de capture ou à ré-entraîner les modèles des clients quand celles-ci étaient trop dissemblables aux conditions initiales.

3.4.2 Apports techniques

La réalisation de cette application m'a permis d'apprendre un grand nombre de choses d'un point de vue technique.

L'environnement Unix

Dans un premier temps, ce stage m'a permis de développer mes connaissances de l'environnement Unix. Je ne connaissais en effet ce système que d'après les enseignements dispensés à l'ENIC, et j'ai pu de ce fait appréhender les différents avantages qu'offre son utilisation, à savoir la stabilité du système, l'existence d'une communauté forte, permettant ainsi d'avoir accès à tout type d'informations face à tout type de problèmes, et la qualité et la gratuité des applications disponibles sous cet environnement.

L'utilisation des scripts, le langage Perl

L'ensemble des modèles a été entraîné à partir de scripts écrits en Perl, qui lançaient de manière automatique une série de programmes. L'utilisation de ces scripts m'a permis de gagner du temps lors de la phase d'entraînement des modèles et m'a permis d'aborder un langage que je ne connaissais pas du tout.

La rigueur du C/C++

Les différents enseignements académiques de l'ENIC m'avaient permis d'appréhender sérieusement le développement orienté Objet, à travers des projets de développement notamment en JAVA. Le développement de cette application m'a permis de m'exercer sur les langages C et C++ qui sont plus contraignants, notamment en matière d'allocation mémoire mais qui se révèlent être puissants, modulaires, structurés et concis.

Ainsi je crois que cette expérience m'a donné des bases solides afin d'appréhender d'autres projets de développement informatique.

L'intelligence artificielle

Cette expérience, à travers le développement de l'application mais aussi grâce au haut niveau technique de l'institut, m'a donné l'occasion d'utiliser les outils mathématiques et statistiques liés à l'intelligence artificielle. J'ai ainsi pu comprendre comment fonctionnait les *systèmes capables d'apprentissages*, même si certaines notions et certains systèmes requièrent un bagage théorique assez étoffé.

Chapitre 4

Analyse

4.1 Le réseau informatique et le Système d'Information

Dans une structure comme l'Idiap, il est important, voire primordial de définir un système d'information complet et de concevoir un réseau informatique performant (en terme de fiabilité, rapidité et ressources machines). En effet, l'Idiap est spécialisé dans les *Computer Sciences*, ce qui implique un besoin important en matière de ressources machines et d'espace disque.

4.1.1 Le LAN

Le service Système de l'Idiap, composé de 5 personnes, assure le fonctionnement d'un réseau hétérogène d'environ 90 machines. La plupart des machines du réseau tourne sous environnement Unix (Solaris 7 et Red Hat 7.3) alors quelques machines sous environnement Windows (Win2000) sont destinées au personnel administratif ou à des tâches bien précises. Chacune des machines du LAN est accessible par sessions SSH (ouverture des sessions distantes sécurisées). Il est donc possible de lancer des processus sur n'importe quelle machine du réseau (et donc de choisir la plus performante à un instant t). Par ailleurs, quelques machines à grosses ressources, les *clusters*, sont dédiées à la gestion de gros processus. Les *jobs* soumis sont alors managés automatiquement par la machine. Ces ressources, aussi impressionnantes soit elles, ne sont jamais suffisantes, et les ressources machines et l'espace disque ne cessent de croître au sein de l'institut.

4.1.2 Une information centralisée

Toutes les informations relatives aux activités réalisées à l'Idiap, sont centralisées au sein d'un Système d'Information On-Line. A travers celui-ci, on trouve un grand nombre d'informations d'ordre technique (liste des machines avec caractéristiques techniques, comment runner un processus sur un cluster, comment se logger sur une machine distante, comment imprimer ...) mais aussi des informations d'ordre administratif (planning des présentations, réservation de salles).

4.2 Les différences majeures entre une entreprise et un institut de recherche

4.2.1 Mesures de performances

Au sein d'une entreprise, la valeur peut se mesurer selon certains critères prédéfinis:

- Les parts de marché.
- Le chiffre d'affaire.
- Le rendement.
- La rentabilité.

La valeur de l'activité de recherche est beaucoup plus difficile à évaluer. Elle concerne avant tout les équipes de chercheurs qui expérimentent, qui manipulent, qui analysent. Elle est évaluée par les pairs de la thématique de recherche ou du domaine scientifique concerné à travers :

- La qualité des différentes publications proposées (mesurable par un taux d'acceptation).
- L'intérêt et l'accueil des présentations effectuées lors des différents séminaires.
- L'évaluation des jurys de thèse amenés à évaluer le travail des doctorants de l'entité.

Mais cette qualité concerne également la qualité des méthodes employées par les chercheurs pour obtenir leurs résultats. Dans ce cas, il est nécessaire d'associer les services d'administration et de gestion qui leur fournissent des appuis et des services répondant de façon adéquate à leurs besoins (4.1). La qualité en recherche concerne donc un ensemble d'acteurs mobilisés de façon collective selon leurs compétences, pour concourir à l'atteinte d'objectifs communs.

Par ailleurs, contrairement à une activité commerciale, la démarche d'une équipe de recherche est une démarche "volontaire", où l'activité n'est pas liée au développement d'un besoin ou d'un produit mais plutot à la production de connaissances autour d'un thème précis. Si les laboratoires d'analyse et de recherche recherchent avant tout "l'excellence" dans leurs investigations, c'est qu'ils en perçoivent les enjeux et l'intérêt. Les bénéfices attendus (en terme économique et en terme de reconnaissance) par la qualité des travaux de recherche doivent compenser les investissements consentis en temps de travail, en équipements et investissements.

4.2.2 Gestion de projet

La gestion de projet en milieu industriel suit toujours une méthodologie très rigoureuse, de par la réalisation du cahier des charges et d'une planning prévisionnel. Elle donc est limitée en terme de temps et souvent en terme de financements. A l'inverse, les projets de recherche n'ont pas de limite dans le temps, puisque la résolution d'un problème soulève souvent d'autres problèmes, et que les questions de budget ne sont pas gérées par les chercheurs, même si l'utilisation "naturelle" de technologies "libres" (spécifiquement dans la recherche liée à l'informatique) réduit considérablement les coûts.

4.3 Le management au sein de l'institut

Au cours de ma spécilisation de quatrième année, j'ai choisi de suivre l'option (Management des Equipes et des Compétences en Environnement TIC). Cette nouvelle option nous a donné quelques clefs intéressantes afin de mieux appréhender les attitudes managériales au sein des entreprises.

J'ai ainsi pu, au cours de ce stage de fin d'étude, essayer d'analyser quels étaient les moyens mis en oeuvre afin de faire travailler ensemble des chercheurs venant du monde entier, tout en étant conscient qu'un institut de recherche scientifique était très différent d'une entreprise traditionnelle, et en particulier pour les questions de management.

4.3.1 La notion d'interculturalité

Au sein de l'institut, environ 20 nationalités sont représentées. La langue anglaise est donc la langue officielle utilisée au sein de l'Idiap : l'ensemble des réunions, des présentations se font dans cette langue. Toutefois, le milieu de la recherche est par nature international, et la collaboration entre les chercheurs de différentes nationalités se fait de manière naturelle.

4.3.2 TAM : Tuesday Afternoon Meetings:

Depuis Octobre 2002, un créneau d'une heure par semaine est reservé afin de permettre à un des membres du staff scientifique (docteurs, doctorants ou stagiaires) de présenter ses activités et l'état d'avancement de ses recherches. Cette présentation est généralement suivie d'une discussion autour du thème abordé. Le but de cette initiative est d'une part d'offrir une visibilité de tous les travaux de recherche effectués au sein de l'institut à l'ensemble des collaborateurs, d'essayer d'en déduire eventuellement de nouveaux thèmes ou axes de recherche, mais aussi de permettre aux chercheurs les moins expérimentés de "s'entraîner" en vue des différents séminaires auxquels ils seront amenés à participer et au cours desquels ils présenteront l'état d'avancement de leurs recherches.

Conclusion

La vérification du locuteur basée sur un mot de passe personnalisé est à présent intégrée à l'application de démonstration. Les deux modes de vérification du locuteur (Texte-indépendant et SV-UCP) sont couplés à la vérification du visage et l'utilisateur peut choisir lequel de ces deux modes il souhaite tester. Les tests effectués ont montré que le système fonctionnait même si les performances ne sont pas encore optimisées. Cette phase d'optimisation (ajustement de la valeur de seuil d'après les résultats récoltés sur un échantillon représentatif de testeurs) n'a pas pu être réalisé pour une question de temps. Toutefois, cette tâche ne requiert que du temps et pourra être réalisée par la suite au sein de l'institut.

Ce projet fût ma première réelle expérience de développement informatique (autre que les projets réalisés à l'ENIC). Les outils que j'ai utilisés, à savoir **Torch** et **QT**, m'ont permis d'accroître mes compétences en langage orienté Objet mais aussi dans la création d'interface. En effet, **QT** constitue une alternative intéressante aux environnements payants et offre l'énorme avantage de porter les applications qu'il permet de développer sur tous les types de systèmes. J'ai pu également progresser dans le domaine de l'intelligence artificielle en général, et dans le domaine de la reconnaissance de la parole en particulier, puisque ces domaines n'avaient été que peu abordés durant les enseignements académiques.

Par ailleurs, les divers stages que j'ai déjà effectués lors de mon cursus m'avaient permis d'intégrer différents types d'entreprises : Multinationales, Start-Up, Administration, ou encore PME. Aussi, cette expérience au sein de l'Idiap constitue ma première expérience au sein d'une institution de recherche. De cette manière, j'ai pu découvrir le monde de la recherche, ses méthodes de travail, et constater à quel point une telle structure est différente d'une entreprise "classique", de par les objectifs, les moyens mis en oeuvre et les contraintes (ou plutôt l'abscence de contraintes).

Pour toutes raisons, je considère cette étude en entreprise comme étant réussie, puisque les objectifs ont été atteints, les notions acquises pendant l'option de spécialisation ont pu être mises en pratique et j'ai pu découvrir un environnement de travail nouveau, qui m'ouvre de nouvelles perspectives de projet professionnel.

Bibliographie

[1] S. Furui. *An Overview of Speaker Recognition Technology.* ESCA workshop on Automatic speaker Recognition, Identification and Verification, pp. 1-9, 1994, Martigny, Switzerland.

[2] *Traitement de la parole.* resses polytechniques et universitaires romandes, 2000.

[3] Thomas F.Quatieri Douglas A. Reynolds and Robert B. Dunn. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, 2000, 19-41.

[4] J. L. Gauvain and C.-H. Lee. Maximum a posteriori estimation for multivariate gaussian mixture observation of markov chains. *IEEE Transaction on Speech Audio Processing*, April 1994, Vol 2, pp. 291-298.

[5] L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of IEEE*, 1989, Vol. 2, pp. 257-286.

[6] A. Viterbi. Erroe bounds for convolutional codes and asymptotically optimum decoding algorithm. *IEEE trans. on Information Theory*, p. 260-269, 1967.

[7] M. F. BenZeghiba and H. Bourlard. User-customized password hmm based speaker verification. *Proceedings of the COST275 Workshop on the Advent of Biometrics on the Internet*, 2002, pp 103-106, Rome, Italy.

[8] M. F. BenZeghiba and H. Bourlard. Confidence measures in multiple pronunciations modeling for speaker verification. *Proceedings of the ICASSP*, Vol 1, pp. 389-392, Montreal , Canada.

[9] A. Constantinescu G. Chollet, J.-L. Cochard, C. Jaboulet, and P. Langlais. Swiss french polyphone and polyvar: telephone speech databases to model inter- and intra-speaker variability. Technical report, IDIAP Research Report, IDIAP-RR-96-01, 1996.

[10] H. Bourlard and N. Morgan. *Connectionist Speech Recognition: A hybrid approach.* Kluwer Academic Publisher, 1994.